ROUTLEDGE HANDBOOKS

The Handbook of Qualitative and Quantitative Content Analysis

Introduction to Classical, Digital, AI-supported, and Automated Data Analysis

Christian Schneijderberg, Oliver Wieczorek, and Isabel Steinhardt

THE HANDBOOK OF QUALITATIVE AND QUANTITATIVE CONTENT ANALYSIS

Translated from German and substantially updated, *The Handbook of Qualitative and Quantitative Content Analysis* is a comprehensive handbook which offers an application-orientated introduction to qualitative and quantitative content analysis methods.

The book provides explanations for beginners from bachelor level onwards on how to select an appropriate qualitative or quantitative content analysis method and how to use the chosen method(s) depending on research interest and amount of data. Part 1 defines the basics of qualitative and quantitative content analysis and empirical research, including research quality conventions, how to do interpretation, and how to introduce AI into your workflow; Part 2 is a practical guide to classical qualitative content analysis and semi-automated quantitative content analysis; and Part 3 introduces Python alongside automated techniques such as correspondence analysis, semantic network analysis, sentiment analysis, and topic modeling using generative and deep learning algorithms. Each of these sections are enriched with extensive examples and cover a range of software applications, including AntConc, MAXQDA, Python, and VosViewer.

This is the ideal resource for anyone interested in content analysis research methods across the social sciences, humanities, and data sciences.

Christian Schneijderberg holds a PhD in sociology and is a senior researcher at the International Center for Higher Education Research at the University of Kassel, Germany.

Oliver Wieczorek holds a PhD in sociology and is a senior researcher in governance and organization at the International Center for Higher Education Research at the University of Kassel, Germany.

Isabel Steinhardt is a professor of sociology of education and head of department of sociology at Paderborn University, Germany.

Taylor & Francis
Taylor & Francis Group
http://taylorandfrancis.com

THE HANDBOOK OF QUALITATIVE AND QUANTITATIVE CONTENT ANALYSIS

Introduction to Classical, Digital, AI-supported, and Automated Data Analysis

Christian Schneijderberg, Oliver Wieczorek, and Isabel Steinhardt

Routledge
Taylor & Francis Group
LONDON AND NEW YORK

Designed cover image: Getty Images

First published in English 2026
by Routledge
4 Park Square, Milton Park, Abingdon, Oxon OX14 4RN

and by Routledge
605 Third Avenue, New York, NY 10158

Routledge is an imprint of the Taylor & Francis Group, an informa business

This book is a translation of *Qualitative und quantitative Inhaltsanalyse: digital und automatisiert*

© 2026 Beltz Juventa in the publishing group Beltz Weinheim Basel

The right of Christian Schneijderberg, Oliver Wieczorek and Isabel Steinhardt to be identified as authors of this work has been asserted in accordance with sections 77 and 78 of the Copyright, Designs and Patents Act 1988.

The Open Access version of this book, available at www.taylorfrancis.com, has been made available under a Creative Commons Attribution-Non Commercial-No Derivatives (CC-BY-NC-ND) 4.0 International license.

Any third party material in this book is not included in the OA Creative Commons license, unless indicated otherwise in a credit line to the material. Please direct any permissions enquiries to the original rightsholder.

We acknowledge support for the publication cost by the Open Access Publication Funds of Paderborn University and the University of Kassel. The authors of this handbook are very grateful for their financial support of about one third each of the open access book processing charges (OABPC). The remaining OABPC third was procured by Christian Schneijderberg.

Trademark notice: Product or corporate names may be trademarks or registered trademarks, and are used only for identification and explanation without intent to infringe.

British Library Cataloguing-in-Publication Data
A catalogue record for this book is available from the British Library

ISBN: 978-1-032-80313-5 (hbk)
ISBN: 978-1-032-80314-2 (pbk)
ISBN: 978-1-003-49639-7 (ebk)

DOI: 10.4324/9781003496397

Typeset in Optima
by Apex CoVantage, LLC

CONTENTS

Preface *vii*

1 Introduction 1

PART 1
Basics of qualitative and quantitative content analysis and empirical research 19

2 Definitions of qualitative and quantitative content analysis, and inductive and deductive research approaches 21

3 Know your data: possibilities and limitations of text, numeric, video, and pictographic data, and primary and secondary studies 41

4 Research quality conventions: a guide for good quality empirical research 61

5 Data interpretation: a practical guide 105

PART 2
Practical guide to classical qualitative content analysis and semi-automated quantitative content analysis **143**

6 Deductive qualitative content analysis 145

7 Introduction to inductive qualitative content analysis 174

8 Introduction to quantitative content analysis 208

9 Deductive quantitative content analysis: a bibliometric literature review 238

10 Artificial intelligence and large language model-powered chatbots to support qualitative content analysis 271

PART 3
Practical guide to automated big data content analysis **299**

11 Automated content analysis: basic concepts and useful tips prior to data collection and data analysis 301

12 Getting started with Python 323

13 Data preprocessing 356

14 Introducing and exploring a dataset statistically 385

15 Automated content analysis using relational methods 407

16 Sentiment analysis 482

17 Topic modeling with Latent Dirichlet Allocation 525

18 Topic modeling with BERTopic 558

References *593*
Index *615*

PREFACE

This first edition of the *The Handbook of Qualitative and Quantitative Content Analysis: Introduction to Classical, Digital, AI-supported, and Automated Data Analysis* in English is a substantially revised and amended version of the German handbook, which was published in 2022. Using chapters of the German handbook in class, we learned how some *things* could have been explained in an easier way, what is missing in the explanation of a method, and what quickly becomes outdated, especially in the fast-paced computational social sciences, advancements in Python programming, and the development of more powerful artificial deep neural networks, which are computationally expensive. What remained the same is our pragmatic understanding of choosing one method or a combination of methods for content analysis according to the research question(s), amount of data, and intended qualitative depth as well as quantitative breath of data analysis and interpretation. In this sense, we believe that methods are tools for systematic research and not exclusive religious belief systems.

The German edition of the handbook contains 12 chapters, while the English edition has been expanded to 18 chapters, although only 4 chapters are entirely new. The new chapters in the English handbook are "Quality conventions: a guide for good quality empirical research" (Chapter 4), "Data interpretation: a practical guide" (Chapter 5), "Large language model-powered chatbots to support qualitative content analysis" (Chapter 10), and "Topic modeling with BERTopic" (Chapter 18).

For the English handbook, we considerably revised the introduction (Chapter 1) and all chapters in Part 1: "Basics of qualitative and quantitative content analysis and empirical research" and Part 2: "Practical guide to classical qualitative content analysis and semi-automated quantitative content analysis." We substantially revised Chapters 11–17 on "Automated content analysis: basic concepts and useful tips prior to data collection and data analysis," "Getting started with Python." We now solely focus on one programming language

and decided to exclude R to keep track with the fast pace of development in the fields of natural language processing and deep learning – "Data processing," "Introducing and exploring a dataset statistically," "Automated content analysis using relational methods," "Sentiment analysis as inductive-quantitative content analysis," and "Topic modeling using Latent Dirichlet Allocation." The substantial revisions in the named chapters focus on providing a basic understanding of a method for quantitative content analysis (Chapters 15–17) and the computational social sciences research process (Chapters 11–14).

In the publication and writing processes of this handbook, we were supported by several people. First, we would like to thank Isabella Czedik-Eysenberg for suggesting improvements to parts of the Python code in Chapters 13–18. We owe a thank you to Kerstin Gackle for the first draft of the English translation of Chapters 1–3 and Chapters 6–10, which we then worked on further. Thank you also to Kate Bird for the English proof-reading of Chapters 4 and 5. At Routledge, we would like to extend sincere thanks to the editors Matt Bickerton, Eleanor Tayor, and Adam Wood for their support and organization of the publication process as well as to the Apex CoVantage copyediting and layout team. Finally, we would like to thank the always friendly and helpful Svenja Dilger of our German handbook publisher Beltz Juventa, who enabled the English publication of the handbook.

Kassel and Paderborn, Germany, September 2025

Christian Schneijderberg
Oliver Wieczorek
Isabel Steinhardt

1
INTRODUCTION

This handbook on content analysis is unique in several ways. First, this handbook introduces both qualitative and quantitative content analysis. Accordingly, you get all you need to know about qualitative and quantitative content analysis in one book. You do not have to search for different handbooks to learn about classical deductive and inductive content analysis as well as digital and artificial intelligence (AI) supported and semi- and fully automated content analysis. In addition to a systematic introduction to the basics of content analysis, you will also learn how to combine different methods of content analysis in a mixed methods research design.

Second, the structure and content of this book is unique, as it spans from the smallest to the largest scales, from the most manual to the most automated types of content analysis available to date. In this unique handbook, the first part of the book also introduces you to differences between data (e.g., text and numbers) and what this means for empirical social research, quality conventions for qualitative and quantitative research, and how to do interpretation. Like in the interpretation chapter, we always use empirical examples from our own research to show you how to do content analysis.

Third, the chapters are structured in such a way that they can be read individually. You can use individual chapters, for example, as a student, to learn about a method on your own or, as an instructor, for teaching purposes. In Part 2 (classical and semi-automated content analysis) and Part 3 (fully automated content analysis, incl. machine learning and Python programming) of this handbook, we introduce each method in a step-by-step manner from the beginning of the research process (i.e., the research design) via data collection and processing to data analysis and interpretation as well as drawing conclusions in the end of the research process. The practical introduction to doing research is complemented by how you present your research in a term paper, bachelor or master thesis or publication.

DOI: 10.4324/9781003496397-1
This chapter has been made available under a CC-BY-NC-ND 4.0 license.

Fourth, the introduction to methods of content analysis is supported by instructions for using analysis software. For this, we provide an introduction to AntConc, qualitative data analysis (QDA) software, Python, and VosViewer and explain how to use each software by using empirical examples.

As you read through the book, you may notice that individual chapters differ in terms of language and structure. This is because we had three main authors who drafted specific chapters. And yet the book is a true collaboration as the authors extensively discussed and edited all of the chapters together.

1.1 Brief overview of the content in the individual chapters

Now that you have an idea of how the book is structured, we would like to give you a brief overview of the content you can expect in each of the chapters.

Chapter 2 "Definitions of qualitative and quantitative content analysis, and inductive and deductive research approaches" outlines what we understand with content analysis, what content analysis has to do with communication, what manifest and latent content are, and which types of content analysis are covered in this book. We will also give you a brief overview of the types of qualitative content analysis – which we believe can be divided into inductive and deductive – and show examples of what an interpretation can look like. In addition, we also provide an overview of quantitative content analysis methods. And you will learn about two rules used in qualitative and quantitative social research: the inclusion rule and the exclusion rule, which will help you to position your research.

Chapter 3 "Know your data: possibilities and limitations of text, numeric, video and pictographic data, and primary and secondary studies" covers the specifics of different types of data, for example, text, numbers and videos, and the associated possibilities and limitations for content analysis. We describe the differences between research-produced and process-produced data – from the design (considering data types, research objectives, and operationalization) and data collection (including sampling, data collection, and data type) to the analysis possibilities (resulting from data storage and types of analysis) and using results and archiving data. Limitations for content analysis also root in the use of data for a secondary study. In comparison to a primary study, data is not collected for answering your research question. However, doing a secondary study means that you do not have to spend time on data collection.

Chapter 4 "Research quality conventions: a guide for good quality empirical social research" introduces the "substantive methods concept" (*substanzieller Methodenbegriff*; Schneijderberg, 2023) for good quality research and provides a step-by-step guide for the entire research process from initial observation to conclusions by focusing on research design, data collection

and processing, and analysis and interpretation. The substantive methods concept – composed of discovery, appropriateness-transparency, and iteration and distancing quality conventions – guides deductive, theory-led and inductive, theory-reflexive approaches to research as well as qualitative small and quantitative big data analysis. In addition to differences associated with different methods, the three quality conventions respect particular aspects of data collection and data type (see Chapter 3). For example, ethnographic observations are documented in field notes by researchers, and interviews are conducted according to a manual guiding the interactive communication between two or more subjects. We believe that all particularities (e.g., data collection and processing) of qualitative methods can be addressed by the discovery, appropriateness-transparency, and iteration and distancing quality conventions. The substantive methods concept can also integrate quality conventions specifying methods for data collected by surveys, quantitative data processing, and statistical analysis. Of course, epistemology, that is, how scientific knowledge is created, and methodology, that is, the logical process of applying a method or methods, differ between qualitative, quantitative, and mixed methods research – but not the basic research process.

Chapter 5 "Data interpretation: a practical guide" shows that interpretation is the single most important part of data analysis, both for qualitative and quantitative research. Data has literally no meaning unless researchers interpret it and attribute meaning to answer a research question. Interpretation is your unique accomplishment and makes a difference between whether you make an original contribution with your research or just summarize and repeat what others say in interviews, write in documents, capture in pictures, or express in a video/film. In content analysis and any other empirical research (e.g., ethnography and surveys), interpretation is based on systematic work with data and theory to produce results and draw conclusions, which are backed by an empirically grounded systematic analysis according to a category scheme. Interpretation cannot be standardized. However, we can provide practical guidance for the interpretation process.

Chapters 6 and 7 on classical deductive and inductive qualitative content analysis focuses on the distinction between the deductive, that is, theory-led, or inductive, that is, data-driven and theory-reflexive, approach. We use the same example in the two chapters on computer-assisted qualitative content analysis to present the fundamental differences between deductive qualitative content analysis (Chapter 6) and inductive qualitative content analysis (Chapter 7). The sample data material is an autoethnographic record by a student from Winter Semester 2020/21, which was an online semester as a result of the Covid-19 pandemic. This practical approach will help you to understand the goals of the respective analysis methods. We also explain the similarities and differences between inductive and deductive data analysis

and the knowledge to be gained in each case. In inductive qualitative content analysis, the categories for data analysis are developed from the respective empirical material. In comparison to inductive content analysis, in deductive content analysis, the categories for the data analysis generated based on theory, conceptual considerations, and/or existing findings from previous empirical studies. The category construction must, of course, be appropriate to the subject under investigation.

Chapter 8 "Introduction to quantitative content analysis" presents an analysis technique that you can use to quantitatively analyze the content of 500 pages or more of text. This technique is suitable for larger interview studies and secondary analyzes of process-produced data (e.g., party platforms or social media). Using the example of a secondary analysis (like in Chapters 6 and 7, the autoethnography on "How are you organizing your daily life as a student at the beginning of the online semester 2020/21?"), the sequential analysis is explained based on the AntConc software (freeware; sociolinguistics or political linguistics) and qualitative data analysis (QDA) software. Guided by the research interest and research question(s), we explain step-by-step how to use search words to systematically understand the text corpus and how to analyze the information found. The semi-automated indexing of the text corpus and quantitative content analysis produces an overview categorized by the search words (keywords), which is also referred to as distant reading. In contrast to in-depth qualitative content analysis, the aim of quantitative content analysis is to record the manifest content and to use this to draw findings, make interpretations, and draw conclusions. However, quantitative content analysis should be followed by in-depth, qualitative content analysis, which is explained at the end of this chapter.

In Chapter 9 "Deductive quantitative content analysis: a bibliometric literature review", we will introduce you to bibliometric literature reviews. You can use this deductive quantitative content analysis method to obtain a systematic overview of publications. To lay the groundwork, we will first explain what publications actually are and what the systematic analysis of literature is based on: citations. As you will need to use databases to carry out analysis, we will then present a selection of databases and their advantages and disadvantages. After these basics are covered, three research questions that can be addressed with the bibliometric literature review will then be presented: (1) the literature review and the question "what are the key publications on a specific topic or in a research field?" (2) mapping and the question "how is a research field (on a specific topic) structured, that is, what connections can be found?" and (3) topic analysis: "what content is discussed in a research field or on a specific topic?"

In Chapter 10 "Artificial intelligence and large language model-powered chatbots to support qualitative content analysis", we provide an overview of how large language models and large language model-powered chatbots

can be used for qualitative content analysis. We explain what large language models are and how they relate to artificial intelligence and chatbots. And to help you really understand how they can be used to analyze qualitative data, we present a literature review that gives you an overview of the February 2025 state of research on the use of large language model-powered chatbots. Then, we will explain how large language models work and provide practical examples of how they can be integrated into your research. Finally, we will provide guidance on the decisions you need to make about the use of large language models.

Chapter 11 "Automated content analysis: basic concepts and useful tips prior to data collection and data analysis" introduces the basic concepts underlying automated content analysis, such as natural language processing and machine learning. We will furthermore define and exemplify different types of machine learning that are important for topic modeling and sentiment analysis approaches. Afterwards, we briefly discuss how to prepare a workflow for automated content analysis and introduce literate programming to structure our coding and documentation process. We then proceed with describing data types and data structures relevant for working with APIs and text as data, before closing the chapter with describing where you may find help with programming tasks online.

In Chapter 12 "Getting started with Python", we provide a gentle introduction into Python. We start with an explanation how to download and install Python, the Anaconda distribution, and Spyder. We will then show you how to install the program packages necessary to conduct automated content analysis. Afterwards, we continue with introducing relevant concepts we will encounter during the successive programming steps, before we summarize what skills we have learned in this chapter and what we will need them for in the following chapters. If you are already familiar with Python, then you may skip this section completely. But if you are a beginner or have started programming recently but are unsure about some of the concepts, you can always consult this chapter and its contents.

Chapter 13 introduces you to preprocessing procedures for big data analysis. These must be applied prior to applying machine learning algorithms for topic extraction and sentiment analysis to ensure the accuracy of the analysis, to reduce biases inherent in the data, and to focus on relevant aspects of the data. In the following, we start with an outline of relevant preprocessing techniques. In this chapter, we demonstrate basic data cleaning procedures, for example how to remove missing values or drop duplicate entries in a data frame. We then proceed with feature extraction techniques on corpus, document, and token level, for example how to create a weighted and unweighted document term-matrix, how to tokenize text documents and filter tokens by their parts of speech, or how to perform named entity recognition.

Chapter 14 "Introducing and exploring a dataset statistically" aims to equip you with the essential tools to explore your big datasets before applying automated content analysis algorithms. To do so, it offers an overview of the dataset and outlines the workflows needed for data preprocessing, exploration, analysis, and visualization. We then demonstrate how to compute descriptive statistics for single variables (univariate) and two variables (bivariate), along with appropriate visualizations.

Chapter 15 "Automated content analysis using relational methods" introduces relational methods for extracting topics from a text corpus. These methods stem from two different paradigms: geometric data analysis and semantic network analysis. Both offer unique advantages. On the one hand, geometric data analysis enables us to integrate multiple data types in a single analysis and allows us to map topics in relation to document metadata. Geometric data analysis encompasses correspondence analysis, multiple correspondence analysis, principal component analysis, and multiple factor analysis, allowing for efficient analysis and visualization of different data types. On the other hand, semantic network analysis captures local meaning structures and quantifies them using centrality measures and community detection algorithms. A key advantage of semantic network analysis is that these local meaning structures are embedded within the broader network structure, which allows us to simultaneously track the evolution of topics and the discourse structure over time.

Chapter 16 introduces you to sentiment analysis and guide you through two distinct approaches. Firstly, we teach you how to apply a lexical approach for assigning sentiment polarities and classes to text documents. Secondly, we show you how to employ a deep neural network for sentiment analysis. Specifically, we show you how to leverage a pretrained BERT architecture for sentiment classification, explain its structure, and training procedure. We also show you how to fine-tune a BERT model to assign custom sentiment values and cover how to evaluate classifiers based on deep neural network algorithms.

In Chapter 17, we introduce you to topic modeling with Latent Dirichlet Allocation (LDA). LDA is a generative model which relies on the bag-of-words approach. LDA assumes that the distribution of topics across documents and tokens results from an underlying process, giving rise to abstracts topics. Against this backdrop, this chapter aims to equip you with the knowledge to understand the LDA algorithm, preprocess the text corpus, automatize topic extraction, select topics for interpretation, calculate and visualize goodness of fit measures, and how to effectively interpret topics.

Chapter 18 introduces BERTopic, a deep learning-based topic modeling approach based on the BERT architecture. To enable you to apply BERTopic, we begin with explaining the algorithms integrated into BERTopic: UMAP for dimensionality reduction; HDBSCAN for topic extraction; and class-based

term frequency-inverse document frequency weighting scheme for identifying the most relevant keyword phrases per topic. You will also learn how to refine your model using deep neural networks, including large language models (LLMs) such as ChatGPT. We then guide you through topic extraction from your text corpus, establishing a topic hierarchy, and reconstructing topic evolution over time. Additionally, we cover topic visualization and coherence scores calculation. We conclude with an overview of BERTopic's advantages and limitations.

1.2 Which content analysis method is the most suitable for you?

Now that you have an overview of this book, you may be wondering which method for content analysis would be the most appropriate for your research. We would suggest that you use a simple heuristic based on three parameters to decide which method might be most suitable for you (see Figure 1.1). The parameters are as follows:

1. Research interest:
    1. Depth of analysis
    2. Breadth of analysis
2. Scope of data:
    1. Amount of data
    2. Sample size
    3. Sampling process

However, the research interest and scope of data heuristic only serve as a general guide to help you decide which method may be best for your content analysis. Before you turn to a specific content analysis method, however, we recommend that you read the two introductory chapters on the method of content analysis (Chapter 2) and the possibilities and limitations of data (Chapter 3). This is particularly the case if you are working with empirical data and/or using content analysis for the first time.

1.2.1 Research interest

We find it important to be pragmatic when selecting a suitable content analysis method for your study. To select a method for your research, you should first consider whether you want to do a qualitative, in-depth analysis or a quantitative, broad analysis.

Depth of analysis: The category *depth of analysis* is aimed at an in-depth, qualitative, and interpretative understanding of empirical data. This also involves working out levels of meaning and immanent meanings that go beyond the manifest text level. This is a time-consuming task that requires close reading, in-depth reflection, and successive analysis and interpretations of specific sections, references, and levels of meaning. You cannot achieve this by simply reading your text material once, because then you, symbolically, only scratch the surface instead of doing content analysis. In empirical social research, in-depth analysis is usually conducted by analyzing either an individual case (e.g., one interview and documents by a single organization) or a small number of cases. This is why the term *small-n studies* is also used for qualitative, in-depth research in methods literature (Ebbinghaus, 2009; Schneijderberg & Götze, 2021; Smelser, 2003).

Breadth of analysis: In contrast to the in-depth analysis and interpretation of empirical material in small-n studies, large-n studies include large numbers of observations with the aims of pattern recognition or statistical inference. There is no fixed demarcation between small-n and large-n studies. Roughly speaking, you can assume that a qualitative study is a large-n study if there are more than 25 interviews or more than 35 shorter interviews when compared to other qualitative studies with a focus on the in-depth exploration of a phenomenon. A large-n study is given, if the assumption of the central limit theorem is met, meaning that techniques can be applied on the data which rely on the approximation of a normal distribution. In statistics, you can assume that a study is a large-n study if there are more than 80 cases, but this depends on the method.[1] The aim of large-n studies is usually to find abstract, more general patterns of a phenomenon. As linguistic features, such as dependencies between words, semantics, or syntactics, are much more complex compared to single variables, a large-n study comprises of tens of thousands text documents at least.

1.2.2 Scope of data

The term *scope of data* was chosen here because there is no direct correlation between the number of documents, the amount of text they contain, and the effort required to analyze the texts. To estimate the scope of data, you must take into account the three factors of amount of data, sample size, and sampling process and therefore also the time that you will need to read and analyze the individual texts that are part of your research. The combination of these three multiplication factors can very quickly increase the amount of time you need for your research.

Amount of data: The amount of data can vary significantly and may not depend on the number of cases you want to analyze. For example, take the case of one interview that lasts approximately one hour. The one-hour interview results in a transcript (i.e., the written record of the interview) of around 50 pages with 1,500 characters (excluding spaces) per page. In contrast, the case of a tweet would only consist of a maximum of 240 characters. As you

can see, the scope per case can vary greatly, which means that you need to consider what you can handle in terms of the breadth and depth of the analysis. For example, the scope of around 312.5 tweets corresponds to the scope of a one-hour interview. In this case, it would be possible to analyze the one-hour interview using qualitative content analysis. But you will need much more data than that of 312.5 tweets for automated content analysis. And if you decide to apply an automated procedure and train an machine learning algorithm from scratch, you need thousands, if not millions of tweets to program a meaningful classifier which is able to extract topics and sentiments from text. Alternatively, you can apply an algorithm pretrained on a similar dataset to analyze a small text corpus. However, keep in mind that this one is trained on millions or billions of text documents. The rule of thumb here is that the more data you have, the more sense it makes to choose a method of content analysis that focuses on the breadth of the analysis rather than the depth of analysis.

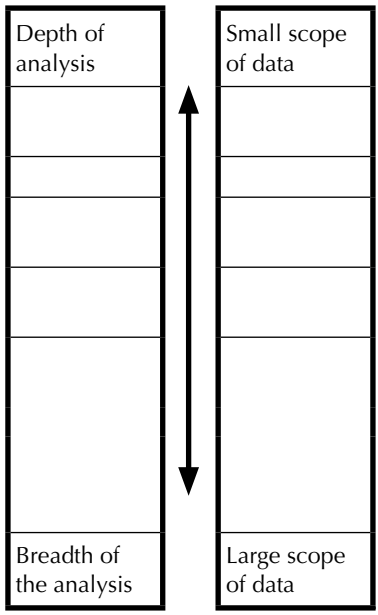
Sample size: Using selection procedures called sampling strategies, you can define which data to collect, for example according to specific criteria to select a certain number of online comments on YouTube from the hundreds of millions of comments that are there online. In addition to sampling strategies for data collection, selection criteria (e.g., categories guiding content analysis) can be applied to reduce the amount of data originally available and then carry out an in-depth analysis. Please always remember to document how you select a sample or reduce the sample size according to clear and comprehensible criteria.

Sampling process: Sampling, like case(s) selection, is very important in empirical research. The sampling process itself can be very time-consuming and require a great deal of thought. For example, it might be interesting in the context of your theoretical framework to investigate students from different subject semesters and of different genders in terms of how well they are integrated into student life. If this is the case, you should consider whether you want to represent the individual groups (discipline by semester by gender) proportionally in your interviews, or whether the groups that are less represented overall in the student body should be represented more strongly in your interviews. You would therefore draw a proportional or anti-proportional stratified sample here. Think carefully about what (potential) differences the groups might have that would allow you to analyze your phenomenon in a systematic way. You could also first select a few people, interview them, and analyze the interviews, and then based on your intermediary analysis, you could select additional people to interview who would offer the greatest possible contrast to those you first interviewed. This method is called theoretical sampling and was developed in the context of grounded theory (Glaser & Strauss, 2017).

1.2.3 Heuristic for selecting a suitable content analysis method

These thoughts underscore that you will make many crucial decisions regarding the amount of data, for example based on the selected sampling procedure

Figure 1.1 Heuristic for selecting a suitable content analysis method.



	Chapter
Deductive qualitative content analysis	6
Inductive qualitative content analysis	7
Inductive quantitative content analysis (semi-automated)	8
Deductive quantitative content analysis (semi-automated)	9
Relational methods (geometric data analysis and network analysis)	15
Sentiment analysis	16
Topic modeling (LDA)	17
Topic modeling using deep learning (BERT)	18

(e.g., sample size, sample population, and stratification by subgroups). These decisions should not be taken lightly. Considering the parameters of research interest, that is, depth of analysis and breadth of analysis, and scope of data, depending on amount of data, sample size, and sampling process, Figure 1.1 presents them as two controls that can help you to understand the relationship between depth and breadth as well as between the amount of data and data selection. In contrast, the sampling process itself and the effort involved vary too much to be depicted in the figure.

If the volume control is at the top – as in the case with an in-depth analysis and a small scope of data – then an inductive qualitative content analysis is the best choice. This means that you either have a small amount of data or have a large amount of data and have sampled a small subset that focuses on a specific topic. In terms of in-depth analysis, inductive qualitative content analysis is most suitable here as it is an interpretative method that requires a lot of time for a small amount of text.

You will need somewhat less time for interpretation with deductive qualitative content analysis as the category system is built up from theory and/or empirical studies and does not have to be developed from the data as with inductive qualitative content analysis. Since you will need less time for interpretation, the breadth of the analysis (i.e., the scope of data) is greater.

As you can see from Figure 1.1, the two qualitative methods are the most suitable for in-depth analysis and a small scope of data. The other methods for content analysis are quantitative methods that you can use to process an increasingly large scope of data as they involve working with semi-automated

and fully automated software procedures. Using the inductive quantitative content analysis as presented in Chapter 8 allows you to include a larger scope of data despite the inductive approach. This is because keywords are used to index the data material, and findings, interpretations, and conclusions can then be drawn on this basis. You can use the bibliometric literature review for deductive quantitative content analysis to analyze a relatively broad scope of data, which is then analyzed in 'medium' depth. Through the deductive selection of data, search criteria are used to automatically analyze metadata from publications. These results must then be interpreted in depth.

Relational methods, including geometric data analysis and semantic network analysis (Chapter 15) go beyond keyword coding. Instead, they extract words or phrases from text documents and use dimensionality reduction techniques to find co-occurring words and phrases. In turn, the words and phrases link specific positions in the potential topic dimensions to text documents. Both, text documents and words or phrases, hint at human-interpretable topics. This provides clues as to which text documents you should analyze in-depth. The depth of analysis of each individual text is generally rather low.

The depth of analysis of individual texts is even lower in sentiment analysis (Chapter 16). Its aim is to classify the sentiment of text documents into discrete classes (e.g., positive, neutral, or negative) or polarity score, based on the linguistic features of texts. The degree of complexity of these linguistic features ranges from low (counting of predefined adjectives) to high (including dependencies, grammar, semantics, morphology). In other words, it aims to characterize the sentiments carried in large text corpora, which we can subsequently link to extracted topics, authors (including their properties such as age, gender, political party, etc.), or sources (e.g., authors of a newspaper outlet). Nonetheless, we must always check the sentiment classifications for anomalies by analyzing a sample qualitatively. Anomalies either arise from the classification process itself or may hint at a previously undiscovered pattern or phenomenon worth studying.

Topic modeling techniques (Chapters 17 and 18) are also used to classify text documents, and as such do not put much weight on single documents to do so. In fact, the underlying algorithms cannot extract topics if our text corpus yields only a few texts. These mathematical beasts are specialized to detect patterns in large text corpora, either by assuming an underlying distribution on how text documents are written (e.g., in regard to the vocabulary in case of the LDA), or structured (vocabulary, word dependencies, morphology, semantics, syntax in case of BERTopic). The information is then used to infer what topics might be present in the data, as well as which words, phrases, and documents might represent those best. To make sense of the topics, these text documents must be interpreted qualitatively. However, if your text corpus contains 100,000 documents, and you extract 25 topics, then you usually end up with five to ten typical documents per topic (250 maximum, or about 0.025% of the documents) for in-depth interpretation. Depending on the length of these documents, this might still take a lot of time. And – last but not least – consider

also the time (programming, training, and evaluation) and resources (energy, hardware) consumed by this approach, especially if you have to train your own topic or sentiment classifier first.

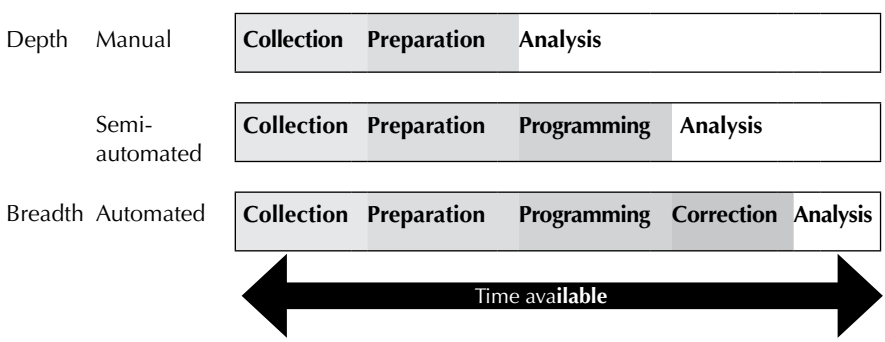
1.2.4 Heuristic for planning the workload of content analysis methods

Based on our experience, you will spend as much time and work on automated, quantitative content analysis on the qualitative analysis of texts. Consequently, it does not necessarily mean that you put less effort into content analysis when you apply automated content analysis tools compared to a qualitative analysis of texts. Instead, you will spend the time you have available in a different way. With a topic model, you may hardly spend any time reading individual texts. You will never be able to read all of the texts (especially if there are hundreds of thousands of them) in full length or even in parts. However, you should allow much more time to prepare and clean the data and for programming. It is important that you take the time to find errors in the data, which undetected would give you distorted results that either cannot be interpreted or in the worst case would give an inaccurate picture of your phenomenon. Above all, you should allow plenty of time to plan and document your work process (i.e., your workflow).

In qualitative data analysis (defined as 'manual,' although we expect that you use QDA software, in comparison to 'automated,' Python-programmed LLM use in Figure 1.2), you will spend most of your time on in-depth analysis and interpretation. Collecting interview data and transcribing it also requires time, but this will take up a smaller proportion of your overall time compared to the time you spend on the actual analysis. We have created a second heuristic (Figure 1.2) to show this connection, namely, how you will use the time for data collection, data processing and preparation (possibly incl. programming), and data analysis and interpretation within your research process.

Now that you have an overview of what to expect in this book, we would like to recommend that you start by reading Chapters 2 and 3 and then work your way through the book using the decision-making aids we have provided.

Figure 1.2 Heuristic for the workload of content analysis methods.



We wish you lots of success and hope you will have fun with the content analysis methods!

1.3 Research process and structure of research papers

Conducting an empirical study is a most demanding endeavor. First, you need to know at least something about the phenomenon under study, for example observed characteristics and social context. Second, you need to select an appropriate method and know how to apply this method for studying the phenomenon. Third, you need to know about theory.

In a methods handbook, we can only introduce to you the know-why for choosing an appropriate method according to the subjects and/or objects of your study and the know-what to do in a research process. The know-how you only learn by using a method when doing research, for example starting to practice research in a small project.

1.3.1 Schematic representation of research with empirical data

The diverse characteristics of data and the associated challenges for the empirical research process are shown later in a step-by-step flow chart of the research process (Figure 1.3), which is then used to give a chapter structure for research papers (Figure 1.4). In order to reduce redundancies, Figure 1.4 presents a keyword-based, schematically organized summary of the previous content. The Flowchart for Research with Empirical Data (Figure 1.3) is a good summary of the information presented in this section and can be used as a guide when you are planning your empirical social research.

1.3.2 Structural outline of research papers (e.g., term papers, bachelor's theses, master's theses, and dissertations)

The broad structure of the content analytical research process can be applied to research papers in a relatively clear and structured way (Step 12: Publish in Figure 1.3). This structural outline forms the core of research papers – be it student research projects, journal articles, or books – and this is independent of their length, although a bachelor's thesis and certainly a dissertation have more subchapters than a term paper. The rough outline of academic papers also fits well with the reading habits of lecturers and examiners, who, as frequent readers, expect a chapter structure similar to that of a suspense novel. You can, of course, always be a little creative, even with chapter and section titles, but you should make it easy for yourself and others, especially if they, for example as instructors, will be evaluating your scientific paper.

In the structure of academic papers presented in Figure 1.4, a distinction is made between deductive, that is, theory-led empirical social research, and inductive, that is, theory-reflexive, explorative, empirically driven social

Figure 1.3 Flowchart for research with empirical data.

Step 1	Outline the topic for your research idea and formulate clearly the aim of the inquiry in writing, at least in bullet points, in your research documentation (appropriateness-transparency quality convention of the research; see Section 4.3.2).
Step 2	Research previous findings on the topic in general (= contextualization) and specifically in order to identify the research gap. Summarize the state of research or lack thereof in writing. This will help you to use empirical and/or theoretical information to specify your research idea (Step 1), justify your research question(s), and if necessary, form your hypothesis.
Step 3	Establish hardware and software requirements for data collection and analysis based on the operationalized materiality of the data type.
Step 4	Clarify data access and, if necessary, conduct sampling.
Step 5	Depending on the research question(s) and identified state of research or lack thereof, choose and justify either a(n) 1. inductive, for example, explorative, empirically driven, or grounded theory-style approach; or 2. deductive, theory-led approach; or 3. mixed methods research design to investigate the social phenomenon.
Step 6	Collect data according to well-founded rules and/or search words, for example in automated content analysis according to a clearly stated and reproducible workflow.
Step 7	Clean and process the data and create a database. If necessary, take into account special requirements (e.g., anonymization) for the creation of student-use and/or research-use files for data reuse by third parties.
Step 8	Analyze the data: • Method-based (e.g., deductive or inductive; see Step 4). • If necessary, explore further iterative data as an art form of data analysis or trial-and-error approach (no p-hacking!).
Step 9	Prepare descriptive analysis (answers to "what" questions; aim: understanding empirical results yourself).
Step 10	Discuss the results as a theory-based analysis and/or reflection of the empirical results for the purpose of theory development (answers to "why" questions; aim: explain, interpret, and infer).
Step 11	Summarize the most important results and, where appropriate, indicate further research steps based on lessons learned.
Step 12	Publish: • research community (e.g., term paper, bachelor or master thesis, and journal article) • general public (e.g., report, newspaper article, blog, or podcast).
Step 13	Archive the data in accordance with the guidelines on good research practice established by the German Research Foundation (DFG).

Figure 1.4 Transfer of flowchart for research with empirical data to a chapter structure for term papers, journal articles, or books.

Deductive Research	*Step(s)*	*Inductive Research*
Introduction	**1**	Introduction
State of research	**2a**	State of research
Theoretical approach	**2b**	
Methodological approach	**3, 4, 5, 6, 7, and 8**	Methodological approach
Descriptive presentation and discussion of the results	**9 and 10**	Descriptive presentation and discussion of the results
	10	Theoretical reflection and theory development
Conclusion	**11**	Conclusion
Appendix	**optional**	Appendix

research. This difference is already evident when transferring research step 2 into written form. In the case of deductive social research, it is not only important to work through the existing state of research in order to be able to clearly explain what you are building on and where the research gap lies. Deductive social research is also guided by a formulated theoretical approach (see also Chapters 6 and 9).

In contrast, inductive empirical social research is, at most, driven by the theoretical assumptions of researchers who seek to grasp an observed social phenomenon. Such assumptions must then also be formulated in a research paper with reference to the state of research or the lack of previous research and thus to suitable theoretical approaches. Neither qualitative nor quantitative inductive social research is theory-free research. In contrast to deductive, theory-led research (see deductive content analysis in Chapters 6 and 9), inductive research (see Chapters 7, 8, 15, 16, 17, and 18) is not guided by theory from the outset, but instead category construction is grounded in the empirical material. The same applies to automated, computational approaches (Chapters 15–18), but additionally incorporate abstract, theoretical assumptions on how the algorithms detect patterns or draw inferences, how the phenomenon

under investigation is linked to latent patterns emergent from the data, and how the link from the complexity of the data to the abstraction is to be made. In other words, if we take automated content analysis seriously, we must dig into theoretical approaches that are not genuinely situated in the social sciences. Nevertheless, these must be linked to existing theory approaches to bridge the gap between detected patterns, algorithmic approaches, and the social phenomenon under investigation. Accordingly, theory is used in a reflexive way (see Section 5.3.3) and, for example, should be used to support the analysis of content and in particular of latent content.

The methods section (also referred to as methodological approach) combines research steps 3 to 9. In principle, this section should always present the (content analytical) method used. Presenting means that you methodologically define, describe, contextualize, and operationalize the essential systematic criteria of a method with a focus on your study – without comprehensively repeating everything from your methods handbook. In the case of qualitative and quantitative social research, you must explain the case selection (i.e., sampling) and the data collection (see Chapter 3) as the social context of data collection, as well as your methodological and systematic approach to data analysis and interpretation. Remember that data cleaning and preprocessing is time-consuming, but essential part of your research endeavor, which might be less daunting and overwhelming in case of small-n qualitative content analysis (see Chapters 6 and 7).

Like in inductive empirical social research in general, inductive content analysis in particular employs theoretical reflection and, if applicable, theorizing or theory development play a key role in the analysis of the results. In theory development, you should refer to existing social theories when discussing the results and argue concisely why these can be only used in part or not at all for an abstract explanation, that is, for a generally valid explanation. Consequently, theory development builds upon existing theories, but expands or modifies them based on the discussion of empirical results. When discussing the results, and even more so in the conclusion you should explicitly contextualize the results of other studies to highlight what you achieved in addition or in contrast to these with your empirical social research.

In empirical social research, more information is usually collected through data mining than can be processed and presented in a research paper. Just as you must select which data is suitable to use for a research paper as a whole, you must also select which data to present in your paper. Given character, word, or page limits (see, for example, the study and examination regulations for your degree program), you can only present a certain number of figures, tables, typical examples, and the like in your text. However, you can include all data-related steps of your research process that is not directly required as evidence or for understanding your text in an appendix. These appendixes can be very extensive, which is why you should consider whether it is necessary (e.g., required by examiners) to print out appendixes with data or if you can just make them available electronically (e.g., via a file folder in a cloud or an online repository)?

1.4 Qualitative and quantitative content analysis according to the interpretative paradigm

In the previous chapters, we outlined the structure of a research process and presentation of your research in a paper. Although both structures are rather generic, our approach to qualitative and quantitative content analysis follows a specific epistemology rooted in the interpretative paradigm. Epistemology is the study of how knowledge is created and how it can be justified. The term 'paradigm' was coined by Thomas Kuhn (1962/1976, p. 186) in his book *The Structure of Scientific Revolutions*. For Kuhn, the term paradigm encompasses the theoretical assumptions and accepted methodological approaches to problem-solving shared by a scientific community.

In this handbook on content analysis, we introduce methods of content analysis according to the interpretive paradigm. The interpretative paradigm aims for understanding and subsequently explaining social phenomena (see Chapter 5). Central to the interpretative paradigm is the reconstruction of patterns found in actions and social processes. The interpretative paradigm has its roots in the hermeneutic tradition and *verstehende* (understanding) sociology, as established by Max Weber (1981) in the beginning of the 20th century and was further developed by scholars of phenomenology, symbolic interactionism, ethnomethodology, and hermeneutics. According to the hermeneutic tradition, truth only arises in the interplay between understanding and interpretation (Gadamer, 2010). Hermeneutics originally comes from biblical exegesis and refers to the interpretation of text through abduction,[2] that is, asking questions of the material and developing various possible interpretations to get closer to what message or meaning a text might contain.

In the interpretative paradigm, society and social reality are subjectively constructed and interpreted through cultural knowledge and attribution of meanings by social actors (Schütz, 1962) and not seen as objective social facts independent of human interpretation. Epistemologically, the interpretative paradigm is based on a constructivist approach to knowledge, which assumes that knowledge is always a (re)construction of the social world. Researchers not only investigate manifest actions but also attempt to understand the subjective motives and meanings that social actors attribute to their actions (Weber, 1981) and produced artefacts.

For the interpretative paradigm, truth is not regarded as an objective, corresponding relationship between a statement and reality (as in positivism or critical rationalism; see Section 10.3). Instead, truth is considered to be dependent on the social context – it is shaped by personal experiences, life circumstances, a community, societies, and cultures, which differ depending on the social space and may evolve over time. Accordingly, there is no one truth; rather, different realities can only be understood intersubjectively. In this sense, truth is not absolute, but results from the degree of understanding that a researcher achieves about the subjective perception of social reality by ethnographic observation, interviews, or process-produced data (e.g., social media posts and transcripts of parliament debates, see Section 3.2). Accordingly, the interpretative paradigm also assumes

that knowledge is never objective and neutral knowledge. Constructed knowledge varies between people depending on the social, spatial, and temporal context. This also means that people do continually interpret and understand the situations, interactions, artefacts, and relationships to be able to act and react, for example considering existing norms and different role models.

In the interpretative paradigm, researchers see themselves as part of social reality, which is why it is not assumed that an objective analysis of the data can take place or that there is one correct interpretation of the data. Instead, it is essential to understand that you as a researcher come with your own contexts, biases, and assumptions, and accordingly, researchers have to reflect on these during data analysis. This applies to qualitative and (automated) quantitative content analysis or combining both in a mixed methods study. In concrete terms, this means questioning why an interpretation was made by you and whether there could be other or additional interpretations of data (see Chapter 5). Such reflexivity about your own position as well as the interpretation of data could be supported by theory.

1.5 Copylefts and copyrights of software

AntConc, MAXQDA, Python, and VosViewer are all either copyleft or copyrighted software. The copylefts (e.g., of freeware (such as AntConc and VosViewer) and of free-use licenses (e.g., GNU 1–3 and Apache 2–0)) and/or copyrights (e.g., MAXQDA, MIT license, and BSD-2-Clause) of the software listed can be used free of charge for certain research purposes. The analysis software used in this chapter and book – including the Python script examples – is exclusively for the purpose of teaching. The authors and Routledge therefore give permission for you to use and modify the programming scripts presented as examples so long as the use and modifications do not serve commercial and otherwise prohibited purposes. As with all copyrighted content and in line with good and honest research practices, you must reference (e.g., by citation) the origin of the Python script either physically (e.g., printed publication) or digitally (e.g., on exchange platforms such as GitHub).

The program code used in this book is subject to the GNU, Apache 2–0, MIT, and BSD-2-Clause free use licenses. As a result, you may use and modify our code as long as these modifications do not lead to commercial or prohibited use of the program code. Furthermore, copyleft licenses require you to cite both the origin of program codes as well as the modifications in the source code either physically or on exchange platforms such as GitHub.

Notes

1 In statistical calculations with multilevel models, the variance can be modeled starting from approx. 20 cases on the first level and more than 10 or 12 cases on the second level (Rabe-Hesketh & Skrondal, 2012).

2 Similarly to induction, *abduction* is defined as an epistemological attitude of discovery (Peirce, 1966, Section 8; Reichertz, 2013). Beyond the argumentation for a specific abductive analysis as an approach to discovery, it is hardly possible to distinguish between abductive and inductive conclusions in data analysis (Frankfurt, 1958).

PART 1

Basics of qualitative and quantitative content analysis and empirical research

Taylor & Francis
Taylor & Francis Group
http://taylorandfrancis.com

2

DEFINITIONS OF QUALITATIVE AND QUANTITATIVE CONTENT ANALYSIS, AND INDUCTIVE AND DEDUCTIVE RESEARCH APPROACHES

This chapter defines our understanding of content analysis, its relation to communication, and the content analysis approaches covered in this book. We will also give you a brief overview of the types of qualitative content analysis – which we believe can be divided into inductive and deductive – as well as an overview of quantitative content analysis. And you will learn about two rules used in qualitative and quantitative social research – the inclusion rule and the exclusion rule – which will help you to position your research.

2.1 Connection between communication and content analysis

The methods and techniques used in social science content analysis are directly linked to the question of inherent and latent meaning of communication, which we try to uncover. "How are you?" You have probably heard this conversation opener an infinite number of times. But if you become aware of this type of conversation opener, then the sentence "How are you?" can have very different meanings. The reason why you ask the question might be that you are genuinely interested in the well-being of the person you are asking. However, you can also use it as phrase because there is an awkward silence that needs to be bridged. Based on the way the question was asked, the tone of voice, the relationship between the people involved, their body language, and the situation in which the question was asked, it is possible to determine what a socially acceptable answer should be. If it was just a phrase, you can answer, "I'm fine, thank you." Or you can reply with a more detailed description of how you are actually feeling if it was not just a phrase. During the conversation, you might provide information about your psychological, emotional, physical, or financial well-being to help the person who asked to understand how you are doing. The person who is asked must therefore provide information to enable

DOI: 10.4324/9781003496397-3
This chapter has been made available under a CC-BY-NC-ND 4.0 license.

understanding. At the same time, many interpretations occur automatically in a communication situation. For example, the answer "The baby finally slept through the night for the first time" to the question "How are you?" is interpreted to mean that because the baby slept through the night, the parents have also finally slept through the night again. This means that the parents are less tired and are doing well. Interpretations are highly dependent on cultural contexts and are often wrong, although this only comes to light when questions are asked (see also Chapter 5).

The question and answer in the previous section clearly show that this is an example of coded communication. The parents are asked a question and are not talking about themselves but about the child. Coded communication means that the meaning of the words alone cannot be used to understand the context. Instead, the communication must be interpreted, that is, coded in order to understand it. For this reason, conversation openers can cause confusion if the message is incorrectly coded. This could happen, for example, if the question and answer do not match because the phrase "How are you" has been incorrectly contextualized. Perhaps you have also experienced a situation like this one where the salesclerk at the bakery asks "How are you?" and the 70-year-old customer talks about her current medical problems because she interpreted the phrase as an honest question. And then while the woman is reporting on her medical condition, the other people present simply don't know where to look or they communicate their discomfort to each other nonverbally by exchanging glances.

Using these examples, we would like to make it clear that communication can be simple but also confusing. Whether in verbal, nonverbal, or written form, basic conditions must be met to understand the communication. First, the language, gestures, or symbols (e.g., in the form of characters) must be understood. Second, contextual knowledge is required. Communication always conveys the cultural context of a society, and one society can be very different from the next. And third, communication depends on the motivation behind words and phrases. To take up the previous example, we understand that "How are you?" can either be an honest question or a phrase, depending on the context and motivation for asking the question.

But what happens when we leave these face-to-face situations, richly saturated with contextual information, and enter media communication spaces such as social media platforms? First of all, we will notice that communication is framed differently and contains less contextual information. To stick with the previous example – if someone asks, "How are you?", the addressee in online communication has no information beyond the communicated text. Unless video communication is taking place (which may well be subject to delays or failures), the speaker's timbre, tone, gestures, and facial expressions are lacking, as are then also important clues needed to interpret the communication.

But even if digital text communication contains less contextual information, we can still assume that communication is subject to a shared meaning and that the participants involved in this communication can reflect on the context in which they are communicating. If we also assume that language is structured and that meaningful forms of expression follow further regularities that can be reconstructed from the flow of communication, then their overall meaning can at least be extracted from large amounts of text. At this point, we enter the field of automated or computer-assisted qualitative content analysis (Bail, 2014; Heiberger & Riebling, 2016).

Using methods from this field, it is, for example, possible to identify recurring topics as well as the sentiments conveyed in digital communication. This would be the first step in a research endeavor. In a second step, topics and sentiment classes alike must be related to the text documents present in a text corpus, and both must be interpreted in tandem by the researchers. Only after these steps, the topics and sentiments can be related to each other and placed in the larger, discursive context – may it be as small as between two persons who know each other intimately, a group, organization, or a society as whole. However, this final step requires that information about the flow of communication is available (e.g., who is talking to whom/about whom/when and why are they talking with or about each other in the way they do).[1]

Similar to the example of the 70-year-old woman at the bakery, confusion can also arise in the case of digital communication if the parties involved incorrectly contextualize the communication. And we can see from discussions in forums and comment columns in online newspapers that ironic statements are often interpreted as serious and may result in personal hostility. Furthermore, information (contextual as well as textual) can be omitted to change the conveyed meaning, modified, or deleted, so it is even more complicated to analyze and understand the meaning of a communication.

But why are we focusing on communication? The reason is because communication is what we analyze in qualitative and quantitative content analysis (Schreier et al., 2019; Stamann et al., 2016). However, to be able to examine communication using content analysis, an intermediate step is usually required: transcription. Unless, of course, you are dealing with written communication such as newspaper articles, research articles, or open questions in a questionnaire survey. If the communication is not already in writing, you will need to take this extra step, for example, transcribe interviews, describe pictures in writing, and write observations down. For mixed forms, such as tweets that consist of text and images, this means that the images will need to first be described and then analyzed together with the text, or you employ models that are able to consider both the visual and textual layer of communication simultaneously. This also applies to emojis, which are often found in digital communication, for example, when you are analyzing chat transcripts. You cannot carry out a content analysis until all of the communication is available as text.

2.1.1 Text-based content analysis of communication

To begin with, text is a sequence of letters that make sense. In a process of communication or writing, letters as symbols take on a context of meaning. This context of meaning is understood if the following basic conditions are met:

1. **Knowledge of the symbols**: If you don't understand the characters, then you can't understand the content either. This doesn't only apply to characters but, for example, also to emojis.
2. **Knowledge of the context**: If you understand the context – because you understand the cultural meaning – then you can interpret the symbols and understand the communication behind them. This is why it is so important in interviews, for example, to ask exactly what is meant and not to take your own interpretation of the spoken symbols in the form of language for granted. In communication, the knowledge of your own socialization is often transported into one (or more) society(ies). This applies not only to more traditional forms of interviews and the writing down of what has been said, but also to media-generated communication such as tweets, comments in forums or Facebook, and blog entries. The implication is that we can often only understand texts if we can contextualize them. For example, we immediately recognize a recipe by the text segments it contains with gram specifications and poems by their rhyme structure (Schwarz-Friesel & Consten, 2014). And to complicate matters, with digital forms of communication, we usually do not know the conditions under which the communication originated, which makes it more difficult to contextualize what is being said.
3. **Knowledge of motivation**: Texts are usually not neutral reports that depict reality but have a goal and a purpose (Schwarz-Friesel & Consten, 2014, p. 8).

   > In more academic terms, language has a persuasive function to move people to action, to make them happy or unhappy, and to convince or persuade them. This persuasive potential of texts results from the instrumental and basic function of language to activate or change the contents of consciousness, to awaken or intensify feelings, and to trigger a driving force for action (Schwarz-Friesel & Consten, 2014, p. 9, translated by the authors).

   This also applies to the analysis of press conferences with companies, politicians, or political parties, the content of which only becomes accessible at a deeper level if the motivation and context are also known and therefore allow for interpretation.

The successful use of content analysis requires that these three basic conditions are met. However, some types of text are easier to understand than others (Kracauer, 1952). For example, modern poems are often difficult to interpret,

which is why we learned to write poetry interpretations in school and needed accompanying booklets to do so. On the other hand, once you have learned to bake, recipes are a simple type of text. At most, old recipes require you to interpret the quantities (or do you know off the top of your head what an ounce is?) (Schwarz-Friesel & Consten, 2014).

2.1.2 Three steps of analysis: understanding the context, understanding the content, and making these clear to an audience

Text-based content analysis of communication can itself be understood as a continuous communication process. As explained in the previous section, textual and symbolic understanding is a prerequisite for the contextual understanding of the social phenomenon being analyzed. The researcher must first understand the context to be able to understand the content of the empirical data through explaining, interpreting, and inferring. And it is only when the researcher has fulfilled the first and second requirements for understanding that the third communication step of audience understanding or making the empirical analysis understandable to an audience can successfully follow.

To simplify the understanding for the audience of academics, it is helpful to use common academic terms – but without trying to sound more sociological than sociology professors, for example. A certain amount of subject-specific language is also recommended in publications that are intended for a specialist or expert audience. If the audience is broader, for example, than readers of a newspaper, then it is advisable to abstain from using foreign words and technical jargon. However, to make the social phenomenon understandable to newspaper readers and to awaken their interest, all aspects of analysis steps 1 and 2 must be taken into account (Table 2.1).

Table 2.1 Analysis as a Three-Step Process of Understanding the Context and Content and Making These Clear to an Audience

Analysis Step 1	*Analysis Step 2*	*Analysis Step 3*
Understanding the social context of the phenomenon	Understanding the content by explaining, interpreting, and inferring	Audience understanding
Subject(s) Object(s) Space/spaces Time/periods	What is being communicated? Manifest content (1st level of meaning) Why is communication taking place? Latent content (2nd level of meaning)	Academia Experts General audience ...

As schematically shown in Table 2.1, a social science study can be broadly divided into three analytical steps. Specified by the research question, the first analysis step involves grasping the social phenomenon based on social, object-related, spatial, and temporal aspects. In sociology, the social context includes cultural, economic, political, and other aspects constituting the social realm. For the content analysis of documents, datasets, and the like, understanding the context makes it possible to determine what is being communicated and then why it is being communicated.

In the second analysis step, the *what*-question of the communicated content aims to capture the obvious, that is, the manifest content. Compared to the manifest content, the latent meaning and evaluation hidden in communication is generally more difficult to grasp. It requires what is often referred to as reading between the lines. To uncover the latent meaning, that is, to get to the semantics *behind* the manifest text level, thorough analysis and interpretation of your data (e.g., in an interpretation group) and/or use of theory, as explained in Section 5.4, is absolutely necessary.

2.2 Definition of qualitative and quantitative content analysis

Let us now proceed and ask what exactly content analysis is. This question is not as easy to answer as it seems. There are many different definitions of what content analysis actually is (Prasad, 2019). At the heart of every type of content analysis is the systematic and rule-based collection and analysis of texts (communication content) through the interpretation of manifest and latent meanings by breaking down the texts, for example, into categories, patterns, classes, topics, or landscapes of topics.

What does this definition mean in concrete terms? To understand it, let us break it down into its individual parts and look at each of these parts. As mentioned before, there are qualitative, quantitative, and automated quantitative content analysis. The aim of quantitative content analysis is to reduce "the unclear social reality that surrounds us ... to its central structures in order to make the patterns that are 'behind things' visible. These patterns are ascribed greater significance than individual cases" (Rössler, 2010, p. 19 ff, translated by the authors). Such latent patterns can be identified through pattern recognition and quantification (Früh, 2011). For quantitative content analysis, it is essential to determine what is to be analyzed before you begin your study, that is, to specify both the units of analysis and the definitions of the units of analysis. The aim is to convert the text material into variables with defined characteristics that will allow you to carry out quantitative analysis such as frequency calculations. These can be codes,[2] which require a certain degree of abstraction, but also phrases, words, word stems, or lemmas, which are recorded as variables or counted within and between texts, or similarities between texts measured as distances in an abstract space, which captures key aspects of the text documents (the so-called embedding space).

Qualitative content analysis emerged in the 1950s. Kracauer (1952) is regarded as the founder of qualitative content analysis. In his article "The Challenge of Qualitative Content Analysis", he explains why the qualitative analysis of content is necessary and what the differences are between qualitative and quantitative research. Kracauer writes that, on the one hand, the interpretation of communication is not objective as it is carried out by subjects who come with their own values. This means that qualitative interpretations cannot be expressed in simple scales (i.e., good, average, or poor) as these would be subjective interpretations. Scales should therefore not be used for interpretations of communication (Kracauer, 1952, p. 632). The second difference Kracauer names concerns the handling of latent, that is, nonobvious content.[3] In contrast to quantitative content analysis, latent content can be identified in qualitative content analysis, thus avoiding the risk of simplifying the data (Kracauer, 1952, p. 633).

The distinction that Kracauer makes here regarding content addresses a further aspect of the definition: the question of manifest and latent content and therefore the question of the aim of the inquiry. Put simply, the aim of the inquiry is to achieve the depth that an investigation should have, that is, the manifest or latent structure of meaning. Let's return to the example of the greeting. If we want to investigate the different types of greetings that exist, then the investigation remains at the first level of meaning, the level that we understand as common sense or manifest content. Przyborski and Wohlrab-Sahr (2014, p. 20, own translation) define common sense as "what competent members of society could immediately infer if they took the time for a systematic reconstruction." If greetings are to be investigated, participant observations could be conducted at the bakery counter to record the different types of greetings that are used throughout the day. Tally lists could also be kept, which would then be quantitatively analyzed. In addition, it would be possible to analyze books, newspaper interviews, chat transcripts, or transcribed video recordings and to use an algorithm to extract the greetings. This could then be used to analyze aspects such as how the types of greetings change during the course of the day and if specific age groups are addressed differently.

If investigations are to be carried out at the second level of meaning, then the latent structures of meaning are of interest, that is, the implicit knowledge that is not immediately apparent. Taking up the example of the bakery counter, it would be interesting to know why the older woman's answer to the greeting "How are you?" led to confusion. Why do we actually know that it wasn't an honest question in this case? And why didn't the woman respond in a way that would have conformed with standard expectations? Here too, participant observations or interviews with the individuals involved could be conducted to find out the answer to the question "Why did the situation lead to confusion?" Of particular interest here would be the shared assumptions of those present, which Mannheim (1982) describes as a space of conjunctive experience (*konjunktiver Erfahrungsraum*), which he defines as behaviors or statements

that do not require explanation because they are based on shared cultural and social experiences and knowledge. This is what was described previously as the context and motivation of communication. Various techniques and methods have been developed and used in quantitative and qualitative content analysis to uncover the manifest and latent meaning structures of the content. This addresses another part of the definition: the systematic and rule-based collection and analysis of texts.

A range of empirical social research methods can be used to collect data that is analyzed using content analysis methods. As described previously, these can be participant observations, interviews in all possible variations,[4] the collection of open questions in surveys, or tweets. A distinction can be made here between research-produced data that you collect specifically for your study (e.g., interviews) and process-produced data that already exists and can be used for your study (e.g., social media posts).

Content analysis techniques refer to the techniques used to break down texts to reveal their content and meaning. Depending on the content analysis technique, categories are formed, topics are extracted, or patterns can be identified. It is precisely these techniques that we want to describe in more detail – in the context of digital and automated options – and provide application instructions for in this book, which is why we will not go into further detail here.

There is no *one* or even the *best* methodological approach to an object of investigation, not even in the case of qualitative and quantitative content analysis. Instead, it depends on what the aim of the inquiry is, as discussed previously. That is why we provide both guidance and instructions in this book on which social science methods of data collection and which associated analysis techniques may be suitable for your study as well as which digital and automated options are available.

When selecting the data collection method and associated analysis techniques for empirical social research, two interrelated questions arise.[5] First, what is the aim of the inquiry? Is the aim of the inquiry focused on an in-depth analysis of the "why" question in order to get to the latent structures of meaning or do you want to give an overview of the "how" question and therefore analyze the manifest content? A combination of both approaches is also possible, but with strict separation of the qualifying and quantifying *logics* of the procedure (see the exclusion rule, which we will explain in Section 2.2.3).

The second central question concerns the focus of the investigation. In other words, which subjects and/or objects are suitable for the empirical investigation of the social phenomenon that you want to study? Subjects of investigation are individuals whom, for example, you conduct an expert interview with (Meuser & Nagel, 2009). Objects of investigation for a qualitative and quantitative content analysis range from documents with text and other audiovisual content to material objects (e.g., a printed book) and their sociotechnical influence on individual and social action. Objects can become the focus of investigation of a social phenomenon either separately or in combination. The combination

of objects is referred to as a dataset, which in the case of a website, for example, consists of text, images, and other symbols (e.g., thumbs up and smiley).

Now that we have given you a basic definition of content analysis, we would like to contextualize the types of content analysis that we will deal with in this book into the methodological canon. We will begin with the types of qualitative content analysis and then move on to new types of quantitative content analysis.

2.2.1 Types of qualitative content analysis: making a distinction between deductive, inductive, and deductive-inductive approaches in a research design

As you found this handbook, then you have probably already searched for the term qualitative content analysis and may have come across various types of qualitative content analysis. As Schreier (2012) points out, the differences between the various types are not as great as they first appear. In our view, there is one central difference that is important for qualitative content analysis: inductive vs. deductive. We believe that all of the types of qualitative content analysis currently described in the literature can be placed in one of these categories (see also Chapter 9, in which we conduct a systematic literature review in the form of a bibliometric literature review on qualitative content analysis). Before we turn to a brief explanation of inductive and deductive qualitative content analysis, we would like to define what we mean with inductive and deductive.

The similarities between qualitative and quantitative methods are based on the fact that induction and deduction are two scientific approaches that are not exclusive but are connected as if on a continuum. In Figure 2.1, a distinction is made between deductive, deductive-inductive, and inductive approaches to data analysis, for example, for the purpose of developing a category system. The approach to the data also determines the interpretation of the results, which is why the following proposed definitions and explanations distinguish between approach and interpretative conclusions with the aim of explanation and/or understanding:

1. Deduction/deductive means drawing a theory-led conclusion from the general for analyzing a individual or particular case. Theory-led hypotheses and a category system is developed prior to going into the field, that is, collecting data. Understood narrowly, deductive research aims at theory testing (Popper, 1935/2002) or, in the deductive-nomological model, at the quasi-natural-law explanation of causal relationships (e.g., Elster, 1989/2006).
2. Induction/inductive means the empirically based interpretation of general patterns based on the observation of individual cases. Some methods of qualitative research and computational social sciences develop their object of investigation via empiricism and attempt to answer the research

questions in a data-driven manner. In quantitative social research, purely inductive data analysis is rejected; at most, it is accepted for developing hypotheses based on assumed knowledge (e.g., Popper, 1972, pp. 1–31). However, induction is established in the quantitative research process as a form of drawing conclusions from data, referred to as inference.

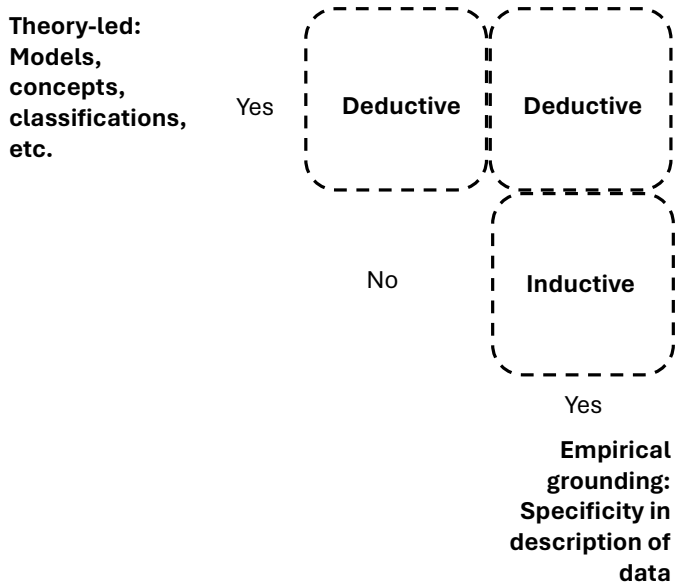
3. Deductive-inductive means a theory- and empiricism-led, combined conclusion based on a general (i.e., theoretical) and empirically driven approach. In the qualitative approach, deductive-inductive means, for example, supplementing and modifying the theory-led construction of a category system with categories derived from the empirical material. The combination of theoretically based generality of the statement and explicit consideration of the empirical environment situates the individual case in the general, whereby findings on the general individual are substantiated. Depending on the research design, mixed methods or method-integrating social research can combine sequentially inductive and then deductive data analysis (e.g., Schoonenboom, 2018; Schoonenboom & Johnson, 2017).

In Figure 2.1, the deductive-inductive or inductive-deductive quadrant, here in short: deductive-inductive quadrant, stands for the combination of theory-led and empirically driven research as an independent research design. However, the deductive-inductive quadrant also stands for the fact that findings from empirical research can evolve in both directions on the continuum. Consequently, the quadrants are to be understood analytically and not as hermetically sealed units. For example, during deductive, theory-led research, when discovering a new phenomenon or social pattern of a phenomenon, this could result in inductively studying said phenomenon or a pattern of it.

All forms of observation of social reality are, initially, assigned to the inductive quadrant (e.g., ethnography, questionnaire surveys, document analysis, and interviews). The aim of empirical research is to achieve the greatest possible accuracy in the description of data to specify actions, structures, groups, etc. All theory-driven research is situated in the deductive quadrant, whereby theory-led research is based on very empirical methodological assumptions, correlations, and proposals about definitions, concepts, models, or grand theories' "generality of statements" (Alexander, 1982, p. 4). It should be emphasized here that scientific observations are always based on the conscious and unconscious assumptions of researchers, that is, they are also guided, at least, partly, by theoretical assumptions, which researchers need to be reflexive about (see Section 4.3.2.3).

Qualitative content analysis is the systematic and rule-based collection and analysis of texts (communication content) through the interpretation of manifest and latent meanings by breaking down the texts into categories. The development of categories is the distinguishing feature between inductive and deductive qualitative content analysis.

Figure 2.1 Deductive-inductive quadrants of research designs and data analysis in empirical social research.



Sources: Alexander (1982); Stokes (1997); Authors' arrangement

In inductive qualitative content analysis, the categories are developed from the material (also known as emergence). Inductive qualitative content analysis is therefore primarily used when little is known about a phenomenon, and empirical studies and theories cannot be used. In short, you would go through the process by collecting data (e.g., interviews) or using secondary data, which you approach with an open mind without hypotheses or presuppositions and then develop the categories based on the material (see Chapter 5 for interpretation and Chapter 7 for detailed instructions of inductive qualitative content analysis). Since the category system arises from the material, new categories are often added during the coding process. This means that coding as assigning text passages to categories should be understood as an iterative process.

In deductive qualitative content analysis, the categories are derived from one or more theories and/or empirical studies. This means that the categories are already defined before you start working with the material. In most cases, for deductive qualitative content analysis, you have the category system in mind when you conduct interviews, for example, because the categories are reflected in your key questions in the interviews. In short, during the coding process, you then search through your empirical material in relation to the categories and assign the text passage to the categories. This is known as coding.

In actual coding, however, this seemingly clear distinction is often less clear. On the one hand, qualitative research assumes that the researcher's assumptions and prior knowledge automatically play a role in the interpretation of data material (Armat et al., 2018). You cannot simply forget what you have already read, seen, or heard in your life – *tabula rasa* is not possible. This is why it is particularly important in qualitative research to be aware of your own assumptions (Strauss, 1987) and to make it transparent to others how the interpretations or categorizations were made. In order to conduct good-quality research, the appropriateness-transparency convention (see Section 4.3.2) must be fulfilled, and intersubjective comprehensibility and transparency must be established in the research process.

In addition, it is very common in the deductive research process for material to be "left over" during coding. This means that material cannot be coded into the categories derived from the theory (Graneheim et al., 2017), either because the material is not relevant to the research question or because it includes aspects that have not yet been highlighted in the theory used. This means that the category system must be expanded so that it includes categories that are developed from the material itself (inductive).

In summary, it can be stated here that both methods – inductive and deductive qualitative content analysis – will always contain characteristics of the other method. However, the central difference comes at the beginning of the procedure. In inductive qualitative content analysis, the categories are derived from the material, whereas in deductive qualitative content analysis the categories are derived from theories and/or empirical studies before the material is analyzed and are then applied to the material. The different types of qualitative content analysis can also be categorized according to this distinction.

Inductive qualitative content analysis:

- **Data-driven qualitative content analysis** (Schreier, 2012): For Schreier, qualitative content analysis focuses on descriptive analysis of the data but also requires interpretation of the meaning of the data. In the data-driven approach, categories are created by summarizing, paraphrasing, and abstracting. For Schreier, it is important to stay close to the data because the categories should reflect the data.
- **Conventional qualitative content analysis** (Hsieh & Shannon, 2005): The type of qualitative content analysis they proposed is strongly based on the grounded theory approach (Glaser & Strauss, 2017), but the difference is that qualitative content analysis is used to describe the phenomenon and not to develop a (substantive) theory.
- **Inductive approach** (Graneheim et al., 2017): The procedure is characterized by a search for patterns. During the analysis, researchers look for similarities and differences in the data, which are then transferred into categories. This involves moving from the data to a theoretical understanding, that is, from the concrete and specific to the abstract and general.

- **Summarizing qualitative content analysis** (Mayring, 2021): Paraphrasing and reduction are used here to develop categories from the material. Mayring distinguishes accordingly between an inductive and deductive approach in a rule-based procedure. If a summarizing qualitative content analysis is carried out, it is an inductive procedure, even if categories that emerge from the material can be assigned to a theory in the course of the analysis or a theoretical framework is created. This gets even more confusing when you read the additional part, that is, structuring to summarizing, which Mayring terms structuring qualitative content analysis (see next bullet-point).

Deductive qualitative content analysis:

- **Structuring qualitative content analysis** (Mayring, 2021) includes the subcategories of formal structuring, content structuring, and typifying structuring. The category system is derived before coding from one or more theories, previously conducted studies, or literature, and the text passages are assigned to the categories based on defined coding rules. If structuring content analysis is carried out, it is a deductive procedure, even if, for example, questions in the interview guide were generated from everyday observations. The inclusion of further categories in structuring content analysis remains largely unclear in Mayring's work, which is why Steigleder (2008) provides instructions for this in her book.
- **Content structuring qualitative text analysis** (Kuckartz, 2014; Kuckartz & Rädiker, 2022): In Kuckartz's methods book, interview material is used to explain qualitative content analysis, which is why the main categories are developed using an interview guide. This interview guide was created through comprehensive literature studies, which in our view includes theoretical work and work with empirical studies. We therefore classify content structuring qualitative text analysis according to Kuckartz as deductive, whereby subcategories can be added to the main categories, which are inductively developed. Kuckartz's inductive category construction is based on the open coding used in grounded theory (Glaser & Strauss, 2017).
- **Content analysis by extraction** (Gläser & Laudel, 2010): Deductive categories are extracted from theory through a theoretical analysis of the problem/phenomenon. These categories are given indicators (i.e., rules) that can be clearly assigned to the category. When the material is coded, the categories can be adapted, and new categories can also be inductively generated.
- **Directed qualitative content analysis** (Hsieh & Shannon, 2005), **deductive approach** (Graneheim et al., 2017), and **concept-driven approach** (Schreier, 2012): Based on theories and, most importantly, gaps in theories or previously untested theories, a category system is developed that is to be tested on text material (usually interviews) with the aim of further empirical or theoretical development. Since theories are to be further developed, text passages that cannot be assigned to a deductive category are of particular importance and are coded into newly developed categories that arise from the material (inductive).

We wanted to show that qualitative content analysis always is connected to either inductive or deductive or deductive-inductive qualitative content analysis (see Figure 2.1). As can be seen from the bullet-point lists, the differences between the types of qualitative analysis are found in the descriptions of the individual analysis steps. In our explanations in this handbook, we do not make such specific types of content analysis distinctions but pursue a pragmatic approach that will allow you to carry out (qualitative) content analysis as simply as possible, but in a rule-based and therefore clear, appropriate, and transparent manner.

2.2.2 Quantitative content analysis: an overview

In contrast to the in-depth analysis of text data using qualitative content analysis, quantitative content analysis aims to identify manifest and latent patterns in communication based on statistical frequencies and correlations. For automated quantitative content analysis, text in documents is put apart and transformed into an abstract feature space, which is then used to calculate frequencies, statistical correlations, similarities, and distances. All of these are then used to cluster or categorize the text documents, either to sentiment classes or topics. To do so, a huge text corpus is needed to infer classes or topics from this abstract space. This is why – as described in Section 1.2 – the volume of data differs greatly as quantitative content analysis can include several hundred to one hundred thousand or more pages.

Unlike the qualitative types of content analysis with their respective theoretical underpinnings, (automated) quantitative content analysis is empirically driven and problem-driven and depends on the development of models, algorithms, and machine learning architectures. Semi-automated and, most importantly, fully automated content analysis methods belong to the fields of computational linguistics (natural language processing, NLP) and artificial intelligence (AI). Fully automated content analysis is used primarily to identify patterns within human language (= natural languages such as German or English), to transcribe spoken language, offer automatic translations, to identify sentiments conveyed in the text, to summarize the content, to extract meaning, or to relate the content between documents, and different data types, for example text and video. We will explore topic modeling and sentiment analysis in-depth in Chapters 15–18 and provide an overview of the underlying concepts in Chapters 11–13. In contrast, semi-automated analysis procedures have been developed for computer-assisted corpus linguistics and for political and sociolinguistics (e.g., Coulmas, 2017; Niehr, 2014), with the aim of identifying manifest patterns of language (e.g., see content frequencies in Chapter 6 and network patterns in Chapter 7). In both semi-automated and fully automated content analysis, the data and text format must be uniformly prepared so that the software can identify and analyze the texts. There are fundamental differences between the quantitative types of text analysis:

1. Semi-automated content analysis is based in roughly equal parts on a mix of human and machine analysis steps and the generation of manifest results. The semi-automated aspect also emphasizes that the analysis and generation of results does not take place in a final run (after a lot of training; see next), but gradually, that is, sequentially. The analysis of each step is in your hands – and is generally not done for you by algorithms and statistical measures (e.g., values for determining topic coherence). The software used (e.g., AntConc, see Chapter 8) usually has a (rudimentary) application interface and requires no programming knowledge. The data volumes are usually still so manageable that with a great deal of time and effort they can be analyzed inductively (e.g., by counting word frequencies). Inductive here means that individual specific statements are used to derive ever more general abstract categories, which in the end makes it possible to make either theoretical statements or – by relating categories to each other – a detailed description of the phenomenon under investigation.
2. In the case of fully automated content analysis, we use software and a programming language (e.g., Python) to preprocess, analyze, and visualize the data in such a way that we, as social scientists, are able to interpret the results and identify latent patterns (e.g., topics). For the analysis, we use unsupervised learning architectures to identify nonmanifest features such as patterns, meanings, and topics in large amounts of text that are too large to process with the capacities of the human brain. However, this does not mean that you press a button and the machine provides you with a fully interpreted model. Instead, it provides us with (more or less) human-interpretable candidates for topics, or sentiment classifications, both of which must be evaluated, along with evaluation metrics. In other words, we must still manually establish a relationship between potential topic-candidates, the evaluation metrics, and text passages, and other patterns. At the same time, we must always be aware of what we are doing, as we are not only dealing with variables or codes, but also with levers and gears that govern the training process of the models themselves. Against this backdrop, you will notice that metrics also contain a great deal of (constructed) meaning and weight and has a great impact on the models to choose, on the texts to be interpreted (qualitatively), or how to make sense of the text documents at all. Another difference is the flexibility of the data types that are suitable for the analysis – which can also cause problems. In contrast to a uniform text corpus (e.g., a large number of newspaper articles on a specific topic or a topic-related exchange on a social media platform), it may be necessary to substantially preprocess a very heterogeneous text corpus (e.g., YouTube comments posted under social science videos) in order to retrieve any information from the text documents, which are then passed to the algorithms to identify underlying patterns, such as topics.

Attempts to analyze texts using semi- and fully automated approaches can be traced back to the 1950s (Deng & Liu, 2018). The problem-centered,

technology-dependent approach is illustrated well by the following statement from Jones (1994, p. 3), which summarizes the statements made by Fred Thompson, a pioneer in the field of computational linguistics, at the 1987 Association of Computing Machinery conference.

> "Work in the field has concentrated first on one problem, then on another, sometimes because solving problem X depends on solving problem Y but sometimes just because problem Y seems more tractable than problem X. We may indeed be seduced by the march of computing technology into thinking we have made intellectual advances in understanding how to do NLP, though better technology has also simply eliminated some difficulties we sweated over in earlier years. But more importantly, better technology means that when we return to long-standing problems they are not always so daunting as before."

In the 1950s, the focus of computational linguistics was to automatically translate texts from Russian into English (research in the service of the Cold War), but this was often technically very challenging due to long computation times (several minutes per sentence) and the lack of systematic programming languages (such as Python) (Plath, 1967). During this period, lexicon- and rule-based approaches were developed, which are reflected in our book in sentiment analysis (translation of words and semantics into positive or negative sentiments; see Chapter 16).

While the limited methods of the 1950s and 1960s meant that any computer performance could be surpassed by trained translators, the methods were expanded in the next 50 years so that today we have automated language completion (e.g., when you enter search terms in Google), spam filters, and topic modeling approaches. Two relevant developments came together in the 1960s and 1970s. Artificial intelligence developed that, for example, brought meanings into relation to each other through semantic networks (Findler, 1979; Schank & Tesler, 1969) and enabled inferences (= statistical reasoning) to meanings. However, for fully automated content analysis, researchers still had to carry out preliminary and additional work, which is only required for semi-automated content analysis today. Given the still limited technical resources (a supercomputer from the 1970s couldn't even keep up with the computing capacity of your cell phone), extensive sets of rules were created to extract meanings from the texts.

The phase of rule-based research of language was replaced by an empiricist paradigm in the 1980s (Murphy, 2012). In the context of this new paradigm, machine learning became established as a separate field of research that focused on pattern recognition and the extraction of information from machine-readable text. A distinction is made between two types of machine learning: unsupervised and supervised machine learning. Unsupervised machine learning can identify structures within data (not just texts!) without the input of researchers

(Solan et al., 2005). Supervised machine learning uses rules and assignments created by human experts (e.g., codes assigned in qualitative analyzes and text passages associated with these codes) to identify patterns. However, here too, the possibilities of machine learning were initially very limited. This was simply because very few machine-readable texts were available (imagine a world without the Internet, in which text occurs almost exclusively in printed form), and the computing capacity of computers was far from sufficient to simulate the learning behavior of a human brain (Benvenuto & Piazza, 1992).

In machine learning, software is trained to identify statistical models based on data. The training data then forms the basis for the software to draw on comparisons to identity similar patterns in other data corpora (e.g., of lists). Since the 1990s, multilayered neural networks and deep learning methods have been increasingly used to identify patterns in large amounts of text data and to group, or rather, classify the text content. At the same time, the Internet quickly spread (as did the amount of machine-readable content), which led to refinements in the algorithms and methods used to solve classification and categorization problems in computational linguistics.

2.3 Combining qualitative and quantitative analysis techniques for content analysis in a mixed-methods research design

As already indicated, data suited for qualitative and quantitative text analysis can be collected in several ways. In addition, data can be combined to illuminate the phenomenon under investigation from various perspectives and achieve an in-depth understanding of the phenomenon. Two rules are central when you use a combination of data analysis methods and content analysis techniques: the inclusion rule and the exclusion rule of qualitative and quantitative social research.

The inclusion rule emphasizes the fundamental compatibility of qualitative and quantitative methods in empirical social research, that is, the possibility to use qualitative methods together with quantitative content analysis or with other methods.

The exclusion rule in empirical social research can be defined as follows:

1. Qualitative methods and associated analysis techniques do not produce quantifiable findings, and
2. Quantitative methods and the associated analysis techniques do not produce any qualitative findings.

The exclusion rule in empirical social research can be explained using two examples of content analysis. Imagine you have a certain amount of texts as data. And you have identified certain terms or keywords based on your research interest and theoretical assumptions. You then go through the texts and code the terms using qualitative data analysis (QDA) software. After completing the

coding process, you now have a certain number of terms used. This number of terms used shows you the frequency but not the meaning of the respective terms in the context of the texts. You can therefore present a quantitative overview of the code. However, the frequency of the terms does not give you in-depth, qualitative insight into the content of the texts. Even if instead of just the term you have systematically coded the sentence in which the term occurs as well as the previous and subsequent sentences, a qualitative content analysis of these would be a further or new analysis step, which is, however, dependent on the previous quantification and is therefore not a qualitative content analysis of the texts.

Mixed methods research (e.g., Burzan, 2016; Creswell & Plano Clark, 2011; Schoonenboom, 2018; Tashakkori & Teddlie, 2003) emphasizes that qualitative and quantitative methods can be used for empirical social research in parallel as well as sequentially. An example of parallel use would be using qualitative content analysis of texts, videos, and symbols on a website in order to gain a deeper understanding of the content of the website. This understanding would then be compared with the results of a parallel questionnaire survey of the users of the website under investigation. If, on the other hand, the special features of Internet-based communication on a website are to be investigated, a sequential research design should be chosen. The content of the website would first be qualitatively analyzed and recorded as results. In a second step, the findings could then be used to formulate empirically based assumptions in the form of hypotheses for the questionnaire survey. However, if there is little knowledge about the users of the website, for example, because all users have pseudonyms or the novelty of a website or its communication options attract an unknown community of users (e.g., age, gender, or social background), information could first be collected through a questionnaire survey so that you could then analyze the content of the website in an informed manner using qualitative content analysis.

In automated quantitative content analysis, it is possible to train one or multiple classifiers that extract topics and sentiments from online communication. In this case, the text content of tweets, forum posts, or the like is linked to attitudes on specific topics and then broken down into a few central dimensions (e.g., in regards to geometric data analysis, Chapter 15), into mixtures of probabilities, which represent the topic composition of a text document (Chapter 17), or the probability to be assigned to different clusters (or be denoted as 'noise', see Chapter 18). A dimension or cluster can be interpreted as a latent unit of meaning, for example, as a topic. In all of these cases, we can utilize differences (and similarities) in the used vocabulary and other linguistic features across a huge number of text documents, which allows the algorithm to recognize these differences and group them accordingly. For example, users of a virtual campus might first talk about studying at their own university, next about transitioning to the job market, then about financing their own studies, and finally about their lecturers. Depending on how "fine-tuned" our algorithm is and how the

underlying learning parameters are set, correspondingly few/many topics and dimensions will be identified. Keep in mind that these dimensions and topics are not necessarily human-interpretable, but candidates for some unit of analysis that might make sense upon closer inspection. This closer inspection is the domain within automated content analysis, where we, as researchers, must employ techniques of qualitative content analysis.

Even if the prevailing attitude here is that automated methods can be used to identify, extract, interpret, and in some cases even predict latent meaning structures without human intervention, the results produced by automated quantitative text analysis methods must still be interpreted by researchers. Some researchers now even intentionally use qualitative social research methods to compensate for the coarseness of the analysis results produced by automated quantitative text analysis and reflect on how automated methods and qualitative methods can be integrated (Andreotta et al., 2019; Dillon et al., 2020; Fawcett et al., 2019; Lauer et al., 2018; Schneiker et al., 2019). This is mainly because of the realization that the results generated by automated methods are often difficult for humans to interpret, that content and topics are implausible, and that patterns are revealed that contradict the intuition of the researchers and as such indicate errors in the data or theoretical approaches.

Conversely, however, we should not assume that automated methods of quantitative text analysis and general machine learning methods pose a threat to qualitative social research (Mills, 2018; Strong, 2014), even if they emphasize the possibility of predicting phenomena. Instead, we argue in this book that automated procedures enrich the canon of research methods in sociology, provided they are used appropriately, complement existing methods, and are always critically reflected on with regard to their limitations. After all, we must not forget that as researchers we always enter the research process with expectations, experiences, and sometimes even biases and that we therefore must structure the process accordingly. This applies to all steps equally: the choice of theory, of research questions, of texts selected, of research methods and their interpretation, which are also reflected in quality conventions for empirical social research (see Chapter 4). Automated methods of quantitative text analysis can help us here by acting as a kind of corrective measure that directs our attention to texts, content, or structures that we would not have paid attention to or that we would have interpreted in a completely different way.

This applies in particular to the aspects of data selection and data collection when we consider the enormous amounts of data that have been produced by users on the Internet or via devices (e.g., your smartwatch or surveillance cameras in public spaces) in recent years (Cao et al., 2020). No one person could sift through these volumes of data on their own, let alone interpret their meaning. In addition, it is equally difficult to gauge the preselection algorithms for data such as Google search or recommender systems on platforms like Spotify created for you based on your surfing behavior (and that of similar users) and how this draws your attention to certain content that is intended to appear

attractive to you (Seaver, 2019). If we assume that the information displayed to us is preselected by algorithms and computer programs and that we ourselves have to make a preselection based on our experiences and decisions, would it not be better to first use methods that do not require us to make a selection, but instead make use of all available (and not preselected!) data? Shouldn't we use data that could even potentially help us to think outside the box and possibly draw our attention to phenomena that we would have otherwise not noticed due to time constraints or the perspective we have adopted? Accordingly, we assume that you can use the insights gained through the use of automated methods to inform your qualitative research, and conversely that you can use the methods of qualitative text analysis to access the latent meaning of the patterns conveyed in communications.

Notes

1 One technique used in political science that makes this step possible is discourse network analysis (Leifeld, 2020). This type of analysis can be used to examine the structure and dynamics of political debates, polarization within political discourse, and the formulation of consensus.
2 The term *code* has several meanings. A code is a meaning assigned by a researcher to a specific passage of text. On the other hand, a code (or program code or source code) can be a sequence of commands in a programming language that can be read and interpreted by humans and can also be read and fully executed by a computer.
3 "While it may be able [sic] to avoid obscure poems, it is much concerned with texts in which latent meanings not only pervade the manifest content, but also are intricately related to the objectives for which the analysis is undertaken. Such latent elements may strongly resist quantification, and occasionally the quantification is actually foregone" (Kracauer, 1952, p. 634).
4 For example, expert interviews, problem-centered interviews, and narrative interviews.
5 For empirical-based examination requirements during your studies – also referred to as student research projects and term papers – and for bachelor's and master's theses, it is generally recommended that you decide on a method and the corresponding analysis technique. Experience shows that in general using more methods does not lead to better results in empirical social research.

3

KNOW YOUR DATA

Possibilities and limitations of text, numeric, video, and pictographic data, and primary and secondary studies

Chapter 3 covers the specifics of text, numeric, video and pictographic data, and the associated possibilities and limitations for content analysis. We describe the differences between research-produced and process-produced data – from the design (considering data types, research objectives, and operationalization) and data production (including sampling, data collection, and data type) to the analysis possibilities (resulting from data storage and types of analysis) and using results and archiving data. Limitations for content analysis also root in the use of data for a secondary study. In comparison to a primary study, data is not collected for answering your research question. However, doing a secondary study means that you do not have to spend time on data collection.

3.1 Introduction

Social science research is always guided by the aim of the inquiry. This aim must be defined by a specific research question and – especially in quantitative studies, a theory, existing research, or a description of a social phenomenon – can be specified by hypotheses. An observation, theory, existing research, and description of a social phenomenon are also used to define the empirical focus of the investigation more closely. Depending on the aim of the inquiry and the focus of the investigation, there are specific possibilities and limitations for data analysis, which are determined both by the qualitative or quantitative quality of the data and access to the data.

The quality of the data refers to the type and characteristics of the data. For example, we can distinguish between data that is available as text, as a recorded video of an expert panel discussion on television, as image or pictographic material, as a data table, or as (structured or unstructured) data types. In social science, also in qualitative studies, it is common to convert data from its original form into another type of data to do content analysis. For the analysis,

DOI: 10.4324/9781003496397-4
This chapter has been made available under a CC-BY-NC-ND 4.0 license.

voice recordings (e.g., interviews or singing) and videos are transcribed or summarized as text. Like numbers, text is therefore the (first or second) most important type of data for the social sciences.

Each type of data is associated with its own methods and approaches to data collection and processing. For example, you will find it difficult to perform statistical calculations on text, video, or image data unless you preprocess and convert it into a machine-readable format (e.g., by machine learning, see Chapters 13 and 14) or invest a lot of time creating your own data table that contains the count characteristics of these files. Data tables are generally not well suited for text analysis, but they do allow you to perform some statistical calculations due to the numerical quality of the information (e.g., calculating correlations, counting frequencies, and reporting percentage distributions).

However, the limitations of quantitative studies also arise from the specific quality of numbers as data. Numbers never speak for themselves even if statistical correlations are significant, which means that numerical values of measured variables are highly unlikely to occur together by chance. On the contrary, the numbers and resulting correlations must be understood (see Table 2.1, manifest content, analysis step 2) by the researchers in the social context (see Table 2.1, analysis step 1) and then explained and interpreted so that the numerical results can be understood by the intended audience. The audience of the study can be other social scientists or individuals without statistical knowledge. In both cases, the language used to present the results should be appropriate for the audience, whereby social scientists also expect clear and comprehensible explanations in publications and presentations that complement the numbers (e.g., in tables) and statistical test significances (see Table 2.1, analysis step 3).

Furthermore, numbers (e.g., estimators, goodness-of-fit measures) are abstract representations of properties of our text contents. As such, they can only capture a small portion of our data. At best, this helps us to focus on aspects linked to our research question, which are otherwise hidden in the data. At worst, these numbers and measures might distract or mislead us or provide a false sense of certainty in regard to our analysis (e.g., the accuracy-measure used for sentiment class prediction in Section 16.5).

However, the conversion of text into numbers changes the quality of the qualitative data. This does not mean that the quality gets better or worse because even if a word has a meaning of its own, the meaning in the context of the study must first be understood before the researcher can explain, interpret, and draw conclusions. The change in quality in the numerical representation of a word or the frequency of the word in a document or a more extensive document corpus (a) removes a word from context and (b) represents a word as a number. Imagine the example of a 5 Euro bill. A 5 Euro bill can be used to purchase goods worth the equivalent of five Euros. However, the 5 Euro bill, that is, the secure against forgery paper with the 5 Euro print, itself is not worth five Euros. The bank issues the 5 Euro bill on the basis of the equivalent value in your account.

Like the difference between the quality and quantity of a 5 Euro bill, the words converted into numbers do not objectively represent the quality of a word in a document or text as a whole but represent a value that can be used for calculations or in statistics. Quality is therefore not an objectification of text, but enables a number-based meta-analysis of text. Analogous to *close* and *distant reading* (Moretti, 2000, 2013), text or words can be assigned the quality of close reading and numbers to the quality of distant reading.

Newer approaches, which rely on deep neural networks (such as BERTopic), posit words and documents onto a landscape-like structure. Here, the numerical representation of words equal coordinates similar to longitude, latitude, and altitude, and the distances between them carry information on their similarities across, their dependencies, morphology, and even meaning. We will get back to this in Sections 11.3, 16.3, and Chapter 18. This representation carries much more information compared to word counts, but it is difficult to interpret those landscapes, and even more difficult to understand how the applied algorithms created these landscapes in the first place.

3.2 Characteristics of research-produced and process-produced data, and primary and secondary studies

Just as important as understanding the distinction between the numerical representation of text data for distant reading and text as analogue data for close reading in social research are the analysis possibilities and limitations of based on the context of data collection. For the context of data collection, a rough distinction is made between research-produced and process-produced data. Research-produced means that data collection is carried out according to research standards and in compliance with quality conventions (see Chapter 4). The primary aim of research-produced data is scientific inquiry. Practical applications of research produced data, for example, for evidence-based policy consulting could be a secondary aim as the empirical evidence behind the consulting is legitimized by research quality conventions.

In comparison to research-produced data, process-produced data is a by-product of social interactions which are stored on purpose or unwillingly in diverse forms. Process-produced data includes files in administration, court records, transcripts of parliament debates, social media posts, newspaper articles, birth and death statistics, grade distributions in schools and universities, etc. The purposes of process-produced data are bureaucracy (e.g., issuing birth and death certificates), democratic transparency (e.g., parliament transcripts), juridical system (e.g., court records), or communication (e.g., social media). Some process-produced data is collected and used for reporting by (federal) statistical offices, which is also referred to as official data, but a lot of process-produced data is archived and may or may not await to be discovered and content analyzed by researchers.

Table 3.1 Characteristics of Data According to the Type of Data and Its Collection

	Research-Produced Data		*Process-Produced Data*		
	Large-n *Studies*	Small-n *Studies*	*Documents as (mass) Data*	*Web-Based Mass Data (big data)*	*Administrative Mass Data*
Design					
Data basis	Survey	Observation/ questioning/discourse/ comparison/similar	Screenplays/books/ party platforms/ music/videos/similar	Terms and conditions of organizations and companies	Laws/regulations/ legal texts
Aim	Answering the research question(s)	Answering the research question(s)	Communication (e.g., entertainment)	Information (data collection as a secondary product or by-product)	Data processing in accordance with legal regulations
Operationalization	Deductive reflection on a theoretical basis	Deductive reflection/ inductive creation of category system	Inconsistent/dependent on the specifics of media	Inconsistent/dependent on the specifics of (social) media	Laws
Data production					
Sampling	Target population/ survey population (intentional and random selection)	Case(s) selection	Document selection	Data processing as data collection	Notification/case (e.g., court)
Data collection	Survey population	Subjects (e.g., interviewees)/objects (e.g., countries)/social phenomena	Documents	Datasets	Database/data archive
Data type	Numbers	Audios/images/texts/ videos/numbers	Audios/images/texts/ videos/numbers	Audios/images/texts/ videos/numbers	Audios/images/texts/ videos/numbers

(*Continued*)

Table 3.1 (Continued)

	Research-Produced Data		*Process-Produced Data*		
	Large-n *Studies*	Small-n *Studies*	*Documents as (mass) Data*	*Web-Based Mass Data (big data)*	*Administrative Mass Data*
Archiving					
Complete	Yes	Yes	Yes	Partial	Yes
Analysis					
Data storage	Researchers	Researchers	Individuals/collectives/ organizations as authors	Data collection by company/information services company	Administrative staff/ members of an institution
Data analysis	Quantitative analysis methods (e.g., multivariate statistics)	Qualitative analysis methods (e.g., thick description)	Qualitative/ quantitative analysis methods	Data processing necessary before quantitative analysis	Analysis for specific purposes
Types of data analysis	Primary and secondary analysis	Primary and secondary analysis	Secondary analysis	Secondary analysis	Secondary analysis
Application					
Result	Research (and some practical) implications (e.g., guidelines for action)/conclusions	Research (and some practical) implications (e.g., guidelines for action)/conclusions	Limited use if the population is unclear and/or not representative	Limited use if the p opulation is unclear and not representative	Practical significance: administration, politics, government, and management purposes

Source: Graff and Baur (2020, p. 262), translated and adapted by authors

Using process-produced data for content analysis means doing a secondary study. Research in secondary studies is limited because a research question and/or hypotheses need to be tailored to the existing data. This also means that secondary studies have to respect the quality of data and deal with certain limitations. In comparison to secondary studies, primary studies do plan and collect data by focusing on the phenomenon under study and with the aim of the inquiry.

In addition to primary and secondary study, for research-produced data, a distinction is made between primary and secondary data analysis (see Box 3.1). In contrast to primary data analysis, secondary data analysis is subject to greater restrictions, for example, providing sufficient but not comprehensive data to study a social phenomenon. The disadvantage of data limitations – the data was collected for another study or is no longer up to date – is offset for researchers by the fact that they do not plan to collect data themselves.

Box 3.1 Brief definitions of primary and secondary data analysis

Primary data analysis: A primary data analysis is the initial analysis of data collected within the research process.

Secondary data analysis: A secondary data analysis is the analysis of data that has already been collected and analyzed by other researchers or by government and companies as part of their standard processes.

As with primary data analysis, the data must be preprocessed for secondary analysis. In case you do a secondary data analysis with research produced data or conduct a secondary study with process-produced data, you have to consider the (institutional) context of data collection, other researchers' research design, prior publications, theory, or biases that influenced the production of data.

Considering the possibilities and limitations of primary and secondary data analyses as well as research- and process-produced data, it is clear that your planning of a content analysis should take into account the results to be achieved by your research right from the start. This does not mean that you know what your results will be, but that you clearly state the aim, how data will be collected, and what the suspected contribution of your results will be, possibly including the specification of a target audience (see Table 3.1). In general, a research design defines a framework that makes the theoretical and methodological approaches as well as the research process transparent and reaching results based on your research question probable. The categories and guidelines for selected types of data collection in Table 3.1 are described in detail later.

In Table 3.1, the category 'data storage' refers to intellectual property rights and data ownership. In contrast to a primary study, a secondary study may involve complicated business aspects. These aspects are particularly clear in

the case of web-based mass data, which can generally only be collected for research with restrictions, sometimes for a fee and usually only in accordance with the conditions of the company who owns the data.

In Table 3.1, column 2 presents the data basis 'survey' (instrument for data collection is a questionnaire) and data type 'numbers' for studies with a large-n of observations as an example of research-produced data. The n symbolizes the number of cases investigated. However, large-n studies are not limited to quantitative social research. Qualitative large-n studies would be, for example, interview studies with a large number of interviewees, whereby not only the number of interviews but also the scope, that is, the length of the interviews, must be taken into account. A study with 135 interviews can certainly be described as a large-n study, as can 35 one-hour interviews that produce one hundred pages plus n pages of transcribed text. This means that interviews could be analyzed both by using qualitative and quantitative content analysis, which is not the case for surveys and their statistical calculations based on numbers.

As a rule of thumb, studies with ten cases or fewer can be described as small-n studies. However, as is common in social research, there are also "large-n in small-n studies" (Schneijderberg & Götze, 2021), for example, if you conduct a questionnaire survey in four countries based on a theoretically justified comparison of the four countries selected as a sample. Another example of a large-n in small-n study would be the survey of all employees in four large companies (<500 employees).

3.2.1 Sampling and case(s) selection for data collection

The possibilities and limitations of collected data very much depends on the sample population and cases selected for study (e.g., Ebbinghaus, 2005, 2009; Seawright & Gerring, 2008; Schneijderberg & Götze, 2021; Tilly, 1984). Regardless of whether you are studying companies, countries, or survey populations, the sample selected for research-produced data must be justified. The justified selection of a suitable sampling approach contains the description of the social phenomenon to be investigated, including respondents or interviewees, or observation sites, and is linked to the research question and, if applicable, the hypotheses.

In English, the phrase to *draw a sample* actually refers to the characteristic of random selection (e.g., by drawing lots), which is mostly used in survey research. In random sampling, the group or documents (i.e., target population) studied is/are either randomly or selected by a quota system to sample a subset of a population or document corpus. Examining the sample (i.e., a smaller number) makes it possible to make data-based statements about the population or document corpus (large number).

Although the phrase drawing a sample seems to be universally valid today, there are very specific forms of sampling procedures for standardized, in particular, large numeric data using methods (e.g., survey), and non-standardized

methods (e.g., see Sections 6.4, 11.6, and 13.5). Except in the case of the completely random sampling of respondents, researchers must consider selection criteria and justify drawing a sample. For example, for a comparative study (e.g., Smelser, 2003) of universities in Austria, Germany, and the German-speaking part of Switzerland, the almost *natural* limitation of the language area is not sufficient as a justification for case selection. The three similar cases also differ in at least one or more characteristics, which are relevant for case selection justification (e.g., Seawright & Gerring, 2008). Although universities in the German-speaking countries have developed over almost two centuries according to the Humboldtian model (Östling, 2020), and in all three cases the higher education system includes universities and Universities of Applied Sciences (also known as Polytechnics), differences exist in higher education governance and because of recent developments (e.g., German Excellence Strategy to create world-class, elite universities, following the example of elite universities in USA).

In general, purposive small-n case selection is preferable to random sampling, especially when there is little or no knowledge about a social phenomenon (Akremi, 2019; Suri, 2011). Single-case or small-n observational or interview studies are suitable for the exploratory investigation of a new or newly discovered phenomenon, that has not yet been researched, and taking a fresh look at existing social phenomena. For example, for exploring a new phenomenon, a "thick description" (Geertz, 2002; see Box 3.2) can be used for in-depth qualitative knowledge production. This combines both the characteristics of (ethnographic) observation of a social phenomenon with the analytical study of social organizations and their supporting structures, that is, based on a systematic description of patterns of human coexistence in groups.[1]

Box 3.2 Brief definition of thick description

Clifford Geertz (2002) introduced the term *thick* description* in the context of cultural observations. As a pattern analysis, thick description is used to understand human actions and everyday routines. The aim of a thick description is to explain and interpret structures as an interplay of social constructivism or network of knowledge and cultural meaning ("web of significance" (Geertz, 2002, p. 12)).

* thick = dense [nature, e.g., forest, fog, and hair]; thick [buildings, e.g., walls]; creamy [gastronomy, e.g., gravy]

For example, for studying a new or sensitive phenomenon, access to interviewees may be difficult. Accordingly, doing snowball sampling (e.g., one interviewee recommends one or more other interviewees) may be the only option, which constructs a nonrandom sample. Still a nonrandom, that is, purposively justification for a selection process of a sample of subjects for interviews can

be justified on careful considerations about (potential) differences between the groups that would allow – or hinder – you to analyze the phenomenon of interest in a systematic way. You could also start with selecting a few interviewees, conduct interviews, analyze the interviews, and – based on your analysis – you could select additional interviewees for interviews who offer the greatest possible contrast to those group of subjects you interviewed first. This approach is called theoretical sampling and was developed in the context of grounded theory (Glaser & Strauss, 2017).

Nonrandom sampling as convenience sampling can also apply when selecting cases (e.g., four large companies that grant you access to documents and employees for interviews) that can be easily accessed but are studied for good reasons. In-depth analysis are usually conducted either with individual case studies – such as one or a few interviews or ethnographic observation – or with the analysis of a few units of analysis, such as text documents or organizations. The analysis of an organization also is considered 'a case' when an organization is analyzed based on interviews. In this 'case,' the interviews are *nested* within an organization.

The possibilities and limitations of large-n and small-n studies as well as a combination of these for the purpose of comparison (e.g., of countries) are summarized in Table 3.2.

3.2.2 Process-produced data: data types and operationalization

The difference between research- and process-produced data is that research-produced data – primarily in text and numerical form – is collected by observing and interviewing individuals and groups. In comparison to research-produced data, normally, researchers have no influence on process-produced data. For example, parties publish text and slogans on their party platforms before elections and individuals, to support a party or a politician, make videos or leave texts of varying (language) quality and (text) quantity on relevant Internet platforms.

The reasons and processes of process-produced data varies considerably. For example, in administrative processes, legal texts define which written documents (e.g., in public administration and court) must be collected and archived with or without a time limit for the purpose of clear bureaucratic processes (Bick & Müller, 1984). In contrast to administrative and bureaucratic documents and documents produced by individuals and organizations (e.g., political parties), the authors of data produced in everyday life are not always known. Data available on YouTube, Facebook, and other social media platforms can originate from fake user accounts. In addition, only certain age and population groups contribute to specific data volumes in online social media, which means that mass data available on the Internet is never representative of the population as a whole (Graff & Baur, 2020; see Box 3.3). However, if all of the users of an Internet platform constitute your population

Table 3.2 Characteristics of Small-n and Large-n Studies

	Large-n	*Small-n*	*Large-n in small-n Comparison*
Ideal sampling	Random selection	Purposive case(s) selection	Combination of purposive small-n case selection and random large-n case selection
Data/methods	Standardized/quantitative data	Unstandardized /qualitative data	Standardized/quantitative data (e.g., questionnaire) Unstandardized/qualitative data of the investigated social phenomenon for cases (e.g., countries)
Strengths/ possibilities	Representativeness	In-depth understanding of the case(s) through thick description	Representativeness for the case (e.g., large-n in countries) Systematization and consideration of contexts for comparison of cases (e.g., through theory and/or analytical framework)
Weaknesses/limits	Findings are limited to the study population Selectivity within the study population is often not considered	Findings are limited to a case or cases, but inform similar cases	Necessary to map the social phenomenon in similar cases (e.g., globally)
Requirements for application	Prior in-depth knowledge of the social phenomenon to be investigated Measurement equivalence is given. Availability of standardized data	Acquisition of in-depth knowledge about the social phenomenon in the context of the case(s) Explorative approach (e.g., for the purpose of developing a hypothesis) to the social phenomenon under investigation	Acquisition of in-depth knowledge of the social phenomenon in specific contexts Cases must be comparable Availability of standardized data

Source: Schneijderberg and Götze (2021, Table 2); translated and adapted by authors

from which a sample is drawn, then this sample can at least achieve approximate representativeness, provided you know the distribution of parameters such as age, gender, and social background of the users, on the basis of which you can differentiate between groups. In this case, you would combine a large-n (users) with a small-n (a platform) analysis.

Box 3.3 Brief definition of representativeness

This is a term used in quantitative research. The group, documents, or other objects studied is/are either the target population or a (randomly or selected by a quota system) sample as a subset of the population. Examining the sample (small number) makes it possible to make data-based statements about the population (large number).

In Table 3.1, process-produced data is differentiated according to the categories of documents as (mass) data that exists in different forms (e.g., books, music, and videos), web-based mass data (big data), and administrative/bureaucratic mass data (big data). The aim underlying the production of data for the different data types illustrates another feature of process-produced data. While research-produced data is collected or produced with the aim of answering research questions, existing in archives or online process-produced data makes it possible for research questions to be answered without data being shaped by theory, methods, or researchers' perspectives.

Documents as (mass) data, such as books, or on party or social media platforms, are produced with the general goal of communication. Despite this common goal, the reason why the documents were produced naturally differs. For example, party platforms are written with the purpose of communicating the political goals of parties in order to obtain votes from voters in elections. In contrast, books in novel form are produced primarily for entertainment purposes, whereas guidebooks are used to inform and support readers in their decision-making. The materiality of books and party platforms is both printed and electronic, digital form (e.g., as PDFs) although both document types can also be available only in print (e.g., old books) or in electronic form. However, the (partial) overlap of the materiality of documents as (mass) data merely means that nondigital documents must be digitized in order to be analyzed. Similarly, documents that exist digitally as video or audio, for example, must also be converted into a digital text form. Social media documents are always digital and do not exist outside the Internet realm.

The digitization of printed material can take place as part of investigating the bureaucracy of administrative processes but is not a characteristic of web-based mass data. In contrast to documents as (mass) data, the materiality of web-based mass data is not a self-contained unit of analysis. Especially web-based mass data is information that is continuously updated in real time, although updated

can mean additional, modified, and/or deleted data. As a result, the volatility and a certain incompleteness of information are characteristics of web-based data to be considered when using process-produced data for research purposes. In some cases, the temporary availability of information is defined by the program (e.g., Snapchat).

The flow of information is established for each specific website in the general terms and conditions of the companies – or organizations – offering the website. Web-based information is organized by the

1. companies (e.g., Google algorithm and X hashtag);
2. individuals contributing content; and
3. users (e.g., as readers) of the content.

What web-based mass data and documents as (mass) data have in common are the characteristics of information, for example, to "closing the gap of knowledge ... between communicator and recipient" (Schulz, 2009, p. 161), and communication as a "(symbolic interaction) between people on a technical basis" (Beck, 2010, p. 16). However, document-mediated information transfer is one-sided communication of the content of the document from the communicator to recipient, which the latter cannot respond to within the document. With web-based mass data, however, reciprocal or interactive communication can occur (e.g., responses to a tweet). In the case of process-produced data, web-based mass data has the unique feature that its information is recorded as data according to certain criteria (Willke, 2004, p. 31).

The web-based nature of data allows for the continuous collection of data about users who contribute content and communication. As a result, the exchange of personal data for information is a unique feature of process-produced web-based data. For example, the Google algorithm adapts the information displayed to the usage habits of the individuals looking for information – as happens with personalized advertisements (Courtois et al., 2018; Powers, 2017). In contrast, explicit metrics were displayed in the form of activity data for information contributors on social media platforms (e.g., messages and followers). YouTube is programmed to show users how often, for example, the music video of a song has been viewed; however, the video as a document and as such a self-contained unit of analysis does not collect any data about users. In addition, the song is not only available as a music video on YouTube but also on the band's website, the music company's website, and other (music) websites in identical form. If there are different versions (i.e., documents), this is indicated by a statement of whether it is an official video, a concert recording, or another version. The number of documents of the song presented in different videos multiplies accordingly, making it possible to systematically compare the lyrics, vocal variations, or the like in these documents.

3.2.3 Data collection for analysis with process-produced data

The example of the music video makes it clear that a document can be part of a website, but that the reverse is not possible. Table 3.1 therefore lists datasets as a category of data to be collected for web-based mass data. For the analysis of datasets, it is necessary to record the web-based mass data in its various versions as an entity and to transfer it to a database and clean it. An entity is nothing other than an individual object (e.g., a music video) within a database that has connections to other entities of the same or different type (e.g., artists or music labels) and has identifiable properties (e.g., length of the piece of music, audio properties, or number of viewers) (Chen, 1976; Elmasri et al., 1985). For these entities, we must take into account that the documentation of web-based mass data only takes place for the time period specified in the research design. Given the real-time nature of web-based mass data, the dataset of the website content – which are the entities of interest for your study – can change less than a second after research archiving (e.g., new social media post). This is also the case for the connections and properties of your entities. And it is not always possible to trace changes to the website content documented for research purposes (e.g., website content deleted or modified because of a breach of the terms and conditions or an online shitstorm).

In contrast to web-based mass data, changes in (administrative and archived) documents as (mass) data are generally traceable, even if tracing them may involve some effort. For example, an image can be compared with another photo, even if the same section of a photo is missing or unusable in a series of photos (e.g., in the case of photos taken by webcams at famous tourist locations) taken once a day over a specified period of time. In the case of photos as document data, it can be clearly documented that a photo is missing or cannot be used (e.g., photos of historical buildings or famous people in archives), and the photo series is therefore incomplete.

In the case of documents as (mass) data, the completeness of the data is not always given. The existence of authors is one key to checking completeness and defining usefulness of documents for research purposes. For example, party platforms can be assigned to parties, books to authors, songs to singers, or bands and images to artists. However, it would be (almost) impossible to find all films or screenplays with researchers in leading and supporting roles as documents to be analyzed. The same definition and selection problem exists, for example, if you wanted to use topic modeling (see Chapters 17 and 18) to investigate films that have a science connection, whereby the restriction to English-language films makes it easier to ensure the completeness of the data. In contrast to documents as (mass) data, we can assume that web-based mass data in itself will be incomplete. This is a result of the self-selection of potential users (limited by lack of Internet access or insufficient knowledge) and the associated unclear boundaries of the user population.

However, process-produced data as a collection of administrative mass data defined by law should be complete. Archiving of administrative procedures and evidence from court cases may require all types of data including audio, images (e.g., photos and pictures), texts, videos, and numbers. Although the text form of transcripts and written information in a file is the predominant document form, video recordings from a surveillance camera and images from a crime scene can be part of the documents archived in criminal proceedings.

Document archiving regulated by law is also relevant for data analysis purposes for practical applications. Such practical applications could include evaluations of administrative processes with the aim of optimizing bureaucracy or the implementation of pattern analysis (including profiles of offenders) in certain criminal cases with the aim of solving crimes. Analysis for a specific purpose are carried out by members of an institution such as police officers and, if necessary, specialized administrative staff. Data analysis may only be accessible to a specific group of people, particularly in the case of personal documents that are very sensitive under data protection laws such as patient files and court proceedings. Accordingly, the problem with administrative mass data is not archiving and completeness according to the number of cases, but whether the data can be accessed and used for research studies. Data protection and anonymization agreements regulate the way in which data is analyzed and results are presented.

Process-produced data is produced on a big scale on social media sites or commercial platforms. If enabled by the business owner, the data can be downloaded using application programming interfaces (APIs). APIs are interfaces between computer programs and databases that make it possible to make specific data requests and download large amounts of data (see Section 11.6). To request data, you must first register as a user with the services to which the API is linked. After registering, you will be sent or assigned a user key by email or on the respective service. This can be used to make data requests on the site itself or in programming environments (for Python, see Chapter 13). If the data accessed is available in the database, it is stored temporarily and must then be saved in separate files or a shared data format.

Another option for collecting process-produced data is web scraping. Web scraping is used to extract and automatically download publicly accessible content from websites specified in advance. Web scraping, on the other hand, automatically downloads publicly accessible content from previously specified websites. However, we do not recommend to use web scraping due to legal, ethical, and privacy concerns (Gold & Latonero, 2017; Krotov et al., 2020).

In both cases, you will have to consider research ethics and legal issues, although this is more so the case with web scraping than with the use of APIs. In the case of APIs, you should read the terms of use very carefully as these contain information about how many data requests you are allowed to make in what period of time and which information you are allowed and not allowed to download. Make sure to always check the data to see whether it can be used

to identify users. In any case, you should anonymize the data and aggregate it (i.e., extrapolate it to a group level) in such a way that the individual users are not the focus of the data.

Ethical concerns arise from web crawling in the research process with regard to the owners of the websites crawled and their users (Thelwall & Stuart, 2006). These include the fact that access to the website during data collection reduces the speed of page retrieval for third parties or can even make it impossible if the number of crawler requests is too high. In addition, there may be problems with the privacy and anonymity of the users whose data is retrieved and stored by the researcher.

In this context, Gold and Latonero (2017, pp. 300–302) rightly emphasize that web crawlers can quickly access sensitive, personal data when (text) data and other content from one or more websites is collected. Furthermore, we should consider that users of web services have rarely read the terms and conditions and therefore often have no idea of the extent to which data is collected and analyzed (Obar & Oeldorf-Hirsch, 2020). Moreover, web crawlers can steal intellectual property, undermine a site's business models, and collect information that violates trade secrets (Krotov et al., 2020).

3.3 Reuse and provision of data

At this point, we would like to briefly explain how you can use open data and how you can obtain data for secondary analysis. We would also like to give you a few tips on how you can open up data you may have collected yourself to other researchers and make it reusable.

You cannot simply download personal data (i.e., data collected by means of a questionnaire survey or interviews) from the Internet. This is because personal data in Europe is subject to the General Data Protection Regulation (GDPR), which provides it special protection. As a result, you can only find this type of research data at research data centers (RDCs). These centers ensure that the data is archived in anonymized form and is only made available for reuse under certain conditions. For example, as a student or doctoral researcher, you will generally have to submit an application to reuse specific data.

In social science data repositories, most of the datasets are quantitative data obtained from large surveys (primarily questionnaire surveys). However, an increasing amount of qualitative data is also being archived and made available for reuse. As anonymization is much more difficult and time-consuming here, many researchers are still reluctant to do this, even though it is possible and in principle desirable (Steinhardt et al., 2021).

In addition to reusing research data, you can also consider whether you would like to make data you have collected available for reuse by other researchers. We would strongly encourage you to do so but understand that this means additional work and that especially in the case of final bachelor's and master's theses, you may not have the resources available. For dissertations, however, archiving and issues of reuse should be considered in any case.

For open data (i.e., data that is freely available on the Internet), it is possible to document and publish a compilation of the data. The data collected cannot usually be made available for reuse. What exactly does this mean? For example, you might obtain data such as tweets or the metadata of publications from social media platforms or databases for your research and analyze these. You can publish documentation that includes which data you have downloaded at which point in time, according to which search criteria and other relevant information. However, you may not publish the data itself (i.e., the tweets or the metadata of the publications) as a dataset as this is not your own data and you are not the owner of the data. In these cases, the owners are the social media platforms or the owners of the databases.

The situation is different for datasets that you have collected yourself through a questionnaire survey or interviews. You can make this data available for reuse in anonymized form, but only if you have obtained informed consent for this at the time of collection. This is also possible retrospectively, but would be much more complicated, which is why it is essential to consider the issue of reuse when you are preparing your survey. For example, if you carry out a questionnaire survey and do not obtain permission to reuse the data when the questionnaire is completed, then it is hardly possible to obtain the permission of all respondents at a later point in time. And since you usually do not know which person corresponds to which case, you cannot eliminate one specific case and therefore cannot make the entire dataset available for reuse.

If you have the informed consent of the respondents, you cannot simply put the data online! Instead, use a research data center for this. The center will help you to anonymize the data so that you do not come into conflict with the General Data Protection Regulation. In addition, contextual information is required for meaningful reuse, and this must also be made available. This is why research data that is stored at a research data center always has a methods report. The report describes how the data was collected, by whom, when, in which format it is available, how it was analyzed in the primary study, which results have been published, and most importantly, which measures have been taken to ensure anonymization.

If you have data that is not subject to data protection or copyright (e.g., corpora of old literature), you can use open repositories to make your data available online. Repositories are large databases that include datasets that have been prepared and annotated. A repository is a data management system. One of the largest is the Harvard Database, in which tens of thousands of datasets are stored for replication purposes and can be cited using a DOI, which stands for digital object identifier. Users can use the search fields to search for specific text corpora.

3.4 Data protection, ethical behavior, and informed consent

The ethical and data protection considerations for qualitative and quantitative social research involve navigating both international and country-specific legal regulations. While some ethical and data protection principles are universally

accepted, for example by professional associations, others vary between countries and jurisdictions. We first give you a short overview of internationally common standards in ethics and data protection for qualitative and quantitative research

3.4.1 Informed consent by participants for data collection

When collecting personal data, such as through interviews, participant observation or a questionnaire, informed consent must be obtained from respondents, either in written form or recorded verbal agreement. What is considered personal or sensitive data varies from country to country. In the European Union (EU), according to the General Data Protection Regulation[2] (GDPR), sensitive data includes racial/ethnic origin, political opinions, religious beliefs, health, sexual orientation, age, sex, etc. In most countries, informed consent must be given in writing by the informant(s) (e.g., interviewees and respondents in a survey).

The informed consent form must be designed so that participants understand the purpose of the study, how their data will be used, and their right to withdraw their participation and data at any time without consequence. The consent form you present to your informant(s) must include details about the purpose of the study, confidentiality measures, data retention policies, the duration of data storage, the transfer of the data, as well as voluntary participation, withdrawal procedures, and contact information of the researcher in charge and data privacy officer at your institution. In certain cultural circumstances, it may be necessary to obtain consent only verbally if the respondents have an oral culture or are illiterate. It is important to always take cultural specificities into account and to obtain consent preferably in the first language of informants to ensure that all aspects have been understood.

You may find that drafting a clear consent form is not as easy as you think. This is because you need to list and take into account the legal requirements of each country regarding data protection. As a result, you will have to wrestle with legalese. Our tip is therefore: If you are writing a term paper or a bachelor, master or doctoral thesis, ask your supervisor if she or he has already drawn up a consent form. If your supervisor is unable to help you, you can also contact your university's ethics committee, if your university has one. A third point of contact may be associations in your discipline, which have often formulated country-specific declarations of consent.

3.4.2 Data protection, data transfer, and anonymity of informants

The safe handling of personal data is governed differently in each country or on a supra-national level:

- European Union (EU): The General Data Protection Regulation (GDPR) governs data protection and privacy. Researchers must comply with stringent

guidelines regarding explicit consent, data minimization, transparency, and data security.
- USA: There is no unified data protection law. Instead, laws like the HIPAA (for health data) or state-level privacy laws (e.g., CCPA in California) might apply. Research institutions often rely on ethical standards from The Common Rule for federally funded research.
- Australia: The Australian Privacy Act (1988) governs personal information, including research data. Specific ethical guidelines include those from the National Health and Medical Research Council (NHMRC).
- Canada: The Personal Information Protection and Electronic Documents Act (PIPEDA) regulates private sector data use. Research ethics are guided by the Tri-Council Policy Statement (TCPS2).
- India: Data use is regulated by the Information Technology Act (2000) and rules on sensitive data. The new Digital Personal Data Protection Act (2023) is being implemented.
- China: Data protection is subject to the Personal Information Protection Law (PIPL) and may impose restrictions on the transfer of data outside China.

As you can see, there are many different regulations, especially when it comes to handling data. It is therefore essential that you inform yourself well and comply with the law. As described in the previous section, you need to be aware of the following aspects of data protection: Data retention policies, the duration of data storage, and the transfer of the data. In the EU the GDPR mandates data retention for no longer than necessary for the purposes of research. Other countries have fixed periods for retention. Please also note that, for example, the EU strictly regulates the transfer of personal data to countries that do not provide equivalent standard protection, for example China and the USA. In addition, data transfers to the USA may require additional safeguards under Standard Contractual Clauses (SCCs) or Privacy Frameworks. Data transfer out of the country is just as tricky, for example Chinese laws such as the PIPL require government approval to export data abroad. If you fail to comply with legal requirements on data protection and transfer, you can be fined or even be sentenced to imprisonment.

In addition to these regulations, it is ethically appropriate to protect participants' data and not share it with third parties without their (written) consent. This means that it is best not to store the data in a cloud (e.g., Dropbox). In addition, the data should always be anonymized or pseudonymized, unless you have participants' consent that this is not necessary, which is rarely the case. Anonymization and pseudonymization means that you delete or rewrite the parts of a transcript that allow conclusions to be drawn about the identity of an individual. Anonymization means deleting personal data and indicating this with parentheses, for example deleting a city name in a transcript and replacing it by [name of a place] (see also Sections 7.2.2 and 8.2.2). Pseudonymization means replacing the name with another name, for example turn a Maria into a Theresa. When pseudonymizing, you should try to find equivalents (gendered)

local names whenever possible, that is, do not choose names from different cultures as substitutes. Another data protection guideline is to collect only the data that is really necessary for your research.

3.4.3 Ethical behavior

Ethical behavior is core for every professional researcher and student, and it is the responsibility of individual researchers and students to adhere to the code of ethics of her/his professional association. For example, the International Sociological Association's (ISA) Code of Ethics[3] states in the preamble:

> "Sociologists work to develop a reliable and valid body of scientific knowledge based on research and, thereby, to contribute to the improvement of the global human condition. The primary goals of the Code of Ethics, a symbol of the identity of the ISA, are (1) to protect the welfare of groups and individuals with whom and on whom sociologists work or who are involved in sociologists' research efforts and (2) to guide the behaviour and hence the expectations of ISA members, both between themselves and toward the society at large. Those who accept its principles are expected to interpret them in good faith, to respect them, to make sure they are respected and to make them widely known.
>
> Each sociologist supplements the Code of Ethics in ways based on her/his own personal values, culture and experience. Each sociologist supplements, but does not violate, the standards outlined in this Code of Ethics. It is the individual responsibility of each sociologist to aspire to the highest standards of conduct."

The ISA code of ethics defines three areas: "1. sociology as a field of scientific study and practice," "2. research procedures" (specifying sponsorship, costs and rewards, and data gathering), "3. publication and communication of data," and "4. extra-scientific use of research results." Please search online for the code of ethics of your (inter)national association of anthropology, economics, education, ethnology, data science, political science, psychology, or any other discipline/research field you are associated with.

In general, when collecting data, you rely on informants to trust you with their information, and vice versa. Therefore, it is crucial to establish a foundation of trust and carefully consider any risks to informants. After all, it is important to avoid harming informants, especially when collecting sensitive data.

In addition to data protection issues, you should consider whether there are any risks related to your research, which may cause an emotional response, psychological discomfort, or re-traumatization. Especially in medical and psychological fields but also when doing research about small children or the elderly, in many countries you have to get ethics approval from research ethics committees (REC) or institutional review boards (IRB) before starting the research. This includes a review of the consent form, the research methodology, and data

collection instrument. Even if doing your research in a country where REC or IRB approval is not mandatory, obtaining ethical approval enhances your credibility and ensures informants about your compliance with ethical practices.

Notes

1 In-depth and systematic knowledge produced by small-n studies can be used, for example, to identify correlations and develop indicators (Smelser, 2003, p. 646). Indicators are measuring instruments (e.g., questions or images used as stimuli to evoke emotions) that address a small part of the phenomenon under investigation and can be operationalized in a questionnaire for a survey. Operationalization is the process of making something measurable and describes the division of an empirically observed social phenomenon or theory into analytical parts that are transferred to the form of easily understandable questions. The sequential mixed methods design described here (Creswell & Plano Clark, 2011; Tashakkori & Teddlie, 2003), that is, the sequential approach of qualitative and subsequent quantitative investigation, can also be carried out the other way around. This is the case, for example, when exploratory large-n questionnaire surveys are used in preparation for the in-depth study (e.g., with interviews) of a population that has been identified (e.g., a new professional category).
2 See: retrieved April 7, 2025, from https://gdpr-info.eu/.
3 See retrieved April 7, 2025, from https://www.isa-sociology.org/en/about-isa/code-of-ethics.

4

RESEARCH QUALITY CONVENTIONS

A guide for good quality empirical research

This chapter introduces the "substantive methods concept" (Schneijderberg, 2023) for good quality research and provides a step-by-step guide for the entire research process from initial observation to conclusions by focusing on research design, data collection and processing, and analysis and interpretation. The substantive methods concept – composed of discovery, appropriateness-transparency, and iteration and distancing quality conventions – guides deductive, theory-led and inductive, theory-reflexive approaches to research as well as qualitative small and quantitative big data analysis. In addition to differences associated with different methods, the three quality conventions respect particular aspects of data collection and data type (see Chapter 3). For example, ethnographic observations are documented in field notes by researchers, and interviews are conducted according to a manual guiding the interactive communication between two or more subjects. We believe that all particularities (e.g., data collection and processing) of qualitative methods can be addressed by the discovery, appropriateness-transparency, and iteration and distancing quality conventions. The substantive methods concept can also integrate quality conventions specifying methods for data collected by surveys, quantitative data processing, and statistical analysis. Of course, epistemology, that is, how scientific knowledge is created, and methodology, that is, the logical process of applying a method or methods, differ between qualitative, quantitative, and mixed methods research – but not the basic research process.

4.1 Introduction: quality conventions for guiding the research process

Discovery by research is fantastic – but it is not fantasy. Some discoveries are made by thinking, others by observing natural, social (including cultural, economic, political, etc.), and/or technical phenomena. Discovery is rarely made

DOI: 10.4324/9781003496397-5
This chapter has been made available under a CC-BY-NC-ND 4.0 license.

by accident, mostly by systematic work and by following a specific method. Without following a method, scientific discoveries are random and questionable. However, a method and its methodology, that is, how to logically apply a method in a research process, is only one part of empirical research. The other part of empirical research is defined by the subjects and/or objects under study, the social (including cultural, economic, political, etc.), spatial, and temporal context, and the fact that research is done by diverse human beings. Accordingly, no one research process is like another, and quality conventions are a means to guide empirical research.

In general, conventions guide the empirical research process and define the requirements for methodological justifications of different parts of the research process. Researchers following quality conventions enable others to evaluate and be convinced of the good quality of a research project and its findings.[1] Quality conventions address how

1. to make discoveries in empirical research,
2. a research process should be made intersubjectively understandable (e.g., justification and appropriateness of research questions, case selection or sampling, and method for data collection, or decisions made during programming), and
3. to proceed in data analysis and interpretation, and reach conclusions by iteration to gain distance from the data (e.g., supported by theory and an interpretation group).

In addition to the commonalities of quality conventions for a research process, differences exist between methods because of the type of data (see Chapter 3): ethnographic observations are documented in researchers' field notes, text documents can be gathered from webpages, interviewing is interactive and requires communication, and statistical information can be collected by surveys or metadata on the Internet. Accordingly, quality conventions must differ for data collection by reactive methods (e.g., interview and survey) and nonreactive methods (e.g., documents and observation) as well as for specificities of data processing and predictive analysis. For example, in addition to guiding the process of selecting a (random) sample group for conducting a survey and of respondents completing a questionnaire, in statistical analysis, you can perform calculations and conduct tests.

However, quality conventions for statistical analysis (e.g., objectivity, reliability, and validity) do not fit qualitative and quantitative content analysis for three reasons: First, content analysis can be used for analyzing text data from various types of documents, for example files, social media posts, and ethnographic field notes, as well as transcriptions of interviews (verbal expressions to text), pictures (observations to text), and videos or films (verbal expressions, sounds/music, and moving pictures to text). Second, content analysis is a method in its own right (e.g., for qualitative content analysis: Kuckartz & Rädiker, 2022; Schreier, 2012), and for data processing as well as analysis and

interpretation, content analysis can be combined or merged with other methods, such as interviews and focus groups. Third, content analysis can be either qualitative or quantitative or combined in a mixed methods study, for example using quantitative content analysis to sample (parts of) documents for in-depth qualitative content analysis.

Finally, content analysis is scalable and, depending on the type of analysis and amount of data, is done by applying classical (manual), semi-automated, and automated procedures for data analysis and interpretation. For example, the methods for quantitative content analysis presented in this handbook are relational methods for content analysis, namely multiple correspondence analysis and semantic network analysis (Chapter 15), sentiment analysis (Chapter 16), and topic modeling (Chapters 17 and 18). They are inductive, that is, data-driven approaches that are also considered unsupervised machine learning algorithms which enable to detect interpretable patterns in complex, large-scale, process-produced data. Accordingly, such inductive quantitative content analysis requires the use of quality conventions for qualitative content analysis (e.g., appropriateness and transparency), at first. In a second step, topic distributions, sentiment polarities, or sentiment classes can be injected into further statistical analysis either as input variables (explanans) or target variables (explanandum). Depending on your research aim, you can use structuration techniques, such as multiple factor analysis (Escofier & Pages, 1994) or socio-semantic network analysis (Basov et al., 2020), or regression techniques (Erhard & Heiberger, 2023) if you seek to predict, for example, the sentiment polarity based on the properties of the discourse or the speakers therein (Abercrombie & Batista-Navarro, 2020). Accordingly, it seems plausible to use quality conventions that focus on the research process and respect qualitative and quantitative data specificities.

Searching for quality conventions that respect qualitative and quantitative data specificities, one of the three authors of this handbook conducted a qualitative comparative content analysis of existing quality conventions (Schneijderberg, 2023) – this was in the aftermath of publishing the first edition of this handbook on qualitative and quantitative content analysis in German, in 2022. As a conclusion of the theory-led comparison between conventions for qualitative and quantitative methods, Christian (Schneijderberg, 2023) suggests the use of a substantive methods concept (in German: *substanzieller Methodenbegriff*) focusing on the research process. The substantive methods concept for any empirical research process relies on three quality conventions: discovery, appropriateness-transparency, and iteration and distancing.

The discovery, appropriateness-transparency, and iteration and distancing conventions address the entire empirical research process and foster the compatibility of both methods and qualitative and quantitative data. In addition to guiding any empirical research process, the three quality conventions support the integration of data-specific quality conventions. For example, for qualitative content analysis, the three quality conventions guide the entire research process, as will be explained in Section 4.3. For ethnographic observations,

interviews, or focus groups, it is necessary to consider further quality conventions guiding the documentation of observations, how to design an interview manual or communicate in the social situation of a focus group. For quantitative data processing and analysis, statistical tests remain relevant and are accompanied by best practices regarding the programming, analysis, and visualization of the results. However, the three quality conventions emphasize that research is a cognitive and social process and, therefore, objectivity seems nothing more than a questionable truth-ideal beyond research practices, as will be explained in the following Section 4.2.

4.2 Quality conventions frame the cognitive and social process of research

In general, research in mathematical, natural, social, and technical sciences as well as humanities is defined as a cognitive and social process (e.g., Barnes, 1982; Becher & Trowler, 1984/2001; Fleck, 1935/2019; Knorr-Cetina, 1999; Latour & Woolgar, 1979; Kuhn, 1962/1976; Whitley, 1984). Cognition refers to the mental process of thinking, understanding, and acquiring new knowledge. Cognition also depends on perception, evaluation, problem-solving, information-processing, and reasoning capacities, which are all relevant in a research process, for example for collecting and analyzing data, discovering patterns, interpreting social actions, and attributing meaning. The adjective 'social' emphasizes that research is done by humans, possibly supported by algorithms and AI programmed by humans. The verb 'to do' indicates that the research actions taken by humans do vary – even though applying the same method – and, therefore, need to be accounted for in a transparent manner. Without transparency about the case or sample, processes of data collection, analysis and interpretation, and intersubjective reliability and perception of appropriateness, research findings will not be socially acceptable, for example, to peers for publication in a journal.

The order in the definition of the research process as, first, a cognitive and, second, a social endeavor is very important. Research as a cognitive and social process makes it very clear that the focus is on the systematic application of a method, use of theory, analysis and interpretation of (empirical) results, etc., which are done by a researcher or group of researchers. Flipping the word order to 'social and cognitive' or 'socio-cognitive' endeavor indicates the supremacy of social prestige and rank over the cognitive performance of the researcher during the research endeavor. In this case, a professor would, *per se*, be considered to do good research without critical examination of the research process, systematic application of a method, empirical grounding of findings and interpretations, etc. By prestige and rank, a professor would also be considered to do better research than a researcher without the status of a professor. We hope that it becomes apparent why the cognitive should trump the social element in a research process, and why the research capacity of an academic must be

evaluated according to quality conventions for research irrespectively of seniority or rank.

The definition of the research process as a cognitive and social endeavor is also very important for another reason: To avoid the false pretenses of data engineering suggested by classical test theory (e.g., Lincoln & Guba, 1985; Popper, 1935/2002, 1972; Yin, 1981). Denying the involvement of humans in the research process, classical test theory solely addresses the objectivity, reliability, and validity of sampling and data production (e.g., surveys) and statistical analysis. Accordingly, for classical test theory it is irrelevant that researchers construct survey questionnaires, that is, humans formulate questions and items and decide on which scales to use (e.g., a Likert scale with either five, six, seven, or ten responses). Likewise, researchers decide on the sampling method (e.g., random or stratified random sampling), data cleaning and data preprocessing procedures, analyze and interpret results, and reach conclusions deduced from the data.

Only recently, and not generally, have the social factors affecting the research process been reflected in quality conventions for statistical research, such as total survey error (e.g., Groves & Lyberg, 2010; Weisberg, 2009) and lack of data transparency (e.g., scientific use files for secondary data analysis), which has led to the so-called reproducibility crisis in quantitative research (e.g., Baker, 2016; Hensel, 2021). Although it is not related to quality, the social practice in statistics of p-hacking is just as alarming, that is, data dredging during analysis to construct statistically significant results or patterns in data that enhance the publication chances of a study (e.g., Brodeur et al., 2020; Friese & Frankenbach, 2020; Gupta & Bosco, 2023).

Statistical tests are also applied in the field of automated text analysis to test the accuracy of algorithms, for example, to correctly identify and predict a certain result, since many studies in the field of machine learning are subject to distortions (Myrtveit et al., 2005; van Atteveldt et al., 2021) and algorithmic misjudgments of abstract categories including sentiments.[2] Furthermore, the Matthews correlation coefficient (Yao & Shepperd, 2020) measures the agreement between the algorithm and human categorizations.[3] In addition, you must always bear in mind that most analyses in the field of automated text analysis are based on corpora that are extracted from a database. Although these corpora usually contain a large number of texts, which can run into the millions, strictly speaking it is a case-by-case analysis. Accordingly, there are problems with transferability to other cases or other data corpora. Cross-validation is one option to overcome such problems (e.g., Browne, 2000), as well as the split of the collected data into training-, validation-, and test-sets to approximate generalizability in machine learning approaches (Grimmer et al., 2021).[4] Nevertheless, the results cannot be generalized due to inequalities in the chances to produce the data due to the digital divide, as well as being represented in the datasets used for model training, statistical inference, and prediction[5] (Baur et al., 2020).

Beyond the special, unique path of classical test theory and quality conventions for quantitative research, the discussion of common quality conventions is not new in social sciences (e.g., Howe & Eisenhart, 1990) and, recently, has been debated for mixed methods research (e.g., Hirose & Creswell, 2023). However, adopting quantitative quality conventions for qualitative research (e.g., Flick, 2007/2018; Mayring, 2021; Miles & Hubermann, 1984, pp. 277–280; Steinke, 2004; Swanborn, 1996) entails severe friction, as we will outline in the following sections.

4.2.1 Same, same but different? The dark side of quality conventions

Despite the differences in quality conventions for qualitative, quantitative, and mixed methods research, when looking at the research process, qualitative and quantitative content analysis in particular and social science methods in general are closer than they appear in the side mirror. The focus of content analysis is on identifying manifest and latent patterns in text data by developing deductive and/or inductive categories (Box 4.1) using digitally supported, semi-automated, or fully automated research procedures. How patterns are identified in the data by analysis and interpretation is at the heart of every content analysis research process. Epistemology, that is, how scientific knowledge is created, and methodology, that is, the logical process that ought to be followed in a method, differs between qualitative, quantitative, and mixed methods research – but not the research

Box 4.1 Category scheme and coding

Definition 'category': A category is a unit to identify, classify, and sort actions, actors, artifacts, phenomena, etc. in qualitative content analysis and interpretation. A category should be distinct, have a clear meaning, and not overlap with other categories. Category definition problems indicate that already defined categories need to be reconsidered.

Definition 'category system': The category system is the result of a category construction process. A category system can be based on categories derived from theory (deductive research approach), categories developed from empirical data (inductive research approach), or a combination of deductively and inductively constructed categories. In general, a category system is composed of several top categories and associated subcategories.

Definition 'top category': A top category is like a first order heading which indicates the overarching content of a text. A top category summarizes two or more subcategories (i.e., similar to second order headings).

Definition 'subcategory': A subcategory specifies one aspect of the top category, for example 'research' and 'teaching' as

process. Schematically, the first part of the research process is formulating research question(s) and/or hypotheses, which are key for the research design (including method(s), use of theory, the literature review, etc.). Data collection is part of the middle segment in a research process – together with data processing (e.g., reviewing transcripts, metric data cleaning, and for secondary analysis).[6] The third part of the research process is data analysis and interpretation, and drawing conclusions.[7]

Looking into the methodological (side) mirror, the processes of qualitative and quantitative research are closer than they appear. However, instead of considering a quantitative research process to be fallible in just as many schematic segments as the similarly composed qualitative research process, or that it might completely fail (i.e., total survey error), researchers focus(ed) on criteria that only cover a small section of the research process: data collection and processing. In standardized, quantitative research, data collection and processing are supposed to be objective (i.e., data, findings, and interpretations are reproduceable; industrial/machine instead of human process), reliable (i.e., data and findings are free of errors; trueness of metrics, construct measures, item constructs, etc. without construction and interpretation need), and valid (i.e., instrument fit for data collection (e.g., questionnaire); not the survey designing nor experiment-performing researchers are responsible for the measurement fit). Considering the human ignorance of

1. human construction and conduct in the initial part of the research process, for example formulating the research question(s) and/or hypotheses, and constructing the research design, including decisions about random or random stratified sampling, and
2. human contributions to standardizing data collection and processing objectivity, reliability, and validity, that is, the second part of the research process,

subcategories of 'academic activities of a university' (top category). A subcategory has meaning on its own and, on a more abstract level, contributes to the overall meaning of the top category. A subcategory can have further subcategories.

Definition 'code' and 'coding': To code means to attribute a category to a segment of text, for example a sentence, several sentences, or a paragraph. How much text you code depends on the category-led construction of a unit of meaning, that is, coherent content in a sequence. Coding is the activity of creating order in document(s) by selecting units of meaning to answer the research question. Noncoded text segments are not considered relevant for answering the research question according to the category system. As in encryption, the coded units of meaning await decoding, that is, content analysis and interpretation by researchers.

3. humans inferring from and interpreting data, significance tests, etc., discussing results, deciding whether hypotheses are supported or rejected, offering explanations deductively guided by theory, drawing conclusions, and considering implications for policy, further research, etc. in the third part of the research process,

we wonder about statements like the following example by Lincoln and Guba (1985, p. 354) from their landmark book titled *Natural Inquiry*, which is an old term for qualitative research:

> But the most serious and central difficulty in the use of qualitative data is that methods of analysis are not well formulated. For quantitative data, there are clear conventions the researcher can use. But the analyst faced with a bank of qualitative data has very few guidelines for protection against self-delusion, let alone the presentation of 'unreliable' or 'invalid' conclusions to scientific or policymaking audiences.

For standardized, quantitative data, the "clear conventions the researcher can use" (Lincoln & Guba, 1985, p. 354) are objectivity, reliability, and validity, which only cover one of three parts of the research process. Accordingly, a quantitative researcher also has "few guidelines for protection against self-delusion" (Lincoln & Guba, 1985, p. 354) in the first and third part of the research process. In addition, not all qualitative methods are standardized with well-formulated methods of data analysis. As we will explain from Chapter 11 onwards, automated quantitative content analysis follows focus on objectivity, reliability, and validity, but also on conventions related to qualitative research at once, as the models employed seek to abstract (and thus objectify) subjective human interpretations for prediction purposes. Alternatively, our machine learning algorithms inform human interpretations after being applied on the data. In this case, the algorithm must, at first, work independently from human interpreters and thus be as objective as possible to produce the same output repeatedly in a concise and understandable manner (hence the objectivity, reliability and validity conventions followed by qualitative quality conventions).

So what is this quality conventions fuss all about? In a nutshell: to ignore and dominate. Ignoring, that is, the nonacceptance of conventions supporting the research process or (ideological) focusing on objectivity, reliability, and validity are used to negate the relevance of plural methods for studying plural social phenomena in societies, in social, spatial, and temporal contexts. The social mechanism of domination is related to, for example, statisticians ignoring the relevance, findings, etc. of ethnographic research. Domination as a social mechanism is rooted in the maintenance of categorical differences to classify a higher and lower, better and worse, or relevant and not relevant.

For example, statisticians and especially programmers who ignore findings from ethnographic, interview, and other qualitative research with reference to

missing objectivity, reliability, and validity cultivate a cumulative disadvantage. The cumulative disadvantage is not accepting findings from qualitative (e.g., ethnographic) research because they do not comply with statistical quality criteria. Of course, *vice versa*, qualitative researchers who ignore findings just because these are quantitative are showing the same questionable behavior. The research process in an ethnographic, interview, or statistical study does vary in terms of the type of data collected and used (e.g., gathered from observations or surveys). However, as explained previously, two-thirds of the research process are the same: Formulating a research question and research design (i.e., initial part of the research process), and analysis and interpretation, and drawing conclusions (final part of the research process). Accordingly, on the one hand, in social research, conventions to consider during the research process are necessary to legitimate empirical and theoretical discoveries. For legitimate scientific discovery, researchers are required to present the appropriateness of their method application as well as their data analysis and interpretation and make transparent how they arrived at a particular finding during their research process.

4.2.2 Conventions to (e)valuate the quality of research

On the other hand, conventions are used to justify and evaluate the *quality* of research. For example, in quantitative research, the validity of an instrument (e.g., a questionnaire for a survey) can be statistically assessed, and statistical tests provide cues for avoiding errors in data and findings (i.e., reliability). However, in qualitative research, the main instrument of data collection, data processing as well as analysis and interpretation is the researcher. Nevertheless, some, like the psychologist Mayring (2021), postulate reliability to be a *measure* for the quality of qualitative research. The *measure* proposed by Mayring (2021) is intercoder reliability, that is, measuring the percentage of similar text codings attributed by two or more researchers during 'qualitative' content analysis. But what does a 75% intercoder reliability mean? It means that in three out of four cases, two researchers attributed the same code to a unit of meaning. Why this seems relevant in quantitative or even in qualitative content analysis is not plausible to us for mainly three reasons: First, intercoder reliability only captures a small part of data processing in the second part in the research process. Second, in terms of data processing, intercoder reliability captures the cognitive capacities of the two or more coders; if the coders are also responsible for data collection, intercoder reliability *measures* the cognitive capacities of the research instruments, which are addressed by the validity convention of quantitative research. Finally, depending on the size of the text dataset, Mayring (2021) recommends only coding 10% to 50% of the material to develop the inductive category scheme for coding the entire text corpus (for definitions see Box 4.1). Accordingly, we must ask what the intercoder reliability *measure* of 75% means when the category system and, therefore, the coding scheme only represents 10–50% of the text? The explanation for qualitative

content analysis is thus: A *measure* has no meaning by itself without analysis and interpretation, that is, reliability does not contribute to the quality of the qualitative research process. For quantitative content analysis we recommend using semi- or fully-automated methods as explained in Chapters 9 and 15–18.

To be clear, just like for reliability, we agree with other qualitative researchers (e.g., Kuckarzt & Rädiker, 2022; Schreier, 2012) that the quantitative quality convention of objectivity cannot be achieved in qualitative research – although objectivity already seems doubtful as a quantitative convention because "objectivity requires that your data, your findings, and your interpretations are the same, regardless of who does the research" (Schreier, 2012, p. 26).

So let us take a look at the third quantitative test convention of validity. According to Flick (2019, p. 483; see also Flick, 2018), in qualitative research, the quality convention "validation through communication" focuses on "transparency of processes" for achieving results. Communicative validation – for example, in an interpretation group – is therefore an integral part of the entire qualitative research process, from the research design to material selection and data collection instruments to the presentation of results.

Validity is a very important criterion for assessing the quality of qualitative research. It is used in two senses: in a narrow sense, referring to the extent to which your instruments help you capture what you set out to capture, and in a broader sense, referring to the overall quality of your study (Schreier, 2012, p. 27).

However, the assessment of the quality of instruments and a specific study cannot be done without considering the key question: What is *good quality* social research? First, good quality is not a fact, but good quality is based on the evaluation of the quality of social research. The sociology of (e)valuation emphasizes that the judgement and justification of good quality is a threefold process: valuation, evaluation, and valorization (e.g., Lamont, 2012; Vatin, 2013). Valuation addresses something that we value, for example, because of our socialization, a belief system, or a value defined by standards and criteria. According to Boltanski and Thévenot (1991/2006), valuation practices express the worth we attribute to interaction, a relationship, an artifact, etc. Such values influence our evaluation of a situation, a product, or a research process.

Accordingly, and despite the analytical distinction between valuation and evaluation, in everyday life and research situations, it is hard to distinguish between valuation and evaluation, which is expressed by the term (e)valuation. (E)valuation is connected to valorization in a circular manner. On the one hand, by doing an (e)valuation, we valorize, that is, attribute worth to a thing, a person, or a research process. On the other hand, (e)valuation is an expression of valorization because of the value expression of evaluation criteria, standards, or beliefs. For example, if you love big cars or a specific luxury brand of artifact, you can spend hours reading about the latest releases of the artifact, get excited just by looking at the aesthetics, forms, colors, etc. and talk about it to likeminded people. If you are not into (big) cars or (expensive) brands, you may (e)valuate them according to their contribution to polluting the planet, working

conditions and workers' pay, use of sustainable materials, prestige compensation for deficient human beings, etc.

Now, let us return to the question of what is *good quality* social research? Well, to be straightforward, there is no single or standard answer to this question because it depends on the following:

1. The particular research process. No research process is like another because the social phenomenon under observation varies (slightly), differences in social, spatial, and temporal contexts exist, access to informants might be limited (e.g., interviews with prisoners or the health impaired), joint team instead of single investigator research, etc. For sure, methods for the study itself and content analysis stay the same, but the execution of the chosen method has to be adapted to the situation of the subjects of the research, of observation, etc., if only a little bit. Even in so-called standardized, survey research, it makes a difference whether a person can answer a question by themself online or in the presence of a researcher applying CAPI – computer assisted personal interviewing. Even when using nonreactive data (e.g., legal documents and social media posts), a research process is affected by the type of documents, quality of information, or the necessity to convert analogous, printed data into digital documents.
2. How a method can be applied by a researcher. In Chapter 2, we explained that only a deductive, inductive, or mixed deductive-inductive approach can be strictly distinguished in a research design. Both qualitative and quantitative social research are just top categories for many different methods. In addition, methods such as qualitative and quantitative content analysis can be combined in mixed methods analysis. The application of methods, that is, the execution of a methodology, depends on the amount of available data, case selection and sampling strategies, and the research question(s) and/or hypotheses formulated by researchers. Accordingly, how a method can be applied depends on researchers' formulation of the research question and hypotheses, whether a full, purposive, or random sample of documents (e.g., texts, pictures, or videos) is analyzed and interpreted, whether researchers have to adapt the research design due to unexpected social circumstances or new research opportunities (e.g., a document source is no longer available or a new Python package is developed, or existing ones modified), etc.
3. How data analysis and interpretation is supported by theory and/or previous findings about the same or similar social phenomena. In general, even statistical significance in quantitative research has no meaning by itself but needs interpretation, that is, meaning attribution by researchers, who use their knowledge about the dataset and social phenomenon under study, previous findings, and theory to analyze and interpret significance (e.g., to avoid data artifacts and false positives; Chakraborty et al., 2021). In deductive, theory-led research as well as inductive, theory-reflecting research, the theory selected by the researcher provides

a particular perspective for data analysis and interpretation. For example, habitus-field theory focuses on the links between objectified relations that exert forces on individuals and their practice that are comparable to physical fields, which in turn impact subjects' habitus and their cultural, social, economic, and symbolic capital via a relational embedding in the respective fields – and more general, the social (Bourdieu, 1977, 1996a, 1996b, 1998; Bourdieu et al., 1994). Tilly (1998/1999) theorizes durable, categorical inequality by identifying social mechanisms of exploitation, opportunity hoarding, adaptation, and emulation that support the organized establishment, maintenance, and modification of inequalities in societies. Self-determination theory helps analyze individuals' intrinsic, identified, introjected, and external (e.g., social and material) motivation and amotivation (Ryan & Deci, 1953/2017).

So what is the answer to the question: what is *good quality* research? The answer is that *good quality* research is always a judgment and an attribution of worth by others (e.g., peers and faculty). Such valorization by others should be guided by quality conventions and based on the (e)valuation of the research process and findings. Quality conventions are institutionalized social orders for coordination and legitimation in a social group (e.g., an academic community) that oppose "personal conveniences" (Boltanski & Thévenot, 1991/2006, p. 146) and collaboration networks as well as co-citation silos (Tight, 2014; Wieczorek et al., 2024).

4.3 Three conventions for (e)valuating quality research

You see that the (e)valuation of research is a tricky issue, which you should be aware of. However, neither the (e)valuation nor the valorization of your research must be your concern when designing and doing your research. You should be concerned with the quality of your research, which means considering quality conventions. In this chapter, we will introduce the three quality conventions discovery, appropriateness-transparency, and iteration and distancing as a practical guide to doing research. The quality conventions support you in planning the entire process of research, provide guidance when encountering challenges while doing research, and are points of reference for justifying your research in case of critique.

The basic definitions of these conventions were developed in a deductive qualitative content analysis of qualitative and quantitative quality conventions (Schneijderberg, 2023). Building on these definitions, our elaboration of the three conventions takes another, general approach to the research process, which is a very different approach than that taken by many quality suggestions for specific research elements and quality-check catalogs (e.g., Eisewicht & Grenz, 2018; Flick, 2018; Howe & Eisenhart, 1990; Hirose & Creswell, 2023; Krebs & Menold, 2019; Lincoln & Guba, 1985; Rocco, 2010; Sandelowski & Barroso, 2002; Steinke, 2004; Strübing et al., 2018; Swanborn, 1996).

4.3.1 The discovery convention: six discovery contexts in a research process

Research is supposed to be about making a discovery and contributing *something* to already existing knowledge. Researchers are expected "to make an 'original' or 'significant contribution' to knowledge" (Lovitts, 2006, p. 163), although what an original or significant contribution to knowledge growth means is rather opaque.[8] Guetzkow et al. (2004, p. 196) define discovery in research vaguely as being about something "new" or "novel", being "innovative" in approaching a social phenomenon or tackling a theoretical problem, a "creative" approach to solving a methodological problem, or thinking in a way that leaves the beaten path, or just as "doing something others have not".

Of course, the adjectives and verbs, and the overall expectation to make a contribution to knowledge express expectations about graduate students' dissertation research and PhD holders. Discovery expectations for a master student's thesis would be already qualified by a 'small' contribution, and for bachelor students the 'new' would focus on the discovery of how to conduct a research process in a systematic, methodological way, and discovery in data.

In addition to the adequate level of discovery expectation, you should be aware that a scientific contribution can be made in various parts or *just one* part of your research:

> question, argument, theory, data, data sources, method, analysis, results – by thinking, saying, or doing something that has not been thought, said, or done before. The novelty can originate wholly or in part from the student, or the student may borrow an existing contribution from another field or discipline and apply it to his or her field or discipline for the first time.
>
> *(Lovitts, 2006, p. 169)*

Accordingly, any research process is potentially full of discoveries, yes, in plural, because the overall findings, most likely, are based on a series of smaller discoveries at different stages of the research process.

Please do not be overwhelmed by the following lists and tables. The different stages of the research process define the sequence in which you proceed in a systematic way during your research. Content analysis is a method that emphasizes a certain logic of how to advance through data analysis and interpretation. Accordingly, this methodo*logical* approach helps you to focus on one stage of your work and a potential discovery. Some of the information presented here does not apply to your research, so please read selectively depending on whether you conduct qualitative or quantitative research, follow a deductive or inductive approach, and have collected new, primary data or are using existing data for a secondary analysis.

In Table 4.1, the different stages of the research process are defined as potential contexts of discovery. We selected the term 'context of discovery' to

emphasize that the different stages of the research process offer distinct opportunities for discovery.[9] An opportunity for discovery may occur in your study in various or just one part of your research process and, sometimes, not at all (e.g., problems with data collection or latent topics cannot be interpreted). The six contexts of discovery in a research process are as follows:

1. Start of the research process: A discovery can be an empirical observation, identification of a theoretical problem or gap in the literature, which should be explained to justify the research question(s) or hypotheses. Of course, this is all embedded in the basic research design.
2. Data:
    a) Data collection for qualitative primary research, which involves a researcher or group of researchers.
    b) Data selection for secondary research, that is, analyzing the fit of existing data in a repository or scientific use file to answer research question(s) and/or test hypotheses.
3. Data processing and/or cleaning, for example during inductive development of categories and coding.
4. Identification and formulation of descriptive results: To answer the research question(s) and/or test hypotheses, based on your analysis in discovery context 3, you have to select results (e.g., relevant, counterfactuals, and significant) for presentation in a term paper, bachelor or master thesis, manuscript for publication, etc. Depending on whether you conduct a qualitative or quantitative study, your presentation of your descriptive analysis should be supported by exemplary quotes (*Ankerbeispiele*), tables, figures, and/or significant results.
5. Interpretation and discussion of results: Opportunities for discovery are rooted in the
    a) presentation of your understanding and explanation of the findings in context, for example of a single case and all cases or a sample,
    b) either theory-led or reflexive use of theory to support interpretations and inferences, and
    c) comparison of your results with findings from similar studies.
6. Conclusions: To answer your research question(s), you should present an overall summary of major findings from your research and, if applicable, attribute meaning to social implications.

In comparison to the six contexts of discovery in a research process, which would apply to qualitative and/or quantitative content analysis, the contexts of discovery are more limited for a quantitative survey study collecting primary data. The limitation originates from the absence of researchers in the data collection process, which excludes the second context of discovery. To specify the

third context of discovery 'data processing and/or cleaning' for a quantitative survey study, we would need to add objectivity, reliability, validity, and other tests to this category (see also Section 4.2.1). The missing context of discovery in deductive survey research is even more curious considering the fact that statisticians have to bring together objectivity, reliability, validity and other significance tests, inductive inference, and deductive hypothesis testing in 'data processing and cleaning' (context of discovery 3) for generating 'descriptive results' (context of discovery 4).

In Table 4.1 and the other tables in Chapter 4, we distinguish between the two substantive epistemologies for empirical social research: The deductive and inductive approaches to research, which can be mixed in a deductive-inductive research design (see Section 2.2). In social sciences, deduction means drawing inferences based on premises or assumptions, that is, which are theory-led. In a deductive research process, theory-led, abstract principles are used to study a phenomenon, which is also referred to as an empirical test of theory. Inductive research avoids a specific theory lens, focuses on the empirical study of a phenomenon, and applies theory reflexively. Accordingly, in social research, induction aims to draw conclusions from a specific phenomenon to understand more general, abstract patterns. Please do not get confused: in mathematics, induction is a term used for deductive reasoning (based on correct premises, observations leading to general principles).

The two substantive deductive and inductive epistemologies for empirical research are not compatible with all theoretical epistemological approaches. For example, the behavioral theories of positivism and critical rationalism as well as the psychological theory of individuals' rational choice can only be used in a deductive research design (e.g., Elster, 1989/2006; Popper, 1972). Social constructivism (e.g., Bourdieu's (1977) habitus-field theory), functionalism, pragmatism, and phenomenology (also known as ethnomethodology) are suitable for both deductive and inductive research approaches. Grounded theory is a solely inductive research approach (e.g., Glaser & Strauss, 2017; Strauss, 1987). As we cannot explain all these theoretical epistemologies in detail in this methods handbook on content analysis, please consult a theory handbook.

In Table 4.1, we use brief and factual formulations. However, some formulations, such as 'fit' in the column 'secondary data analysis', indicate the need for researchers to justify appropriateness and make choices about their research design transparent. For the purpose of analytical distinction, we do not mix the detailed definitions of the three quality conventions of discovery (Table 4.1), appropriateness-transparency (Tables 4.4–4.6, see Section 4.3.2), and iteration and distancing (Table 4.7, see Section 4.3.3). Nevertheless, you should be very aware that the six contexts of discovery are connected to either one or both of the other quality conventions in a research process (Table 4.2), for example, that discovery requires appropriateness and transparency (e.g., about data collection), and emerges from iterations and distancing (e.g., during data analysis

Table 4.1 Contexts of Discovery in an Empirical Research Process of Qualitative, Quantitative, and Automated Content Analysis

	Context of d. 1	*Context of Discovery 2*		*Context of d. 3*	*Context of d. 4*	*Context of d. 5*	*Context of d. 6*
	Start of Research Process	*Data Collection (Primary)*	*Secondary Data Analysis*	*Data Processing*	*Descriptive Results*	*Interpretation & Discussion of Results*	*Conclusions*
Qualitative, inductive, primary study	Empirical observation, theoretical problem(s), follow-up research, gap(s) in literature, etc. expressed by research question(s) and/or hypotheses, and formulated in research design.	Data collection process (e.g., ethnography)	-	Development of categories & coding	Answering research question(s) and/or (non)confirmation of hypotheses. The presentation of the descriptive analysis is supported by exemplary quotes (*Ankerbeispiele*), tables, figures, significance tests, etc. (whatever applies)	Understanding and explaining in context: single case and/or all cases. Reflexive use of theory to support interpretation, comparison with findings from similar studies	Summary of findings focused on the research question, attribution of meaning to social implications, and, possibly, indications for future research
Qualitative, inductive, secondary study		-	Fit of data for answering research question(s) and/or testing hypotheses				

(*Continued*)

Table 4.1 (Continued)

	Context of d. 1	*Context of Discovery 2*		*Context of d. 3*	*Context of d. 4*	*Context of d. 5*	*Context of d. 6*
	Start of Research Process	*Data Collection (Primary)*	*Secondary Data Analysis*	*Data Processing*	*Descriptive Results*	*Interpretation & Discussion of Results*	*Conclusions*
Qualitative, deductive, primary study		Data collection process (e.g., videos)	-	Coding according to categories (noncoding, i.e., omission of data without category)		Understanding and explain in context: single case and/or all cases. Theory-led interpretation, comparison with findings from similar studies	
Qualitative, deductive, secondary study			Fit of data for answering research question(s) and/or testing hypotheses				
Qualitative, deductive & inductive, primary study		Data collection process (e.g., focus groups)	-	Coding according to categories & development of additional categories for coding			
Qualitative, deductive & inductive, secondary study		-	Fit of data for answering research question(s) and/or testing hypotheses				
Quantitative, text data, inductive, primary study		Data collection process (e.g., interviews)	-	Key words or sentiments search, automated coding, identification of themes, topic modeling (incl. perplexity test), etc.		Reflexive use of theory to support interpretation, comparison with findings from similar studies	
Quantitative, text data, inductive, secondary study		-	Fit of data for answering research question(s) and/or testing hypotheses				

and interpretation). We will come back to the interconnectedness of the three quality conventions in the following sections.

The six contexts of discovery in a research process are aligned with the presentation of your research in a term paper, bachelor and master thesis, or manuscript for publication. In Table 4.3, schematic headings indicate where you are expected to document your research and present your findings. The first context of discovery, the start of research, includes more than just the presentation of your research question and/or hypotheses. In your research design, the research question and/or hypotheses need to be motivated, that is, explained with reference to the social phenomenon, problem to be solved, etc. and situated in the relevant literature. By reviewing the relevant literature, you present your awareness of similar research, you show how your planned research was conducted in another spatial context (e.g., another country), temporal context (e.g., ten years ago), or social context (e.g., another group), and which methods and theories might have been applied for the study. If you state a discovery in the first context, you will have to justify and substantiate your claim – nobody will believe you made a discovery just because you say so! The relatedness of a discovery to other research also applies for the data and methods section, discussion of results, and conclusions. The sections 'descriptive results' and 'interpretation' focus on your research, only.

Finally, considering the core idea of the discovery convention, we would like to emphasize that writing up a research process means reporting, justifying, and making others understand what your research findings are – novel or not: all findings are discoveries. Please be aware that nothing is obvious or self-explanatory, not even statistical significance. Accordingly, it is your job as a researcher to present your discoveries to an audience that has not read the same literature you have, that did not collect and analyze your data, that did not think your thoughts when theoretically reflecting and interpreting data, constructing categories and/or inferring results from data. Therefore, everything you neglect to explain and fail to make a reader understand provokes alternative interpretations or even raises doubts about your research.

Table 4.2 Schematic Connectedness of the Six Contexts of Discovery to the Appropriateness-Transparency and Iteration and Distancing Conventions

Contexts of discovery		*Appropriateness-transparency convention*	*Iteration and distancing convention*
1	Start of research process (i.e., research design)	X	X
2	Data collection (primary study) or Secondary data analysis	X	X
3	Data processing	X	X
4	Descriptive results	X	X
5	Interpretation & discussion of results	X	X
6	Conclusions	X	X

Table 4.3 Suggestions for Presenting the Research Process and Possible Discoveries According to Schematic Headings (e.g., in term paper, bachelor and master thesis, and manuscript for publication)

Contexts of discovery		*Heading*
1	Start of research process (i.e., research design)	Introduction Literature review Deductive research: theoretical approach
2	Data collection (primary study) Secondary data analysis	Data and method(s), possibly including limitations
3	Data processing	Case and/or sample descriptions Construction of instruments (e.g., interview manual, survey questions and items)
4	Descriptive results	Descriptive results
5	Interpretation & discussion of results	Interpretation & discussion of results [inductive approach: theory]
6	Conclusions	Conclusions

And doubts are legitimate because a research process, in most cases, is not a clinical or straightforward endeavor but demands many decisions from researchers and, often, compromises as well because, like the research process, data collection is also a cognitive and social process. Such decisions and compromises ought to be appropriate (e.g., methodologically and socially justified) and transparent as they affect all research processes and, therefore, are highly relevant when considering the good quality of research and empirically grounded discoveries of qualitative and quantitative content analysis.

4.3.2 The appropriateness-transparency convention

In comparison to the discovery convention, the interdependence of cognitive and social elements of the research process is more present in the appropriateness-transparency convention. The hyphenation emphasizes that appropriateness and transparency go hand in hand in a research process. To recall: Appropriateness addresses the fit of the research design, including the methodological and/or theoretical approach to studying a social phenomenon, of empirical observation and results, that is, data analysis and interpretation. Transparency emphasizes that other researchers must be able to understand, reenact (in their mind), and reproduce your research process, including possible limitations, and results, although their data interpretations might differ.

The application of appropriateness and transparency has to level several real-world obstacles. Unlike statistics, in inductive quantitative research and qualitative research, in general, transparency has to solve several problems,

for example, limited access to a social group, intellectual property rights applying to documents, and anonymity and data protection of informants.[10] Here, two meanings of data protection have to be considered: Data protection to avoid revealing the identity of a subject giving information in verbal, pictographic, video, and text documented form. For the appropriateness-transparency convention, the second meaning, the protection of raw data, is of similar relevance to assuring anonymity of subjects and groups, for example, when studying any kind of abuse, rare diseases, minority groups, political refugees, and internal, that is, not publicly available documents, which only a researcher or a small research team can study.

In Table 4.4, the connection between appropriateness and transparency is indicated with the formulation of questions *WHY* a method, theory, case selection, analysis and interpretation, conclusion, etc. are appropriate (i.e., 'why did you do it?') and *HOW* you applied a method, theory, etc. and dealt with ethical issues (see Box 4.2), limitations, etc. throughout your research process (i.e., 'how did you deal with it?'). The appropriateness-transparency convention applies, first, for your research design – assuming that you have discovered a gap in the literature, identified a

Box 4.2 Information and results vs. ethics: a conundrum?

Although you will be weighing the pros and cons of using critical information and results, please be 100% assured: This is not a conundrum! Even in cases of minor doubt, always rate ethical considerations and data protection higher than reporting the information in your results. Even if you cannot ground the most striking insight in your data, which results in a finding being less meaningful – it is not worth it. Nowadays, social media and other platforms provide so much information that a skilled data scientist can connect even small dots with little effort. Such a scandal would harm your personal credibility as well as the credibility of empirical social research, which will lower the willingness of people to participate in studies.

In general, ethical considerations are present from the beginning to the end of a research process (e.g., Miles & Huberman, 1994, pp. 288–296). For example, prior to recruitment of interviewees, you need to provide them with information about the study and get their signed informed consent. During the interview, you depend on their collaboration and, for example, need to respect cultural sensitivities of their environment, signals for no-go areas for follow-up questions, etc. Once your fieldwork and data analysis are done, upon reporting your findings (e.g., in a term paper, your bachelor or master thesis, or a manuscript intended for publication), it is your responsibility to repay the informants confidentiality, that is, do not disclose their identity, locations, etc., and analyze and interpret their information with all fairness.

theoretical problem, made an empirical observation worth studying, or your higher education instructor gave you an assignment. Please be aware that selecting data for secondary analysis is a part of research design – even though you might still need to 'harvest' it (e.g., copy it online, download it in the form of documents, or access it via an API – the application programming interface) – because the feasibility of your study does depend on it.

4.3.2.1 Appropriateness-transparency convention: research design

Unlike the second context of discovery (data collection, primary or secondary study), all other contexts of discovery correspond to distinct stages of the research process, which are identified in the categories in Table 4.4 (research design), Table 4.5 (data collection and processing), and Table 4.6 (results, interpretation, and conclusions). In the column 'Elements of ...', we focused on major issues in any research process. Please be aware that some issues only concern inductive or deductive approaches, or qualitative or quantitative methods. As we do not know what kind of research you plan to do, please select the relevant 'elements' and associated 'appropriateness and transparency' requirements from Tables 4.4, 4.5, and 4.6 for planning your research process and arrange the 'elements' and 'appropriateness and transparency' requirements in a single table.

For constructing your research process in a table, please consider the contexts of discovery (Table 4.1) and schematic headings for a manuscript (Table 4.3), no matter whether the manuscript is a term paper, bachelor, master or PhD thesis, or a scientific publication. Writing a research design is not additional or optional work and, to a considerable extent, the research design is the nucleus for several sections of your manuscript: 1 introduction (including research question(s)), 2 literature review, 3 theoretical approach (in the case of deductive research), and 4 data and method(s) including case and/or sample descriptions, construction of instruments (e.g., interview manual, and survey questions and items). Appropriate and transparent conduct in part 3 of the research process, that is, data analysis and interpretation, and drawing conclusions is not part of the research design – because you do not have data and, most likely, no glass ball for scrying the research process from data processing onwards – and will be discussed in Section 4.3.3.

Now, let us walk you through the application of the appropriateness-transparency convention for the three parts of the research process. Table 4.4 highlights key elements of research design. In Table 4.4, the 'appropriateness and transparency' column requires you to state the 'why are you doing it?' in a particular way and to explain how you plan to conduct or did conduct your research. For example, the appropriateness *WHY* question is not concerned with the requirement that you have a research question – having a research question is a cognitive-social convention in research. Appropriateness addresses *WHY* a

specific research question is asked, and *HOW* does this research question capture the purpose of the research, *HOW* is it related to the observation of a new social phenomenon, etc. The same applies to the formulation of hypotheses.

For the literature review, justifying the literature studied is a challenge in two very different ways. First, as a student or early career researcher, you do not have time to read every paper and book on a method, topic, theory, or whatever is relevant for your study. Accordingly, you have to make a compromise and select your readings. The second challenge is associated with the fundamental goals of a literature review. These are as follows:

1. To familiarize yourself with a method, theory, topic, and context relevant for your study.
2. To learn how other researchers studied (specific other aspects of) your phenomenon or a similar phenomenon, what challenges they encountered and solutions they found, etc.
3. To build on and embed your research in the relevant, read literature.
4. To define the purpose of your research (i.e., research design = context of discovery 1; Table 4.1), for example study a phenomenon in another social, spatial, and/or temporal context, address a research gap identified in the literature, to systematically grasp an observation, replicate another study (e.g., to aid comprehension of findings), to apply another method promising additional insights into a phenomenon, and test a theory.

The literature review is a good example for the horse and carriage connectedness of appropriateness and transparency in one quality convention. Transparency in the case of a literature review means that you have gained in-depth understanding of your readings and (a) can make your understanding comprehensible to others and, most importantly, (b) that what you read enables you to appropriately apply it in your research process. The time-consuming effort of reading literature and understanding the content you read are a prerequisite for the quality of your research and so that others may (e)valuate your research as *good quality* research. Moreover, especially as a novice to empirical research, serious efforts in reading and understanding the literature you read will reduce your insecurity about what you are doing.

Having clarified the aim of a literature review, we return to the first challenge of making a compromise by selecting literature. To not compromise your literature review, the literature selection could be based on others' expertise. For example, in a course, a lecturer will provide (introductory) literature based on their n-yearlong research.[11] You can use the bibliography of a relevant article or book to construct your reading list or conduct a systematic literature review yourself, as described in Chapter 9. Another source for selecting literature is

using published literature reviews to guide your reading. Please do not think that you can take shortcuts based on what others have summarized: If you do not read and understand the literature yourself, you can, at best, only superficially apply a method, a theory, or approach subjects, and you will make mistakes during research.

If you are unwilling to do the reading yourself, the risk of making mistakes is even higher if you rely on a generative pretrained bot, which means that you are voluntarily exposing yourself to artificial intelligence (AI) hallucinations (see Chapter 10). In general, we do not object to AI use for supporting your research. However, you are doing the research (and get a grade or publication) and not the bot. As with a literature review published by others, you yourself have to read the literature selected and summarized by the bot. Otherwise, you cannot be sure about the fit between your actual research topic and the bot-selected literature or that the bot wrote an appropriate summary and conveyed a proper understanding of the content of reviewed publications. As the literature review is the basis for applying the method(s) and theory, understanding the phenomenon under study, data collection and a theory-led or theory-reflexive discussion of results, not doing the reading yourself will compromise your entire research process.

Based on the reviewed literature and considering your research question, you can justify your design choice for a deductive, inductive, or mixed deductive-inductive approach to your phenomenon of study. In addition to the core justifications presented in Table 4.4, for more details on the application of the appropriateness-transparency convention in deductive, theory-led research approaches, please see Chapter 7 (qualitative content analysis) and Chapter 9 (quantitative content analysis). Inductive research approaches are presented in Chapter 6 (qualitative content analysis), Chapter 8 (semi-automated quantitative content analysis), and for fully automated quantitative content analysis see Chapter 15 (multiple correspondence analysis and semantic network analysis), Chapter 16 (sentiment analysis), and generative topic modeling as well as topic modeling with different embedding-layers in Chapters 17 and 18. Each of the chapters also justifies case(s) selection and/or applied sampling strategies for data collection. The latter is further elaborated for automated inductive quantitative content analysis in Chapter 11. Again, as for the literature review, it is imperative that you take your time to read about the basics of a method and understand the application of a method to know how to apply it for research.

Prior to reading one of the said chapters, to inform your selection of the appropriate content analysis method, please consult introductory Chapter 1. In Section 1.2.3, we provide cues so you can make the best choice for a content analysis method depending on the research question and amount of data (see Figure 1.1). The elaboration of the crucial 'element' of theory (deductive

Table 4.4 Application of the Appropriateness-Transparency Convention: Research Design

Research process	*Elements of research design (Please select what applies to your research)*	*Appropriateness – why are you doing it? – and transparency – how do you plan to do it? (Please select what applies to your research)*
Research design (start of research process)	Research question(s)	Justification of research purpose and explanations with reference to observation of a social phenomenon, literature, theory, etc.
	Motivation of hypotheses	Deduction from theory, similar studies, and/or observations.
	Literature review	Expert, quantitative, or AI selection of publications considered relevant for the phenomenon, theory, and/or method(s) of study.
	Deductive, theory-led study	Theoretical fit to answer research question(s), grasp social phenomenon, etc. (see Chapter 7). Justification of theory test or look at data according to a particular theoretical lens. Category system (see Box 4.1) deduced from theory, which might have been used in previous studies, including possible alterations.
	Inductive, data-driven study	The method is *per se* inductive (e.g., multiple correspondence analysis and topic modeling). Discovery of a new phenomenon (see Chapter 6) or taking a fresh look at a phenomenon which has been studied before, for example in another social, spatial, and/or temporal context (see Section 5.3.1). Appropriateness of theory or theories for reflexive understanding of empirical results.
	Mixed deductive-inductive study	Justification of combined theoretical and empirical approach. Category system deduced from theory amended by top and/or subcategories developed during data coding.
	Method(s)	Fit to answer research question(s), to approach data sources (e.g., Internet and interviewees), and/or type(s) of data for qualitative and/or quantitative (content) analysis.
	Case selection and/or sampling strategy for data collection, primary study	Methodological, theoretical, or empirical case(s) selection strategy (e.g., snowball selection of interviewees, comparison of social media samples, and full document corpus), possible ethical conundrums (see Box 4.2).
	Data selection, secondary study	Fit of data, data type(s), and description(s), access (e.g., API), and dealing with limitations (e.g., process-generated social media posts, literature corpus, and anonymized interviews from repository).

approach) will be presented in Section 4.3.2.3, addressing the role of theory in the research process.

4.3.2.2 Appropriateness-transparency convention in part 2 of the research process: data collection and processing

In your research design, you are required to make transparent and justify why something that you plan to do in your empirical research process is appropriate. For example, in part 1 of the research process, you justified your selection of a method and a sampling strategy or selection of cases. From part 2 of the research process onwards, you will see how well your planning works and whether you need to make modifications during data collection (e.g., unexpected recruitment problems or limited access to documents).

Accordingly, in Table 4.5, transparency requires you to respond to the question 'how did you do it?' and appropriateness to the question 'why did you do it?' In a best-case scenario, everything you planned works out really well. Nevertheless, other researchers (and your instructor) are interested to learn how you executed your planning and why things worked in the specific situation of data collection. In case things do not (all) work as planned, you are required to explain how you modified your plan, for example (slightly) deviated from interviewee characteristics for reaching the set number of interviews or selection of text data from a similar social media platform, and why the modifications did not jeopardize your research. If you cannot argue why the modifications were appropriate, the research needs to start over by rewriting your research proposal. This may be a setback, but it is quite common in empirical research.

As we argue throughout this methods handbook, content analysis always depends on the qualities of data (in particular, see Chapter 3). The qualities of the data refer to the type and characteristics of the data. For example, we can distinguish between data that is available as text, as a recorded video of an expert panel discussion on television, as image material, as a data table, or as (structured or unstructured) data types. Each type of data is associated with its own methods and approaches to data collection and processing. In comparison to the standardization of survey responses by (variable) scale numbers, all other text data used in social research can and must first be converted from a raw form into another type of data that is suitable for qualitative, semi-automated quantitative, and fully-automated quantitative content analysis. For the analysis, voice recordings (e.g., interviews or singing) and videos are transcribed or summarized as text. Text can also be translated into numbers and thus removed from the raw form even further than in the case of transcripts or summaries (see Chapters 11–15). Accordingly, in Table 4.5, we distinguish between reactive (e.g., interviews) and nonreactive forms (e.g., observations) of data collection.

In content analysis, we often work with non-reactively collected, that is, selected data that was process produced. While research-produced data is

collected or produced with the aim of answering one or more research questions, process-produced data can be drawn from archives (e.g., parliamentary debates and historical documents) or the Internet (e.g., social media posts and PDFs on websites). The quality of process-produced data is not defined by the researcher but, for example, by the legal requirements on files (e.g., for administrative records or lawsuits) or communication formats of social media platforms (e.g., characters or word limits and available emojis). For more information about data collection for analysis with process-generated data see Section 3.2.3.

In comparison to research-produced data, which is produced according to the aim of the inquiry and tailored to the phenomenon under study, using process-produced data means conducting a secondary study. The term secondary study expresses that data was not collected for your research process – if you collect data, this is a primary study. A secondary study is also the analysis of data that has already been collected and analyzed by other researchers, for example made available as a scientific use file in a data repository. In comparison to process-produced data, when doing a secondary study with research-produced data, you can rely on documented decisions made during observations, interviews, etc. in a research process.

For more details on the appropriateness and transparency of data corpus construction, processing, and management, see Chapter 9 for deductive-quantitative content analysis (bibliometric literature review) and Chapter 11 for automated inductive-quantitative content analysis. Deductive category system construction will be elaborated in Section 4.3.3 and for inductive category system construction see Section 5.3.

The final line in Table 4.5 addresses the appendix to a paper, which is standard in quantitative research. The appendix to a paper is relevant for different reasons. Any paper, no matter whether a term paper, bachelor thesis, or journal article has character or word limitations. Such focusing of content presentation requires that large tables, which support empirical insights, etc. have to be presented in the appendix of the paper. For qualitative research, the category system, for example, including definitions and exemplary quotes for every top and subcategory should be presented in the appendix, if it is too large for the paper. If your analysis is supported by large language models embedded in generative artificial intelligence (AI) interfaces, for transparency, prompt engineering and exemplary quotes of AI responses need to be presented in the appendix of a paper. As we discuss in Chapter 10, a generative AI, for example ChatGPT, never produces the same answer to the same prompt. However, we should assume that AI produces similar answers or responses that point in the same direction. Accordingly, in the appendix, the range of responses the AI generated as well as the researcher's justification for the analysis selected support and contribute to the interpretation of empirical results. Similar to AI support in qualitative research, unsupervised topic modeling always produces at least slightly different results (see Chapter 17).

Table 4.5 Application of the Appropriateness-Transparency Convention: Data Collection and Processing

Research process	*Elements of data collection and processing (Please select what applies to your research)*	*Appropriateness – why did you do it? – and transparency – how did you do it? (Please select what applies to your research)*
Data collection, primary study (context of discovery 2; Table 4.1)	Observation (e.g., ethnography and participant observation)	Field notes, follow-up reflections, etc.
	Reactive format of data collection (e.g., interview, survey, and video)	Documentation of situation, first insights. Limitations and follow-up notes (e.g., in research diary)
	Process-produced data (e.g., files, pictures, posts, and videos)	Transcription (e.g., manually or using AI), organization of documents, etc.
	Data management	Store in secure place, anonymization, limited access, etc.
Data selection, secondary study (context of discovery 2; Table 4.1)	Data corpus	Construction of document corpus or quantitative dataset, modification of transcripts (e.g., delete sections of interviews or focus groups that are not relevant to answer research questions), etc.
	Data management	Store in secure place, anonymization, limited access, etc.
Data processing and/or cleaning (context of discovery 3; Table 4.1)	Amount of data	Sufficiency to answer research question(s) and possible limitations
	Software use	Type(s) of software, specific features, etc.
	Qualitative and quantitative inductive content analysis, primary and secondary study	Manual, semi-, or fully automated construction of categories and coding
	Qualitative deductive content analysis, primary and secondary study	Coding (see Chapter 8)
	Quantitative inductive and automated content analysis, primary and secondary study	Pre-processing (see Chapter 11) Workflow Algorithms and scripts/code
	Appendix	Category system (detailed top and subcategories, including definitions and exemplary quotes – if not included in main text) Additional and supporting data analyses In case of AI-supported analysis (see Chapter 10), prompts and exemplary quotes (*Ankerbeispiele*)

4.3.2.3 Appropriateness-transparency convention: results, interpretations, and conclusions in theory-led and theory-reflexive approaches

In Tables 4.4 and 4.5 we always made a distinction between the deductive, theory-led approach and the inductive, theory-reflexive approach. We chose not to explain the application of theory in the respective Sections 4.3.2.1 and 4.3.2.2 because of the salient role of theory throughout the research process. We figured that it will be easier for you to understand the role of theory by presenting it in a coherent manner instead of asking you to connect the uses of theory for the three parts of a research process of research design, data collection and processing, and analysis and interpretation.

In Table 4.6, almost all elements of data analysis and interpretation are explicitly related to theory. Moreover, the use of theory in the empirical research process is funneled from the research design via data collection and processing to the analysis and interpretation of empirical results. According to our experience, novices to empirical research struggle the most with theory use in part 3 of the research process. On the one hand, interpretation as your act of meaning attribution to the empirical findings is most challenging, as we explain in detail in Chapter 5. In Chapter 7, we present an example interpretation for a deductive qualitative content analysis. We will return to discussing interpretation in the following Section 4.3.3 on the iteration and distancing convention.

On the other hand, analysis as an activity means that you need to present your understanding of what you see in data, draw inferences, explain results, etc. In the remainder of Section 4.3.2.3, we will focus on this challenge by distinguishing between deductive theory-led and inductive theory-reflexive approaches to empirical research.

Before we address the role of theory in the research process, we need to define what theory is.[12] According to Stinchcombe (1968/1987, p. 5), "the reason for having theories of social phenomena is to explain the pattern in observations of the world." In other words: For the analysis of phenomenon x, an appropriate theory would specify features of x and patterns of subjects' social behavior or object uses of x. However, a theory about x does not express a social fact but an assumption that may be very broad (e.g., a philosophical idea and general proposition) or theorize a very specific empirical description. Such an assumption about the y of x can capture "the empirical consequences of theories" (Stinchcombe, 1987, p. 50) based on observations, or observation may be the beginning of theorizing (Alexander, 1982; Swedberg, 2016). For example, the latter theorizing is very close to observed social reality. Such theorizing means attributing an abstract description to a feature or features of everyday behavior or assuming the meaning of an empirical observation for study.

In this context, theorizing may not be intended to develop a theory but to grasp observations by issuing definitions, classifications, correlations, or conceptual ideas for analyzing and interpreting patterns about the y (i.e., assumptions) of x (i.e., phenomenon) for empirical study. You see that a theory can be

Table 4.6 Application of the Appropriateness-Transparency Convention: Results, Interpretations, and Conclusions

Research process	*Elements of data analysis and interpretation (Please select what applies to your research)*	*Appropriateness – why did you do it? – and transparency – how did you do it? (Please select what applies to your research)*
Descriptive results (context of discovery 4; Table 4.1)	Inductive: category-led analysis	Groundedness of results in data
	Deductive: category- and/or hypothesis-led analysis	Theory-led empirical data analysis Confirmation, partial confirmation, or rejection of hypotheses
Interpretation & discussion of results (context of discovery 5; Table 4.1)	Number of observations in data and their interpretation, including alternative interpretations and counterfactuals (more than one interpretation is always possible! See Chapter 5)	Decisions on which (most plausible) interpretations to follow and which (possible but implausible) interpretations to ignore
	Theory-led discussion of results (deductive)	Theoretical relatedness and robust theory-empirical link
	Theory-reflexive discussion of results (inductive)	Theoretical relatedness and contribution to broader understanding of phenomenon
	Comparison to findings and interpretations from similar studies	Commonalities and differences of findings, specificities, limitations, etc.
Conclusions (context of discovery 6; Table 4.1)	Answer(s) to research question(s)	Clear-cut answer(s) (positive and/or negative), possibly new definition(s), and/or concepts
	Social (incl. cultural, economic, political, etc.) implications	Specification of sector and context, transferability, practical meaning, etc.
	Future research	Limitations

anything ranging from an observed empirical correlation – either by you or in a published study – classifications via sector-specific theories (e.g., political theory addressing "appropriate forms of government"; Harrington, 2005, p. xxi) and thematic theories (e.g., social inequality; Tilly, 1998/1999) to theories helping to understand and explain social behavior on a very abstract level (e.g., habitus-field theory; Bourdieu, 1977).

In general, for your first empirical research project, we would recommend a deductive and not an inductive approach because an inductive, theory-reflective approach is more demanding as we will explain in a minute. For a first-time, deductive, theory-led approach to empirical research, we would recommend

using either a well-elaborated thematic or general social theory to provide guidance for your research. In addition, theories used in other empirical studies provide examples for how to apply theory in your research design.

For the use of theory in a research process, we suggest a basic distinction between the deductive and inductive approaches as follows:

1. The theory-led, deductive approach is characterized by a straightforward role of theory throughout the research process (Figure 4.1). Figuratively speaking, the phenomenon under study is only regarded through the lens of theory. Accordingly, in the deductive approach, theory is the focal point in the empirical research process.
2. The theory-reflexive, inductive approach is characterized by a two-stage role of theory in the research process: First, theory or theories need to be reflected for the research design and, second, theory reflection is essential for the discussion of empirical results and reaching conclusions (Figure 4.2). Figuratively speaking, the phenomenon under study is looked at from different theory perspectives. Accordingly, in the inductive approach, the phenomenon under empirical study is the focal point in the research process.

In the theory-led, deductive approach, the theory you chose should

1. be explicit in the formulation of the research question. The research question is the first explicit theory-phenomenon iteration, although the phenomenon under study may (or may not) have been observed first without looking through the chosen theoretical lens. Iteration means the repetition of an action or a process to construct a sequence of outcomes. In a deductive research process, the schematic sequence of outcomes is based on eight or nine theory-phenomenon iterations.
2. provide assumptions (e.g., about actors, activities, and structures) to 'grasp the phenomenon under study' (i.e., the second theory-phenomenon iteration). The theoretical elaboration of the phenomenon is an immanent part of any manuscript about a theory-led study (see Table 4.3). In Chapter 7, student engagement theory (Tinto, 1997) is used for a step-by-step explanation of how to do deductive qualitative content analysis.
3. be discussed in the literature review, which also has to address the phenomenon under study and similar or different social, spatial, and temporal contexts. In the literature review, you should advocate for the use of your chosen theory, ideally referencing similar studies using the same theory, or argue why other theories are less suitable for grasping the phenomenon of your study. The literature review is the third theory-phenomenon iteration.
4. inform the formulation of hypotheses, in case you choose to specify the research question(s). Hypotheses can be useful for focusing your research but are not a requirement for an empirical study – that is why hypotheses are indicated as optional in Figure 4.1 and the relevance of hypotheses for the following elements of the research process is indicated by dotted

arrows. The theory you chose does not have to be the only basis for formulating hypotheses. Findings from other studies or your observation of a phenomenon can be formulated, that is, theorized as assumptions (i.e., the possible y of phenomenon x) and expressed as hypotheses. The hypotheses are the fourth theory-phenomenon iteration.

5. lead the construction of the category system (see Box 4.1) by capturing analytical characteristics of the phenomenon under study (fifth theory-phenomenon iteration). A category is a unit to identify, classify, and sort actions, actors, artifacts, phenomena, etc. during content analysis and interpretation. A category system is composed of several top categories and associated subcategories. Subcategories are analytically distinct characteristics which help to grasp the top category.[13] Operationalization of categories/characteristics is necessary in survey research but not content analysis.
6. guide data processing (i.e., coding; sixth theory-phenomenon iteration), analysis, and interpretation (see Chapter 5), and their presentation as descriptive results (seventh theory-phenomenon iteration). The theory-based category system should organize your presentation of the descriptive results. If you have hypotheses, your presentation of descriptive results should be organized according to the sequential test of these hypotheses and with explicit references to the category system. In deductive research, only theory-led inferences based on data analysis and interpretation are considered quality research. For the presentation of descriptive results, the seventh theory-phenomenon iteration focuses on the use of theory expressed by categories and hypotheses. Nevertheless, as the word descriptive expresses, this section in your paper describes empirical findings and inferences and does not discuss them; discussion is required in the section discussion of results (i.e., in the eighth theory-phenomenon iteration). In Figure 4.1, the black arrows indicate the theory lens throughout the research process; the grey arrows in Figure 4.1 are a reminder of that particular theoretical grasp of the phenomenon under study. If you generate nontheory-led results during your data analysis and interpretation, you either missed a part of the theory or discovered a blank spot in the theory. Regarding the latter, you should present those findings as 'limitations' of your theory-led research in the conclusion or use these new insights to reflect on your theory-led findings in the section discussion of results.
7. guide discussion of results (eighth theory-phenomenon iteration). In Figure 4.1, the grey arrow is the reminder of the theoretical grasp of the phenomenon under study. In the discussion of the results, your chosen theory, theorized assumptions from other studies (e.g., used for hypothesis construction), contradictory empirical findings from other studies, etc. guide the argument and explain your findings. The theory-led act of distancing yourself from the data in the discussion of your findings is also a process in which you help others understand your analyses and interpretations in a more abstract way in comparison to the presentation of descriptive results.

8. guide drawing conclusions and deducing social (incl. cultural, economic, political, etc.) implications. Again, the grey arrow in Figure 4.1 emphasizes the view through the lens of theory at the phenomenon under study, that is, that theory is the focal point throughout the empirical research process (nineth theory-phenomenon iteration). Moreover, the direction of the grey arrow from the box 'conclusions and implications' toward the 'theoretical grasp of the phenomenon under study' expresses the requirement for you to present concluding, empirically grounded findings adding to, questioning, or confirming existing knowledge about a phenomenon in context (see contexts of discovery 6 in Table 4.1).

In comparison to drawing arrows to represent the guiding line of theory in a deductive research process, an inductive research approach is characterized by two stages of reflexive use of theory (i.e., two phenomenon-theory iterations). For an inductive approach to research, the phenomenon under empirical study is the focal point of the research process, and the two theory-reflexive stages look at the phenomenon under study from different theory perspectives. The two stages fulfill two different functions for scientific discovery. In Figure 4.2, the first function is represented by 'Ideas about the phenomenon under study' in the first part of the research process. Even if you discovered a completely new phenomenon, you would not have noticed it without comparing the phenomenon to already known phenomena. Accordingly, in the first theory-reflexive stage you should write down all your assumptions about the (newly discovered) phenomenon of study, theoretical approaches to studying similar phenomena (e.g., from previous research), what you read in relevant, possibly remotely-related literature, discussed with peers or in seminars, etc. In addition to having theorized assumptions about the (newly discovered) phenomenon, it is not possible to clear your head from what you know – the literal *tabula rasa* or clean slate – to inductively approach a (newly discovered) phenomenon.

The goal of theory-reflexive stage 1 is to become aware of what you know and assume about a phenomenon so that you can take a proper empirical look at the phenomenon under study. Possibly, your empirical look might be sharpened by testing hypotheses (flagged as optional in Figure 4.2), although others might argue that asking a research question is less directive than hypothesis testing.[14] For the research design and in a paper, the theorized assumptions should be made explicit in the justification of the 'research question(s)' and your arguments in the 'literature review' about why your discovery is new or why it is appropriate to inductively approach a phenomenon which has been previously studied (see Section 4.3.2.1).

Your stage 1 reflexive use of theory will also affect part 2 of the research process: Data collection and processing, which is made explicit in Figure 4.2 by the three overlapping and differently dashed partitions of the research process. For example, in qualitative content analysis, coding units of meaning in documents and constructing a category system will not be free from your theorized assumptions (Part 1: research design). Accordingly, some analytical characteristics of

your assumptions will, most likely, become top and subcategories, but only if they are discovered in the empirical data. Again, the phenomenon under empirical study is primary, and theory is secondary in the research process, therefore, inductive research is and has never been free or bereft of theory.

In automated inductive quantitative content analysis, theory-reflexivity affects data collection and processing in different ways. Data collection for a study means in most cases selecting documents and constructing a data corpus for sentiment analysis (Chapter 16) or topic modeling (Chapters 17 and 18). In multiple correspondence analysis, theoretical assumptions guide, at least partly, the category construction, relating the text data with the document metadata (Chapter 13). Any such effects of theoretical assumptions on the inductive approach to data should be made explicit in the methods section of a paper but already be discussed in the literature review (theory-reflexive stage 1).

Stage 2 reflexive use of theory is led by empirical results in roughly two different ways. First, your theorized assumptions from stage 1 (dashed partition 1) should be reconsidered in the light of empirical findings about the phenomenon under study (dashed partition 3); this could include affirming or rejecting hypotheses. Second, empirical findings about patterns of actors in actions or social structures should be reflected by phenomena appropriate to the theory. For the reflexive use of theory, you need to outline the key assumptions of a theory and then apply them to your empirical findings to 'grasp the phenomenon under study'.

The empirical focus of an inductive approach means that only your findings should be reflected by applying theory and not that you have to explain all features of a theory or theories – this would be too much. As many theories exist, please choose a theory carefully so as not to overload your discussion of your empirical results: The goal is the theory-reflexive discussion of empirical findings and not writing an introduction to the theory. For example, if your phenomenon is a group of actors, appropriate theories could be on class, culture, education, gender, identity, motivation, and profession. However, to avoid drowning in theory, please select a broader social theory that connects patterned actions, for example habitus-field theory (Bourdieu, 1977). In Section 5.4, to reflect the interpretations generated from an inductive content analysis, we use the four institutionalized functions of universities as sieves for social mobility, incubators of identities, temples for legitimate knowledge, and hubs for cultural, economic, political, religious, etc. elites as proposed by Stevens et al. (2008).

Similar to iteratively analyzing and interpreting empirical data, the theory-reflexive approach to your findings helps you to put some distance between yourself and the data and supports you in gaining a more abstract 'grasp of the phenomenon under study' (see also Section 4.3.3). In Figure 4.2, 'grasp of the phenomenon under study' signifies the end of the research journey which started with you having some 'ideas about the phenomenon under research'. In a paper (see Table 4.3), the 'grasp of the phenomenon' is purely empirical in the section 'descriptive results', a mix of empirical and theoretical considerations in the section 'discussion of results' and abstract or more generic in the section 'conclusions and implications'.

Comparing the two figurations of the theory-led and theory-reflexive research processes, you may be surprised to see that the theory-reflexive, inductive approach is more straightforward in its theory use than the theory-led, deductive approach. Of course, the ideal of the deductive research process is that theory engineers all elements in a research process, like the structure of a paper (Table 4.3). However, in the theory-led research process, theory is used in an iterative manner – the researcher has to continuously iterate between theory and the empirical phenomenon under study (e.g., for statistical inferences) – as described in this section. Accordingly, the actual research practice of eight to nine theory-phenomenon iterations makes Figure 4.1 look like someone has accidentally dropped some matches (i.e., the arrows).

4.3.3 The iteration and distancing convention

In the previous Section 4.3.2.3 on the appropriateness-transparency convention, when presenting the difference between a theory-led and theory-reflexive research approach, we already addressed the iteration and distancing convention. In a deductive, theory-led research process, a certain distance from the phenomenon under study is maintained right from the start as the phenomenon is approached by looking through the lens of theory. In comparison to the deductive approach, in an inductive research process, the phenomenon is the focal point, and the two-stage reflexive use of theory is a means of gaining distance from the data.

You need to gain distance from your data because data has literally no meaning unless researchers interpret it and attribute meaning to data to answer a research question. Analysis and interpretation are your unique accomplishments and make a difference between whether you make a contribution with your research or just summarize and repeat what others say in interviews, write in documents, capture in pictures, or express in a video/film. In content analysis and any other empirical research (e.g., ethnography and surveys), analysis and interpretation are based on systematic work with data and theory, which requires time and effort.

Distancing refers to the cognitive-social effort to literally gain distance from your phenomenon, particularly your data about the phenomenon. Content analysis is about a systematic search for manifest and latent patterns based on data analysis and interpretation, but it is neither politics nor social work. Failing to distance yourself from your data and, even worse, from the subjects providing the data, will blur your researcher's eye and make you blind to the manifest and latent content of data. Images, documents, interview transcripts, or data from ethnographic observations may concern sensitive, devastating, touching, unfair, etc. social realities and individual situations. Of course, you can express your empathy during data collection (e.g., in an interview) and you should address your emotions and be reflexive about them during data analysis and interpretation. This applies both to the deductive and inductive research approach.

Figure 4.1 Drawing the lines of theory through the research process for a deductive approach to empirical research (solid black arrows = theory-led analysis; dotted arrows = optional hypotheses; grey arrows = theory-led answers).

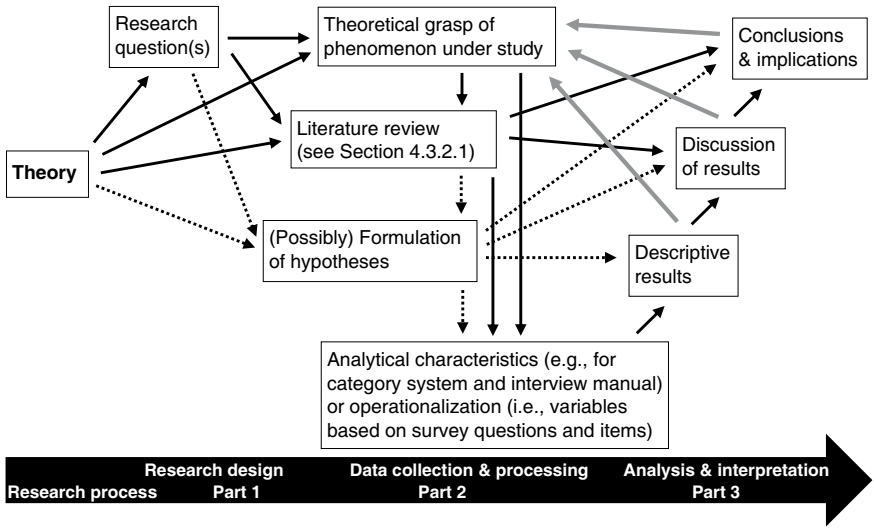
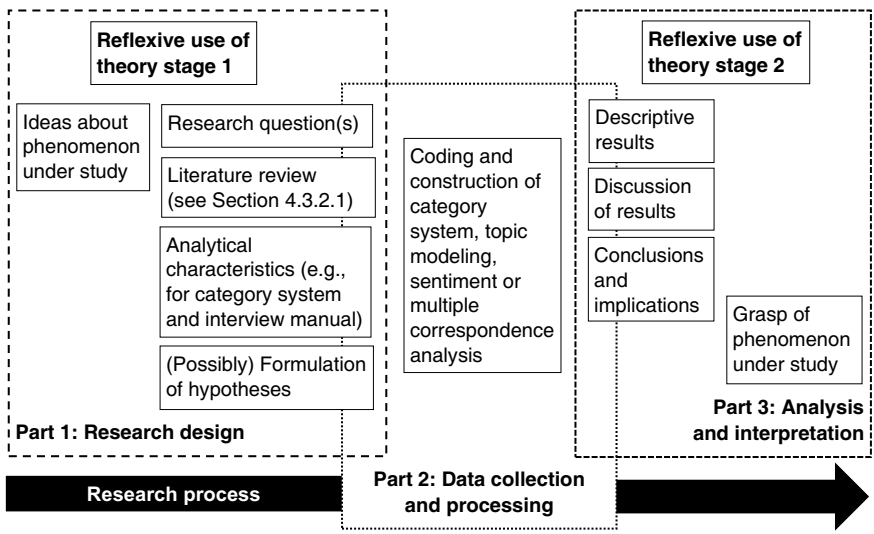


Figure 4.2 Two-stage reflexive use of theory for an inductive approach to empirical research.



4.3.3.1 Iteration and distancing: An emotional example

To gain distance from emotionalizing data, you should iterate in data analysis and interpretation. To reiterate: Iteration was defined as the repetition of an action, or a process to construct a sequence of outcomes.[15] For example, acknowledging the significance of your emotional reaction to data content, the first iteration in data analysis and interpretation would be writing down your reactions to get to the bottom of what content got you. To find out what got you, you may need to go over the coded units of meaning or your entire data several times, that is, conduct an iterative process of analysis and interpretation. In your research diary, please note each insight from your content analysis and refer to the relevant units of meaning for how you got it – otherwise you will get lost in the serendipity of data interpretation. Having done several iterations of data analysis and interpretation and noticing that no further insights emerge from your effort of content analysis, please put data aside and focus on the findings in your research diary. By comparing your findings from each iteration of data analysis and interpretation, you start a more abstract iteration in data analysis and interpretation. Again, it is imperative that you take notes because the formulation of your findings is a shield against going in analytic and interpretation circles.

The iteration process has come to an end with you reaching a conclusion or conclusions about what got you about your phenomenon under study. The conclusion distancing you from your data is a result for reflexive use in further data analysis and interpretation and, accordingly, contributes to your discoveries (in context(s) 4, 5, and/or 6; Table 4.1). The issue of emotions is just an example for the explanation of the iterative process of data analysis and interpretation. Methodologically, the iterative process of content analysis and interpretation has to be repeated for all top and subcategories separately, and then, in combination, all top categories to understand and explain patterns in your phenomenon under study. Please present the core of this iterative process in the appendix of your paper (see Table 4.6) so that readers (e.g., your instructor) (a) become aware of your efforts, (b) can (e)valuate the appropriateness of your analysis and interpretation, and (c) are assured that you did not take the first idea and explanation that came into your head as a conclusive finding.

4.3.3.2 Iteration and distancing: the data coding process

Iteration and distancing efforts, the appropriateness of empirical grounding and you yourself being sure that you found the best solution is also very relevant during the coding process. Coding is part of data processing according to the category system. In a deductive research process, the category system was constructed based on the 'theoretical grasp of the phenomenon under study' (Figure 4.1).[16] To code means to attribute a category to a segment of text, for example a sentence, several sentences, or a paragraph. How much text you code depends on the category-led construction of a unit of meaning, that is,

coherent content in a sequence. Coding is the activity of creating order in documents (e.g., interview transcripts, files, and social media webpages) by selecting units of meaning to answer the research question. Noncoded text segments are not considered relevant for answering the research question according to the category system. As in encryption, the coded units of meaning await decoding, that is, content analysis and interpretation by researchers. For more elaborate examples of the coding process, please see Section 6.3 (inductive qualitative content analysis) and Section 7.3 (deductive qualitative content analysis).

Meeting the demands of the iteration and distancing convention can best be understood by looking at the end of the theory-led coding process. Assuming the theoretical robustness of your category system, in general, we would recommend using the following signals as guidelines for being finished:

1. If you are going through the text segments that are not coded and realize that no unit of meaning can be coded based on your category scheme. This is referred to as empirical saturation in qualitative research.
2. When reading the documents, you do not generate new ideas or additional insights to answer your research question(s).
3. You are stuck, hopelessly (e.g., going in circles, round and round and round).

Please be assured that the requirement that *you do your best* does not mean that your data processing, analysis, and interpretation must be perfect. At some point, you also need to finish the research process, answer your research question, present results, write up, and hand the paper over (e.g., to your professor or an editor).

4.3.3.3 How not to get caught in the iteration loop

Iteration and distancing should not be a never-ending process. Unfortunately, we cannot provide an easy rule of thumb, such as: do three iterations of going through all units of meaning of a code, do three iterations of noted analyses and interpretation, and then you can draw conclusions. As we do not know the amount of data you are dealing with (e.g., content analysis of four interviews or thousands of social media posts), whether you are doing a deductive or inductive, or a qualitative or quantitative content analysis, we formulated some questions in Table 4.7 that will support your conclusion of what we named an iteration loop.

In Table 4.7, some questions are formulated to give a yes or no answer, so you can make a check mark once you complete the task in the research process. However, at this point in Chapter 4, you are well aware that the *yes* is just the beginning of the answer which justifies the empirical grounding of your results and appropriateness of discoveries. *How*, *which*, *why*, and *what* questions directly require answers related to the transparency-appropriateness convention (Tables 4.4, 4.5 and 4.6) – please, also make a check mark once you answered them!

Table 4.7 Schematic Loops and Questions to Be Answered for the (E)valuation according to the Iteration and Distancing Convention

Iteration loop	*Element in research process*	*Appropriateness why and transparency how of iteration and distancing*		
		Deductive, theory-led approach	*Inductive, data-driven and theory-reflexive approach*	
			Qualitative	*Quantitative*
1 (context of discovery 1)	Initial description of phenomenon	How have all relevant features of your chosen theoretical approach been applied to the phenomenon for empirical study (theory-phenomenon iteration (TPI) 2; see Section 4.3.2.3)? Does your theoretical approach miss or focus on certain characteristics of your phenomenon for empirical study?	What observation(s) contributed to the idea(s) about the phenomenon for empirical study? What are your theoretical assumptions about the phenomenon for empirical study (reflexive use of theory stage 1; Figure 4.2)?	
2 (context of discovery 1)	Literature review	Concerning the phenomenon of empirical study, have you considered relevant previous studies, theories, and methods, and are you aware of how these do (not) inform your research design?		
3 (context of discovery 3)	Data processing	Were all top and subcategories deduced from theory applied during coding (TPI 6)? During coding, which blank spots did the theory-based category system reveal in data or vice versa?	During the coding process, in which units of meaning do top and subcategories originate? What are the empirical properties covered by the category system (e.g., exemplary quotes)? With reference to your research question and/or hypotheses, which parts of the empirical material are not relevant?	Concerning explained variance, coherences, and predictive performance of your findings, which is the most suitable number of topics (see Chapters 18 and 17), number of dimensions in a multiple correspondence analysis (Chapter 15), and accuracy in the prediction of sentiment classes on a holdout test set in a sentiment analysis (see Chapter 16).

(*Continued*)

Table 4.7 (Continued)

Iteration loop	*Element in research process*	*Appropriateness why and transparency how of iteration and distancing*		
		Deductive, theory-led approach	*Inductive, data-driven and theory-reflexive approach*	
			Qualitative	*Quantitative*
4 (context of discovery 4)	Descriptive results	Have all categories of the category system been analyzed and interpreted? Do subcategories analytically describe the top category? Do the top categories describe the phenomenon under study?		Why can certain topics be analyzed and interpreted, and others not? What are the (hyper)parameters that enable an algorithm to extract coherent topics or to assign sentiment classes with minimal bias? If the models are biased, are they biased systematically (e.g., predict too positive sentiment classes), and is the cause for this bias inherent to the algorithm, the data, or the interpretations injected into the algorithm? Do the algorithms inherit the interpretations of the human annotators/interpreters who fed the algorithms with ascribed meaning to texts? Do the algorithms fluctuate and display erratic behavior in assigning topics or sentiment classes on the documents?
5 (context of discovery 5)	Discussion of results	Focusing on descriptive results (TPI 7), does the eighth TPI consider TPI 2 (theoretical grasp of phenomenon), TPI 3 (literature review), and possibly TPI 4 (hypotheses)?	Have empirical results been analyzed and interpreted according to reflexive use of theory (stage 2; Figure 4.2)?	
6 (context of discovery 6)	Conclusions and implications	Are results considered according to the ninth theory-phenomenon iteration?	Does the concluding 'grasp of the phenomenon under study' consider the research journey from the initial 'ideas about the phenomenon' via stage 1 and 2 of the reflexive use of theory?	

4.4 Conclusions: discovery, appropriateness-transparency, and iteration and distancing quality conventions

By now, you may be close to cognitive system overload caused by quality conventions for empirical research. But despite the cognitive overload, please don't look up from this textbook yet. You should never forget that doing empirical research is very demanding: You need to understand a phenomenon in its

Figure 4.3 Discovery, appropriateness-transparency, and iteration and distancing quality conventions in deductive qualitative content analysis according to key characteristics.

	Part 1: Research design	**Part 2: Data collection and processing**	**Part 3: Analysis and interpretation**
	Research process (epistemology and methodology)		
Discovery	Phenomenon Research question(s) Hypotheses Method(s)	Select documents Data preparation	Descriptive results Discussion of results
Appropriateness -transparency	Selected theory Literature review Instrument Cases selection	Coding Category system Secondary study limitations	Theory use Empirical grounding Findings in context Compared to literature
Iteration and distancing	Theory application Reviewed literature Grasp of phenomenon	Application of category system during coding as analysis and interpretation	Inferences Conclusions Limitations Implications

Figure 4.4 Discovery, appropriateness-transparency, and iteration and distancing quality conventions applied to inductive qualitative content analysis according to key characteristics.

	Part 1: Research design	**Part 2: Data collection and processing**	**Part 3: Analysis and interpretation**
	Research process (epistemology and methodology)		
Discovery	Phenomenon Research question(s) Hypotheses Method(s)	Select documents Data preparation Category system	Descriptive results Discussion of results
Appropriateness -transparency	Assumptions Literature review Instrument Cases selection	Coding Top and subcategories Secondary study limitations	Theory-reflexivity Empirical grounding Findings in context Compared to literature
Iteration and distancing	Theory-reflexivity Reviewed literature Idea of phenomenon	Coding and category construction as analysis and interpretation	Conclusions Limitations Implications

social, spatial, and temporal context, know and apply a method and a theory. So relax – take some mental steps backwards to get a distanced look at all the details on discovery, appropriateness-transparency, and iteration and distancing quality conventions presented in seven tables and two figures in this chapter.[17] The explanation of all these details is necessary for you to understand what it takes for (good) quality research.

Figure 4.5 Discovery, appropriateness-transparency, and iteration and distancing quality conventions applied to inductive quantitative content analysis according to key characteristics.

	Part 1: Research design	**Part 2: Data collection and processing**	**Part 3: Analysis and interpretation**
	Research process (epistemology and methodology)		
Discovery	Phenomenon Research question(s) Hypotheses Method	Select documents Data preparation Metadata categories Statistical tests	Descriptive results Discussion of results
Appropriateness -transparency	Assumptions Literature review Instrument Sample/cases selection	Statistical indications Meta data Secondary study limitations	Theory-reflexivity Empirical grounding Findings in context Compared to literature
Iteration and distancing	Theory-reflexivity Reviewed literature Idea of phenomenon	Analysis and interpretation (e.g., no errors)	Inferences Conclusions Limitations Implications

Figure 4.6 Discovery, appropriateness-transparency, and iteration and distancing quality conventions: compatibility with deductive survey research according to key characteristics.

	Part 1: Research design	**Part 2: Data collection and processing**	**Part 3: Analysis and interpretation**
	Research process (epistemology and methodology)		
Discovery	Phenomenon Research question(s) Hypotheses Survey	Field work Data cleaning Statistical tests	Descriptive results Discussion of results
Appropriateness -transparency	Selected theory Literature review Questionnaire Sampling	Reliability Validity Other significance tests	Theory use Empirical grounding Findings in context Compared to literature
Iteration and distancing	Theory application Reviewed literature Grasp of phenomenon	Analysis and interpretation of tests (e.g., no errors)	Inferences Conclusions Limitations Implications

Then, when you are actually *doing* research, you notice that the elements of the three schematic parts of the research process align with one, two, or all three quality conventions, as we summarize for deductive qualitative content analysis in Figure 4.3, for inductive qualitative content analysis in Figure 4.4, and for inductive quantitative content analysis in Figure 4.5. The characteristic key differences between Figure 4.3 and Figure 4.4 are explained by either the theory-led (Figure 4.3) or the theory-reflexive approach (Figure 4.4). The differences between key characteristics in inductive content analysis are the differences between qualitative small data (Figure 4.4) and quantitative big data (Figure 4.5), which enables statistical operations, as well as training and fine-tuning of algorithms (see Chapters 11–18).

In addition to differences associated with different methods, quality conventions have to respect particular aspects of data collection and data type (see Chapter 3). For example, ethnographic observations are documented in field notes by researchers and interviews are conducted according to a manual (i.e., instrument; see appropriateness-transparency convention in Figures 4.3 and 4.4) guiding the interactive communication between two or more subjects. We believe that all particularities (e.g., data collection and processing) of qualitative methods can be addressed by the discovery, appropriateness-transparency, and iteration and distancing quality conventions. The compatibility or connection of methods for empirical social research by means of the three quality conventions is the fundamental idea of the substantive methods concept (*substanzieller Methodenbegriff*). Accordingly, the discovery, appropriateness-transparency, and iteration and distancing quality conventions are also suitable for mixed or merged methods studies.

Of course, the epistemology, that is, how scientific knowledge is created, and methodology, that is, the logical process of applying a method, differ between qualitative, quantitative, mixed methods, and algorithm-driven machine learning research – but not the research process schematically composed of three parts. Accordingly, the substantive methods concept – composed of discovery, appropriateness-transparency, and iteration and distancing conventions – can also integrate quality conventions specifying methods for data collected by surveys, quantitative data processing, and statistical analysis (Figure 4.6). For example, the contexts of discovery (Table 4.1) are the same for content analysis and survey research. The adequacy-transparency convention can be applied just as well to the process of selecting a (random) sample group for conducting a survey and to respondents completing a questionnaire. Beyond the specificities of statistical analysis (i.e., performing calculations and conducting tests; see Section 4.2.1), Figure 4.6 shares key characteristics with deductive qualitative content analysis (Figure 4.3) as well as inductive, automated quantitative content analysis (Figure 4.5). How nice.

Notes

1 Please be aware that we use the terms 'findings' and 'results' as synonyms.

2 In the case of previous evaluations by human coders, we would speak here of a *ground truth*, which refers to an open and obvious truth. Caution is also required here. For example, it is easy to have a statement such as 'The chair is red' checked

if there is a chair or several chairs with different colors in a room or in an image file and you can look to see if this chair is really red. However, the more abstract the concepts to be checked become (e.g., emotions and macro-theoretical concepts), the more interpretation and subjectivity flow into the viewpoint that your computer or algorithm will use as objectively "true" for its own evaluation. In other words, as the degree of abstraction increases, the ground truth is also more socially constructed and dependent on the evaluators (see Grosman & Reigeluth, 2019; Jaton, 2021). Accordingly, you cannot assume that the results of your automated analysis can be generalized. On the contrary, many algorithms conceal and disguise the basis on which the categorizations were calculated, which further reduces the reliability and validity of the applications.

3 You may have heard about the correlation coefficient in one of your statistics courses. This takes values in the range from +1 to –1. In our case, +1 would indicate a perfect match between the algorithm and human categorization, 0 a completely random match, and –1 a complete discrepancy between the algorithm's prediction and the human's categorization attempts.

4 This is a statistical procedure in which a dataset is divided into a number of (e.g., ten) samples and the results for one group (e.g., the assignment of statements to sentiments) in one sample can also be found in all of the other samples.

5 This includes inequalities in users' access to databases, differences in technical equipment and Internet access, and knowledge of how to use the Internet and its applications.

6 In the schematic of a research process, actually doing content analysis is situated in the second and third part of the research process. You can apply content analysis for analyzing and interpreting field notes from an ethnographic study, interview transcripts, free text responses in surveys, and visual data, for example pictures (either as an image or transcript of what researchers observe in a photo or painting) and videos (i.e., verbal, sound, image, and mix of data types).

7 To be clear: the schematic distinction between the three parts of a research process is an analytical distinction to facilitate understanding quality conventions for social research. In an actual research process some parts overlap, for example the construction of an inductive category scheme is both data processing (part two of the research process), and analysis and interpretation (part three of the research process; for more on interpretation, see Chapter 5).

8 For example, researching an original topic as well as a disregarded or emerging social group would be considered original in social sciences.

9 We are aware of earlier uses and definitions of 'context of discovery' by Reichenbach (1938, 1951) and Popper (1935/2002). Here 'context of discovery' is defined rather broad as a context where a discovery may happen all throughout the research process. Accordingly, our use of 'context of discovery' has nothing whatsoever to do with the efforts of Reichenbach and Popper to establish empiricism as a philosophy of science.

10 Intellectual property rights in particular and informants' anonymity and data protection should not be taken lightly. Infringements of intellectual property, such as not respecting copyrights and copylefts (see Section 1.5), can get you sued. For example, you can quote some sections from another text with proper citation and referencing. In a similar way, you can transcribe sequences from films, but you should not make the entire transcript of a film available – the intellectual property rights belong to the authors and, possibly, a film studio. For sure, you can find thousands of pirated film scripts on the World Wide Web, which you can use for quantitative content analysis, but do not provide links to where you found them online or provide others with access to your film corpus.

11 For sure, expertise increases as a correlation between years as a higher education teacher and reading time – assuming that they are a studious scholar who is continuously involved in reading and understanding (old and new) literature.

12 In social sciences, there are many different definitions of what a theory is (e.g., Abend, 2008).
13 If theory-based categories do miss important characteristics of a phenomenon, categories developed from empirical data can be added to the category system in a mixed deductive and inductive approach. For novices to content analysis, we do not recommend mixing deductive and inductive approaches because of the big challenge of combining theory-led and theory-reflexive approaches in one research project.
14 A hypothesis is always directive, that is, makes a statement such as 'if ... then ...', 'more time for research means less time for being bored' and 'a positive correlation between a university degree and quality of life'.
15 Repeating, that is, iterating an action, is comparable to practicing, for example how to catch a ball, paint, read, walk, write, etc. The intended goal of practice, that is, iteration in data analysis and interpretation, is to find patterns in social action, their motivation, meaning, values, etc. Similar to how the goal of practice is to master reading, walking, or writing, the goal of data analysis and interpretation is to discover empirically grounded manifest and latent patterns about the phenomenon under study during content analysis.
16 To reiterate: For empirical research beginners, we do not recommend mixing deductive and inductive approaches. Accordingly, even though your category system derived from theory (deductive research approach) might reveal deficits during the coding process, please abstain from developing categories from empirical data (inductive research approach), that is, producing a combination of deductively and inductively constructed categories. If you decide to add inductive categories, it is essential to revisit and most likely recode the already coded parts of the text. This is a learning effort for your second empirical research project.
17 The beginning of doing empirical research is no different from learning anything else. Just imagine learning to catch a ball: You need to understand the procedure (e.g., someone throwing the ball toward you), hand-eye coordination (e.g., see ball and get hand ready to catch), and the biomechanics (e.g., close hand when the round thing touches your skin). In addition, you need to be aware of dangers (e.g., distortion of fingers and not standing in front of something that can break, in case you miss the ball) and your environment (e.g., not to harm others or risk the ball going downhill), etc. In case you overthink the entire process of catching a ball, the ball has passed you and hit the ground long before you even move a finger. With practice, your ability improves, and you may become able to catch the ball without looking at it.

5
DATA INTERPRETATION
A practical guide

5.1 Introduction

Interpretation is the single most important part of data analysis, both for qualitative and quantitative research. Data has literally no meaning unless researchers interpret it and attribute meaning to data to answer a research question. Interpretation is your unique accomplishment and makes a difference between whether you make an original contribution with your research or just summarize and repeat what others say in interviews, write in documents, capture in pictures or express in a video/film. In content analysis and any other empirical research (e.g., ethnography and surveys), interpretation is based on systematic work with data and theory, which requires time and effort.

Interpretation in a research process is a methodological and, therefore, substantially different activity from everyday interpretations of situations, for example, reacting in an appropriate manner in a social situation. Interpretation should also not be confused with summarizing text, identifying a topic in a text, or drawing conclusions. In qualitative and quantitative content analysis it is quite the other way around: Conclusions are the result of interpretation, which are backed by an empirically grounded systematic analysis according to a category scheme. A category scheme is the very first result of your research – no matter whether you do inductive, empirically led content analysis (see Chapter 6), or deductive, theory-led content analysis (see Chapter 7).

As emphasized in the literature (e.g., Kuckartz & Rädiker, 2022; Mayring, 2021; Schreier, 2012), interpretation is based on a systematic process of data analysis, which varies considerably depending on the type of data (see Chapter 3), social phenomenon, context, etc. Accordingly, interpretation cannot be standardized. However, we can provide practical guidance for the interpretation process. In this chapter, we will support you in your interpretation and, furthermore, in interpreting manifest content as well as latent content,

DOI: 10.4324/9781003496397-6
This chapter has been made available under a CC-BY-NC-ND 4.0 license.

which is also referred to as the meaning hidden between the lines or behind the manifest, obvious wording of a text (see Table 2.1).

In the following Section 5.2, we start with a simple interpretation exercise to provide you with ideas about what interpretation as a basic activity is. In Section 5.3, we continue by introducing more empirical text data and providing examples for content analysis and interpretation to answer the research question. In order to conduct content analysis by interpretation, in this chapter, we elaborate seven core issues to be considered during the research process:

1. An interpretation is your contribution. Doing an interpretation will, most likely, produce a lot more text than the original unit of meaning (e.g., a sentence).
2. In general, more than one interpretation of a unit of meaning is possible. In addition to the most plausible interpretation, alternative, and/or counterfactual interpretations of the unit of meaning should also be considered.
3. To answer a research question, several interpretations drawn from unities of meaning across the text document(s) should be considered, ordered, and reflected on.
4. Interpretations need to be situated in the specific social, spatial, and temporal context of your research.
5. As a result of points 3 and 4, different interpretations and contexts have to be aligned.
6. In content analysis, interpretations are an integral part of several stages of the research process, such as category construction, making meaning from coded text segments, discussing results, and drawing conclusions.
7. Theory is an essential tool to support data analysis and interpretation.

The practical examples will help you understand that interpretations are part of different stages of the research process and that types of interpretation vary according to scope. For example, an interpretation can be considered narrow in scope during inductive category construction and coding the text (Section 5.3.1). The interpretation of small segments of a text, for example, an exemplary quote (*Ankerbeispiel*) or key text sequence, is a very good way to practice doing interpretation. It will also help you to gain a more in-depth understanding of your data, assuming that you know your data very well already. In the middle stage of interpretation, you would do in-depth content analysis and interpretation of a top category including two or more subcategories (Section 5.3.2). Broad scope interpretation would mean interpretation supported by theory (Section 5.3.3) and offering conclusions as answers to your research question. To elaborate how theory can support interpretation, in Section 5.4, we present three types of theories: concept, proposition and comparison, and typology.

In Section 5.5 titled "Overcoming interpretation *Angst* by doing interpretation", we take a step back from the examples highlighting the actual interpretation efforts. In an abstract manner, we conceptualize the beginning of the

content analysis and interpretation process as having to deal with a surplus of data and meaning. Better understanding the overall content analysis and interpretation process will help you to create working strategies, apply a pattern for in-depth data analysis to advance from manifest to latent content meaning, and define intermediary goals for the continuous development of your research and toward finishing your project.

5.2 Data interpretation for content analysis: an inductive example

The interpretation example is just one line. Accordingly, the first interpretation steps are understanding the line and all the manifest content provided in the text data. The context is presented later. But: Don't look ahead! The line is as follows:

"That's the beauty of science, Kate. There's no luck! Follow your evidence."

Analyzing the line bit by bit, several manifest content interpretations and questions relevant for interpretation work are plausible:

1. Seemingly a person is addressing another person named Kate.
2. The sender of information could be a chat-bot?
3. Is the sender a colleague, who Kate does research with, or a teacher (in school) or a professor (at university)?
4. The line, segmented in three sentences, expresses something positive. Positive significance is related to "beauty", "luck", and "evidence". The noun "science" is qualified by the adjective "beauty".
5. The positive meaning of the line could be irony or sarcasm. However, this is not clear as the words are printed and the tone of voice indicating the opposite meaning of what is said is missing.
6. "That's the beauty of science, Kate" is a statement – not a question or a conclusion.
7. "There's no luck!" is an emphatic expression underlined by the exclamation mark. The negative wording seems to be used to support the statement in the first sentence and the third sentence. The key message of the line does not alter the meaning because of the second sentence.
8. "Follow your evidence" is a request; it seems intended to motivate Kate.
9. Kate is told that a specific "beauty of science" is related to following the evidence.
10. The "beauty of science" seems not related to "luck".
11. Has the "beauty of science" anything to do with discovery? No manifest content explicitly suggests that the line addresses scientific discovery.
12. Kate has some "evidence"; however, Kate is urged to continue and find more "evidence" to her "evidence". Kate's "evidence" seems to be not enough "evidence" for or on whatever.

13. Why should Kate "follow" the evidence? Is the "evidence" lined up piece by piece, hidden from the *inspector* or moving away?
14. ...

The three dots mean that we could go on with the interpretation of manifest content embedded in the line, but we think 13 interpretations suffice for our cause. First, you notice that interpretation produces a lot more text than what we have as data. Second, several of the interpretations are very close to the text – gathering factual evidence. However, some interpretations are not plausible, and other interpretations are formulated as questions.

Accordingly, and this is the third point, it is necessary to reorganize our interpretation (attempts). For example, points 1–3 are concerned with the unknown sender, who/which addresses Kate. We are clear about Kate being the addressee of the line. Some of the words said to Kate seem clear (e.g., points 6–8), but some remain vague (e.g., points 9–13). By writing this (in your research diary), you start to structure the interpretation efforts and produce first results – these results should be noted in the research diary as well. Whether these interpretation results are intermediary or rejected later in the research process cannot be anticipated yet. This means that you have to come back to these results and re-evaluate them at the end of the interpretation process. One way or another these (intermediary) results will contribute to the transparency and reliability of your research (see quality conventions in Chapter 4), help others to understand your inferences, and underscore how close or explicitly the interpretation is based on data, for example, the line "That's the beauty of science, Kate. There's no luck! Follow your evidence."

Of course, fourth, the interpretation needs to be situated in the specific social, spatial, and temporal context of your research. Unlike this interpretation example, you are aware of these contexts. The line was spoken by professor Dr. Randall Mindy to astronomy PhD student Kate Dibiasky in the film *Don't Look Up* (Box 5.1; director: Adam McKay; scriptwriters: Adam McKay and David Sirota). Unfortunately, the line is not part of the film version and only part of a, seemingly, earlier version of the film script – you may search for the PDF of the script on the Internet. We selected the line for the interpretation example because it did not give away its origin and it expresses a necessary, however not always adequate illusion of (natural sciences) research (e.g., Knorr Cetina, 1999; Merton, 1942/1973; Mitroff, 1974).

By including the film context in our interpretation, some of the 13 interpretations can be ruled out (e.g., point 2, as it is not a bot speaking to Kate), and others become qualified, such as point 3, now that you know that the sender is a professor in a university research lab.

In what follows, please be aware that we are using a science fiction film as an interpretation example. On the one hand, films have long been considered a

viable source of empirical evidence for sociological analysis (e.g., Denzin, 2004; Kirby, 2013; Morin, 1956/2005). On the other hand, films conscript interesting stories to fit a plot or storyline, and in the plot stereotyping, for example, science and scientists, is applied to connect audiences quickly to the character(s), theme(s), and genre of a film (e.g., Critchton, 1999; Frayling, 2005; Schweinitz, 2006/2011).

Fifth, when doing an interpretation, most likely, this means that not all data is limited to one part of a document, just like a scene in a film, but the social context has to be grasped by analyzing different parts of the document, well, different scenes of the film. Please be aware that *Don't Look Up* is a science fiction film, and we are already interpreting, at most, truly possible social events in the USA (spatial context) in the second and third decades of the 21st century (temporal context).

Box 5.1: Summary of film *Don't Look Up*

"Kate Dibiasky (Jennifer Lawrence), an astronomy grad student, and her professor Dr. Randall Mindy (Leonardo DiCaprio) make an astounding discovery of a comet orbiting within the solar system. The problem – it's on a direct collision course with Earth. The other problem? No one really seems to care. It turns out that warning mankind about a planet-killer the size of Mount Everest is an inconvenient fact to navigate. With the help of Dr. Oglethorpe (Rob Morgan) [NASA head of planetary defense unit], Kate and Randall embark on a media tour that takes them from the office of an indifferent President Orlean (Meryl Streep) and her sycophantic son and Chief of Staff, Jason (Jonah Hill), to the airwaves of The Daily Rip, an upbeat morning show hosted by Brie (Cate Blanchett) and Jack (Tyler Perry). With only six months until the comet makes impact, managing the 24-hour news cycle and gaining the attention of the social media obsessed public before it's too late proves shockingly comical – what will it take to get the world to just look up?" (IMDB 2021)

5.3 Interpretation as a systematic activity based on a category system, coding empirical data, and theory

These events unfold as Kate and Randall follow the evidence by adding more observations and doing the math on the comet's trajectory. Shockingly, Earth lies on the comet's trajectory. Professor Mindy contacts a scientist at NASA, who then connects him to the NASA expert on comets Dr. Oglethorpe, who is head of the Planetary Defense Coordination Office in Washington, DC. Kate and Randall have conferred with NASA headquarters and their scientists, who have checked and confirmed their calculations. This makes the comet an issue

of national security, and Dr. Oglethorpe arranges for Kate and Randell to meet with the US president and her advisors.

Before we go on, we have to return to square one of the research process: What is the research question? As usual, the research process begins with the research question(s) or hypotheses, which guide the entire research design and process. Due to the didactical approach to introducing interpretation in this chapter, the research question has so far remained implicit; now we state it explicitly: "How is legitimate scientific knowledge produced?" The research question contains two issues: the production of knowledge, for example, scientific evidence in the form of an observable fact (i.e., Comet Dibiasky), and the social acceptance or rejection of the scientific evidence and its implications for political or economic behavior (i.e., its legitimacy). We selected the research question because the legitimacy of knowledge is pivotal to the discussion of good research quality, the justification of science in society (e.g., Lewandowsky et al., 2016; Rekker, 2021), and (should be) a core issue in scientific research education and training. In addition, the research question is motivated by personal curiosity, with higher education and science being one of our core research areas for more than a decade.

To answer the research question, not all parts of the film are relevant. Moreover, to answer the research question, you are required to distinguish between very relevant and not so relevant parts of the film, that is, the data. Being very familiar with your data, the process of bringing order to the data should not be difficult for you. In inductive qualitative content analysis, the ordering process is supported by the construction of categories and coding data, which will be explained in the following Section 5.3.1. The construction of categories and codes as well as the selection of relevant film segments is already part of the interpretation process and, therefore, should be well documented in your research diary.

Guided by the research question, and in the interest of brevity, we focus on the six key segments of the film debating the legitimacy of scientific knowledge. Even without the encouraging line by Randall: "That's the beauty of science, Kate. There's no luck! Follow your evidence," Kate has followed her evidence, and her scientific evidence has been peer reviewed, checked, and confirmed by NASA – the US *National Aeronautics and Space Administration*. Nevertheless, when Kate and Randall report to the different audiences, it becomes evident that a fact is only a fact if (we have added the emphasis on public and private elite universities and provided additional information in square brackets):

2:16:01 until film end[1]	*TELEPHONE CONVERSATION*
Dr. Oglethorpe:	What are the most updated estimates on its [the comet's] trajectory?
Randall Mindy:	Kate.
Kate:	Um ... I've been running it all day. I keep getting the same result. A direct hit of Earth in six months and 14 days.

(*Continued*)

(Continued)

Randall:	Me too.
Dr. Calder [NASA]:	Matthew, what are your calculations?
Matthew:	Um, Sout program is saying six months, 14 days.
Dr. Calder:	We got six months and 14 days.
Dr. Oglethorpe:	And it's roughly five to ten kilometers wide. Which ... which, uh...
Randall:	Isn't that... an extinction level event?
Dr. Calder:	Well, let's not be dramatic here.
Dr. Oglethorpe:	Okay, uh ... First move is to get the school field trip here to D.C.
Dr. Calder:	Let's calm down.
Dr. Oglethorpe:	I need the person who first had eyes on the comet and the person in charge.
Randall:	What?
Dr. Oglethorpe:	... share this data immediately with *Cambridge, Caltech*, and IAU [International Astronomical Union].
Dr. Calder:	Wait a minute, this is classified information.
Randall:	What is happening?
Dr. Calder:	I serve at the discretion of the president.
Cut	**[...]**
2:02:45 until film end	OVAL OFFICE, WHITE HOUSE
President Orlean:	The point is you can't go around telling people there's a 100% chance they will die. You know, that's just nuts. And we should get our own scientists on this. No offense, but you're just two people who walked in here with ...
Dr. Oglethorpe:	Dr. Oglethorpe, I've been at NASA for fifteen years. And Dr. Mindy is a tenured professor of astronomy at *Michigan State* where Ms. Dibiasky is a doctoral candidate.
President Orlean:	Good.
Jason:	I am sorry, did you just say *Michigan State*?
Dr. Oglethorpe:	Exactly. They have an excellent astronomy department.
Jason:	Yes, come on bro.
Kate:	Are you kidding? Do you want to see our goddamn SAT [standardized scholastic aptitude test for university admission] scores?
Cut	[...]
2:01:27 until film end	OVAL OFFICE (CONTINUED)
President Orlean:	Okay. At this very moment I say we sit tight and assess.
Jason:	Sit tight and assess.
President Orlean:	Let's get some other people on this. Some *Ivy Leaguers*. And you and I will review the NASA plans. OK?
Dr. Oglethorpe:	I'd greatly appreciate that.
Kate:	Am I to understand correctly that after all the information you've received today, the decision you're making is to 'sit tight and assess?'
Jason:	I'm sorry. Who is she?
Kate:	Who the fuck are you? Aren't you, her son?

(*Continued*)

(Continued)

Jason:	I'm the fucking Chief of Staff. Boy with the dragon tattoo. So, I'm doing fine. [The dragon tattoo is supposed to symbolize power, strength, wisdom or the like. In an earlier version of the script, but not the film, Jason says that he went to *Cornell Business School*, class of '05, Jason also adds "Miss *State School*", referring to Kate.]
Cut	[...]
1:55:34 until film end	INTERNATIONAL NEW YORK HERALD OFFICE
Chief editor:	What other confirmation do we have on this, Adul?
Adul:	I've spoken to Dr. Oglethorpe in D.C. He confirmed it and all the math. Then, I showed it to Dr. Franks at *Columbia*, and he almost fell out of his chair. So, look, I have to admit. I was very skeptical at first, but this is all quite credible.
Randall:	The calculations are credible.
Chief editor:	Jesus. And the president really said to sit tight?
Randall:	Yes, that's correct.
Kate:	'Sit tight and assess'.
Ms. Tanken:	She's probably worried about the midterms.
Cut	[...]
1:34:38 until film end	OVAL OFFICE
President Orlean:	I think, it was Winston Churchill who said or maybe it was Bobby Knight who said, "The great labor of leadership is to lay down one's yoke of pride and admit that mistakes were made".
Jason:	You were deceived, as was I, but the end result was, we had our scientists who went to, uh, prestigious schools of *Harvard*, *Princeton*, et cetera. They confirmed the data. So, we would like to offer you a presidential apology.
Randall:	Really?
Jason:	And we are prepared to mobilize in a historic fashion to save this planet.
Man's voice:	The best news I have heard in a long time.
Kate:	So, basically, what you're saying is you're about to lose the midterms because you got caught sending pictures of your cooch to your porn star sheriff. So, now it's to your advantage to act on the comet?
President Orlean:	Right
Cut	[...]
1:11:33 until film end	BOJO MAMBO'S SHRIMP RESTAURANT (IN THE EVENING)
Dr. Oglethorpe:	That's insane. Like, completely insane.
Kate:	It will never work.
Randall:	I had the exact same reaction.
Kate:	This is a nine-kilometer-wide planet killer we are talking about, not roaming charges [referring to BASH cellular phones].

(*Continued*)

(Continued)

Randall:	Look, I ... I was confused and outraged as well that they turned the mission [to destroy the comet with nukes] around too. But they've [BASH] got Gary Talcamont out of *Stanford* and Lisa Inez from *Princeton* on their team. I mean, this is cutting-edge next generation technology.

Source: Film *Don't Look Up* on Netflix; authors' transcript

In the following scene we learn that Inez and Talcamont have received a Nobel Prize, the most prestigious award in science. Talcamont presents his nano-scanners, which are supposed to investigate the comet's integral structural points. Based on this on-site information gathering, microtargeted quantum fission explosives developed by Inez and her team are supposed to slice the comet into 30 smaller pieces that can be harvested from the ocean. The reason for the abortion of the mission to destroy the comet with nuclear explosives – see conversation between Kate, Dr. Oglethorpe, and Randall in Bojo Mambo's shrimp restaurant – was that the scientists from the private, billionaire-owned corporation BASH discovered rare minerals worth almost 140 trillion dollars. At first, Randall was stunned by the fact that the president and her billionaire friend are willing to take the risk of the comet destroying earth for private profit, because the cutting-edge technology developed by the scientists from *Princeton* and *Stanford* had not yet been tested under real-world conditions. As we learned in the conversation, Randall's fascination for and trust in science made him withdraw his objections. At the end of the film, a long shot of Earth from space shows the comet's impact. We see giant plumes of magma, shockwaves across the planet, debris, and massive explosions destroying Earth and all of humankind, almost. A BASH spaceship navigates its way out of the catastrophe in search of another inhabitable planet in space.

5.3.1 Inductive construction of categories as data analysis and interpretation

Despite the extinction level event (ELE), systematic data analysis as the basis for interpretation has to continue. To do so, we construct categories for a category system based on the manifest content of the previous six film segments. A category is a unit to identify, classify, and sort actions, actors, artifacts, phenomena, etc. (see also Box 5.2). For example, actors or subjects at a university can be classified as students and academics. Academics could be used as a top category for all university employees doing research and teaching. Under the top category academics, according to rank, we can classify subcategories of professors, research assistants, lecturers, and other academic staff. The latter could be further subcategorized according to pre- and postdoctoral academic staff. Functionally differentiated 'academic actions' (top category) at universities are 'research' (subcategory 1) and 'teaching' (subcategory 2). The actor category

'student' can distinguish the subcategories of 'bachelor student' (subcategory 1), 'master student' (subcategory 2), and 'doctoral student' (subcategory 3). The example category system now consists of two top categories (i.e., academic actors and actions) and associated subcategories.

For content analysis of a text or a film about academia, the different categories can be used to classify one or several sentences or a paragraph that you as a researcher identify as a unit of meaning. To classify in qualitative research is also referred to as to code. In Table 5.1, we apply the category system to code a small part of the *Don't Look Up* film script. In the example, we can apply three subcategories 'professor' (top category: 'academics'), 'doctoral student' (top category: 'student'), and 'research' (top category: 'academic activity').

In Table 5.1, the category/code 'research' has been applied several times – please be aware that the number of codings per category is irrelevant in qualitative content analysis. The 'research' category is very broad, and the coded units of meaning vary in their significance: The "estimates" (Kate) or "calculations" (Dr. Calder) of the comet's "trajectory" is 'research' done based on mathematical formulas using a computer program (e.g., "Sout"; the program(s) used by Kate, on which she "has been running [calculations] all day", and Randall

Table 5.1 Example for Coding Text for Content Analysis

2:16:01 until film end	*TELEPHONE CONVERSATION*	*Categories*
Dr. Oglethorpe:	What are the most updated estimates on its [the comet's] trajectory?	Research
Randall Mindy:	Kate.	Professor Doctoral student
Kate:	Um ... I've been running it all day. I keep getting the same result. A direct hit of Earth in six months and 14 days.	Research
Randall:	Me too.	Professor Research
Dr. Calder [NASA]:	Matthew, what are your calculations?	
Matthew:	Um, Sout program is saying six months, 14 days.	Research
Dr. Calder:	We got six months and 14 days.	
Dr. Oglethorpe:	And it's roughly five to ten kilometers wide. Which ... which, uh ...	Research
Randall:	Isn't that ... an extinction level event?	Research

Source: Film *Don't Look Up* on Netflix; authors' transcript

is/are not specified). Several mathematical formulas were presented earlier in the film, when Randall used a blackboard, chalk, and an astronomy handbook to do the calculations with the group of doctoral students. The second meaning of 'research' is interpretation of data (the comet is "roughly five to ten kilometers wide") and drawing conclusions (this is "an extinction level event"). The question mark at the end of the sentence is not considered important as it emphasizes the surprise or shock but does not question the fact.

You might have noticed that we already entered a process of interpretation when explaining the two different meanings of 'research'. This is the sixth core issue on interpretation to remember when constructing categories based on empirical data and/or theory. Now, as a researcher as well as a student, you need to decide whether to keep the rough category or subdivide 'research' into further, more precise subcategories of 'research' (see Box 5.2). Both will work for the content analysis of the *Don't Look Up* film script, which has about 30,000 words (60 pages, assuming 500 words per page; 150 pages, assuming 200 words per page), of which several thousand originate from the repetition of

Box 5.2 Category scheme and coding

Definition 'category': A category is a unit to identify, classify, and sort actions, actors, artifacts, phenomena, etc. in qualitative content analysis and interpretation. A category should be distinct, have a clear meaning, and not overlap with other categories. Category definition problems indicate that already defined categories need to be reconsidered.

Definition 'category system': The category system is the result of a category construction process. A category system can be based on categories derived from theory (deductive research approach), categories developed from empirical data (inductive research approach), or a combination of deductively and inductively constructed categories. In general, a category system is composed of several top categories and associated subcategories.

Definition 'top category': A top category is like a first order heading which indicates the overarching content of a text. A top category summarizes two or more subcategories (i.e., similar to second order headings).

Definition 'subcategory': A subcategory specifies one aspect of the top category, for example 'research' and 'teaching' as subcategories of 'academic activities of a university' (top category). A subcategory has meaning on its own and, on a more abstract level, contributes to the overall meaning of the top category. A subcategory can have further subcategories.

Definition 'code' and 'coding': To code means to attribute a category to a segment of text, for example a sentence, several

role names and scene instructions, which are of minor or no importance to the text analysis. Please be aware that we are inductively constructing categories, and the example is just a very small fraction of the entire text of the *Don't Look Up* film script.

> sentences, or a paragraph. How much text you code depends on the category-led construction of a unit of meaning, that is, coherent content in a sequence. Coding is the activity of creating order in document(s) by selecting units of meaning to answer the research question. Noncoded text segments are not considered relevant for answering the research question according to the category system. As in encryption, the coded units of meaning await decoding, that is, content analysis and interpretation by researchers.

With more text, we would recommend introducing a more fine-grained subcategorization of 'research' because you interpret the meaning of 'research' during the coding process. Sure, the more detailed coding of text will take a little more time, but it will be less time than having to go through all the coded units of the meaning of 'research' and do the meaning attribution then. In addition, the more fine-grained coding will provide more order to the data. Of course, if you decide to reorganize or add categories, it is essential to recode the already coded parts of the text. Therefore, for inductive qualitative content analysis it is advised to develop categories and test coding on key segments of a document or selected exemplary quotes (*Ankerbeispiele*).

Again, we assume that you know your data – in this case the *Don't Look Up* film script – very well, for example because you purposively selected the film as a popular document for analyzing a Hollywood-view on the debated legitimacy of scientific knowledge in society. In the process of familiarizing yourself with the data by reading the document in the sense of "reading film" (Denzin, 2004, p. 237) several times, you will identify several key segments and exemplary quotes to get started with your category system. During the reading process, please take notes in your research diary, for example copy and paste the quote (including page(s) and or timestamps), and write a few words about why you selected the particular quote and what you want to use it for. At the end of the systematic reading process, you might have collected several exemplary quotes for the same purpose, which is good. You can compare these exemplary quotes to select and define a category, compare similarities and differences of content, which might lead you to deciding on a more fine-grained categorization or construction of subcodes. For transparency reasons (see Chapter 4 on quality conventions), it is important to also document such decisions in the annex of your term paper, bachelor thesis, or publication.

Now, we should talk about Matthew in Table 5.1. Matthew is the NASA researcher confirming Kate and Randall's 'research'. According to our category system, Matthew could be coded as 'academic staff' – although not at a university but at a research facility. In the example in Table 5.1, we did not code Matthew because confirming findings about the comet is his sole role in the document/film and, for content analysis, the 'research' facts are represented in the remainder of the film by Dr. Oglethorpe, who is the NASA government liaison. The title 'Dr.' identifies him as someone who did 'research' for a certificate and, possibly, for a living for some time but now is "Head of the Planetary Defense Coordination Office in Washington D.C." (02:17:08 until end of film and p. 9 in PDF of nonofficial film script). By not coding Matthew, we will not use the subcategory 'academic staff' in our content analysis. This example shows that not all subcategories or top categories are actually useful for the content analysis and interpretation, although it seemed plausible to define them in the first place. Even if we would have coded Matthew, we would not have used the information 'academic staff' in the interpretation. The nonuse of categories initially considered relevant is not uncommon in inductive qualitative research – and also in deductive content analysis.

The example in Table 5.1 could be considered a key segment for the development of categories and to test coding. In addition, the interpretation of the key segment, which could also be considered a unit of meaning, generated a first, possibly intermediary or partial result from content analysis. Accordingly, the construction of a category system (including coding) and interpretation is both data processing and analysis. The category 'research' has helped to provide one clue to answering the research question "How is legitimate scientific knowledge produced?": Estimations or calculations were done by more than one researcher, and three researchers (Kate, Randall, and Matthew) had the same findings. In academic terms, the legitimacy of this scientifically-produced knowledge is based on peer review.

In addition to peer review, more precisely, the replication of the ELE result, scientific legitimacy is granted by rank and status. In the academic hierarchy the professor (i.e., Randall) outranks the statuses of PhD (e.g., Calder and Oglethorpe), researcher (i.e., Matthew), and doctoral student. So Randall saying "Kate" instead of answering Oglethorpe's question could be considered giving credit to Kate for having discovered the comet. Nevertheless, Randall could have remained silent and ignored the hierarchy or social expectation in the situation. So an additional, secondary interpretation is that Randell is deferring to Kate by granting her the right to speak first. Then, he confirms the doctoral student's findings ("me too") and legitimates them with his rank. The story is different for NASA director Dr. Calder, who has different things to do (e.g., serve the president (02:14:50) and justify budget cuts to NASA (02:17:36)), and does not do research on her own in (astro-)physics because she is a medical doctor

("anesthesiologist", p. 50 in nonofficial script PDF; missing in film). Accordingly, she must rely on Matthew's findings and Dr. Oglethorpe's judgment, who are her subordinates.

Now, let us return to inductively constructing categories from the transcribed *Don't Look Up* film sequences. Similar to science as an institution and the university as an organization in which 'research' is conducted, other institutions[2] also support legitimacy claims. Looking at the subjects and the institutions they represent, we can create a list of institutions for an overview:

- Kate and Randall: Michigan State University, public, top-tier, research institution.
- Dr. Frank, Dr. Inez, and Dr. Talcamont: Ivy League universities, private, elite institutions.
- Dr. Calder, Dr. Oglethorpe, and Matthew: NASA – National Space Agency of the USA, part of US public administration.
- President Orlean, Jason, etc.: political institutions (i.e., US president, the White House and Oval Office, political party, etc.).
- BASH billionaire owner Peter Isherwell: economic institution, corporate America.
- Chief editor, Adul, and Ms. Tanken: Media.

Based on this list, we have to decide whether we continue with a rough set of categories consisting of the institutions or construct a more detailed category system consisting of top categories and subcategories. For example, the category 'US political institutions' could be considered as doing politics, which is justified by the research question about the legitimacy of scientific knowledge production. Accordingly, based on the text data of the film *Don't Look Up*, the top category 'US politics' could be constructed inductively containing six subcategories.

- US politics:
    - Presidency (including Chief of Staff and others at the White House)
    - Advisors (e.g., Ivy Leaguers)
    - Public administration (e.g., NASA)
    - The ruling party (e.g., including their members and followers)
    - Top donors (e.g., BASH billionaire)
    - Voters

To apply the categories during the data coding process, you need to analyze and interpret the text and identify units of meaning. To support your decision-making it is imperative to be very clear what each category stands for. Accordingly, you can define each category in an abstract manner, for example, by addressing the concerns, functions, and means of politics as an institution:

- Politics:
  - Functions and means:
    - Government
    - Public administration (e.g., NASA)
    - Laws
    - Budget
    - Etc.
  - Concerns and means:
    - Popularity
    - (Midterm) Elections
    - Scandals (e.g., President Orlean being "caught sending pictures of [her] cooch to [her] porn star sheriff", whom she nominated for the US Supreme Court)
    - Campaign funding (e.g., by donors)
    - Etc.

In addition to guiding the coding, clear-cut category definitions facilitate further data analysis and interpretation. Some examples of concerns, functions, and means of the other institutions present in the *Don't Look Up* film are as follows:

- Economics
  - Profit (about 130 trillion US dollars)
  - Harvest rare materials (i.e., become more independent from China)
  - Jobs and income
- (Social) Media:
  - News or information
  - (Secured, peer-reviewed) fact presentation
  - Entertainment (e.g., The Daily Rip)
  - Social mobilization
- Science/research:
  - Discovery of new evidence
  - Exact calculation of ELE
  - Destroy Comet Dibiasky with how many nuclear explosives?
  - Develop and apply cutting-edge technology (e.g., microtargeted quantum fission explosives)

As you can see, the construction of categories and subcategories is an activity which is based on decision-making by researchers and students. Your decision to inductively construct a category must be justified by your knowledge about

the researched phenomenon in its social, spatial, and temporal context. Inductive research means that it is driven by the data which you process based on your existing stock of knowledge. Your knowledge could be informed by familiarizing yourself with the topic prior to starting your research, expertise about the phenomenon under study, theory, observation of similar phenomena, etc. Of course, in your decision-making what you like better or whatever feelings you may have about an issue should be irrelevant.

5.3.2 The interdependence between data processing (i.e., category construction and coding) and content analysis and interpretation

Your knowledge also grows during content analysis and inductive data processing (e.g., category construction and coding). Reading the *Don't Look Up* film script, you are aware that the academic community at large has accepted the finding that Earth lies on the trajectory of a comet. However, the conclusion is disputed: destroy the comet to prevent an ELE (e.g., promoted by Dr. Oglethorpe and Kate) or harvest the rare materials by applying "cutting-edge next generation technology" (Randall) developed by researchers from Princeton and Stanford, who are cooperating with the research team of the private corporation BASH?

In the present case, for example, due to a previous study, the construction of the category 'scientific knowledge production' is based on your knowledge that, in the US, whatever research is produced in private, Ivy League universities is worth more, that is, possesses higher legitimacy than research produced in public universities (e.g., Abbott, 2002; Stevens et al., 2008). Considering this knowledge, the top category 'scientific knowledge production' could be constructed as containing two subcategories for research institutions which distinguish between public and private facilities. In the following example, we use numbers as an option to identify the hierarchy of top categories and subcategories in your category system during the coding process:

- Top category 'scientific knowledge production' (astronomy and technical engineering)
    - Subcategory 1: 'University research'
        - Subcategory 1.1: 'Public university'
        - Subcategory 1.2: 'Ivy League, private university'
    - Subcategory 2: 'Nonuniversity research'
        - Subcategory 2.1: 'NASA public research facility'
        - Subcategory 2.2: 'BASH private research facility'

For example, subcategory '1.2: Ivy League, private university' should be applied to the statement by President Orlean: "Let's get some other people on this.

Some Ivy Leaguers. And you and I will review the NASA plans. OK?" In the section of the film transcript presented, the subcategory-led coding of Orlean's statement is justified by containing coherent content that is presented in a continuous sequence, that is, a clearly identifiable unit of meaning.

Once all text data relevant to answering the research question is coded (see Box 5.3), for its analysis and even more so for data interpretation, we recommend amending the categories with notes or definitions that support meaning making, for example, in your research journal and/or the MAXQDA software interface where you can use the memo function for the categories. With the research question in mind, for the top category 'scientific knowledge production' subcategories that support meaning making, i.e., interpretation could be the hierarchy of facts and/or legitimacy for decision-making about what to do with Comet Dibiasky:

- High (individually and as cumulative advantage):
    - Scientific facts produced by Ivy Leaguers
    - Political facts produced by president, White House administrators, party politicians, in *Don't Look Up* movement, etc.
    - Private, corporate research facts (i.e., BASH)
- Low: Scientific facts produced by professors and (PhD) researchers from a public university (i.e., Michigan State) and public administration (i.e., NASA)

Box 5.3 When are you finished with inductive category development and coding?

The straightforward answer is: it is hard to say. Assuming you have systematically constructed the category system, in general, we would recommend using the following signals as guidelines for being finished:

1. If you are going through the text segments that are not coded and realize that (a) no category could be developed from them to answer the research question, and (b) no unit of meaning can be coded based on your category scheme. This is referred to as empirical saturation in qualitative research.
2. When reading the documents, you do not generate new ideas or additional insights from content analysis and interpretation.
3. You are stuck, hopelessly (e.g., going in circles, round and round and round).

Please be assured that the requirement that *you do your best* does not mean that your data processing, analysis, and interpretation must be perfect. At some point, you also need to finish the research process, answer your research question, present results, write up, and hand the paper over (e.g., to your professor or an editor).

Such a binary high-low evaluation could be connected to the concerns, functions, and means of the institutions relevant in the film *Don't Look Up* to qualify the hierarchy of facts and/or legitimacy for decision-making about what to do with Comet Dibiasky. The structure or process of data analysis and interpretation could start with the sub-subcategories (e.g., '1.1: Public university' and '1.2: Ivy League, private university') to qualify how the legitimacy of 'scientific knowledge production' (top category) depends on the type of university where it is produced (i.e., subcategory 1). Connecting to other categories, the legitimacy of knowledge produced by Ivy Leaguers is enhanced by 'US politics' and BASH's promise of financial profit.

5.3.3 Theory to support interpretation – learning by example

We assume that the scripted film sequences and the emphasis we added has made it sufficiently clear that scientific evidence is not a fact when not presented by someone endorsed by an Ivy League university or without their confirmation. Moreover, although humanity is facing extinction from Comet Dibiasky, cutting-edge technological advancements made by Ivy Leaguers – so far fully academic and not applied under real conditions of a fast-advancing, debris-clouded comet – seems sufficient to seek political and economic profit.

Before we continue, we would like to make it explicit that the remainder of this section is not written in an explanatory handbook style. We wrote it as an example of how you could write it in a term paper and bachelor, master, or PhD thesis. For the example interpretation of the six segments from the science fiction film *Don't Look Up*, we add some more social context and theoretical considerations to discuss why a fact is only a fact if ...

To understand the legitimacy of knowledge produced by Ivy Leaguers, we need to be aware of key figures of the US higher education landscape and the role of the university in the social context of the reproduction of elites in the US. When presenting the findings to the US president and her advisors, the scientific evidence is not recognized because the discovery was made by a doctoral candidate and a professor at a public university. As part of the US elite, the president and her chief of staff only trust scientific evidence produced in private, so-called Ivy League universities (i.e., Brown, Columbia, Cornell, Harvard, Princeton, and Yale University, University of Pennsylvania, and Dartmouth College). Michigan State is not a private university but a public university, and part of what is sometimes referred to as the Ivy League Plus universities (e.g., Chetty et al., 2017; Sandel, 2020). These 69 private and public universities[3] account for about 2% of all higher education institutions in the US – the eight Ivy League universities account for about 0.2% of US higher education institutions.

In addition to social context, this is the seventh core issue: Theory is an essential tool to support interpretation. As we are working inductively, that is, a data-driven analysis of a single case, and want to broaden our

perspective to the more general societal level, theory is introduced to reflect on the empirical findings. Theory has the function of taking a more abstract or distant look at our empirical material as well as empirically grounded interpretations. Here, our science fiction finding under scrutiny is that scientific evidence is only beautiful when delivered by Ivy Leaguers and accepted by their elite kin.

Among various theories for analyzing elites (e.g., Mills, 1956/2000; Tilly, 1998/1999; for a more recent overview see Beckert, 2022), we selected a theoretical approach from the sociology of higher education. For the conceptualization of the institutionalized functions of the private Ivy League universities, Stevens et al. (2008, p. 128) used the four metaphors sieves, incubators, temples, and hubs:

1. Hubs for elites with adequate economic, political, religious, etc. background.[4]
2. Sieves for the entry and regulation of upward mobility, which can lead to elite positions.[5]
3. Incubators for the formation and coalescence of elite identities.[6]
4. Temples for the provision of legitimate forms of cultural knowledge.

For answering the research question "How is legitimate scientific knowledge produced?", the institutional function of the 'temple' is key. The 'temple's' function for the provision of legitimate forms of culturally acceptable knowledge and observable facts (i.e., the comet) is relevant, at first, as we are interpreting the beauty of scientific evidence. These Ivy League universities in an exclusive, "commercial idea system" (Abbott, 2002, p. 226) are considered by elites – such as political elites represented by President Orlean and her son, the economic elite represented by BASH's billionaire owner, and the media elite represented by the newspaper chief editor or The Daily Rip presenters – as the "model for generating new ideas – or at least new representations of ideas" (Abbott, 2002, p. 226). Accordingly, in a circular, self-perpetuating manner, the 'temple's' ideas produced by scientists in Ivy League universities are constructed and reconstructed by applying the hub, sieve, and incubator functions of these institutions, as defined by Stevens et al. (2008).

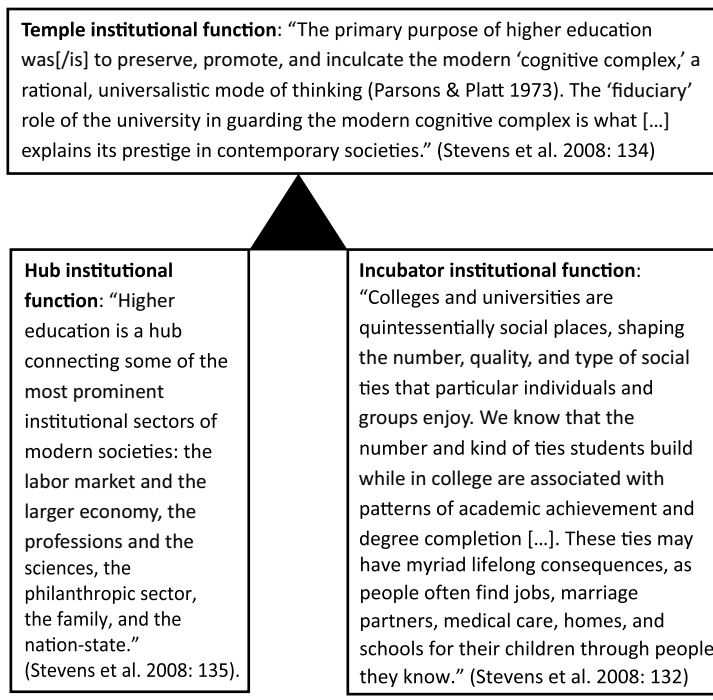
Based on the theory, and going beyond manifest content of the text in the six transcribed science fiction film segments, the latent content can be interpreted as that elite socialization in Ivy League universities – for example, of the president, newspaper chief editor, and billionaire – has established or maintained a critical stand or, even more, a functional distrust towards legitimate scientific knowledge produced by outsiders like Kate and Randall, and institutions of lower social evaluation according to the binary elite – non-elite criteria, in the present case the University of Michigan. Accordingly, the 'temple' function coalesces with the 'hub' and 'incubator' function of Ivy League universities.

For our science fiction case, the 'sieve' function is not relevant to answer the research question about knowledge legitimacy.

An additional interpretation could be that the knowledge produced in Ivy League universities is considered legitimate without critical examination. Such self-fulfilling legitimacy is beyond the question about good or bad quality science – in the analyzed science fiction case the noncritical assessment and plotted consequential failure of the high-tech produced by Nobel Prize winners Dr. Inez (*Princeton*) and Dr. Talcamont (*Stanford*) helped to destroy planet Earth.

Accordingly, three of the four institutional functions of 'temple', 'hub', and 'incubator' are part of the "protective belt" (Lakatos, 1970/1997, p. 118) of elite knowledge construction and legitimacy.[7] The Ivy Leaguers' problem shift appears progressive, at least theoretically, and seems more easily acceptable than the ELE caused by Comet Dibiasky, which is even rejected as pseudoscientific by the dominant political elites and their followers in the general US public (e.g., the *Don't Look Up* movement). Figure 5.1 presents, in a schematic way, the socially interdependent and self-relying, perpetual legitimacy of elite knowledge production, which connects the elites from politics, the economy, and science in the science fiction film *Don't Look Up*.

Figure 5.1 Temple-Hub-Incubator-Triangle of legitimate scientific knowledge.



5.4 Theory supports interpretation: concept, proposition and comparison, and typology

Theory has two basic functions for empirical research. First, theory is used to guide the entire research process from research design via data analysis and interpretation to conclusions. Such theory-led research is referred to as the deductive research approach. For example, the content analysis of the *Don't Look Up* film could have been led by the previously presented theory on elites focusing on the functions of higher education according to Stevens et al. (2008).

Second, in an inductive research approach, theory is used to reflect on empirically grounded findings. In inductive research, the reflexive use of theory is situated toward the end of the research process, that is, after securing descriptive results and before drawing conclusions. In a paper, this section will be discussion of the results. Therefore, it is a mistake to think that inductive research is atheoretical. Moreover, we knew about the theory on elites focusing on the functions of higher education according to Stevens et al. (2008) and chose not to use it to lead, but to reflect on our empirical findings. We took this decision to conduct a content analysis of a Hollywood science fiction film guided solely by our research question and not by the lens or looking glasses of theory. Of course, we could have combined the deductive and inductive approach (see Chapter 2), but we wanted to take a look with fresh eyes at the image, stereotypes, roles, capacities, etc. attributed to science and knowledge production in a major film production. Methodologically, this decision could support the production of new insights but does not necessarily have to lead to new discoveries (see also Chapter 4).

In general, theory supports the goal of empirical research to understand, explain, and interpret patterns in social behavior and processes in context. In social sciences, social behavior and processes are grasped, for example, by the "development of categories, insights, frameworks, propositions, models and theories" (Smelser, 2003, p. 654). To be sure, no one would expect from a bachelor, master, or PhD student that they develop their own theory, but you are expected to use theory according to your deductive, inductive, or combined research design.

Accordingly, one focus in this chapter is on how to use theory to grasp the results of your empirical research. The additional examples we offer are proposition-led comparison with results of similar studies, conceptualization, and developing a typology. We will explain the different theoretical qualities of proposition, concept, and typologies in the next respective sections.

5.4.1 Theoretical propositions and comparison to support answering your research question

The results of your study and the interpretation of your empirical data have to provide one, most likely more than one answer to your research question. The research question guiding the previous analysis of a science fiction film is "How

is legitimate scientific knowledge produced?" The research question contains two issues: the production of knowledge and the social acceptance or rejection of scientific evidence, that is, the legitimacy of what to do with Comet Dibiasky.

Analyzing the legitimacy of knowledge in *Don't Look Up*, it neither depends on the "beauty of science" that Kate follows her evidence, nor that "there's no luck!". Legitimacy is not about facts or scientific evidence but about social support and dominance. Stinchcombe (1968/1987, p. 162) defines the idea or proposition of legitimacy as follows:

> "A power is legitimate to the degree that, by virtue of the doctrines and norms by which it is justified, the power-holder can call upon sufficient other centers of power, as reserves in case of need, to make his power effective."

The definition of legitimacy shows that the term *legitimacy* is a word that designates a conceptual proposition for understanding, explaining, and interpreting observed empirical patterns of social phenomena.

Accordingly, scientific, solid, robust, and peer-reviewed facts about the planet-Earth-killer Comet Dibiasky and the social support that Kate and others can muster are not powerful enough. The power leading to the abortion of the mission to blow up Comet Dibiasky with nuclear bombs, the decision to harvest the rare minerals for jobs and profit, and to ignore the Extinction Level Event (ELE) for Earth is legitimized in *Don't Look Up* primarily by the elected president, her party members, business, and private university Ivy League scientists. Accordingly, the last sentence is the answer to the research question "How is legitimate scientific knowledge produced?" Of course, in a term paper, bachelor or master thesis, you would need to argue more elaborately about justifying the answer(s) to your research question and refer to empirical evidence in your descriptive and analytical discussion of observed social patterns. References should be made to your category system and coded key segments in text(s) as well as supported by exemplary quotes, for example discussed with reference to Table 5.1.

In addition to the theoretical proposition, a comparison between your empirical findings and results from similar studies supports audiences' understanding, explanation, and interpretation of the social phenomenon under study (e.g., Ebbinghaus, 2005; Mills et al., 2006; Smelser, 1976/2013, 2003). According to Tilly (1984), a comparison between similar (or dissimilar) cases seeks to generate knowledge about commonalities (i.e., universalizing comparison) and differences (i.e., variation finding) or uniqueness (i.e., individualizing comparison) of social patterns and processes (e.g., by process-tracing). The systematic consideration of the specific social, spatial, and temporal contexts assumes that comparative cases vary by some degree, and therefore the causes of effects and their origin also vary.

To make a comparison, it is first necessary to provide an answer to the question of cases being "*comparable with respect to which properties or*

characteristics, and incomparable (i.e., too dissimilar) with respect to which other properties or characteristics" (Sartori, 1991, p. 246). In our interpretation example, the case is the science fiction film *Don't Look Up*. Comparable cases with a comet (or asteroid) being an ELE for Earth are *Armageddon* (1998; director: Michael Bay; scriptwriters: Jonathan Hensleigh, J.J. Abrams, and Tony Gilroy) and *Deep Impact* (1998; director: Mimi Leder; scriptwriters: Bruce J. Rubin and M. Tolkin).

In *Armageddon*, NASA observations discover a Texas-sized comet on a collision course with Earth, with an ELE in 18 days. In *Deep Impact*, a high school student and an astronomer discover a comet on a collision course with Earth. Comet Wolf-Biederman (11 kilometers in diameter) will hit earth in one year. Neither in *Deep Impact* nor *Armageddon* is the legitimacy of the scientifically confirmed evidence contested, as occurs with the elites in *Don't Look Up*. In comparison to *Don't Look Up*, *Armageddon* is avoided by the world's best deep-sea oil driller (Bruce Willis) and his team, who plant a nuclear bomb deep within the comet. After the best training possible in the short period of 18 days, and due to their many years of practical experience, the team overcomes severe obstacles on the comet (e.g., a crash landing and debris rock storms). Nevertheless, the US hero (Bruce Willis) has to sacrifice himself to ensure that the nuclear bombs detonate.

In comparison to the failed application of Ivy Leaguers' nano-scanners and microtargeted quantum fission explosives in *Don't Look Up*, in *Armageddon*, the bomb splits the comet, avoiding the ELE for Earth. In *Deep Impact*, the mission to destroy the comet is partially successful. Encountering severe difficulties (e.g., harder than expected surface, debris, and sunlight) when trying to drill holes into the comet, the mission crew sets off the nuclear bombs, which only split the comet into two pieces. Sacrificing themselves, the crew flies the spaceship into a crater in the larger comet piece to detonate the nuclear bombs. However, the smaller comet piece (2.5 kilometers in diameter) crashes into the Atlantic Ocean. On the continents surrounding the Atlantic, a monster tsunami kills millions who did not manage to reach higher ground or shelters. In *Don't Look Up*, Comet Dibiasky (about 9 kilometers in diameter) crashes into the Pacific off the coast of Chile, and the ensuing tsunami and shock wave cause an ELE for the entire planet Earth.

In *Deep Impact*, the legitimacy of scientific evidence about the ELE is plotted as a global effort: A joint Russian-US team is sent into space to destroy the comet (please be aware that not many countries had the facilities to launch rockets into space and accommodate orbiting space stations at the end of the 20th century). Moreover, aware of the fact that the destruction of an approaching comet has never been attempted before and is highly risky, a plan to secure the future of mankind is crafted, in case the mission fails. For example, martial law is imposed in the USA, and a lottery selects 800,000 people to join 200,000 preselected individuals (e.g., artists, military, politicians, and scientists) in underground mountain shelters called the Ark. In *Don't Look Up*, only a small number of the elites close to President Orlean (n = 600) are deemed worthy to escape the ELE into an unknown future in space.

To summarize the comparison and to provide answers to the research question: "How is legitimate scientific knowledge produced?", we recommend using a table. The table does not necessarily have to be part of your term paper or bachelor or master thesis. The summary of the comparison does require you to be very explicit about the results of your interpretation. In addition, this exercise supports distancing yourself from the empirical data as well as what you think you have achieved at the end of your empirical research, i.e., the table is a method for making you reflect on your results. In Table 5.2, we put the three film titles as columns and Tilly's (1984) categories for a comparison of social patterns and processes in the rows, so we can add further lines if they are needed for analysis.

5.4.2 A theoretical concept and its subconcepts guiding the answering of the research question

Using a proposition, such as that for legitimacy, is very good, but only supports a focused theory-led or theory-reflective presentation of empirical results. We can use the same theory that helped us interpret the data to also organize the presentation of results of the transcribed sequence from *Don't Look Up* in the previous Section 5.3. Theory supports the presentation of results through its function of abstracting from empirical material as well as from the empirically grounded interpretations. Here, we use the term 'elite belief system' to describe the concept of understanding the beauty of/belief in a fact to only be a fact if it is produced or at least confirmed by Ivy Leaguers'. According to Stinchcombe (1968/1987, p. 38), a concept must define the "forces" (e.g., structures, actors, and activities) in society in general or in a certain social context more specifically. The forces at work according to the concept of the elite belief system are the four institutionalized functions: sieves, incubators, temples, and hubs (Stevens et al., 2008), which we use as four subconcepts for the theory-led understanding, explanation, and interpretation of the results of our empirical analysis and interpretation:

1. Hubs (elite belief system subconcept 1): Ivy League universities are hubs for elites with the right economic, political, religious, etc. background. As hubs, Ivy League universities are social connection points for past, present, and future elites. The central function of the hub is a self-perpetuating (almost) closed system of elite belief construction and reconstruction.
2. Sieves (elite belief system subconcept 2): Connected to the hub, the elite belief system strictly regulates entry into it and options for upward mobility, which can lead to elite positions. The sieve function assures that the elite belief system is in (mind) place at entry or established during the period of enrollment at the hub of Ivy League universities.
3. Incubators (elite belief system subconcept 3): Ivy League universities are incubators for the formation and coalescence of elite identities. If not

Table 5.2 Comparison of the Legitimacy of Scientific Knowledge in Three Science Fiction Films

	Don't Look Up *(2021)*	Deep Impact *(1998)*	Armageddon *(1998)*
Commonalities (universalizing comparison)	Comet as an extinction-level event (ELE)	Comet as an ELE	Comet as an ELE
	NASA, science, and US president and other politicians	NASA, science, and US president and other politicians	NASA, science, and US president and other politicians
	Solutions: drill and nuclear bombs	Solutions: drill and nuclear bombs	Solutions: drill and nuclear bombs
Differences (variation finding)	Legitimacy of scientific evidence produced in public university is heavily contested and, in the end, overruled	In general, no contest of legitimacy of scientific evidence; quibbles over differences in how to proceed	
	Academic solution to science fiction real world problem	Academic and practical solutions applied in science fiction problem-solving	
	Programmed and remote-operated drones (fail to) do the job (i.e., cutting-edge lab technology)	Humans do the job on site (e.g., by applying technology)	
	Competition between nations (US government and economy versus China and Russia)	Cooperation between nations	
Uniqueness (individualizing comparison)	A portion of the elites survives (n > 600)	In the USA, 200,000 elite individuals (e.g., artists, military, politicians, and scientists) are preselected for survival; similar action suggested for other countries	No special measures for elites
	Almost all humans die	Larger parts of humanity survive	All of humanity survives; only the hero dies
	Earth is devastated	Parts of the Earth are devastated	No harm to flora and fauna
	Science differentiated between Ivy League and the rest	-	-
	'Private' science, i.e., by Ivy Leaguers and commercial, is considered most legitimate	-	-
	(Social) media battle for legitimacy		
	Habitable planet in space – mind the Brontarocs![8]	-	-

already socialized by cultural, economic, politics, etc. elite identity, elite identities are forged by belonging, common experience, rituals, social networks, etc. as well as certificates issued by Ivy League universities.

4. Temples (elite belief system subconcept 4). As "Truth-Spots" (Gieryn, 2018), Ivy League universities, their labs, etc. are places that make people believe. Accordingly, the scientists working in the temples are exclusively entitled to provide 'true', legitimate forms of astronomical, catastrophic, cultural, economic, environmental, political, social, technical, etc. knowledge for the elite belief system.

Dr. Inez and Dr. Talcamont from the Ivy League temples of *Princeton* and *Stanford* do seem enamored by the academic beauty of their scientific evidence on nano-scanners, which are supposed to investigate the comet's integral structural points, and microtargeted quantum fission explosives, which are supposed to slice Comet Dibiasky into 30 smaller pieces – and so are the political elites (e.g., President Orlean and her son), the economic elite (e.g., BASH owner), and the media elite (e.g., newspaper chief editor). The we-identity (elite belief system subconcept 3), temple (elite belief system subconcept 4), and sieve (elite belief system subconcept 2) functions at work are manifest in the lines spoken by President Orlean ("we should get our own scientists on this. No offense, but you're just two people who walked in here") and her son Jason ("we had our scientists who went to, uh, prestigious schools of Harvard, Princeton, et cetera. They confirmed the data").

The elite belief system – which is also working in/on Randall ("But they've [BASH] got Gary Talcamont out of *Stanford* and Lisa Inez from *Princeton* on their team. I mean, this is cutting-edge next generation technology") – categorically distrusts (scientific, peer-reviewed) knowledge produced outside the temple of private Ivy League universities. In the present case, the non-elite spot is the public University of Michigan, which is part of the top 2% research universities in the USA. In addition, the elite identities forged at the Ivy League hubs make President Orlean, the BASH owner, and others consider the knowledge produced in Ivy League universities legitimate without (much) critical examination and real-life testing to prevent Comet Dibiasky from destroying planet Earth.

In the film, non-elite humans seem torn between disinterest and ignorance, fear of dying and protest, job opportunities offered by harvesting the rare minerals of Comet Dibiasky, by ideological distrust in science represented by the president-led *Don't Look Up* movement, etc. in the specific social context of the USA.[9] Seemingly, the science fiction elites – following their belief system – do not need to compromise in face of the ELE, just take the risk for potential profit and without a backup plan (e.g., missiles to destroy Comet Dibiasky). In case the Ivy Leaguers' cutting-edge, next generation technology fails, there is the escape spaceship bringing 600 of them to another habitat – where most of them arrive after 22,740 years of flight. Upon cryogenic defrosting and a naked descent into the habitable world, President Orlean gets eaten by a colorful,

ostrich-like science fiction beast called a Brontaroc. This closing scene of the film almost seems to be an indecorous, almost vicious retaliatory endnote to elites' responsibility for the total destruction of Earth.

5.4.3 A typology guiding the answering of the research question

A third analytical option to support the interpretation of empirical data is the construction of types and typology (e.g., Collier et al., 2008, 2012). An example of types are public and private universities, which can be further differentiated into all public and private US elite universities (e.g., Ivy League Plus) or only the eight private Ivy League universities. A typology represents a specific part of a social reality but is not necessarily fully representative. Considering that there are more than 3,000 higher education institutions in the USA (e.g., (community) colleges, four-year universities, and research universities), the 69 public and private elite universities account for 2% and the eight Ivy League universities for 0.27% of all US higher education institutions. A type (e.g., public university) is an analytical category that may become part of a typology, and a typology should be collectively exhaustive for a social phenomenon (e.g., higher education institutions).

Collier et al. (2012, Annex 1) define a typology as an "organized system of types that breaks down an overarching concept into component dimensions and types." In a typology, types need to be different in at least one characteristic. Such a characteristic can be manifest content or contingent, possibly latent content defining a dimension of a social phenomenon, which is mutually exclusive in comparison to the other types in a typology.[10] Further, Collier et al. (2012, Annex 1) distinguish between a classificatory typology – either for a deductive study or based on inductive, empirically based types – and a conceptual typology. Here, to answer the research question "How is legitimate scientific knowledge produced?", we conceptualize a typology based on Stinchcombe's (1968/1987, p. 162) legitimacy proposition: "A power is legitimate to the degree that, by virtue of the doctrines and norms by which it is justified, the power-holder can call upon sufficient other centers of power, as reserves in case of need, to make his power effective." To recall, legitimacy of knowledge is not an issue of factuality, proof, or scientific 'truth', but of dominance.

In *Don't Look Up*, dominant, legitimate scientific knowledge about the comet and the ELE is not produced by PhD student Kate Dibiasky, tenured professor Dr. Randall Mindy, or NASA researchers (e.g., Dr. Oglethorpe) but by Ivy Leaguers who are part of a corporate research group (i.e., BASH), whose billionaire owner is among the top donors to the political party President Orlean belongs to. The elites represented in *Don't Look Up* can be grasped by a typology of four elite group types. In Table 5.3, the elite group types are characterized by their primary means of domination and relevant subjects. Regarding the subject category, we want to emphasize that elite groups are not homogeneous but, to a certain extent, heterogeneous,[11] for example, Ivy League scientists and public,

top two percent research, elite university scientists. Therefore, in our example, the elite group called 'science' (type 4) would need to be separated into two subtypes which share the same means for action and subjects. However, in social reality, the *rest* of US higher education institutions would need to be considered as subtypes as well, for example, when studying the legitimacy of knowledge produced by a community college professor.

The three categories of elite group, means, and subjects plus the four elites as types represented in the previously transcribed film sequences create Table 5.3 with four rows by five columns. The typology in Table 5.3 explicates "the meaning of a concept by mapping out its dimensions, which correspond to the rows and columns in the typology. The cell types are defined by their position vis-à-vis the rows and columns" (Collier et al., 2012, p. 218).

Based on science fiction empirical evidence in *Don't Look Up*, the short answer to the research question is that legitimate scientific knowledge about the consequences of Comet Dibiasky is produced by Kate, Randall, and Dr. Oglethorpe and is confirmed by Ivy League scientists (e.g., Dr. Franks at *Columbia*). However, the ELE happens because Ivy League scientists (i.e., Dr. Talcamont from *Stanford* and Dr. Inez from *Princeton*), who are part of the BASH corporation's research group, and said private interest research produce scientific knowledge about cutting-edge technology (theoretically) allowing them to turn the threat into a promise of profit for BASH and other US firms. The promise of profit is used to create hope for jobs among people in the USA. The legitimacy of the profit-not-threat scientific knowledge is supported by President Orlean, her party, the *Don't Look Up* movement, and parts of (social) media.

Table 5.3 Typology of Elites' Legitimizing Scientific Knowledge Production in *Don't Look Up*

	Type 1	*Type 2*	*Type 3*	*Type 4*
Elite group name	Business/ corporation (e.g., billionaire BASH owner)	Media (e.g., Herald newspaper and TV show)	Politics/ politicians in democracy (e.g., President Orlean and her son)	Science (e.g., Kate and Randall (and Dr. Oglethorpe) vs. Ivy Leaguers)
Means	Money, profit, and jobs	Communication, information, and news	Votes, polls, law, and public administration	Research and teaching (incl. public science communication)
Subjects	Entrepreneur, owner, and manager	(Chief) editor, writer, presenter	Elected representatives, administrators, and consultants	Professor, researcher, and academic

Let us now take a step back and look at the basic building blocks of a typology for deductive data analysis and interpretation. A typology does not need to be fancy. It should be theoretically and/or empirically grounded, depending on whether you pursue a deductive or inductive research approach (see Chapter 2). A typology as well as a theoretical concept or proposition supports the analysis and interpretation of empirical data and the formulation of results. Theories help to understand and "explain the pattern in observations of the world" (Stinchcombe, 1968/1987, p. 5) by taking a more abstract, distant look at the social phenomenon under scrutiny.[12]

The same applies for typologies.[13] In Table 5.3, the cells constructed for types 1–4 (columns) and the categories elite group, means, and subjects in the rows are "mutually exclusive" (Collier et al., 2008, p. 157). Accordingly, the two-step analytical strategy for data analysis and interpretation consists of two options:

1. The columns guide data analysis and interpretation. Accordingly, each type is studied separately according to the categories, and then the results are systematically compared.
2. The rows guide data analysis and interpretation. Accordingly, for the categories of means and subjects the types of elite groups are directly compared, possibly in a sequence to avoid mixing categorical differences. In a subsequent step, the results of the analysis and interpretation should be analyzed and interpreted separately.

A typology represents a specific way of analyzing and interpreting data, for sure. The example typology in Table 5.3 originates in research and theories of elites. This is justified by the selected science fiction document data, the research question, and is made explicit in the text and Table 5.3. Our analysis and interpretation example meets the social sciences quality criterion of transparency (see Chapter 4). Without the explicit, transparent use of the theory-based typology, no matter whether it is used from the beginning of the research process or to reflect on inductive results, the plausibility of the analysis and interpretation can be easily doubted.

Of course, by using brief examples we can only provide an introduction on how to construct a category system for data analysis and interpretation. For more elaborate category systems, their application in coding data, its analysis, and interpretation, please see Chapter 6 for inductive qualitative content analysis and Chapter 7 for deductive qualitative content analysis.

5.5 Overcoming interpretation *Angst* by doing interpretation

In the previous interpretation examples, we selected the lines and sequences for interpretation. And we talked about science fiction and stopped interpreting once we had made our point. However, in research reality, it is only you and your big pile of documents, including all the text data. Accordingly, in the following sections, we present a scheme for doing interpretations. Let's face it:

Data interpretation can make you panic, paralyze you, or worse, both. Panic and/or paralysis when facing the task of having to do an interpretation is rooted in the mental state of *Angst*. In German, *Angst* is more severe than 'anxiety' and Wallace (1984, p. 95) suggest that *Angst* "would be better rendered by the psychiatrist's paraphrase, 'intense fear or dread lacking an unambiguous cause or a specific threat'." However, interpretation is not a monster, such as the Medusa, the emblem of *Angst* in Greek mythology with living snakes or serpents instead of hair who turned all who looked at her into stone. In ancient Greek, Medusa means guardian or protector, who is at the gates between death and life. The latter refers to the fair Medusa before her metamorphosis (Box 5.4).

Box 5.4 Medusa's hair is replaced by serpents

"Because, O Stranger, it is your desire to learn what worthy is for me to tell, hear ye the cause: Beyond all others she [Medusa] was famed for beauty, and the envious hope of many suitors. Words would fail to tell the glory of her hair, most wonderful of all her charms – A friend declared to me he saw its lovely splendour. Fame declares the Sovereign of the Sea [Poseidon] attained her love in chaste Minerva's temple. While enraged she turned her head away and held her shield before her eyes. To punish that great crime Minerva changed the Gorgon's splendid hair to serpents horrible. And now to strike her foes with fear, she wears upon her breast those awful vipers – creatures of her rage." (Ovid 1922)

So how does this make the Medusa myth a fitting analogy for the need to overcome interpretation *Angst*? You, as a student and scholar, are expected to craft the metamorphosis from data to meaning. In less aloof words: without interpretation, the meaning of data remains protected or guarded from understanding and explanation.

5.5.1 Surplus of meaning in data: symbols, context, and motivation

Staring at text data, like staring at Medusa, can turn your mind into stone. To avoid the metaphoric stone-mind, the key to comprehending the data and doing interpretation is finding a systematic way of dealing with the surplus of meaning (*Bedeutungsüberschuss*) embedded in a text. Such a surplus of meaning originates in the symbols, context, and motivation embedded in text data and the communication it stems from. In Section 2.1.1, we defined communication symbols, context, and motivation as follows:

1. *Knowledge of the symbols*: If you don't understand the symbolic meaning of characters and text, then you cannot understand the content either. This

not only applies to characters but, for example, also to emojis as pictogram symbols.

2. *Knowledge of the context*: If you understand the context – because you understand the cultural meaning – then you can interpret the symbols and understand the communication behind them. This is why it is so important in interviews, for example, to ask exactly what is meant and not to take your own interpretation of the spoken symbols in the form of language for granted.
3. *Knowledge of motivation*: Texts are usually not neutral reports that depict reality but have a goal and a purpose, which is most clear in interview situations: for example, you are asking questions and getting responses from someone you selected for the interview on purpose. This also applies to the analysis of press conferences with companies, politicians, or political parties, whose contents only becomes accessible at a deeper level if the motivation and context are also known and therefore allow for interpretation.

Knowledge about the motivation for communication addresses the issue of why communication is taking place. The why question refers to the latent content of communication and the second level of meaning. Accordingly, when starting to interpret data, the surplus of meaning also originates in the qualitative need to go beyond the question of what is being communicated, that is, the manifest content at the text level of meaning (Table 2.1). Compared to the manifest content, the latent meaning and evaluation hidden in communication is generally more difficult to grasp. It requires what is often referred to as reading between the lines. To uncover the latent meaning, that is, to get to the semantics *behind* the manifest text level, thorough analysis and interpretation of your data (e.g., in an interpretation group) and/or use of theory, as explained in the previous Section 5.4., is absolutely necessary.

Box 5.5 The limit-concept for unlimiting interpretation

"One must already have the work of myth behind one in order to be able to apply oneself to work on myth and to perceive it as the stimulus to exertion directed at a material whose hardness and power of resistance must have unfathomable origins. The limit-concept of the work of myth could be what I have called the absolutism of reality; the limit-concept of work on myth would be to bring myth to an end, to venture the most extreme deformation, which only just allows or almost no longer allows the original figure to be recognized. For the theory of reception this would be the fiction of a final myth, that is, of a myth that fully exploits, and exhausts, the form." (Blumenberg, 1984, p. 112)

5.5.2 The analytical limit-concept to get started with interpretation

Borrowing from Blumenberg (1984; Box 5.5), we suggest applying an analytical limit-concept to get started with interpretation despite the surplus of meaning embedded in text data.[14] Analogous to Medusa turning you into stone, looking at data and not being able to interpret it makes you succumb to the "absolutism of reality" Blumenberg (1984, p. 112). Accordingly, the limit-concept would affect you by the 'work' of data – the Medusa death myth and its metaphoric stone-mind.[15]

To face the Medusa's polymorphous serpent hair and deal with the meaning surplus, we suggest grabbing one meaning-resembling serpent at a time. This serpent need not be the greatest, the most promising, or the like – it should be the first meaning you get hold of to avoid a stone-mind. The key function of this first serpent is being a door opener to interpretation, that is, helping you to engage in the limit-concept of work on data to bring the data myth to an end. Once you opened the door for interpretation, you can move across Medusa's head from one serpent to the next. In doing so, the initial surplus of meaning in data is compartmentalized in several analytical parts – the serpents – which can then contribute to an overall interpretation of the phenomenon under research. In this stepwise interpretation process the Medusa is no longer the protector of meaning because your interpretation work contributes to understanding and explaining a social phenomenon.

Of course, you can start the interpretation process inductively from empirical scratch as explained in Section 5.2 for the line "That's the beauty of science, Kate. There's no luck! Follow your evidence." However, grabbing the first serpent is not a random act but a methodological approach following the three quality conventions discovery, appropriateness-transparency, and distancing and iteration (see Chapter 4). For doing interpretation, you need to be aware of your work so far, and build on what you know from the research planning process and data collection, such as

1. the research question(s) you want to answer,
2. your field notes, including hunches, ideas, etc.,
3. in the case of a deductive research design, the theoretical approach,
4. your categories and possible subcategories for coding the empirical data (see Section 5.3.1)
5. in the case of inductive research design, a theory (e.g., proposition or concept) to support interpretation of results (see Sections 5.3.3 and 5.4),
6. findings from previous research on similar phenomena, and
7. the social, spatial, and temporal context of your empirical study.

5.5.3 Distancing yourself from data by a three-step analysis process supporting interpretation of latent meanings

Please be aware that data will not allow for innumerous interpretations. Nevertheless, more than one or two interpretations of data are always possible and, therefore, expected by any audience. Interpretation is not a question of true and false, but of plausibility, transparency, and intersubjectivity – just like working on myth by meaning making or making sense of what we observe in myths. Attributing meaning to myths very much depends on your closeness and distance to myths. Here, closeness is understood both as a true believer in a myth as well as knowing myths as the subject of your research very well. Wallace (1984, p. 99) makes us aware of the fact that "myth and rationality are inseparable from each other, functionally, from the beginning: the one making room for the other to do its thing." In other words, what seems mythical might be the result of rationalization and, conversely, rationalization might contribute to myths. To separate myth and rationalization of myth it is essential to know your social phenomenon and data very well. This is what Blumenberg (1984, p. 112) expressed with "one must already have the work of myth behind one in order to be able to apply oneself to work on myth."

To avoid being trapped in close proximity to your (mythical) social phenomenon under study, distance separates the researcher on myths from the believer in myths. The understanding, explanation, and interpretation will always remain close to the myth itself, more precisely, the empirical grounding of interpretation in data as "distance presents itself in the myths themselves as the acceptance of 'significance' [*Bedeutsamkeit*]" (Blumenberg, 1984, p. 111). For researchers, the acceptance of significance of and in distant myths is not without pitfalls (*Fallstricke*). To achieve distancing from data and to avoid a serendipity of data interpretation, for a systematic content analysis of myths or any other text data, we recommend an interpretation according to units of meaning and clearly differentiating between manifest and latent meaning of content (see Table 5.2).

Miles and Huberman (1994, p. 94) define units of meaning as coherent content that is presented in a continuous sequence. A unit of meaning can be one sentence or line, such as "That's the beauty of science, Kate. There's no luck! Follow your evidence." This does not exclude the possibility that a similar or complementary content unit of meaning exists elsewhere in the text. Accordingly, a unit of meaning can be a sequence of sentences, such as the transcribed film conversations before and after the black line indicating the 'cut', that is, the beginning of a new relevant film scene.

While your understanding of the manifest content of the unit of meaning is methodologically located at the text level, your understanding of the latent content is directed behind the manifest text level and the explicit meaning of the

actual words. An in-depth analysis of content embedded in units of meaning is particularly challenging when the meaning of the information in the text data is ambiguous. Ambiguous means that the content of text data is not manifest or explicit, at first. To disambiguate the meaning of a coded text segment, for example an exemplary quote, and work on the understanding, explanation, and interpretation of meaning(s), we recommend a three-step process:

1. Comprehension by reformulating the seemingly manifest content of a coded unit of meaning in your own words:
    a) *Analysis*: This first part of comprehension analysis may seem unnecessary and a repetition of what seems explicit in the original text. However, by writing down the content of a unit of meaning, your understanding becomes transparent and intersubjective, and you avoid the risk that your understanding of a unit of meaning only sounds plausible in your head. In addition, through writing you will analyze the content to a much greater extent. Depending on the content of the original text, your reformulation of content in your own words can be longer than the original text passage(s).
    b) *Interpretation*: During and upon finishing comprehension analysis, some initial interpretations of data are possible. As your comprehension analysis is very close to the original text, the interpretation of data is as well. This should not imply that a key or fundamental insight into the interpretation of meaning might surface already during the first part of the process.
2. Description of content by further qualifying the meaning in your own words:
    a) *Analysis*: Descriptively summarize your comprehension analysis text more concisely (again) and systematize your initial understandings. In comparison to the narrative character of the comprehension analysis, you should choose a more analytical and factual writing style for the descriptive analysis.
    b) *Interpretation*: Based on comprehension and descriptive analysis, you can assume to have gained more distance from the original text and to what seemed significant in the text data. Again, key or fundamental insight into the interpretation of meaning might surface already during the second part of the process but cannot be expected.
3. Discussion of manifest and/or latent content by naming patterns, identifying types, explaining and interpreting core statement(s), as well as theory-based reflection:
    a) *Analysis*: The last part of the three-part process of understanding is the most comprehensive and demanding step. In your formulations, you must literally discuss the analyzed manifest and latent content, provide

alternative explanations, make suggestions for interpretation, and enable an imagined reader to clearly understand the empirical material and its meaning. The theoretical approach used to explain, interpret, and reflect on the text data should be suitable for the empirical data material as a whole and not just for one unit of meaning.

b) *Interpretation*: The analytical discussion of manifest and/or latent content you, first, comprehended, and then, second, described signifies the greatest distance you, individually, get from the original text. Accordingly, your interpretation could result in the definition of patterns, identification of types, etc. in a unit of meaning or, in the end, of all units of meaning.

The three-step process is very time-consuming and, therefore, only feasible for the content analysis and interpretation of short texts and for selected coded units of meaning. In addition, the three-step process is worth the time and effort to gain clarity about seemingly ambiguous meanings of text segments, that is, to make sure and erase doubt. For the application of the three-step process in an example of data interpretation, see Section 6.4.1 "Explaining and interpreting as part of inductive qualitative content analysis".

5.6 Final remarks on interpretation in classical, semi-automated, and automated qualitative and quantitative content analysis

Kracauer (1952) makes it clear that the interpretation of communication is not objective as it is carried out by subjects who possess particular knowledge and experiences, and come with their own values. In communication, these values are often not manifested but transported via evaluation words or remain hidden in the text. In the worst case, such "latent meanings not only pervade the manifest content, but also are intricately related to the objectives for which the analysis is undertaken" (Kracauer, 1952, p. 634). In contrast to quantitative content analysis, latent content can be identified and interpreted in qualitative content analysis.

In Chapter 4 of this handbook, we explained that appropriateness-transparency and iteration for distancing are the two quality conventions for interpretation. In qualitative research, the focus is on how manifest and, above all, latent content can be extracted from data. This is done by interpreting the material. Researchers must be aware that the interpretations they make are always linked to their own assumptions and their own prior theoretical knowledge as well as their everyday knowledge and their own life experiences (Strauss, 1987). As a researcher, you cannot put your mind into a *tabula rasa* state and approach the data 'objectively' (Steinhardt, 2015).

Instead, it is crucial to make your own assumptions and interpretation approaches clear (Charmaz, 2011). It is therefore important to explain why you interpreted what you did and how you interpreted it so that a person reading

your interpretations can understand them and assess them as meaningful. To achieve intersubjectivity, it is always helpful to discuss the results of interpretation with other people and to find out their interpretation, for example, which inductive categories they developed and why. In fact, for interpretative methods, interpretation in groups is central to the process. Interpretation in a group or with another person is very useful for qualitative content analysis, but also of utmost importance for the interpretation of sentiments or topics in fully automated quantitative content analysis procedures.

Notes

1 In general, in analogy to pages in documents, the reference would be to the exact time of the film sequence from the start of the film. However, the red dot on the Netflix timeline is not exact but covers a period of about one minute. Accordingly, we chose to refer to the time until the end of the film, which is presented in hours:minutes:seconds at the end of the timeline on screen at the right-hand side.

2 North (1990, p. 3) defines institutions as "the rules of the game in society or, more formally, are the humanly devised constraints that shape human interaction."

3 See *Association of American Universities*, retrieved October 28, 2024, from https://www.aau.edu/who-we-are/our-members.

4 For example, Epstein (2002, pp. 50, 55) summarizes the association of Ivy League universities with the New England aristocratic and old money upper-class White Anglo-Saxon Protestant (WASP) community.

5 According to Stevens et al. (2008, p. 129), theoretically, "schooling is central to stratification in modern societies, as the allocation of occupational positions is done largely on the basis of educational attainment. The notion that formal education might serve as a meritocratic mechanism of social mobility has been an animating ideal for many social scientists and social reformers for over a century." In empirical facts, "students from the lowest-income families are less likely to attend the nation's most selective private colleges than its most selective public colleges... . The fraction of students from families in the bottom quintile rises monotonically as one moves down selectivity tiers, ranging from 3.8% at Ivy-Plus colleges to 7.1% at 'Selective Private' colleges to approximately 21% at for-profit colleges. Conversely, the fraction of students coming from the top 1% falls from 14.5% to 2.4% and 0.4% across these tiers." (Chetty et al., 2017, p. 15)

6 Mills (1956/2000), in his book *The Power Elite*, emphasizes with the metaphor of two *Harvards* the double standards of the sieve, incubator, temple, and hub: The one for elite or old boys' networks and the one for everyone else. "That is why in the upper social classes, it does not by itself mean much merely to have a degree from an Ivy League college. That is assumed: the point is not Harvard, but which Harvard? By Harvard, one means Porcellian, Fly, or A.D.; by Yale, one means Zeta Psi or Fence or Delta Kappa Epsilon; by Princeton, Cottage, Tiger, Cap and Gown, or Ivy. It is the prestige of a properly certified secondary education followed by a proper club in a proper Ivy League college that is the standard admission ticket to the world of urban clubs and parties in any major city of the nation. To the prestige of the voice and manner, constructed in such schools, local loyalties bow, for that experience is a major clue to the nation-wide upper class that is homogeneous and self-conscious" (Mills, 2000, p. 54).

7 Lakatos (1970/1997, p. 135) defines a protective belt surrounding a problem as a "positive heuristic", which "consists of a partially articulate set of suggestions or hints on how to change, develop the 'refutable variants' of the research programme".

8 A Brontaroc is a colorful, ostrich-like beast invented for the film *Don't Look Up* (see: retrieved November 26, 2024, from https://web.archive.org/web/20220103050623/https://variety.com/2021/film/news/adam-mckay-dont-look-up-ending-spoilers-1235142363/).
9 For non-science fiction on-topic publications see Dimock and Wike (2020), Lewandowsky et al. (2016), Oppenheimer (2021), and Rekker (2021).
10 A typology can be used for qualitative and quantitative content analysis or be the (intermediary) result of both types of social scientific research. According to Collier et al. (2012: Annex 1), in quantitative content analysis, a typology can be unidimensional (i.e., in a table, "cell types are created based on a single categorical variable") or multidimensional (i.e., in a table, "cell types are created by cross-tabulating two or more variables").
11 For non-science fiction research on elites, see for example Abbott (2002), Beckert (2022), Chancel et al. (2017), Epstein (2002), Mills (1956/2000), Piketty (2020), Sandel (2020), Stevens et al. (2008), and Tilly (1998/1999).
12 Please remember that the plausibility of your analysis and interpretation depends on your choice of theory, because "what appears, concretely, to be a difference in types of scientific statements – models, definitions, propositions – simply reflects the different emphasis within a given statement on generality or specificity" (Alexander, 1982, pp. 2–3). In the previous section, we used the proposition on legitimacy formulated by Stinchcombe (1968/1987, p. 162) as an example for a specific theoretical statement. The theoretical specificity limits the interpretation of results to the definition of legitimacy. The theoretical specificity of the legitimacy proposition can be an advantage as it provides focus for our interpretation example. This specificity made it easier to select science fiction film cases comparable to *Don't Look Up* (i.e., *Armageddon* and *Deep Impact*). The systematic comparison of the three cases makes the analysis and interpretation less specific and more general, but not too general, as we had three rather different cases on the legitimacy of scientifically produced knowledge about a comet-induced ELE.
13 The conceptual typology in Table 5.3 can be used for both qualitative and quantitative content analysis. For deductive quantitative content analysis, the example Table 5.3. is a typology, "in which the cell types together form the dependent variable, and the dimensions that establish the rows and columns are the independent variables" (Collier et al., 2012, Annex 1).
14 "This point of departure is something that he entitles the 'absolutism of reality,' a situation in which 'man comes close to not having control of the conditions of his existence and, what is more important, believes that he simply lacks control of them.' Blumenberg calls this a 'limit-concept,' which while it may never have been fully present in reality is a necessary extrapolation, a 'limiting case,' which makes sense of what we do observe in myth and in the rest of human history" (Wallace, 1984, pp. 94–95).
15 Please be aware that Blumenberg (1979a, 1979b/1985) comprehends myth and metaphor in the traditional way as means to illustrate a specific reality or issue as well as an independent form of understanding.

Taylor & Francis
Taylor & Francis Group
http://taylorandfrancis.com

PART 2

Practical guide to classical qualitative content analysis and semi-automated quantitative content analysis

Taylor & Francis
Taylor & Francis Group
http://taylorandfrancis.com

6
DEDUCTIVE QUALITATIVE CONTENT ANALYSIS

As described in Chapter 2, we distinguish between two types of qualitative content analysis: inductive and deductive content analysis. In this chapter, we will now describe deductive qualitative content analysis. First, we present a flowchart for deductive qualitative content analysis. Then, we use Tinto's engagement theory as a specific example to demonstrate how deductive qualitative content analysis can be carried out using autoethnographic data.

6.1 Flowchart for deductive qualitative content analysis

Deductive qualitative content analysis is a method to analyze text data (e.g., interviews, group discussions, newspaper articles, and tweets) using a deductively created category system. Deduction is the derivation of empirically verifiable statements (hypotheses) from theory and/or empirical studies. This means that the categories are applied to the data and are not created from the data. Accordingly, deductive qualitative content analysis is used when there is a clear focus for the analysis and categories can be operationalized from theories and/or empirical studies that are based on preliminary work on the phenomenon to be investigated (Elo & Kyngäs, 2008, p. 109). Deductive qualitative content analysis therefore makes it possible to generate hypotheses and to test these using text data.

As a result, every deductive qualitative content analysis starts with a comprehensive literature review to examine the state of the art on the topic of interest (for instructions on how to examine a comprehensive literature review, see Chapter 9). It is crucial to have a good overview of the research that has been done in your field of interest, as this will help you to develop your research question and, most importantly, the deductive category system that you will use. The state of research refers to studies that have been carried out and theories that

DOI: 10.4324/9781003496397-8
This chapter has been made available under a CC-BY-NC-ND 4.0 license.

have been used to explain a certain phenomenon related to your topic. Or you may be interested in a particular theory and want to test it. Even then, it is essential that you familiarize yourself with the empirical studies that have applied this theory. Even if you want to test a theory, it makes sense to look at studies that have already worked empirically with the theory. In this way, you can find helpful information about categories for your deductive qualitative content analysis, because the theory test will take place on your qualitative data.

Once you have familiarized yourself with the state of research on the topic you want to investigate, the second step (Option a) is to develop the deductive category system. This step is specifically about developing categories that are as unambiguous as possible, free of overlaps, and assigned according to clear and comprehensible rules (see Section 6.3). Or as shown in Option b, you select the data sample and collect the data. Accordingly, Option a is best suited for the analysis of secondary data, whereas Option b is dependent on primary data. This means, for example, that you would conduct interviews before creating a category system. It is important that you have knowledge of the current state of research in order to create a guide (see brief definition in Box 6.1) for the interviews and to choose the best type of sampling (i.e., a selection of people to be interviewed that makes sense in relation to the research question). When creating the interview guide, you draw on the state of research that you have prepared in Step 1 (see Figure 6.1). You derive your questions from the theory and/or the empirical studies you have read. This means that a semi-structured guide is the best option for a deductive qualitative content analysis (see Box 6.1).

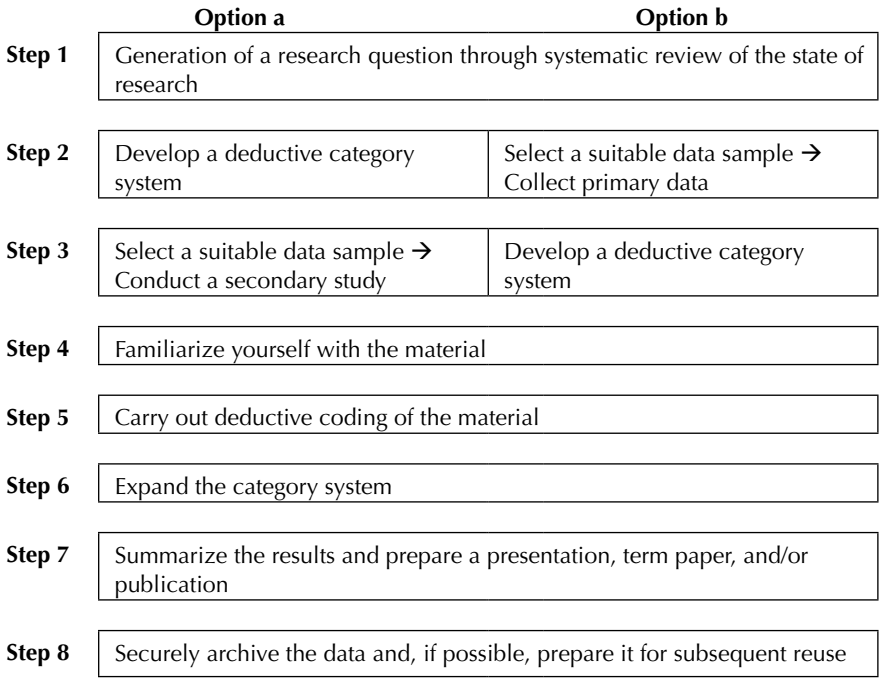
Sampling is generally understood as the cases selected to be investigated. For qualitative content analysis, these can be, for example, interviews, participant observations, newspaper articles, publications, political documents, or tweets. The cases are not selected at random, but according to their representativeness in terms of helping the researcher understand

Box 6.1 Brief definition of interview guides

There are different types of interviews, which differ primarily in terms of their structure. The structure of interview guides therefore also differs accordingly.

a) **Highly structured guide**: The questions are formulated and are always asked in the same way.
b) **Semi-structured guide**: There are core questions that are asked in all interviews, but these are not necessarily asked in the same order. In addition, it is also possible to ask questions that arise during the course of the interview.
c) **Less structured guide**: Only the introductory question is formulated, and there are specific subject areas that can be addressed. The questions arise from the conversation.

Figure 6.1 Flowchart for deductive qualitative content analysis.



the study phenomenon (Gill, 2020). This means that you should select the cases that can provide the richest information with regard to your research question (Gill, 2020). Three common sampling strategies are presented next: sampling based on specific criteria, theoretical sampling, and snowball sampling.

Purposive sampling (also called purposeful, judgmental, or selective sampling): In this sampling procedure, criteria are developed based on the current state of research and should help you select the broadest or narrowest possible sample. If, for example, the aim is to investigate how students are affected by the Covid-19 crisis in terms of their social integration at universities, then a broad sample would involve selecting students from as many different degree programs and semesters and with as many different characteristics (e.g., age, gender, physical limitations, and social status) as possible. This is because these could all be factors that have an effect on social integration. But if, for example, it isn't possible to interview so many students for a thesis or doctoral thesis, then you must choose a focus (e.g., one subject and only first-semester students) and decide whether the other factors (such as gender or age) should also be as equal or diverse as possible. If you decide on a very narrow focus (one gender, small age-range, one subject etc.) then you can make in-depth statements

about this sample, but your study will not have a wide range (see Box 6.2). If you include several factors (more than one gender, different age groups), you have greater scope for comparison, which gives your study a somewhat wider reach.

Theoretical sampling: In theoretical sampling, which originates from grounded theory (Glaser & Strauss, 2017; Strauss, 1987), a case is first selected and then directly analyzed in order to use the results to select the next case. However, this sampling strategy is not used in deductive qualitative content analysis and is only listed here for the sake of completeness.

Box 6.2 Range of your empirical results

The range of statements you can make with your results, that is, how transferable your results are to other cases, other contexts, or other research methods, varies. In principle, empirical results have a greater range when they are theoretically thought through and abstracted. The range of conclusions also increases with the amount of data, since more cases generally mean a broader representation of a social phenomenon. If the range is large, there is a high degree of transferability of your conclusions to other social phenomena, contexts, etc.; if the range is small, there is a low degree of transferability.

Snowball sampling: In snowball sampling, you start with a case that has been selected based on criteria and is believed to be a rich source of information that can be used to answer your research question. At the end of the interview, you ask the interviewee to name other people who might also be of interest in answering the research question. This is how you select the next case.

Once the sampling and data collection have been completed and the category system has been developed, the fourth step is to familiarize yourself with the material. In Option b, it may seem strange to you that you should familiarize yourself with the material that you collected yourself. However, you will find that even if you conducted the interviews yourself, that these read very differently in transcribed form than you remember. That's why we would always advise you to transcribe the interview in its entirety. You become familiar with the material by asking questions such as "Who is telling? Where is this happening? When did it happen? What is happening? and Why?" (Elo & Kyngäs, 2008, p. 109). It makes sense to write down your initial ideas immediately in order to document your first analytical impressions.[1] In the methodology literature, familiarizing yourself with the material often goes hand in hand with summaries of the case (Kuckartz, 2014). However, this seems very time-consuming to us and is not necessary for a deductive qualitative content analysis.

Once we have gained an overview of the material, we start with the deductive coding of the material (Step 5). We will describe exactly how this works in detail using an example in Section 6.5. Basically, the coding process consists of assigning the available text material to the categories. The QDA-software

MAXQDA can be used for this (which we will use later), but there are also other QDA-software such as Atlas.ti or NVivo, which have the same functions.

In deductive qualitative content analysis, top categories are developed from theory and/or empirical studies and usually also a level of subcategories. Since theories are always being developed through new empirical cases, it can be assumed that inductive categories (usually subcategories) will be added to the deductive categories. Inductive categories are categories that are developed from the material and extend the theoretical assumptions accordingly (see Chapter 7). However, this should only be done if the subcategories serve to answer the research question. It is unlikely that new top categories will emerge. The category system in its final form is therefore determined only at the end of the coding process, that is, when all the material has been coded. As this is an ongoing process, it makes sense to check at the end whether the coded text passages, especially from the material coded first, still fit the categories and subcategories. In other words, you check the fit between the text passages and categories again once all of the material has been coded using the deductive category system. In a deductive qualitative content analysis, it is normal that text passages are not coded. These uncoded passages are reviewed at the end of the coding process and discussed for their relevance to answering the research question. It is also normal that there may be categories that you have developed from theory to which no text from the interviews can be assigned. This is exciting for the theoretical discussion, as there may be a deviation from the theoretical assumptions.

6.2 State of research and research question: Step 1

In the following, we would like to provide you with instructions for deductive qualitative content analysis and will give a specific example to show you how category systems are developed and coded. The first step is to generate a research question derived from the existing state of research. Specifically, our example examines how students experienced their initial time studying in the second Covid-19 semester (winter semester 2020/21). The beginning was difficult as it was not possible to enter the social space campus because the universities were all closed at the time (Steinhardt, 2021). All teaching was switched to online teaching, which meant that many activities were not possible such as entering a physical seminar room or lecture hall, meeting fellow students on campus, getting to know each other during orientation week, or having a casual conversation in the hallway, in the library, or with the person sitting next to you. And it was no longer possible to have casual conversations or informal exchanges with lecturers. All of this can be categorized as interaction at the university. But how did this social interaction take place for students in their first semester who did not have the opportunity to use the physical space of the university? This is the research question we want to address later.

For the deductive qualitative content analysis, we conducted a systematic literature review (see Chapter 9) in 2021 to look at studies that address the

question of how students dealt with the conditions of the Covid semester in Germany. We found various studies that conducted surveys of students and focused on study conditions (Arndt et al., 2020; Händel et al., 2020a, 2020b; Kreulich et al., 2020; Lörz et al., 2020; Traus et al., 2020). These studies show that students missed interacting with each other and with the lecturers most of all. But they also missed the university as a social space itself. We also found studies in which lecturers had the opportunity to comment on and report or reflect on the conditions of teaching during the Covid pandemic (Arndt et al., 2020; Autor:innengruppe AEDiL, 2021; Kreulich et al., 2020). The studies found are empirical summaries that primarily provide insight into the situations and challenges posed by the changed conditions during the Covid pandemic. Theoretical reflections on interaction and the university as a social space can be found in a related article by Steinhardt (2021). This study discusses Tinto's (1975, 1997) student engagement theory and uses the theory to explore how a social space can be created for students in online teaching (Steinhardt, 2021).

We decided to apply Tinto's theory (1975, 1997) as it encompasses the idea of social space. In the context of higher education research, Tinto's theory is a widely used approach (Braxton et al., 2000) to explain why students drop out and to provide ways that higher education institutions can counteract this phenomenon. The central starting point is that the tendency of students to drop out varies depending on how actively they are involved in their university and how well their goal commitment fit with the institution they are at. Rather than repeat Tinto's theory, we will focus on the finding that when students are more involved in university life, they acquire more knowledge, develop their skills, and are less likely to drop out (Tinto, 1997, p. 600).

We know from various studies conducted in Germany that the institutional involvement and goal commitment of students differs based on their social, institutional, and academic involvement (Dahm & Lauterbach, 2016; Müller & Braun, 2018). Accordingly, it is assumed that a student's tendency to drop out is reduced if they are more actively involved in university and university life, for example, through peer groups or interaction with lecturers. The university as a social space plays a particularly important role as a physical space in which social exchange (e.g., conversations in the hallway or getting a coffee together), interaction with other students (e.g., studying together), and interaction with academic staff and professors are possible. As a result of the Covid pandemic and the closure of universities, this social space as a physical space was lost. We are therefore interested in what aspects of the (missing) social space students describe, what they are lacking, and how they themselves (don't) feel integrated.

Drawing on Tinto's theory and the results of the empirical studies lead us to the following assumptions. We assume that students have changed their perception of university social life as a result of the elimination of the physical space. First, we understand university life to mean interaction with fellow students and lecturers, as well as interaction in the seminar rooms, but also participation, for example, in a department or in university groups such as student councils. Second, social space is crucial for university life, which is why it is of

interest to what extent the lack of social space (e.g., the library or other study spaces) plays a role in getting started at university. All of these were factors that were eliminated during the Covid pandemic, which is why it is crucial for the research question to analyze whether the students mention the lack of physical space and what significance is given to the elimination.

6.3 Develop a deductive category system: Step 2

Now that we have grounded our research question in the theory, we can derive categories. It is essential that the categories are clearly distinguishable from one another and have clear definitions (Graneheim et al., 2017; Mayring, 2021). We already saw in the previous section that there are two central areas: interaction and social space. These will also become our top categories, and in a second step they will be assigned subcategories, which must be clearly distinguishable from one another.

For deductive qualitative content analysis, we use MAXQDA to support our work (see note in Box 6.3 and Section 1.3 on copylefts and copyrights). You can use MAXQDA (as well as the other QDA-programs like atlas.ti or NVivo) to digitally organize your files and data and to code text material such as interviews, group discussions, and documents, but also videos and images. MAXQDA also has an interface that allows you to load tweets directly into the program. It is licensed software, which means that if it is not available as a campus license at your university, you will have to purchase a student license. There is also a teaching license for MAXQDA so that you can work with the program in a seminar for one semester free of charge. The program runs the same on Windows and Mac computers.

Box 6.3 Note about MAXQDA 1

MAXQDA has a good manual and online tutorials that explain how to use it step-by-step. This is why we have not included our own instructions here. You can find the manual on the MAXQDA home page under "Online Manual." www.maxqda.com/

Box 6.4 Note about MAXQDA 2

It is important to know that categories are referred to as codes in MAXQDA. This is misleading since a code is actually the coded text passage in qualitative content analysis.

The next step in our procedure is to develop the category system (see note in Box 6.4). Since MAXQDA allows you to import an Excel table directly as a category system, we created Table 6.1 in Excel. To do this, the Excel table must

Table 6.1 Deductive Category System

Code	*Memo*
Interaction with fellow students	First top category
Interaction with fellow students \ Peer group	Students report contact with their peer group, a lack of contact with their peer group, or that they have not yet had the opportunity to build up a peer group as they have only just started studying. A peer group is a social group that students feel they belong to.
Interaction with fellow students \ Study group	Students report that they have set up study groups, that they miss having study groups, or that they can't set up digital study groups. In contrast to a peer group, a study group is set up for a specific purpose such as preparing for a presentation or an exam.
Interaction with fellow students \ Social life	Students report, for example, that there aren't any student parties or activities planned by the student council.
Interaction with fellow students \ Conversations in the hallway	Students report that they miss having conversations in the hallway and casual conversations, for example, before the start of a seminar or at the library, where they could talk about their seminars with fellow students. It is also possible that students would report on having casual conversations in breakout sessions or other digital formats.
Interaction with lecturers	Second top category.
Interaction with lecturers \ Availability	Students report on their contacts and the availability of their lecturers, for example, via email or Zoom office hours.
Interaction with lecturers \ Communication	Students report on the (lack of) communication with lecturers, including information given or missing about the courses.
Interaction with lecturers \ Integration in the context of seminar	Students report on how they were involved in seminars, for example, in Zoom sessions. Or they report that the lecturers expected them to simply work through a set of tasks.
Interaction with lecturers \ Taking time in seminars	Students report that lecturers asked them about the challenges they are facing, for example, regarding caregiving responsibilities, a poor Internet connection, or other challenges.
University as a social space	Third top category
University as a social space \ Study spaces	Students describe the current situation with regard to study space. This can include both the lack of adequate opportunities to study at home and the lack of opportunities to use study spaces at the university.
University as a social space \ Library	Students report on the effects that the loss of the library as a social space has had on them (The problem of not being able to access literature is not relevant here).
University as a social space \ Social life in university spaces	Students describe the lack of social life at the university, for example, the lack of parties at the university. Please note: It is the loss of physical spaces that is important here and not the interaction with fellow students ; see the category "Interaction with fellow students \ Social life."
University as a social space \ Work	Students report on their work at the university, for example, as staff in the cafeteria or as student assistants.

contain the word "Code" as the first heading so that MAXQDA recognizes that the category names are found underneath. Subcategories can be inserted in the "Code" column using a backslash (i.e., \). The second column must contain the heading "Memo," below which the definitions of the categories are listed. To start the import, select the "Import Codes and Memos from Excel Spreadsheet" function on the "Codes" tab in MAXQDA. MAXQDA then automatically creates the categories and assigns the corresponding definition under "Memo." It is important to note that the category name must not be longer than 63 characters in total (including spaces); otherwise, it will be truncated. Once you have imported the category system, you can open the properties window by right-clicking the code name. You can then assign colors for the different categories. Assigning colors makes it easier for you to have a good overview of your categories and coded text passages. If you code your material, you can also add exemplary quotes (*Ankerbeispiele*) under Memo to make the definition of the categories even clearer. Exemplary quotes are text passages that represent the category or subcategory in a particularly clear way (Mayring, 2021).

As can be seen from the deductive category system developed here, there are three top categories and one level of subcategories. A second level of subcategories could be added. However, as we do not yet know at this stage how the students will assess the aspects, we will leave this open for the time being – and this will allow us to be able to better compare the different assessment levels with each other. The second level of subcategories would then be added on an ongoing basis in regard to the data.

As you can already see from the definitions, in this example we are not only concerned with manifest content, but also with latent content (Elo & Kyngäs, 2008, p. 109). If you opt for a deductive category system, you must also decide whether to code only manifest content or also latent content. Manifest content is described and can therefore be understood directly from the text, whereas latent content must be interpreted (Graneheim et al., 2017).

An example of manifest content would be the following: "I wasn't able to start a study group during the semester." This would be assigned to the deductive category 'Interaction with fellow students \ Study group'. In contrast, latent content requires you to interpret text passages (see Chapter 5). It is important that latent content is also based on the text and is not 'wild' speculation. An example of a latent text passage would be as follows: "I didn't know what to do during the lecture." From the context of the passage, it could be interpreted that the uncertainty arose because of a lack of information provided by the lecturer. This passage would then be assigned to the category 'Interaction with lecturers \ Communication'.

6.4 Select a suitable data sample → Conduct a secondary study: Step 3

As shown in the flowchart, there are two options for deductive coding. For our example, we opted for a secondary study of text material, which means that

we draw on data that was not collected for this deductive qualitative content analysis. This is because it was important to us to show you the challenges you face when you carry out a secondary study. When you collect primary data, you can decide which questions you ask in interviews, for example. You would just need to familiarize yourself with the topic and formulate key questions based on theory or studies that have already been conducted. These key questions can then also become the top categories of your deductive category system.

In the following example, we have decided to analyze the same material as we do in the inductive qualitative content analysis (see Chapter 7). This will allow us to show you how different inductive and deductive coding can be and how the category systems and coding lead to different results.

The selected data is an autoethnographic record, commonly known as journal entries (or diary entries), of the second online semester caused by the Covid pandemic. The teaching and learning research project of an autoethnography titled "Two Weeks of Studying in Winter Semester 2020/21" was part of a research methods lecture in a bachelor's degree program at a university in Germany. The content-related focus of the autoethnography was how the second Covid wave, which started at the beginning of November 2020, shaped studying in general and for new students the start of their studies. The research question guiding the study for the students was as follows: "How are you organizing your daily life as a student at the beginning of the online semester 2020/21?" Over a period of 14 days, students were asked to write about 10 autoethnographic records. The students received an example text of an autoethnographic record from another research project that served as a guide, and this was supplemented by methodological literature, explanations of autoethnography in a video, and guidance from the lecturer.

In a secondary study, you do not initially know what is contained in the material. This means that it is quite possible that you will not be able to answer all of the questions that you ask based on the state of research. Central to secondary study is understanding the context of the data. For this reason, repositories that make qualitative data available for secondary studies always provide very detailed data descriptions. These usually include the following points (Steinhardt & İkiz-Akıncı, 2020 as an example of a German research methods report):

Organizational framework

- Description of the context of data collection such as a research project or, as in our example, a teaching and research project

Methodological approach

- Data collection method, for example, narrative interview, guided interview or, as in our example, autoethnography
- Sampling based on criteria, theoretical sampling, or snowball sampling (see Sections 3.2.1 and 6.1)

- Method used to analyze the primary data, that is, to what extent and with what results the data has been analyzed
- Data protection issues: whether consent has been given for secondary use

Preparing the data

- Preparing the text material, for example, the transcription rules for interviews Anonymization (see Section 6.2.2)

6.5 Familiarizing with the data: Step 4

Now that we have clarified the data we are using, we will familiarize ourselves with the material. As described, this is an autoethnography written by a student. This means that the student describes daily student life and issues related to daily student life in the ten entries we are looking at here as an example (see Table 6.2). The autoethnography is completely anonymized (see Section 7.2.2), and we have consent for its use in this book.

The first step of familiarizing is to read through all of the material to get an overall impression. As you would expect from journal entries, the texts are structured according to the student's daily routine and focus on conditions related to studying. In Table 6.2, the student's text is divided into text units of meaning for each day, which can be used as a reference (i.e., paragraph + number) or for an exemplary quote when analyzing the text.

Miles and Huberman (1994) define a unit of meaning as a coherent description of a course (e.g., lecture or seminar), an action, or a context of action. A new unit of meaning was created in Table 6.2 if, for example, a new part of the day that included a new action began and the action changed in a clearly recognizable way such as the following: Getting up early because of nervousness on the first day of university (paragraph 1/1), making coffee (paragraph 1/2), and drinking coffee at the computer while checking emails (paragraph 1/3). In contrast to the first day of university, the three text elements on the second university day are no longer relevant as individual actions, but only in combination as a single unit of meaning, which is also shown by the coherent presentation in paragraph 2/1. It is also clear that, for example, a lot happens on some days and therefore more units of meaning can be identified in the autoethnographic text (especially day 1) and that on other days less noteworthy things happen in students' lives, which is why only four units of meaning can be identified (e.g., days 5, 6, 8, and 9). As you can see, creating units of meaning is based on topics (e.g., actions) and must be well justified for third parties, but it is not complicated science.

The entries you read previously describe when the student got up, when and what they studied for class, and what tools they used to study online. The student also reports on interactions with other students, whereas interactions

with lecturers hardly ever occur. The data also reveals two aspects that are mentioned at a high frequency in students' autoethnographies.

1. *Interactions with parents and siblings* are also described as a result of the Covid pandemic and associated online studies at home: "I went down to the kitchen and made myself a coffee, as I do every morning. My mom was still in the kitchen and asked me what my first lecture was going to be. I briefly explained that it was the [major] lecture and that it started at [morning]" (paragraph 1/2).
2. There are descriptions of the process of studying and being a student such as "I then tried to log in to Moodle to have another look at what to expect in the lecture today. I had already read through the text on the weekend, but to be on the safe side I checked again" (paragraph 1/4).

As these aspects have little to do with our research question, we will initially not consider them in the deductive coding.

Table 6.2 Autoethnography: Commencement of Studies in Winter Semester 2020/21

Paragraph	*Journal Entry*
1	**Monday, [month] [day], 2020**
1/1	My alarm clock rang at 7:30 am, and when I got up I didn't feel so well rested. I was nervous because today is my first day at university and I couldn't sleep as well as usual.
1/2	I went down to the kitchen and made myself a coffee, as I do every morning. My mom was still in the kitchen and asked me what my first lecture was going to be. I briefly explained that it was the [major] lecture and that it started at [morning].
1/3	With my coffee in hand, I went up to my room, turned on my laptop, and checked my emails. The emails included several notifications about the places I had received in the modules I had enrolled in for this semester.
1/4	I then tried to log in to Moodle [online teaching-learning platform] to have another look at what to expect in the lecture today. I had already read through the text at the weekend, but to be on the safe side I checked again.
1/5	I then put on some nicer clothes and got ready to take part in the Zoom meeting.
1/6	When it was [morning], I wanted to log in to Moodle to get to the Zoom meeting link but I couldn't because all of the new students were also trying to log in and Moodle had frozen up. I'm in a WhatsApp group for the first-year students in our program and one of the students had saved the Zoom link and sent it to our group, so we were still able to join the Zoom lecture on time.
1/7	My first Zoom lecture was about autoethnographies, and we were given the task of writing one ourselves. We are supposed to record 10 days for an autoethnography and really try this experiment.

(*Continued*)

Table 6.2 (Continued)

Paragraph	*Journal Entry*
1/8	After this lecture, my university day was already over as I only had one lecture on this day.
1/9	At lunch, I talked about the first lecture and how interesting it was.
1/10	In the afternoon, I decided to start by sorting and organizing everything I've received so far in terms of texts, assignments, and information for the modules I'm taking at university. I set up a subfolder for each module and printed out and organized the literature I had already received. I then decided to read the next text for another module to prepare myself a little.
1/11	After dinner, I decided to read through the examination regulations because you should internalize these so that you are aware of all your rights during the examination phase.
1/12	At 11 pm, I decided to go to bed, but before I finally turned off the light, I checked my schedule for tomorrow in the [university] app on my cell phone.
2	**Tuesday, [month] [day], 2020**
2/1	My alarm went off at 7:30 am. I grabbed my phone and checked to see if I had received any emails and opened my schedule on the [university] app to see what lecture I had first today.
2/2	Then I went down to the kitchen and made my coffee before I went back upstairs to turn on my laptop and check Moodle to see if I'd received a notification and if there was anything new.
2/3	Then at [morning] my first lecture started: an introduction to [major]. I had already read through the lecture text the evening before and was therefore able to follow the lecture well.
2/4	After these 90 minutes, I didn't have any more lectures until [afternoon], so I focused on something else and continued to search the Internet for apartment ads and replied to some of them. I'm still looking for an apartment right now, and the current situation with the Covid pandemic is making it more difficult than I thought.
2/5	Around [afternoon], we (the first-year students) noticed that we didn't have the Zoom link to our lecture for [afternoon]. Shortly before the lecture started, we realized that this lecture in [minor] was to take place asynchronously. This means that the lecturer uploads the lecture to Moodle, and we can watch it when we have time. I decided around [early evening] to watch half of it, that is, the introductory part and a bit of the first topic we're going to cover.
2/6	After this introduction, I ate something and prepared for the seminar, which took place at [evening]. The seminar started at [evening].
2/7	I then put my notes away and joined a Zoom event ... [of the student group] [from country] in [city name], who welcomed the first-year students, introduced [their student group], and said they would be happy to have us as members.
2/8	I went to bed at around 11 pm after looking at my schedule and setting my alarm for 7:30 am.

(*Continued*)

Table 6.2 (Continued)

Paragraph	*Journal Entry*
3	**Wednesday, [month] [day], 2020**
3/1	My alarm clock rang at 7:30 am. I got up, not yet fully rested, and went to the kitchen to make myself a coffee. I sat down at the kitchen table and talked to [sibling] about my first two university days. He's currently on fall break and still thinks it's weird that we don't go to the same school anymore.
3/2	After I finished my coffee, I took a water bottle up to my room, sat down at my desk, and turned on my laptop to take part in the Zoom course that started at [morning]. In the [major] preparatory course, we had talked about what to expect in the coming semester.
3/3	Afterwards, we had the seminar for [major] that started at [morning]. At lunch, I talked about my first university week and about the fact that I'm going to have the practical course group for the [additional course outside of field] module on Thursday and that I really want to watch the lecture for assignment [B], even though it will be about assignment [A], but if I have any questions, then I could ask them.
3/4	After dinner, I pulled myself together and decided to start the lecture today after all so that I could make better progress tomorrow. I then got out my schedule and looked at what else was due and what needed to be done soon.
3/5	I find it is a big change from my [school] and the long vacation, to start studying again now, but I am really excited about what lies ahead and I'm hoping that I will be really happy with this new bachelor's degree in [major].
3/6	I went to bed at 10:30 pm and didn't set an alarm clock because I can sleep in tomorrow and the practical course group isn't until the afternoon.
4	**Thursday, [month] [day], 2020**
4/1	I slept in until [morning] because there is nothing on our schedule, but we were told on Moodle that we will have the practical course groups for the module [additional course outside of field]. I am in practical course group [A] and therefore have this on Thursday [afternoon].
4/2	After getting up, I went to the kitchen and made myself a coffee like I do every morning. My [sibling] was already awake, and we talked about what we were going to do today. I then briefly explained to him that I wanted to get straight to the practical course for [additional course outside my field] because first I wanted to be well prepared, and second, I still had to take care of the lecture and practical course [B] for Monday.
4/3	When I got back to my desk, I turned on my laptop and checked my emails, both my personal and [university] emails. There was news about a viewing for an apartment in [city name]. After contacting them briefly, I had set up an appointment for a viewing on Saturday.

(*Continued*)

Table 6.2 (Continued)

Paragraph	*Journal Entry*
4/4	Then I checked Moodle to see if there was anything new and then sat down to the lecture for [additional course outside my field], which really takes a lot of time as I have no previous knowledge in this subject. Throughout the day, I noticed that I kept thinking about this because I really have difficulties with the tasks we are given. To get a better understanding, I've ordered the book for [additional course outside my field], and I hope that it will provide the help I'm looking for.
4/5	In the afternoon, I went over my first exercise again, which we had already handed in on Monday in order to be prepared to take part in the practical course group. At [afternoon] then, this course started; at the beginning of each practical course, [number] student numbers are drawn, and these students are then [tested] at the end of the hour. My number wasn't drawn this time, and I was kind of happy about this as I don't feel so confident in the material yet.
4/6	Then my study day was over, I closed my laptop and spent the rest of the day doing other things that I hadn't had time for during the week because of the new focus of my daily life, my studies.
5	**Friday, [month] [day], 2020**
5/1	I don't have any lectures on Fridays and can do follow-up work for the modules I am in, but I can also do some preparatory work and print out and organize all the texts I've received so that I have a clearer view of the next week and what's coming up. So, I got up a little later than usual because I'm finding it hard to get up earlier again after the long vacation, even though I worked during this time.
5/2	At [morning] I made myself a coffee, sat down at the kitchen table, and opened Moodle on my cell phone to see if there was anything new. After going through my emails, it was already lunchtime.
5/3	At lunch, I talked with my family again about the viewing appointments I had received. Tomorrow, we want to go to [city name] and look at the three apartments because so far the search for an apartment has been difficult as a result of the pandemic, which means that as a foreigner from [name of country], I'm only allowed to be in Germany for a certain amount of time. I then got out the documents I wanted to take with me to the viewing, wrote down all the addresses, and to make sure I had an idea of the time plan, I looked at the routes to get there. In the afternoon, I continued to browse the real estate sites on the Internet, but with no success, so I hoped to get lucky with one of the three upcoming viewings on Saturday.
5/4	The book that I had ordered for the module [additional course outside my field] was also delivered today, so I immediately read the first two chapters in order to get a better grip on practical course 02. In the evening, I checked Moodle to see if anything had changed during the day, set an alarm clock to get up early and go to [city name] for the viewing appointments so that I can move to [city name] for my studies.

(*Continued*)

Table 6.2 (Continued)

Paragraph	*Journal Entry*
6	**Monday, [month] [day], 2020**
6/1	This Monday marks the start of the second week of my studies in [major]. My alarm went off at 9 am and I got up, made myself a coffee in the kitchen, and immediately sat down at my laptop to watch the video on Moodle for my first lecture as I hadn't found the time to do so on the weekend. It turned out that this wasn't the best idea because I was causing myself unnecessary stress.
6/2	In the lecture, we talked briefly in a breakout session about the past week and our experiences with regard to our autoethnography task and discussed how we can record our information in order to write it down afterwards. It was interesting that everyone really has their own approach, and we then talked about the fact that it is sometimes difficult to reach the specified word count because our daily life in terms of studying is limited on the one hand because of the pandemic, that is, that it is an online semester, but also that it feels monotonous because we can't do anything else because of the pandemic such as getting to know fellow students, which is normally a part of daily student life.
6/3	In the afternoon, I received the final rental contract for my first apartment, which will allow me to move to [city name] at the beginning of next year and continue my studies there in the hope of a normal and non-digital semester at [university] in [city name].
6/4	Then I watched part of the lecture of the [additional course outside my field] module and took notes. At the end of the day, I watched the podcast for another lecture and was done with everything related to university that day. I set my alarm clock, looked at my schedule, and went to bed.
7	**Tuesday, [month] [day], 2020**
7/1	My alarm rang at 7:45 am. I woke up and made myself a coffee. Then I sat down at our big table in the living room and opened my laptop to check my emails and look at Moodle. Our professor had decided to start the lecture a little later, at [morning], so I was able to get everything ready and take a quick look at the previous class before we then started at the set time.
7/2	After the lecture, I sat down to look at the rental contract for my new apartment and read it thoroughly before I signed it and took it to the post office to send it to the landlord in [city name].
7/3	At lunch, we talked about what I still have to do today for university and what things I can already take care of for my courses, but also for my apartment.
7/4	Like every Tuesday, I don't have any classes until [early evening] because we get the lecture for [minor] uploaded and can look at it on our own and in this way structure our daily life as a student a bit ourselves.

(*Continued*)

Table 6.2 (Continued)

Paragraph	*Journal Entry*
7/5	Unfortunately, it didn't work so well today because it was online on Moodle a bit too late, and it wasn't possible to look at the lecture before the seminar because the lecture was too long. We had also found out that we could ask the professor to upload the slides to Moodle earlier, but somehow that didn't work. Our seminar at [early evening] ended a little earlier because it was a kind of introduction to [major].
7/6	So I had a few more minutes before I attended another Zoom meeting organized by [student group] from [city name]. This involved auctioning off the [first-year students] as a way for them to be accepted [into the student group] – it was a lot of fun and you got to know a lot of new people. As a result of the circumstances during the pandemic, you unfortunately don't get to know that many people in this first semester, which is entirely online, so I was pleasantly surprised that so many online meetings were planned. This makes it possible to at least make some connections. This auction went on a little longer, so I didn't get to bed until after midnight and just quickly set my alarm clock.
8	**Wednesday, [day] [month], 2020**
8/1	My alarm went off at 7:45 am and I hit snooze and then didn't wake up until five minutes before the lecture, so I had to hurry and quickly make myself a coffee and sit down in front of my laptop. But I was still in time for the lecture and didn't miss anything.
8/2	I then had a lecture until [noon], and at lunch everyone asked me about the event I attended yesterday evening. I briefly told them what I had to do... . We also talked about the study grant that every ... student can receive if they submit an [application]. But to apply for this, I need several documents, including the rental contract, which I had only sent off this week and hadn't received back yet.
8/3	After lunch, I didn't have any more lectures or seminars, so I was free in the afternoon and could do some preparatory and follow-up work for my courses. I had decided to read through some of the texts and finish watching the [additional course outside my field] lecture that was uploaded to Moodle. The [additional course outside my field] module had turned out to be more difficult and a bigger challenge than I had expected. I kept finding myself thinking about this module throughout the day, so in the evening I decided to talk to my partner for assignments in the practical course group about it. She agreed and said that she found the rapid increase in expectations in the exercises a bit too extreme.
8/4	By the end of the day, I had had enough of university stuff and had decided not to read any more texts, so I just made myself a to-do list for tomorrow and set my alarm clock to get up in time so that I could get something done in the morning.

(*Continued*)

Table 6.2 (Continued)

Paragraph	*Journal Entry*
9	**Thursday, [day] [month], 2020**
9/1	My alarm went off at 9:30 am and I got up to make myself a coffee. I sat down at the kitchen table and looked at my messages and checked my emails on my phone. As I had already made my to-do list yesterday, I knew exactly what I had to do today. So I went upstairs and sat down at my desk. I tidied it up a bit so that I had enough space to work productively and sat down and opened my laptop.
9/2	The first thing I did was to read through some texts and use the book for the [additional course outside my field] to prepare for the afternoon practical course session. Somehow this module takes up most of my time and I would actually like to spend more time on the other modules, also when I was following up on the various modules for [major], I realized that I already know a lot of the methods described as I had [name of school] [main subject].
9/3	In the afternoon, I printed out the missing texts as I like to underline everything I find important in a text, and I don't have a tablet and reading on a laptop is quite difficult in my opinion. At [afternoon], we had the practical course, and this time my student number was on the list of [examinees], which made me a bit nervous as I had some difficulty with the exercise this time. [The exam] went well though and I passed it. I was asked a few comprehension questions that I had no problem answering as he had addressed these in class and I had made notes on the explanations.
9/4	After this practical course, we had dinner together as a family. I told them about the [exam] I had passed and what these plus points mean in terms of the written examination. Before I went to bed, I didn't set an alarm clock because I don't have any lectures or other scheduled university courses or events on Fridays.
10	**Friday, [month] [day], 2020**
10/1	I got up at 10 am and made myself a coffee and a snack. Then I went upstairs and completely cleaned up my room before I sat down at my desk to do things for university. I find that it's easier to concentrate when things are tidy and not disorganized. Somehow it seems more structured, and I make better progress with what I want to achieve on the day.
10/2	So I opened my laptop and checked Moodle to see what was new there, and I logged into my [university] email to see what emails I had received. Once I had a rough overview, I started preparing for Monday and had a look at what was coming up in the week ahead. I quickly realized that I still had to do some work to follow up on my last classes and that at the same time I also had to start preparing for the new topics for the coming week. So I had enough to do on the weekend, as it turned out.

(*Continued*)

Table 6.2 (Continued)

Paragraph	*Journal Entry*
10/3	After lunch, I sat down to fill out the student grant form so that I could at least get started with it. It turned out that I could fill it out online and sign it with my electronic signature. So I sat down to work on the form and filled it all out and uploaded the parts where I didn't need help from my parents.
10/4	During the afternoon, my mom and I decided to go to [city name], the city nearest us, and take care of some errands, including some things for my new apartment, which I will move into in January because I will be studying in [city name].
10/5	When we got back home, I checked Moodle again and printed out the new texts that had been added and filed them in my folder so that I had everything in one place. In the evening, I made myself a to-do list for the coming week and listed out the documents I still needed to apply for the study grant.

Source: Autoethnography: Daily Life of a Student in the Online Winter Semester 2020/21

6.6 Deductive coding: Step 5

As already stated, there is a very good guide to working with MAXQDA on the company's website *(www.maxqda.com/)*. We will therefore not describe here how to install the program or how it is structured. To understand the process of deductive coding, it is important that the category system was imported into MAXQDA via an Excel list. We also uploaded the autoethnography, which is our example, as a document in MAXQDA. Before you continue working, you must import all of the documents to be analyzed into MAXQDA. To open the documents, follow these steps:

1. Go to the 'Import' tab in the MAXQDA menu.
2. Click the first icon 'Texts, PDFs, Tables'.
3. Select the documents from your computer that you want to analyze by clicking the file name in the respective folder.
4. Complete the import process by clicking the 'Open' button.

Two tips for importing documents: First, it is best to create a document group first (e.g., 'Journal entries' (based on content) or 't1' or 't2' (based on time of data collection), or 'University of x' or 'University of y' (based on place)). Second, you can also add documents to a project directly from Windows Explorer or the macOS Finder by clicking and dragging them into the 'List of Documents'.

The documents to be analyzed appear in the MAXQDA standard setting in the top left window ('List of Documents'). Double-click a document name to open the document in the 'Document Browser' window where it can be coded. If you want to activate all of the documents (later), you can do this by

right-clicking the root in the 'List of Documents' and selecting the 'Activate All Documents' function or even faster by clicking the symbol to the left of 'Documents'.

If you have activated the document, it will be displayed in the 'Document Browser' so that you can work with it. This means that you can highlight text passages and assign them to a category by dragging the highlighted text passage to a 'Code' in the 'List of Codes' window. After coding, the coded text passages are displayed in the 'List of Coded Segments', which is a kind of results window (see also the section titled 'The MAXQDA Interface'). This list can now be exported by clicking the Export icon under 'List of Coded Segments'. You can then choose between a Word or Excel list. We have chosen the Excel list as it is easier to work with. The list shows the color coding, the category name, and the coded text passages (including the paragraph number). For us, only the category and the corresponding text passage are important here. Downloading the Excel table is particularly useful if you are working with a MAXQDA teaching license or at a computer lab at the university and you can't save your work permanently in MAXQDA on the computer. Otherwise, you can continue to work in MAXQDA and, for example, continue to work in the 'Smart Coding Tool' or directly in the 'Document Browser'. For our analysis, we performed all of these steps in MAXQDA and now show the quotes we coded for each category in Table 6.3.

Based on the color coding, we can immediately see that there are no purple highlights, that is, no coding in relationship to the university as a social space. The explanation for this is simple: the coded autoethnography is written by a first-semester student. As a result, the student had not yet been on the university campus and had no experience with the university as a social space.

Overall, the coding shows that this case is about someone who seems to have quickly settled into the new situation of being a student. There are reports of a "first-year student group" that helps each other with problems via WhatsApp and which the person feels they belong to as a group member as they repeatedly talk about "we". There are reports of working in study groups, and there is also a "study partner", which means that the student is already integrated into the university and things are starting off smoothly. The text also describes how the student visits and is included in a student group, meaning that they have started with a digital social life. This is particularly emphasized as it becomes clear in the exchange with other fellow students that a "normal" student life on campus is not possible right now.

We would also like to point out some special characteristics of the deductive coding. The text passages in the category 'Interaction with lecturers \ Communication' are all implicitly formulated statements that do not directly indicate from the wording that they are about communication with lecturers. For example, "Shortly before the lecture started, we realized that this lecture in [minor] was to take place asynchronously. This means that the lecturer uploads the lecture to Moodle and we can watch it when we have time." Interpreting the text passage, the students (the "we" form is used) only became aware that the course would take place asynchronously shortly before it began. This lack of

Table 6.3 Coding All Relevant Quotes With Deductive Category Scheme

Category (= MAXQDA code)	*Coded Text Passage (= MAXQDA segment)*
Interaction with fellow students \ Peer group	When it was [morning], I wanted to log in to Moodle to get to the Zoom meeting link but I couldn't because all of the new students were also trying to log in and Moodle had frozen up. I'm in a WhatsApp group for the first-year students in our program and one of the students had saved the Zoom link and sent it to our group, so we were still able to join the Zoom lecture on time. (paragraph 1/6)
	Around [afternoon], we (the first-year students) noticed that we didn't have the Zoom link to our lecture for [afternoon]. Shortly before the lecture started, we realized that this lecture in [minor] was to take place asynchronously. This means that the lecturer uploads the lecture to Moodle and we can watch it when we have time. (paragraph 2/5)
Interaction with fellow students \ Study group	At lunch, I talked about my first university week and about the fact that I'm going to have the practical course group for the [additional course outside of field] module on Thursday and that I really want to watch the lecture for assignment [B], even though it will be about assignment [A], but if I have any questions, then I could ask them. (paragraph 3/3)
	I slept in until [morning] because there is nothing on our schedule, but we were told on Moodle that we will have the practical course groups for the module [additional course outside of field]. I am in practical course group [A] and therefore have this on Thursday [afternoon]. (paragraph 4/1)
	In the lecture, we talked briefly in a breakout session about the past week and our experiences with regard to our autoethnography task and discussed how we can record our information in order to write it down afterwards. (paragraph 6/2)
	The [additional course outside my field] module had turned out to be more difficult and a bigger challenge than I had expected. I kept finding myself thinking about this module throughout the day, so in the evening I decided to talk to my partner for assignments in the practical course group about it. She agreed and said that she found the rapid increase in expectations in the exercises a bit too extreme. (paragraph 8/3)
Interaction with fellow students \ Social life	I then put my notes away and joined a Zoom event ... [of the student group] [from country] in [city name], who welcomed the first-year students, introduced [their student group], and said they would be happy to have us as members. (paragraph 2/7)

(*Continued*)

Table 6.3 (Continued)

Category (= MAXQDA code)	*Coded Text Passage (= MAXQDA segment)*
	It was interesting that everyone really has their own approach, and we then talked about the fact that it is sometimes difficult to reach the specified word count because our daily life in terms of studying is limited on the one hand because of the pandemic, that is, that it is an online semester, but also that it feels monotonous because we can't do anything else because of the pandemic such as getting to know fellow students, which is normally a part of daily student life. (paragraph 6/2)
	So I had a few more minutes before I attended another Zoom meeting organized by [student group] from [city name]. This involved auctioning off the [first-year students] as a way for them to be accepted [into the student group] – it was a lot of fun and you got to know a lot of new people. (paragraph 7/6)
	As a result of the circumstances during the pandemic, you unfortunately don't get to know that many people in this first semester, which is entirely online, so I was pleasantly surprised that so many online meetings were planned. This makes it possible to at least make some connections. This auction went on a little longer, so I didn't get to bed until after midnight and just quickly set my alarm clock. (paragraph 7/6)
Interaction with lecturers \ Communication	Shortly before the lecture started, we realized that this lecture in [minor] was to take place asynchronously. This means that the lecturer uploads the lecture to Moodle and we can watch it when we have time. (paragraph 2/5)
	In the [major] preparatory course, we had talked about what to expect in the coming semester. (paragraph 3/2)
	I slept in until [morning] because there is nothing on our schedule, but we were told on Moodle that we will have the practical course groups for the module [additional course outside of field]. I am in practical course group [A] and therefore have this on Thursday [afternoon]. (paragraph 4/1)
	Our professor had decided to start the lecture a little later, at [morning], so I was able to get everything ready and take a quick look at the previous class before we then started at the set time. (paragraph 7/1)
	Like every Tuesday, I don't have any classes until [early evening] because we get the lecture for [minor] uploaded and can look at it on our own and in this way structure our daily life as a student a bit ourselves. Unfortunately, it didn't work so well today because it was online on Moodle a bit too late, and it wasn't possible to look at the lecture before the seminar because the lecture was too long. (paragraph 7/4 and paragraph 7/5)

(*Continued*)

Table 6.3 (Continued)

Category (= MAXQDA code)	*Coded Text Passage (= MAXQDA segment)*
Interaction with lecturers \ Integration in the context of the seminar	In the afternoon, I went over my first exercise again, which we had already handed in on Monday in order to be prepared to take part in the practical course group. At [afternoon] then, this course started; at the beginning of each practical course, 13 student numbers are drawn and these students are then tested at the end of the hour. My number wasn't drawn this time and I was kind of happy about this as I don't feel so confident in the material yet. (paragraph 4/5)
Interaction with lecturers \ Availability	We had also found out that we could ask the professor to upload the slides to Moodle earlier, but somehow that didn't work. (paragraph 7/5)
Interaction with lecturers \ Taking time in the seminar	We had also found out that we could ask the professor to upload the slides to Moodle earlier, but somehow that didn't work. (paragraph 7/5)

information was interpreted as communication between the lecturer and the students. After all, a lack of communication is also a form of communication. It is not important here what caused the communication problem.

6.7 Expand the category system: Step 6

As you may have already suspected from the few text passages that were coded during deductive coding, there are still many text passages that have not yet been coded. The theoretical perspective you choose will determine what and how much will still need to be coded. In our example, this is a strong focus on the social space and becoming integrated into and engage with the university, but not on the teaching and learning process or the interaction with the family. As we do not want to carry out a comprehensive analysis of what is in the material (which would be an inductive qualitative content analysis), but rather deliberately focus on specific aspects, this is not a problem. Nevertheless, it is important to keep in mind that for testing theories it is essential to look at the remaining material so that you can expand theories if necessary. For example, if the purpose of deductive qualitative content analysis is to add new dimensions to a model (e.g., Tinto's model in our example), then the remaining data must be considered (Graneheim et al., 2017).

In the next step, we will therefore look at the uncoded text passages and check whether these passages should be used to expand the category system. This only makes sense if it adds value to answering the research question. We must therefore first examine which passages in our example might refer to the

university as a social space and integration into the university. To determine this, it is important to read the material again, especially the text passages that have not yet been coded. As described previously, we can then directly exclude text passages that refer to interaction with the family. This includes, for example, "My mom was still in the kitchen and asked me what my first lecture was going to be." We also exclude descriptions of daily life such as "My alarm clock rang at 7:30 am, and when I got up I didn't feel so well rested. I was nervous because today is my first day at university and I couldn't sleep as well as usual" or "I went down to the kitchen and made myself a coffee, as I do every morning." If we exclude the passages about daily life, we are left with descriptions of student life and passages about finding an apartment.

We will now continue to work with the passages about student life and finding an apartment and analyze the extent to which they refer to the university as a social space and integration into the university. If we take the passages about finding an apartment, we can see that the student is looking for an apartment in the city where the university is located. The student wants to be able to start studying on campus and despite the digital semester wants to pursue all possibilities to move to the city where the university is located as soon as possible. Without living in the city where the university is located, it is more difficult to study and participate in student life, that is, to get engaged. In this respect, we generate the category 'City of university as a social space' and define it as follows: "Students report on their arrival at the city the university is located, for example, in regard to finding an apartment." According to the definition, we will go through the text again and code the corresponding text passages, which can be seen in Table 6.4.

In this example, it becomes clear that the student has had difficulties finding an apartment because of the pandemic, and not because of financial or other problems. It is clear that the family is providing support and this is evident from the context of the passage: "At lunch, I talked with my family again about the viewing appointments I had received. Tomorrow, we want to go to [city name] and look at the three apartments." The aim of the apartment search is to be in the city where the university is "so that I can move to [city name] for my studies".

In addition to the text passages on finding accommodation, there are also text passages about organizational aspects related to studying such as, "I checked my schedule for tomorrow in the [university] app on my cell phone." At the beginning of their studies, in particular, many students find it difficult to find their way around. Everything is new and doesn't usually have a lot of structure. Finding your way around is a crucial step in arriving at university and getting involved. Therefore, it is relevant for answering our research question. Text passages that refer to the organization of their studies can reveal a lot about how well the respective students have started their studies. For this purpose, we create the category 'Organizing their studies' and define it as follows: "Students report on how they are organizing their studies, for example, how they keep track of things or how and where they obtain materials for their studies."

According to this definition, we will go through the text again and code the corresponding text passages, which can be seen in Table 6.5.

It is clear from the text passages that the student has no problems finding his or her way around. The text passages show that the student is very structured

Table 6.4 Deductive Coding in the Category "City of University as a Social Space"

Category (= MAXQDA code)	*Coded Text Passage (= MAXQDA segment)*
City of university as a social space	After these 90 minutes, I didn't have any more lectures until [afternoon], so I focused on something else and continued to search the Internet for apartment ads and replied to some of them. I'm still looking for an apartment right now, and the current situation with the Covid-19 pandemic is making it more difficult than I thought. (paragraph 2/4)
	There was news about a viewing for an apartment in [city name]. After contacting them briefly, I had set up an appointment for a viewing on Saturday. (paragraph 4/3)
	At lunch, I talked with my family again about the viewing appointments I had received. Tomorrow, we want to go to [city name] and look at the three apartments because so far the search for an apartment has been difficult as a result of the pandemic, which means that as a foreigner from [name of country], I'm only allowed to be in Germany for a certain amount of time. I then got out the documents I wanted to take with me to the viewing, wrote down all the addresses, and to make sure I had an idea of the time plan, I looked at the routes to get there. In the afternoon, I continued to browse the real estate sites on the Internet, but with no success, so I hoped to get lucky with one of the three upcoming viewings on Saturday. (paragraph 5/3)
	In the evening, I checked Moodle to see if anything had changed during the day, set an alarm clock to get up early and go to [city name] for the viewing appointments so that I can move to [city name] for my studies. (paragraph 5/4)
	In the afternoon, I received the final rental contract for my first apartment, which will allow me to move to [city name] at the beginning of next year and continue my studies there in the hope of a normal and non-digital semester at [university] in [city name]. (paragraph 6/3)
	After the lecture, I sat down to look at the rental contract for my new apartment and read it thoroughly before I signed it and took it to the post office to send it to the landlord in [city name]. (paragraph 7/2)
	During the afternoon, my mom and I decided to go to [city name], the city nearest us, and take care of some errands, including some things for my new apartment, which I will move into in January because I will be studying in [city name]. (paragraph 10/4)

Table 6.5 Deductive Coding in the Category 'Organizing Their Studies'

Category (= MAXQDA code)	*Coded Text Passage (= MAXQDA segment)*
Organizing their studies	The emails included several notifications about the places I had received in the modules I had enrolled in for this semester. (paragraph 1/3)
	I then tried to log in to Moodle to have another look at what to expect in the lecture today. (paragraph 1/4)
	When it was [morning], I wanted to log in to Moodle to get to the Zoom meeting link but I couldn't because all of the new students were also trying to log in and Moodle had frozen up. (paragraph 1/6)
	[...] but before I finally turned off the light, I checked my schedule for tomorrow in the [university] app on my cell phone. (paragraph 1/12)
	My alarm went off at 7:30 am. I grabbed my phone and checked to see if I had received any emails and opened my schedule on the [university] app to see what lecture I had first today. (paragraph 2/1)
	[...] and check Moodle to see if I'd received a notification and if there was anything new. (paragraph 2/2)
	I went to bed at around 11 pm after looking at my schedule and setting my alarm for 7:30 am. (paragraph 2/8)
	I then got out my schedule and looked at what else was due and what needed to be done soon. I find it is a big change from my [school] and the long vacation, to start studying again now, but I am really excited about what lies ahead and I'm hoping that I will be really happy with this new bachelor's degree in [major]. (paragraphs 3/4 and 3/5)
	Then I checked Moodle to see if there was anything new and then sat down to the lecture for [additional course outside my field], which really takes a lot of time as I have no previous knowledge in this subject. (paragraph 4/4)
	At [morning] I made myself a coffee, sat down at the kitchen table, and opened Moodle on my cell phone to see if there was anything new. (paragraph 5/2)
	In the evening, I checked Moodle to see if anything had changed during the day. (paragraph 5/4)
	I set my alarm clock, looked at my schedule, and went to bed. (paragraph 6/4)
	My alarm rang at 7:45 am. I woke up and made myself a coffee. Then I sat down at our big table in the living room and opened my laptop to check my emails and look at Moodle. (paragraph 7/1)
	So I opened my laptop and checked Moodle to see what was new there, and I logged into my [university] email to see what emails I had received. Once I had a rough overview, I started preparing for Monday and had a look at what was coming up in the week ahead. (paragraph 10/2)
	When we got back home, I checked Moodle again and printed out the new texts that had been added and filed them in my folder so that I had everything in one place. (paragraph 10/5)

and that they need to use a number of tools (e.g., Moodle, apps, and email programs) to keep up with everything that has to do with his or her studies. The student has organized the individual steps required for his or her studies, which shows us that they have a high level of self-organization. This brings us to the next and final step of our flowchart.

6.8 Consolidate the results: Step 7

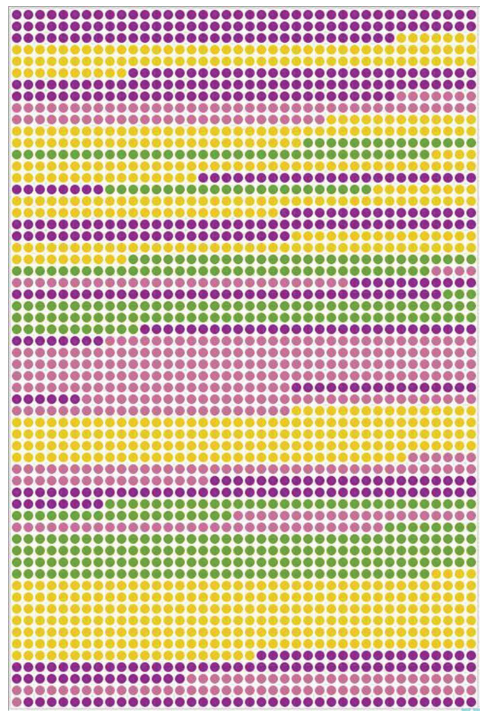
In the final step of the deductive qualitative content analysis, we will bring the results together. You can do this in MAXQDA by first creating a color sequence of the coded text passages using the 'Document Portrait' visualization tool. However, the 'Document Portrait' can only be created for one document. You can select the 'Document Portrait' from the menu ribbon ('Visual Tools' 'Document Portrait') or by selecting 'Document Portrait' in the context menu of a document. You can see what the 'Document Portrait' of our example looks like in Figure 6.2.

Based on the color sequence of the categories (only the top categories are shown here), you can see that the topics of "organizing your studies" and "interacting with fellow students" occur most frequently in the coded autoethnography. However, the portrait does not provide any more information than this. This is because a qualitative content analysis is not used to make statements of frequency (see the exclusion rule in Chapter 2). For the case analyzed here, we have obtained the following results:

- The student appears to be doing well in his or her studies as evidenced by the fact that they do not describe any problems with organizing his or her studies. On the contrary, there are many indications of how the student has his or her student life under control thanks to digital tools.
- The student also seems to be well integrated socially in his or her studies as they refer to both a peer group and study groups. By joining a student group, the student is also taking part in a digital social life.
- The student doesn't mention the social space of the university as a physical location, but this isn't surprising since the student has not yet studied on campus, but instead started directly with the online semester. However, the student has a desire to be integrated into the place of study, which is reflected in his or her (successful) search for accommodation in the city where the university is located.

With recourse to Tinto and the existing studies, the ad hoc hypothesis could be generated that the student is very likely to have few difficulties in his or her studies due to the good fit between their characteristics and the expectations at university.

Figure 6.2 Document Portrait of deductive coding.



6.9 Comparison of the cases

In a deductive qualitative content analysis, this would not be the end but only the beginning because this type of content analysis thrives on the comparison of cases. It is only by comparing several cases that you would arrive at a comprehensive result for your research question. In our case, we would analyze all of the material in the autoethnographic recordings (see Chapter 8), that is, all 50 autoethnographies. This would involve a text volume of approximately 250 pages, which is demanding with deductive qualitative content analysis, but is certainly feasible for a master's thesis (see Chapter 4 and the Quality Conventions).

For a bachelor's thesis, you could consider drawing a sampling from the data, but this must be well justified. For example, you could consider only analyzing cases of students who have a part-time job or care responsibilities and therefore have another commitment in addition to their studies. The assumption behind this could be that this makes it more difficult for them to engage into university life.

By comparing the autoethnographies, it would very likely be necessary to continue to develop the category system and adapt it at the subcategory level.

You would probably have to develop positive and negative subcategories. For example, under the category 'Interaction with fellow students \ Peer group', you would create two subcategories, which would then read 'Interaction with fellow students \ Peer group \ Existing' and 'Interaction with fellow students \ Peer group \ Not existing'. According to this category, the other categories would also receive further subcategories. In addition, the process of coding the autoethnographies could result in the addition of further categories, which in turn would lead to a review of the autoethnographies already coded. This is because you need to make sure that the newly developed categories do not appear in the autoethnographies that have already been coded. It can always happen that you paid too little attention to one aspect at the beginning. This is why it can be useful to analyze your material together with others, especially at the beginning of the coding process. This also meets the quality conventions discovery and appropriateness-transparency. Communicative validation – for example, with your supervisor for a qualification paper or with fellow students or colleagues – is therefore an integral part of the entire qualitative research process, from the research design to material selection and data collection instruments to the presentation of results.

If we had coded all of the autoethnographies, then we could use the Document Portrait to make a quick visual comparison between the documents, for example, to make an in-depth comparison of a category. We could then compare similar color schemes or very different color schemes, which is called a minimum or maximum comparison (Glaser & Strauss, 2017).

The comparisons also make it possible to uncover previous gaps in theoretical assumptions and in the state of research you have documented. In our example, the aspect of the digital social space, that is, a social space without a physical university, could exist and this space could help students to successfully integrate and engage (Steinhardt, 2021). And with this, the theory would be expanded.

Note

1 For empirical work, it is always advisable to keep a research journal in which you note down everything that has to do with your research. You can write down all of your ideas, thoughts, approaches to interpretation, interesting literature that you still want to read, and the like. It doesn't matter whether the research journal is analog or digital (e.g., the notes app on your smartphone) (see also Box 6.2).

7

INTRODUCTION TO INDUCTIVE QUALITATIVE CONTENT ANALYSIS

Content analysis is the systematic and rule-based collection and analysis of texts (communication content) through the interpretation of manifest and latent content and their meaning. The meaning is determined by breaking down the texts into categories using coding. Inductive qualitative content analysis is used when you want to understand the content of texts such as documents, group discussions, and interviews by focusing on empirical data and/or when little is known about a phenomenon. In short, you would go through the process by collecting data (e.g., interviews) or using secondary data (example used in the chapter: students' autoethnography), which you approach with an open mind without strict guidance by theoretical presuppositions and then develop the categories based on the material. In inductive qualitative content analysis, theory is used in a reflexive way (see Section 4.3). The category system is not predetermined for inductive qualitative content analysis, but it is instead constructed with reference to the empirical material for the organization of data for analysis and interpretation (see Section 5.3).

7.1 Introduction

In empirical social research, inductive qualitative content analysis is a methodological analysis that focuses primarily on the text and, if applicable, the associated data (e.g., icons and images from social media platforms). The term *inductive* is used in methodological literature to refer to the development of categories from empirical data. The inductive construction of categories means the construction of reliable analytical references, which then guide the inductive qualitative content analysis as a structuring element. In contrast to inductive qualitative content analysis, deductive qualitative content analysis

DOI: 10.4324/9781003496397-9
This chapter has been made available under a CC-BY-NC-ND 4.0 license.

is a method used to analyze data (e.g., group discussions, interviews, party platforms, tweets, and newspaper articles) using a category system created (= deduced) based on theory and previous empirical studies (see Chapters 4, 5, and 6).

The systematic and rule-based collection and analysis of texts (communication content), including the interpretation of manifest and latent content (see reminder in Box 7.1) and understanding of their meanings, involves several steps. Inductive qualitative content analysis is used when you want to understand the content of texts such as documents, group discussions, and interviews by focusing on empirical data and/or when little is known about a phenomenon. In short, you would go through the process by collecting data (e.g., interviews) or using secondary data (example used in the chapter: students' autoethnography), which you approach with an open mind without strict guidance by theoretical presuppositions and then develop the categories based on the data. In inductive qualitative content analysis, theory is used in a reflexive way (see Section 4.3). The category system is not predetermined for inductive qualitative content analysis, but it is instead constructed with reference to the empirical material[1] for the organization of data for analysis and interpretation (see Section 5.3).

Box 7.1 Three steps of understanding content

Analysis Step 1: Understanding the context of the investigated phenomenon [(a) subject(s) and/or (b) object(s), in (c) social (including cultural, economic, political, etc.) spatial, and temporal contexts].

Analysis Step 2: Understanding content by interpreting, inferring, and explaining.

1. Manifest content [What-question of communication (1st level of meaning)?
2. Latent content [Why-question of communication (2nd level of meaning)?

Analysis Step 3: Enable understanding of the audience [summarize results specific to the audience, organize for presentation, and/or write-up for publication].

7.1.1 Definitions for category, category system, and coding

The category system for inductive qualitative content analysis is not predetermined but is supposed to emerge from the material in an iterative process (see Section 4.3.3). In the second part of the research process (i.e., data processing), systematically coding the material involves the repetition (in sociolect: iteration) of the same action – don't get caught in the iteration loop (see Section 4.3.3.3)! Based on the initial coding, the codes are transformed into categories, and

categories compose an entire category system for the thematic organization and structuring of the analysis. Here are the basic definitions:

1. Category: A category is a unit to identify, classify, and sort actions, actors, artifacts, phenomena, etc. in qualitative content analysis and interpretation. A category should be distinct, have a clear meaning, and not overlap with other categories. Category definition problems indicate that already defined categories need to be reconsidered.
2. Category system: The category system is the result of a category construction process. A category system can be based on categories derived from theory (deductive research approach), categories developed from empirical data (inductive research approach), or a combination of deductively and inductively constructed categories. In general, a category system is composed of several top categories and associated subcategories.
    a) Top category: A top category is like a first order heading which indicates the overarching content of a text. A top category summarizes two or more subcategories (i.e., similar to second order headings).
    b) Subcategory: A subcategory specifies one aspect of the top category, for example 'research' and 'teaching' as subcategories of 'academic activities of a university' (top category). A subcategory has meaning on its own and, on a more abstract level, contributes to the overall meaning of the top category. A subcategory can have further subcategories.
3. Code and coding: To code means to attribute a category to a segment of text, for example a sentence, several sentences, or a paragraph. How much text you code depends on the category-led construction of a unit of meaning, that is, coherent content in a sequence. Coding is the activity of creating order in document(s) by selecting units of meaning to answer the research question. Noncoded text segments are not considered relevant for answering the research question according to the category system. As in encryption, the coded units of meaning await decoding, that is, content analysis and interpretation by researchers.

7.1.2 Reflexive use of theory in inductive qualitative content analysis

For an inductive approach to research, the phenomenon under empirical study is the focal point of the research process, and the two theory-reflexive stages (see Figure 4.2) consider the phenomenon under study from different perspectives. For example, in Chapter 6 and here, in Chapter 7, we use the same empirical example to explain how to proceed in a methodological way. So we know what results deductive, theory-led research produces, in Chapter 6 Tinto's (1997) student engagement theory. Nevertheless, it is plausible that the inductive approach to researching the impact of the Covid-19 event on the beginning

of university studies might produce new insights, that is, show new patterns of students' social behavior. Here, it is the event of the Covid-19 pandemic that justifies taking a fresh look at a well-studied social context of beginning students (context of discovery 1, see Sections 4.3.1 and 4.3.2.1). Even if you discovered a completely new phenomenon, you would not have noticed it without comparing the phenomenon to already known phenomena. Accordingly, in the first theory-reflexive stage you should write down all your assumptions about the (newly discovered) phenomenon of study, theoretical approaches to studying similar phenomena (e.g., from previous research), what you read in relevant, possibly remotely-related literature, discussed with peers or in seminars, etc. The goal of theory-reflexive stage 1 is to become aware of what you know and assume about a phenomenon so that you can take a proper empirical look at the phenomenon under study.

Your stage 1 reflexive use of theory will also affect part 2 of the research process: Data collection and processing. For example, in qualitative content analysis, coding units of meaning in documents and constructing a category system will not be free from your theorized assumptions (part 1: Research design). Accordingly, some analytical characteristics of your assumptions will, most likely, become top and subcategories, but only if they are discovered in the empirical data.

Stage 2 reflexive use of theory is led by empirical results in roughly two different ways. First, your theorized assumptions from stage 1 should be reconsidered in the light of empirical findings about the phenomenon under study (in a paper under the heading 'descriptive results', see Table 4.2). Second, empirical findings about patterns of actors in actions or social structures should be reflected by phenomena appropriate to the theory. For the reflexive use of theory, you need to outline the key assumptions of a theory and then apply them to your empirical findings to 'grasp the phenomenon under study'. You see, in inductive research, the phenomenon under empirical study is primary, and theory is secondary in the research process; therefore, inductive research is and has never been free or bereft of theory.

7.1.3 Step-by-step flowchart of the research process

Before you start planning your research, you should consider the scope of data and time required for the analysis. Research using empirical material is quite demanding and time-consuming. This is because you need to become familiar with, organize, and understand the data to be analyzed (e.g., the text) in order to be able to identify, explain, and interpret the manifest and latent content. The empirical material does not have to be very extensive. The autoethnography used as data material in this chapter is comprised of ten journal entries from two weeks on the topic "How are you organizing your daily life as a student at the beginning of the online semester 2020/21?"; originally, the autoethnography was about five and a half A4 pages long (with font size Arial 11).

Putting your qualitative research interest aside for the time being, you should use these rules of thumb as a parameter to balance the time you have for research and the number of pages required for your qualification paper:

1. Bachelor's or master's student research project or term paper (approx. 10–20 pages): maximum one text for practice purposes
2. Bachelor's thesis (approx. 50 pages): maximum of three texts
3. Master's thesis (approx. 80 pages): maximum of five texts
4. Dissertation (approx. 250 pages): about ten texts and not too many more

For planning your inductive qualitative research as well as the writing of the paper, we provide some orientation in a flowchart (Figure 7.1). The flowchart for inductive quantitative content analysis in Figure 7.1 is not explained in detail

Figure 7.1 Flowchart for inductive qualitative content analysis.

Step 1

Formulate the research interest as a question based on
1. literature analyzed,
2. your own previous empirical studies, and/or
3. other prior knowledge (e.g., personal experience and theoretical assumptions),
4. and prepare the research design (including methods and justification of the case selection).

Step 2

Prepare the data:
1. Data quality and social context of data collection
2. Anonymization of data

Step 3

Prepare the inductive qualitative content analysis:
1. Overview of text material and summary of content.
2. Structuring analysis of the data.
3. Inductive category development.
4. Coding of the data.

Step 4

Carry out the qualitative content analysis:
1. Identify manifest and latent content.
2. Analyze and interpret manifest and latent content.
3. Reflexive use of theory

Step 5

Summarize the results and prepare a presentation, term paper, and/or publication.

Step 6

Securely archive the data and, if possible, prepare it for subsequent reuse.

here already to avoid redundancies. The individual steps are explained in the following both theoretically and methodologically as well as by using examples. One exception is step six, which is of a general nature. If you are working with personal data, you must not only ensure the anonymity of the interviewees (see Section 7.2.2), but also the security of the data when archiving. It is essential that you store audio files, lists with contact details, transcripts that have not been anonymized, and the like in a secure, possibly password-protected location (e.g., server and/or USB stick). The aspect of reusing data is only relevant for you if you are collecting empirical data as part of a seminar or other larger research projects (see Section 3.3).

7.2 Some basics before starting with inductive qualitative content analysis

7.2.1 Social context of data collection

For qualitative social research in general and inductive qualitative content analysis in particular, the social context in which the empirical data was collected is of great importance (see Section 2.2). In sociology, the term social context summarizes cultural, economic, political, etc. aspects of a context. There are two main reasons why the context of data collection is so important. First, as the reference point of inductive qualitative content analysis, the empirical material was collected in a social context with specific spatial (e.g., Berlin, a market in rural Cameroon, or Malaysia) and temporal characteristics (e.g., 21st Century and beginning of 20th Century). The social context of data collection of the present empirical study is the autoethnography of a student who began their studies online at a university in Germany in Winter Semester 2020/21. Second, almost as reciprocal support for the first reason, the social context is the epistemological and analytical focus for inductive qualitative content analysis.

As emphasized in the introduction of this chapter, the inductive analysis and interpretation technique and qualitative content analysis are not a *tabula rasa* approach to the empirical data because as a social researcher you usually have all kinds of experiential, research-based, hearsay, and theoretical prior knowledge about the phenomenon being studied (e.g., Armat et al., 2018; Strauss, 1987). An inductive analysis technique does not mean that your research is free of theory either. During theory-reflexive stage 1 you should become aware of what you know and assume about a phenomenon so that you can take a proper empirical look at the phenomenon under study. This should be reflected in your research design. Stage 2 theory-reflexivity is an important step in the inductive qualitative data analysis and interpretation process in the third part of the research process (Step 4 in Figure 7.1; see also Figure 4.2).

The social context of data collection also establishes that text-based communication is not something natural like day and night or wind and rain, but that text communication always requires construction and reconstruction. The sociological construction of text is usually performed by human subjects. People

communicate on their own initiative and/or in response to previous communication (see Section 2.1). In some cases, a text's social context of origin is vague (e.g., symbols drawn in a cave) and not documented (e.g., in an archive). This must be made clear in the description of the context of data collection and considered in the analysis (see Chapter 3).

In the context of possibilities and limitations arising from the nature of data used for empirical social research, it should be pointed out here that, in this introductory chapter, we are doing a secondary data analysis. We will analyze the journal entries of a student that were created for another purpose using the method of autoethnography. The secondary data analysis is possible as the student signed a declaration of consent (see Section 3.4.1). Prior to signing the declaration of consent, students were informed about the project, data use, and received a written assurance of anonymity (see Section 3.4.2), as discussed in the following section.

7.2.2 Anonymization

The anonymization of autoethnographies, group discussions, interviews, and other personal identifying data serves three main purposes. Anonymization

1. offers individual protection for the subjects providing the information;
2. allows the autoethnographer, interviewee, etc. to speak/write freely about the chosen topic; and
3. obscures the recognizability of persons, places, and institutions.

In comparison to publications for teaching purposes, complete original texts are not included in papers (e.g., term papers, bachelor thesis, and scientific articles). Ideally, you present text documents (e.g., transcripts of interviews) in the (electronic or digital) appendix of your paper (appropriateness-transparency quality convention, see Section 4.3.2). A first step toward anonymization is the assignment of identification numbers (idno for 'ID(entity)' and 'no' from English 'number'). We have chosen to use idno in this book because of its brevity, but you could use another identifier such as Student_05. If you conducted three interviews, you could also choose Interview_1, Interview_2, and Interview_3 for anonymization. It would also be possible to assign other real name aliases, whereby, if relevant for your study, you should respect the biological gender in heterogeneous groups of interviewees. A sociobiologically neutral comparison of statements in your flow text would read as follows: in contrast to the statement in Interview_1, Interview_3 argued for ... and against ... In addition to the use of the identification number, you should always refer to the places in the original text both in the flow text and in quotations (e.g., idno05: paragraph 7/6; see Table 7.1).

In the next example, in addition to 'idno05', we use the gender-neutral reference to 'student' or 'they/them' in the text. The autoethnography of the student idno05 is one of 50 autoethnographic recordings voluntarily provided by students for teaching and research purposes from a course with around four to

Table 7.1 Autoethnography: Commencement of Studies in Winter Semester 2020/21

Paragraph	*Journal Entry*
1	**Monday, [month] [day], 2020**
1/1	My alarm clock rang at 7:30 am, and when I got up I didn't feel so well rested. I was nervous because today is my first day at university and I couldn't sleep as well as usual.
1/2	I went down to the kitchen and made myself a coffee, as I do every morning. My mom was still in the kitchen and asked me what my first lecture was going to be. I briefly explained that it was the [major] lecture and that it started at [morning].
1/3	With my coffee in hand, I went up to my room, turned on my laptop, and checked my emails. The emails included several notifications about the places I had received in the modules I had enrolled in for this semester.
1/4	I then tried to log in to Moodle [online teaching-learning platform] to have another look at what to expect in the lecture today. I had already read through the text at the weekend, but to be on the safe side I checked again.
1/5	I then put on some nicer clothes and got ready to take part in the Zoom meeting.
1/6	When it was [morning], I wanted to log in to Moodle to get to the Zoom meeting link but I couldn't because all of the new students were also trying to log in and Moodle had frozen up. I'm in a WhatsApp group for the first-year students in our program and one of the students had saved the Zoom link and sent it to our group, so we were still able to join the Zoom lecture on time.
1/7	My first Zoom lecture was about autoethnographies, and we were given the task of writing one ourselves. We are supposed to record 10 days for an autoethnography and really try this experiment.
1/8	After this lecture, my university day was already over as I only had one lecture on this day.
1/9	At lunch, I talked about the first lecture and how interesting it was.
1/10	In the afternoon, I decided to start by sorting and organizing everything I've received so far in terms of texts, assignments, and information for the modules I'm taking at university. I set up a subfolder for each module and printed out and organized the literature I had already received. I then decided to read the next text for another module to prepare myself a little.
1/11	After dinner, I decided to read through the examination regulations because you should internalize these so that you are aware of all your rights during the examination phase.
1/12	At 11 pm, I decided to go to bed, but before I finally turned off the light, I checked my schedule for tomorrow in the [university] app on my cell phone.
2	**Tuesday, [month] [day], 2020**
2/1	My alarm went off at 7:30 am. I grabbed my phone and checked to see if I had received any emails and opened my schedule on the [university] app to see what lecture I had first today.

(*Continued*)

Table 7.1 (Continued)

Paragraph	*Journal Entry*
2/2	Then I went down to the kitchen and made my coffee before I went back upstairs to turn on my laptop and check Moodle to see if I'd received a notification and if there was anything new.
2/3	Then at [morning] my first lecture started: an introduction to [major]. I had already read through the lecture text the evening before and was therefore able to follow the lecture well.
2/4	After these 90 minutes, I didn't have any more lectures until [afternoon], so I focused on something else and continued to search the Internet for apartment ads and replied to some of them. I'm still looking for an apartment right now, and the current situation with the Covid pandemic is making it more difficult than I thought.
2/5	Around [afternoon], we (the first-year students) noticed that we didn't have the Zoom link to our lecture for [afternoon]. Shortly before the lecture started, we realized that this lecture in [minor] was to take place asynchronously. This means that the lecturer uploads the lecture to Moodle, and we can watch it when we have time. I decided around [early evening] to watch half of it, that is, the introductory part and a bit of the first topic we're going to cover.
2/6	After this introduction, I ate something and prepared for the seminar, which took place at [evening]. The seminar started at [evening].
2/7	I then put my notes away and joined a Zoom event [...] [of the student group] [from country] in [city name], who welcomed the first-year students, introduced [their student group], and said they would be happy to have us as members.
2/8	I went to bed at around 11 pm after looking at my schedule and setting my alarm for 7:30 am.
3	**Wednesday, [month] [day], 2020**
3/1	My alarm clock rang at 7:30 am. I got up, not yet fully rested, and went to the kitchen to make myself a coffee. I sat down at the kitchen table and talked to [sibling] about my first two university days. He's currently on fall break and still thinks it's weird that we don't go to the same school anymore.
3/2	After I finished my coffee, I took a water bottle up to my room, sat down at my desk, and turned on my laptop to take part in the Zoom course that started at [morning]. In the [major] preparatory course, we had talked about what to expect in the coming semester.
3/3	Afterwards, we had the seminar for [major] that started at [morning]. At lunch, I talked about my first university week and about the fact that I'm going to have the practical course group for the [additional course outside of field] module on Thursday and that I really want to watch the lecture for assignment [B], even though it will be about assignment [A], but if I have any questions, then I could ask them.

(*Continued*)

Table 7.1 (Continued)

Paragraph	*Journal Entry*
3/4	After dinner, I pulled myself together and decided to start the lecture today after all so that I could make better progress tomorrow. I then got out my schedule and looked at what else was due and what needed to be done soon.
3/5	I find it is a big change from my [school] and the long vacation, to start studying again now, but I am really excited about what lies ahead and I'm hoping that I will be really happy with this new bachelor's degree in [major].
3/6	I went to bed at 10:30 pm and didn't set an alarm clock because I can sleep in tomorrow and the practical course group isn't until the afternoon.
4	**Thursday, [month] [day], 2020**
4/1	I slept in until [morning] because there is nothing on our schedule, but we were told on Moodle that we will have the practical course groups for the module [additional course outside of field]. I am in practical course group [A] and therefore have this on Thursday [afternoon].
4/2	After getting up, I went to the kitchen and made myself a coffee like I do every morning. My [sibling] was already awake, and we talked about what we were going to do today. I then briefly explained to him that I wanted to get straight to the practical course for [additional course outside my field] because first I wanted to be well prepared, and second, I still had to take care of the lecture and practical course [B] for Monday.
4/3	When I got back to my desk, I turned on my laptop and checked my emails, both my personal and [university] emails. There was news about a viewing for an apartment in [city name]. After contacting them briefly, I had set up an appointment for a viewing on Saturday.
4/4	Then I checked Moodle to see if there was anything new and then sat down to the lecture for [additional course outside my field], which really takes a lot of time as I have no previous knowledge in this subject. Throughout the day, I noticed that I kept thinking about this because I really have difficulties with the tasks we are given. To get a better understanding, I've ordered the book for [additional course outside my field], and I hope that it will provide the help I'm looking for.
4/5	In the afternoon, I went over my first exercise again, which we had already handed in on Monday in order to be prepared to take part in the practical course group. At [afternoon] then, this course started; at the beginning of each practical course, [number] student numbers are drawn, and these students are then [tested] at the end of the hour. My number wasn't drawn this time, and I was kind of happy about this as I don't feel so confident in the material yet.
4/6	Then my study day was over, I closed my laptop and spent the rest of the day doing other things that I hadn't had time for during the week because of the new focus of my daily life, my studies.

(*Continued*)

Table 7.1 (Continued)

Paragraph	*Journal Entry*
5	**Friday, [month] [day], 2020**
5/1	I don't have any lectures on Fridays and can do follow-up work for the modules I am in, but I can also do some preparatory work and print out and organize all the texts I've received so that I have a clearer view of the next week and what's coming up. So, I got up a little later than usual because I'm finding it hard to get up earlier again after the long vacation, even though I worked during this time.
5/2	At [morning] I made myself a coffee, sat down at the kitchen table, and opened Moodle on my cell phone to see if there was anything new. After going through my emails, it was already lunchtime.
5/3	At lunch, I talked with my family again about the viewing appointments I had received. Tomorrow, we want to go to [city name] and look at the three apartments because so far the search for an apartment has been difficult as a result of the pandemic, which means that as a foreigner from [name of country], I'm only allowed to be in Germany for a certain amount of time. I then got out the documents I wanted to take with me to the viewing, wrote down all the addresses, and to make sure I had an idea of the time plan, I looked at the routes to get there. In the afternoon, I continued to browse the real estate sites on the Internet, but with no success, so I hoped to get lucky with one of the three upcoming viewings on Saturday.
5/4	The book that I had ordered for the module [additional course outside my field] was also delivered today, so I immediately read the first two chapters in order to get a better grip on practical course 02. In the evening, I checked Moodle to see if anything had changed during the day, set an alarm clock to get up early and go to [city name] for the viewing appointments so that I can move to [city name] for my studies.
6	**Monday, [month] [day], 2020**
6/1	This Monday marks the start of the second week of my studies in [major]. My alarm went off at 9 am and I got up, made myself a coffee in the kitchen, and immediately sat down at my laptop to watch the video on Moodle for my first lecture as I hadn't found the time to do so on the weekend. It turned out that this wasn't the best idea because I was causing myself unnecessary stress.
6/2	In the lecture, we talked briefly in a breakout session about the past week and our experiences with regard to our autoethnography task and discussed how we can record our information in order to write it down afterwards. It was interesting that everyone really has their own approach, and we then talked about the fact that it is sometimes difficult to reach the specified word count because our daily life in terms of studying is limited on the one hand because of the pandemic, that is, that it is an online semester, but also that it feels monotonous because we can't do anything else because of the pandemic such as getting to know fellow students, which is normally a part of daily student life.

(*Continued*)

Table 7.1 (Continued)

Paragraph	*Journal Entry*
6/3	In the afternoon, I received the final rental contract for my first apartment, which will allow me to move to [city name] at the beginning of next year and continue my studies there in the hope of a normal and non-digital semester at [university] in [city name].
6/4	Then I watched part of the lecture of the [additional course outside my field] module and took notes. At the end of the day, I watched the podcast for another lecture and was done with everything related to university that day. I set my alarm clock, looked at my schedule, and went to bed.
7	**Tuesday, [month] [day], 2020**
7/1	My alarm rang at 7:45 am. I woke up and made myself a coffee. Then I sat down at our big table in the living room and opened my laptop to check my emails and look at Moodle. Our professor had decided to start the lecture a little later, at [morning], so I was able to get everything ready and take a quick look at the previous class before we then started at the set time.
7/2	After the lecture, I sat down to look at the rental contract for my new apartment and read it thoroughly before I signed it and took it to the post office to send it to the landlord in [city name].
7/3	At lunch, we talked about what I still have to do today for university and what things I can already take care of for my courses, but also for my apartment.
7/4	Like every Tuesday, I don't have any classes until [early evening] because we get the lecture for [minor] uploaded and can look at it on our own and in this way structure our daily life as a student a bit ourselves.
7/5	Unfortunately, it didn't work so well today because it was online on Moodle a bit too late, and it wasn't possible to look at the lecture before the seminar because the lecture was too long. We had also found out that we could ask the professor to upload the slides to Moodle earlier, but somehow that didn't work. Our seminar at [early evening] ended a little earlier because it was a kind of introduction to [major].
7/6	So I had a few more minutes before I attended another Zoom meeting organized by [student group] from [city name]. This involved auctioning off the [first-year students] as a way for them to be accepted [into the student group] – it was a lot of fun and you got to know a lot of new people. As a result of the circumstances during the pandemic, you unfortunately don't get to know that many people in this first semester, which is entirely online, so I was pleasantly surprised that so many online meetings were planned. This makes it possible to at least make some connections. This auction went on a little longer, so I didn't get to bed until after midnight and just quickly set my alarm clock.

(*Continued*)

Table 7.1 (Continued)

Paragraph	*Journal Entry*
8	**Wednesday, [day] [month], 2020**
8/1	My alarm went off at 7:45 am and I hit snooze and then didn't wake up until 5 minutes before the lecture, so I had to hurry and quickly make myself a coffee and sit down in front of my laptop. But I was still in time for the lecture and didn't miss anything.
8/2	I then had a lecture until [noon], and at lunch everyone asked me about the event I attended yesterday evening. I briefly told them what I had to do [...]. We also talked about the study grant that every [...] student can receive if they submit an [application]. But to apply for this, I need several documents, including the rental contract, which I had only sent off this week and hadn't received back yet.
8/3	After lunch, I didn't have any more lectures or seminars, so I was free in the afternoon and could do some preparatory and follow-up work for my courses. I had decided to read through some of the texts and finish watching the [additional course outside my field] lecture that was uploaded to Moodle. The [additional course outside my field] module had turned out to be more difficult and a bigger challenge than I had expected. I kept finding myself thinking about this module throughout the day, so in the evening I decided to talk to my partner for assignments in the practical course group about it. She agreed and said that she found the rapid increase in expectations in the exercises a bit too extreme.
8/4	By the end of the day, I had had enough of university stuff and had decided not to read any more texts, so I just made myself a to-do list for tomorrow and set my alarm clock to get up in time so that I could get something done in the morning.
9	**Thursday, [day] [month], 2020**
9/1	My alarm went off at 9:30 am and I got up to make myself a coffee. I sat down at the kitchen table and looked at my messages and checked my emails on my phone. As I had already made my to-do list yesterday, I knew exactly what I had to do today. So, I went upstairs and sat down at my desk. I tidied it up a bit so that I had enough space to work productively and sat down and opened my laptop.
9/2	The first thing I did was to read through some texts and use the book for the [additional course outside my field] to prepare for the afternoon practical course session. Somehow this module takes up most of my time and I would actually like to spend more time on the other modules, also when I was following up on the various modules for [major], I realized that I already know a lot of the methods described as I had [name of school] [main subject] [...].

(*Continued*)

Table 7.1 (Continued)

Paragraph	*Journal Entry*
9/3	In the afternoon, I printed out the missing texts as I like to underline everything I find important in a text, and I don't have a tablet and reading on a laptop is quite difficult in my opinion. At [afternoon], we had the practical course, and this time my student number was on the list of [examinees], which made me a bit nervous as I had some difficulty with the exercise this time. [The exam] went well though and I passed it. I was asked a few comprehension questions that I had no problem answering as he had addressed these in class and I had made notes on the explanations.
9/4	After this practical course, we had dinner together as a family. I told them about the [exam] I had passed and what these plus points mean in terms of the written examination. Before I went to bed, I didn't set an alarm clock because I don't have any lectures or other scheduled university courses or events on Fridays.
10	**Friday, [month] [day], 2020**
10/1	I got up at 10 am and made myself a coffee and a snack. Then I went upstairs and completely cleaned up my room before I sat down at my desk to do things for university. I find that it's easier to concentrate when things are tidy and not disorganized. Somehow it seems more structured, and I make better progress with what I want to achieve on the day.
10/2	So I opened my laptop and checked Moodle to see what was new there, and I logged into my [university] email to see what emails I had received. Once I had a rough overview, I started preparing for Monday and had a look at what was coming up in the week ahead. I quickly realized that I still had to do some work to follow up on my last classes and that at the same time I also had to start preparing for the new topics for the coming week. So I had enough to do on the weekend, as it turned out.
10/3	After lunch, I sat down to fill out the student grant form so that I could at least get started with it. It turned out that I could fill it out online and sign it with my electronic signature. So I sat down to work on the form and filled it all out and uploaded the parts where I didn't need help from my parents.
10/4	During the afternoon, my mom and I decided to go to [city name], the city nearest us, and take care of some errands, including some things for my new apartment, which I will move into in January because I will be studying in [city name].
10/5	When we got back home, I checked Moodle again and printed out the new texts that had been added and filed them in my folder so that I had everything in one place. In the evening, I made myself a to-do list for the coming week and listed out the documents I still needed to apply for the study grant.

Source: Autoethnography: Daily Life of a Student in the Online Winter Semester 2020/21

five times as many students. The majority of the course participants either did not take part in the voluntary exercise or wanted to keep their journal entries to themselves. The record of which student is anonymized by an idno was documented in an Excel file saved in a secure location. If the file is stored on a drive or in a cloud that several people have access to, it is essential that the file is password protected.

For anonymity purposes, interventions in the original empirical text should be made uniformly and be clearly visible. In the example in Table 7.1, square brackets are used. In general, common (writing-related) conventions should guide the researcher's practical decision to mark the researcher's interventions in the text (e.g., three dots for omissions). Accordingly, an [...] was chosen for deleted text. Concise alternate terms should ideally be chosen, whereby it is irrelevant in terms of content whether, for example, [city name] or [name of city] is used.

Anonymization should also be guided by content-related aspects. For example, you should omit the name of the degree program and anonymize it by using [major] and [minor] as placeholders. Journal entries for various compulsory and compulsory elective courses in the student's degree program and that the student is attending an elective course that does not belong to either the major or minor are indicated by [additional course outside of field]. The daily schedule is also relevant for a journal entry. Instead of just deleting class times to prevent them from being searched for, since the Internet doesn't forget and search engines are good, we replaced them with relatively clear times of day such as [morning], [afternoon], and [early evening]. By anonymizing the names of the city and university, the probability of the student being found is even further reduced as there are over 100 universities among the approximately 400 higher education institutions in Germany and more than 8,000 degree programs.

Of course, you may overlook a certain word or description that would make the place or university clear to *those in the know.* This can happen and is disturbing. But if the empirical social researcher has made a demonstrable effort to anonymize the data, it is excusable. For example, as a reader of this book, you could search to see which university the data and autoethnography might come from. You could start with the CVs of the three authors. This would limit the possibilities to a few universities in Germany. But even if you continue the search and find a German publication by Isabel Steinhardt on autoethnography (Autor:innengruppe AEDiL, 2021), it might be that a colleague from another university allowed us to analyze the data under their supervision. As you can see, it is generally not worth the effort to go looking out of curiosity, nor is it of interest. Even the students in the course do not know who consented to their autoethnography being used for research and teaching. Of course, the students themselves know. But if they do not explicitly (subsequently) provide self-disclosure or confirm inquiries, fellow students and other third parties will never know.

In some cases, the text can only be anonymized by making more significant changes. However, the core of the empirical information should remain the same and not be distorted by the changes. For example, in the following text

excerpt, the word order of the sentence was changed so that the passage reads smoothly: "[...] [of the student group] [from country] in [city name]" (Table 7.1, paragraph 2/7). An omission in the text is indicated by inserting [...], and the researcher can also add relevant information using brackets: [from country] in [city name]. The information [from country] was inserted instead of an omission as all universities have student associations based on country and/or geographical regions (e.g., Latin America) in addition to student organizations such as student councils and departmental student representatives involved in university governance. Knowing this, you can assume that the text has been sufficiently anonymized. If you are unsure or in doubt and have difficulty assessing a piece of information, you should always be careful and anonymize it or even partially delete information that could lead to the identification of informants.

Although you will be weighing the pros and cons of using critical information and results, please be 100% assured: This is not a conundrum! Even in cases of minor doubt, always rate data protection and ethical considerations higher than reporting the information in your results. Even if you cannot ground the most striking insight in your data, which results in a finding being less meaningful – it is not worth it. Nowadays, social media and other platforms provide so much information that a skilled data scientist can connect even small dots with little effort. Such a scandal would harm your personal credibility as well as the credibility of empirical social research, which will lower the willingness of people to participate in studies.

7.2.3 Structuring analysis of the text data

No matter if you use Atlas.ti, MAXQDA, NVivo, or another qualitative data analysis (QDA) software – before you start with data processing, you must first import the empirical material into the QDA software. You can find short introductions as well as tutorials for each of these programs on the provider's website or via an Internet search engine (e.g., data-protected via *https://duckduckgo.com*).

You can structure the text data at the same time as you are identifying central topics in the empirical text material. There are basically two processes for structuring the data:

1. Structuring based on the organization given in the text (e.g., a new numbered paragraph every time the speaker changes during an interview) or
2. structuring the text using units of meaning as done in Table 7.1.

In the case of QDA software use, the first structuring process is not necessary as the QDA software takes over this work of assigning numbers to paragraphs. To structure, for example, interview transcripts according to units of meaning, you can use the paragraphs as a guide. However, you must assume that a unit of meaning can also begin in the middle of a paragraph – which can be identified by a manifest change of topic – continue for another whole paragraph and then

end after the first sentence in the following paragraph. As Miles and Huberman (1994) emphasize, units of meaning are coherent content that is presented in a continuous sequence. This does not exclude the possibility that a similar or complementary topical unit of meaning exists elsewhere in the text.

7.2.4 Sample text: autoethnographic journal entry (commencement of studies in Winter Semester 2020/21)

Building on the discussion of structuring the text for analysis, the entire students' autoethnography text is presented next (Table 7.1). This sample text will help

1. you to understand how the categories were inductively developed, and
2. then you can try to inductively develop your own categories from coding the information in the text (= data).

The student wrote one full paragraph per day as text for the voluntary autoethnography assignment. In Table 7.1, the student's text is divided into text units of meaning for each day, which can be used as a reference (i.e., paragraph + number) – analogous to paragraph markers in the qualitative data analysis (QDA) software (see Section 8.2.3).

Miles and Huberman (1994) define a unit of meaning as a coherent description of an event (e.g., lecture or seminar), an action, or a context of action. A new unit of meaning was created in Table 7.1 if, for example, a new part of the day that included a new action began and the action changed in a clearly recognizable way such as the following: getting up early because of nervousness on the first day of university (paragraph 1/1), making coffee (paragraph 1/2), and drinking coffee at the computer while checking emails (paragraph 1/3). In contrast to the first day of university, the three text elements on the second university day are no longer relevant as individual actions, but only in combination as a single unit of meaning, which is also shown by the coherent presentation in paragraph 2/1. It is also clear that, for example, a lot happens on some days and therefore more units of meaning can be identified in the autoethnographic text (especially day 1) and that on other days less noteworthy things happen in students' lives, which is why only four units of meaning can be identified (e.g., days 5, 6, 8, and 9). As you can see, creating units of meaning is based on topics (e.g., actions) and must be well justified for the understanding by third parties, but it is not complicated science.

7.3 Inductive qualitative content analysis process

7.3.1 Gaining an overview: summarizing the content of text document(s)

In the analysis described here, the coding of manifest content of the empirical material (autoethnography in Table 7.1) should always be guided by the research question "How are you organizing your daily life as a student at the beginning of

the online semester 2020/21?" For clarity, you should briefly summarize the main content and topics. This means that summarizing is the first analysis step (see Step 3 in Figure 7.1). Depending on your own preferences, your working style, and the requirements formulated by the lecturer (in the role of teacher and/or examiner), you can summarize the text you have examined in continuous text or using bullet points. If you choose to use bullet points, you could supplement these with an exemplary quote. The central manifest content of the autoethnography in Table 7.1 is briefly summarized here according to seven topics:

1. Getting started with studies and becoming familiar with the demands of the students' degree program (e.g., procedure and preparing for courses, as well as using online tools such as Moodle and Zoom).
2. Organizing the new everyday life as a student (e.g., getting up, using time between classes, and making to-do lists).
3. Getting to know new people in the digital space (e.g., at classes and events, via messenger, and in student groups).
4. Living at home (e.g., interactions with family members and eating together).
5. Moving out of home to the place of study (e.g., finding accommodation and completing the rental contract).
6. Financing of studies (e.g., study grants and bureaucracy).
7. Completing the autoethnography (e.g., exchange with fellow students and reaching the required word count).

You should be able to easily recognize these seven topics, but they are not completely free of overlaps. For example, the exchange with other students about autoethnography in paragraph 6/2 is methodologically interesting for Topic 7 both because of the autoethnography method (keyword: reflexivity of research) and because of becoming familiar with the demands of the degree program (Topic 1).

Before we begin, just a gentle reminder: Just as important as careful and systematic analysis and interpretation of data is the documentation of what you did and why you did it in your research journal (see definition in Box 7.2). Make notes in your research journal to document potential information from the empirical material derived during coding that could be used as an inductive category as well as initial ideas you have on contexts, meanings, and approaches for interpretation. From our own experience, we can say definitively that without documentation in a (digital and/or analog) research journal, you will forget many good ideas again, unless you have an eidetic memory, as the analysis process is cognitively very demanding.

7.3.2 Coding the empirical material

If you thought that coding was a quasi-automatic process or that artificial intelligence should make decisions for you (see Chapter 10), well, unfortunately, you are wrong. To learn inductive qualitative content analysis, you will have to understand the coding process and then make decisions (almost) every

minute during the activity of coding. Coding for inductive qualitative content analysis is usually carried out manually (= by hand). Although your QDA software could definitely autocode individual words such as 'email' and 'Zoom' (see Chapter 8), this no longer works for the subcategories such as 'not connected to courses' or 'messages' as these are derived from the context of the statement and are therefore not automatically recognized by the QDA software. Before you can start manual coding, you must decide how (and how much text) to code. The decision on how much text to code is primarily limited to the following options.

Box 7.2 Brief definition of a research journal

The purpose of a research journal is to ensure that you do not forget good ideas you have for your research (e.g., category development, interpretation, and theoretical reflection) and to create transparency in the sense of documenting your research process and the associated decisions (see quality conventions in Chapter 4). You will soon realize that inductive qualitative content analysis involves simultaneously carrying out a number of cognitive tasks. For example, anonymizing, summarizing, and structuring the empirical material are important steps in the analysis and do not just serve to organize the text data.

From experience, we can tell you that you will be very annoyed if you can no longer think of an obvious idea for interpreting the data or you spend a lot of time trying to find – or not finding – an important quote in the literature you have read. You should, of course, organize your literature and reading in a reference management program (e.g., Zotero). Depending on your preferred working style, you can keep a research journal electronically (e.g., in Word) or on paper (e.g., a small book).

1. **Coding individual words:** Coding individual words, that is, small-scale coding with codes of identical individual words can be useful for an in-depth analysis, for example, if you wanted to look at three or five student autoethnographies and analyze the regularity of checking and writing 'emails' as a means of maintaining social contacts. By coding the single word 'email', you can activate the code in the QDA software (see also Section 8.4.2), and the corresponding passages will be displayed in the autoethnography. Depending on the previous settings in the QDA software, the line or paragraph numbers can then be used as references for citations (see explanation in Section 7.3.2 and Table 7.1).
2. **Coding sentences:** In addition to or as an alternative to single-word coding, you can code entire sentences. For example, the sentence "in the lecture, we talked briefly in a breakout session about the past week and our experiences with regard to our autoethnography task and discussed how we can

record our information in order to write it down afterwards" (paragraph 6/2) contains content captured both by 'face-to-face' and 'connected to courses' subcategories (see Table 7.2). In principle, you can assign several codes to a sentence, which means the coding scheme defines how often a sentence is coded as empirical material. However, the average sentence length makes it relatively unlikely that all two subcategories can be coded for a sentence – except for the half to full-page nested sentences made possible by German grammar, which, however, rarely occur in journal entries or in transcripts of interviews or social media communication.

3. **Coding paragraphs:** Coding sentences has certain limitations that can be overcome through coding paragraphs. In contrast to single-word and sentence coding, when you code paragraphs the aim is to capture the content of more abstract contexts of meaning in the empirical material. Thanks to the numbering, the subcodes define a clearly recognizable hierarchical relationship. Accordingly, the codes also represent specific 'social contacts' (top category) during your studies, for example in a 'student group' and/or 'face-to-face' communication. The code hierarchy from concrete to abstract therefore runs from 'face-to-face' communication and in a 'student group' to 'social contacts'.
4. **Coding units of meaning (see also Section 7.3.2):** Units of meaning are coded independently from grammatical rules of text organization (e.g., sentences and paragraphs). They can address the levels from the category (= highest aggregate level) to the last level of the subcategories, as explained later for a new, inductively created subcategory 'student organization' (Table 7.3) as an example. Coding units of meaning serve to differentiate between larger contexts in the empirical data, which were broadly identified as six topics in Section 7.3.1.

The four coding options should not be seen as either-or decision. Each type of coding provides information that fits in a predefined scope, which ranges from a word or a sentence to an entire paragraph or unit of meaning. No additional results for manual coding are presented here. Examples of coding can be found in this book in Chapter 6 and in Section 8.4.5.

7.3.3 Inductive category development

By identifying units of meaning and coding manifest content, you will become very familiar with the content of your data. How your in-depth knowledge of the data should guide inductive data analysis and interpretation for category construction is explained using the example of Topic 3 (i.e., getting to know new people in the digital space). To get started with analysis and interpretation for inductive category construction (see Section 5.3.1), we rely on the Topic 3 summary as an initial search guide for working through the empirical material. To learn coding for inductive category construction, you can choose one of the other topics yourself for practice purposes and work through it in the same way as the example Topic 3 explanation.

If you carefully review the text material in Table 7.1, you could identify the following text passages in units of meaning for category construction related to Topic 3 'social activities to get to know new people':

- "Checked my emails" (paragraph 1/3), "if I had received any emails" (paragraph 2/1), "both my personal and [university] emails" (paragraph 4/3), and "going through my emails" (paragraph 5/2).
- "WhatsApp group" (paragraph 1/6).
- "[student group]" (paragraphs 2/7 and 7/6).
- "breakout session" instead of "getting to know fellow students, which is normally part of daily student life [without the Covid-19 pandemic]" (paragraph 6/2).
- "My partner for assignments in the practical course group" (paragraph 8/3).

In the winter semester 2020/21, the opportunities to make contacts and get to know new people are limited to text 'messages', phone calls, and online meetings due to Covid-19 social distancing rules. Nevertheless, we see that the student is actively involved in online communication. The autoethnographic journal entries show that 'face-to-face' communication with fellow students takes place primarily through the contacts made in and outside the courses. Therefore, 'social contacts' is chosen as top category to which all other categories are assigned in Table 7.2. It is important to remember that even though coding and inductive category construction are part of your empirical analysis, only the final category system and the subsequent activity of coding the text material is the basis for answering your research question.

We can see that the inductively developed categories are very close to the text. The subcategory 'Connected to course/study partner (e.g., group)' is differentiated rather by chance in such a way that it becomes clear that in colloquial language the term study group is used, even if it consists of two people. This example shows that based on our data, we have to decide which categories to assign during coding and how detailed subcategories define features of a top category.

Table 7.2 Example of Social Contacts (Top Category) and Subcategories Used for Coding the Text Data in Table 7.1

Top category	*Subcategories (= codes)*
Social contacts	Email communication
	Messenger communication (cell phone)
	Face-to-face communication in a course
	Connected to a course (e.g., partner in assignment (or group))
	Student group (= not connected to courses)

Attributing easy-to-understand category names does not mean that a category is self-explanatory to a third party (e.g., your instructor). In an analog social context, 'face-to-face' would be understood as 'face-to-face' communication without any (digital) technology use. However, 'face-to-face' is a technical term in sociolect for personal communication on site – in both analog and digital locations. Without a pandemic (e.g., Covid-19), daily student life is characterized by analog 'face-to-face' communication in courses (e.g., lectures and seminars) as well as in contexts 'not connected to courses', but with a direct connection to studies (e.g., in 'student groups'). 'Face-to-face' communication 'connected to courses' can include communication between students and with lecturers. For example, in a lecture that was offered via 'Zoom' in Winter Semester 2020/21, the lecturer made use of 'breakout sessions' – in an analog space, this would be group work – where students were to exchange ideas or work on specific assignments.

Once all text data relevant to answering the research question is coded, for its analysis and even more so for data interpretation, we recommend amending the categories with notes or definitions that support meaning making, for example, in your research journal and/or the MAXQDA software interface where you can use the memo function for the categories.

Now, just one final question concerning the inductive construction of a category system remains open: When are you finished with inductive category development and coding? The straightforward answer is thus: It is hard to say. Assuming you have systematically constructed the category system, in general, we would recommend using the following signals as guidelines for being finished:

1. If you are going through the text segments that are not coded and realize that (a) no category could be developed from them to answer the research question, and (b) no unit of meaning can be coded based on your category scheme. This is referred to as empirical saturation in qualitative research.
2. When reading the documents, you do not generate new ideas or additional insights from content analysis and interpretation.
3. You are stuck, hopelessly (e.g., going in circles, round and round and round).

Please be assured that the requirement that you do your best does not mean that your data processing, analysis, and interpretation must be perfect. At some point, you also need to finish the research process, answer your research question, present results, write up, and hand the paper over (e.g., to your professor or an editor).

7.3.4 Examples for research decisions during analysis

As you can see, the construction of categories and subcategories is an activity which is based on decision-making by researchers and students. Your

Table 7.3 Subdivision of the Unit of Meaning according to the Top Category 'Social Contacts' and Subcategory 'Student Group' (paragraph 7/6; Table 7.1)

Paragraph	*Journal Entry*
7/6/1	So I had a few more minutes before I attended another Zoom meeting organized by [student group] from [city name].
7/6/2	This involved auctioning off the [first-year students] as a way for them to be accepted [into the student group] – it was a lot of fun, and you got to know a lot of new people.
7/6/3	As a result of the circumstances during the pandemic, you unfortunately don't get to know that many people in this first semester, which is entirely online, so I was pleasantly surprised that so many online meetings were planned. This makes it possible to at least make some connections.
7/6/4	This auction went on a little longer, so I didn't get to bed until after midnight and just quickly set my alarm clock.

decision to inductively construct a category must be justified by your knowledge about the researched phenomenon in its social, spatial, and temporal context. Inductive research means that it is driven by the data which you process based on your existing stock of knowledge (i.e., theory-reflexive stage 1, see Figure 4.2).

You may also have to make pragmatic research decisions regarding content during the analysis process, which have to be justified according to the appropriateness-transparency quality convention (see Section 4.3.2). Depending on how narrowly or broadly we want to answer the research question "How are you organizing your daily life as a student at the beginning of the online semester 2020/21?," family social contacts should also be included in the analysis. The student appears to maintain close social contact with their family. The clearly close relationship is illustrated in Table 7.1 by the social contacts with the mom (paragraphs 1/2 and 10/4), the sibling (paragraphs 3/1 and 4/2), and indirectly at lunch (paragraphs 1/9, 3/3, 5/3, 7/3, and 8/2). If we answer the question broadly, we will have to consider that even without the restrictions due to the Covid-19 pandemic, students sometimes live with their family part of the time and sometimes permanently. Consequently, the decision to participate in the courses online and (for the time being) not to move to the place of study would have to be analyzed as analogous to the commuting distance between home and place of study.

We see that the decision between a narrow and broad analysis can be based on content-related reasons, for example defining the beginning of studies a new stage in life, including new social contacts and altered family support. The decision for a narrow or broad analysis of the data via inductively developed codes can be made based on both the cognitive interest and on the relationship

between the analysis and the scope of the written text. If you are writing a term paper for a seminar, there are usually specifications regarding the scope of the text (e.g., in the study and examination regulations and the guidelines provided by the degree program on written papers). This can be 10 to 12 pages for a course in a bachelor's degree, 15 to 20 pages for one in a master's degree, and correspondingly more pages for a final thesis. If you are at the bachelor's level and only have limited space to present your analysis, you should choose a narrow answer to the research question because you would not be able to present the elaborate manifest and latent content analysis in the space allowed, which would then be very frustrating.

Explaining the inductive category development for the other topics found in the empirical material would also go beyond the scope of this introduction to inductive qualitative content analysis. For the purpose of practicing inductive category development, you can choose one of the other six topics listed previously in Section 7.3.1. If you do this exercise, it is possible that you will discover an additional topic (Topic 8) in the empirical material that was overlooked. In an inductive qualitative content analysis, it can never be ruled out that a Topic or further categories in the text material has been overlooked. Social researchers face a cognitive challenge in the analysis process as they must use, connect, organize, and classify a great deal of empirical information.

In addition to making decisions during coding and inductive category construction, in the process of familiarizing yourself with the data by reading the text several times, you will identify several key segments and exemplary quotes. During the reading process, please take notes in your research journal (see Box 7.2), for example copy and paste the quote (including paragraph numbers and/or timestamps), and write a few words about why you selected the particular quote and what you want to use it for. At the end of the systematic reading process, you might have collected several exemplary quotes for the same purpose, which is good. You can compare these exemplary quotes to select and define a category, compare similarities and differences of content, which might lead you to deciding on a more fine-grained categorization or construction of subcodes. For transparency reasons (see Chapter 4 on quality conventions), it is important to also document such decisions in the annex of your term paper, bachelor thesis, or publication.

7.4 Qualitative content analysis: understanding manifest and latent content and making it understandable for third parties

7.4.1 Three-step understanding process of interpretation during qualitative content analysis

Building on the inductive development of categories and associated codes (Table 7.2), inductive qualitative content analysis allows you to interpret and explain manifest and latent content of data (see also Chapter 5). You can

imagine working on your analysis and interpretation of empirical material as a three-part[2] process of understanding.

1. **Comprehension analysis (reformulating the manifest content in your own words):** The first part may seem unnecessary and a repetition of the original text. However, it will prevent a situation in which your understanding of the original text only sounds plausible in your head. In addition, by writing down what is in your head, you will analyze the manifest content to a much greater extent. Depending on your preference and working style, you can record the comprehension analysis as a flow text or renumbered according to manifest content identified (see Table 7.3). Depending on the *content* of the original text, the reformulation can also be longer than the original text.
2. **Descriptive analysis (identifying the meaning in your own words):** For the descriptive analysis, you should summarize the text more concisely (again) and bundle your initial findings. In comparison to the narrative character of the comprehension analysis, you should choose a writing style for the descriptive analysis that you would use to write a description for a third party in your term paper, thesis, or dissertation.
3. **Discussion of manifest and latent results (empirically grounded and theory-reflexive naming of patterns, interpreting core statement(s), for example exemplary quotes):** The last part of the three-part process of understanding is the most comprehensive and demanding step to formulate. For the discussion of results, you must literally discuss the manifest and latent content and provide alternative explanations, make suggestions for interpretation, and provide the reader with ways to clearly understand the empirical material and its manifest and/or latent meaning. Theory is used to interpret and reflect on the empirical material as a whole and not just for one unit of meaning.

7.4.2 Systematic understanding through interpretation of manifest and latent content

We will use the unit of meaning 7/6 from Table 7.1 – which is assigned to the inductively created category of *student organizations* (category: *social contacts*) – as an example of qualitative content analysis. The student already reported in paragraph 2/7 that they attended an online welcome event for students from their home country. The example was also chosen to show that, despite anonymization, as a researcher you can use the information provided by autoethnographers, interviewees, and other informants. If you have this information, you should therefore use it in the analysis (e.g., summary or interpretation), but just always make sure that you do not inadvertently reverse the anonymization in the text. After you finish writing the description of results section, it is important that you go over the text again critically with a focus on the

anonymization. This may mean that you have to delete interesting and carefully formulated findings at this point as the protection of the informants must take precedence (see Chapter 3) for your work to be in accordance with the ethical guidelines of empirical social research (see Section 4.3.1).

For the identification of patterns and the interpretation of manifest and latent content in the data, it is important for qualitative content analysis that you

1. dig deeply into the text material and use all relevant empirical information;
2. always make the connection to the original text (= empirical grounding in data) clear (to avoid third parties getting the impression that you are *making* up results to make the analysis more interesting);
3. follow a seemingly static, but step-by-step and therefore systematic procedure for in-depth content analysis; and
4. document and make it clear both for yourself and for third parties (e.g., lecturers and thesis supervisors) how you arrived at your interpretations and results.

7.4.3 Examples of interpretation and reflexive use of theory

If you further subdivide the text into units of meaning, it can make it easier for you to dig deeply into the data and also to break down the text data, that is, to subdivide it into different units of meaning. Your decision on how to subdivide the text will depend on the complexity of the content. The aim of further subdividing the units of meaning is always to facilitate analysis and interpretation. As an example, we have subdivided the unit of meaning 7/6 into sentences (Table 7.3). If the unit of meaning had been a longer text sequence (e.g., several sentences), it might have been best not to divide the unit of meaning into so many parts. Dividing the units of meaning should always increase your own ability to understand the empirical data.

In example 7/6, the first sentence (= paragraph 7/6/1) is a transitional sentence that contains the name of the student group and the information that the meeting took place online as manifest information. In terms of content, your curiosity as an empirical social researcher might be piqued by sentence 2, which contains information about the unspecified – and possibly unusual for you – introductory activity of "auctioning off the [first-year students]." This introductory activity was fun for the student as they stayed in the meeting until "after midnight" (paragraph 7/6/4). After midnight is an indeterminate but manifest piece of information as we know that the student usually goes to bed at around 11 pm (paragraphs 1/12 and 2/8, Table 7.1) or 10:30 pm (3/6). Sentence 3 is a piece of information, or rather, a self-reflection, that the student inserted into their journal report of the introductory activity. An interpretation could be that the Covid-19 protective measures make it much more difficult than usual or expected, but not impossible, to get to know people during this new stage of life (i.e., their studies). We can also interpret from paragraph 7/6/3 that the

student was pleased that more people took part in the online meeting than they had expected.

Further manifest content found in the unit of meaning 7/6 is that the student is actively seeking contact with other students – both individuals and groups. The 'student group' is the group that the student has access to by virtue of their nationality. As a result, the student group is a way for the student to make contacts. As we noted previously in the summary (Section 7.3.1) and inductive category construction (Section 7.3.3), the student is actively pursuing their student life at the place of study. Therefore, an interpretation would be going too far, for example, to emphasize nationality as an exclusion criterion for other students or the like. This would contradict the pattern previously identified inductively (top category 'social contacts') and would go beyond the text.

However, the text represents the frame of reference for the inductive qualitative content analysis applied here. It is also manifest that the student is visiting the student group for the second time (paragraphs 2/7 and 7/6). From this, we can conclude that the student had a positive experience of getting to know the group and therefore decided to join the meeting again. The student was welcomed back by the group, which indicates longer-term involvement, and by participating in the ritual of the auction, the student has potentially become a member of the group for the longer term. The student only writes here that they "got to know a lot of new people" (sentence 7/6/2) in the student group, whereby it remains unclear a) how the auction was carried out and b) if being auctioned off as initiation ritual to become a member of the student group might have been irritating in any way.

7.4.4 Interpretation of latent content

While your understanding of the manifest content of the unit of meaning is methodologically located at the text level, your understanding of the latent content is directed behind the manifest text level and the explicit meaning of the individual words. In order to overcome the comprehension barrier of "auctioning off the [first-year students]," we must consider the latent content and make it clear through plausible interpretations with reference to the empirical material, its social context of data collection, and the category system. Your understanding of latent content should be plausible, and interpretations should be offered so that third parties can understand (e.g., lecturers and readers) a specific passage.

As a rule, empirical material offers more than just one possible interpretation. You should neither be cognitively lazy and satisfied with the first interpretation that comes to your head, nor should you expect to have to offer all imaginable interpretations. For self-assurance and to show that you have made an effort to decipher the empirical material if you can discuss two or more interpretations, for example, including contradictory interpretations, and develop an independent argument based on these. Interpretations not used in your

content analysis could be presented in the appendix of your paper to satisfy the appropriateness-transparency as well as iteration and distancing quality conventions (see Section 4.3).

In Table 7.1 and Table 7.3, the text data does not provide any further information that helps you to understand and interpret the latent content of "auctioning off the [first-year students]." The manifest information is a starting point as it describes the auction as an initiation ritual for accepting new students into the group with the goal of "getting to know people," and the phrase "a lot of fun" is used to describe the ritual.

Your understanding of daily life provides you with initial clues to understand the social significance of initiation rituals, for example, that these rituals are symbolic acts and as such follow rule-governed and festive procedures. Predefined rules of auctions include at least the roles of auctioneer, seller of an *item*, and more than two buyers of the *item*. For example, when auctioning a painting, the auctioneer opens with the minimum price set by the seller. The formal procedure that follows involves the buyers of the painting placing their bids until no higher bid is made. The auction formally ends with the symbolic blow of the hammer. It is completed when the highest bidder exchanges the money for the painting.

In the initiation ritual of auctioning off first-year students, the student is clearly in the role of the subject to be auctioned off. We can assume that they introduce themselves so that the group can get an idea of who they are and, for example, pitch themselves by talking about their skills, interests, and the like. Since the auction was fun for the student, it could also be that they, like the other first-year students, had to include something funny, tolerably embarrassing, or something similar when they introduced themselves or that they had to solve a task (e.g., describe or demonstrate ...). The role of the questioners is presumably taken on by members of the student group who bid on the first-year students (in the plural). In this context, the interpretation of "got to know a lot of new people" could be that the members also pitched themselves. For being auctioned by an individual mentor, monetary analogous qualitative offers for "driving up the price further" in the auctioning off of the first-year students could mean that, for example, comparing the interests, abilities, and task-solving skills of the bidders and the subject are the basis for deciding the outcome of the bidding competition. Considering the significance of the initiation ritual, there is a relatively high probability that a member with a formal role (e.g., chair) played the role of auctioneer and symbolically accepted new members into the student group as part of the ceremonial auction.

As already emphasized several times in this chapter, inductive qualitative social research is theory-reflexive. In contrast to deductive qualitative content analysis, however, theory does not guide the investigation from the outset (see Section 4.3.2.3). Accordingly, social science theory(ies) can and should be used to support the analysis and interpretation of latent content. In the context of (bidding) symbols and (initiation) rituals, for example, you can use

Geertz's (2002) classic theory on understanding cultural systems and associated practices, as well as studies based on this. Geertz (2002) defines culture in a socio-constructive way, that is, the interactive reconstructing maintenance, modification, and new construction of cultural practices and associated social structures.

Even if we do not know how the first-year students were auctioned off in reality, the theoretical approach taken by Geertz (2002) helps us to sociologically understand and interpret the analogies offered in the previous paragraph (the auctioning of a painting), for example the social code and the symbolic meaning of the "fun" initiation ritual. The student does not provide us with any further information about the initiation ritual, but the autoethnographic journal entry offers several possibilities for interpreting the establishment of social contacts in the digital space. For example, the student engages in social contacts in digital space with "fun", despite the restrictions of online meetings for 'getting to know people'.

The student's focus on "establishing social contacts" cannot conceal the fact that an auction is also a practice of valuation through (social) pricing, as the comparison with the auctioning of a painting made clear. According to the theory of the sociology of valuation (e.g., Lamont, 2012; Nicolae et al., 2019; Vatin, 2013), an object or – in the case of auctioning off first-year students – a subject is evaluated. However, since auctioning off first-year students involved several subjects, the evaluation as a comparison of (qualitative) values would arise if one student achieves a higher auction value than another student. But since the student emphasizes how "fun" the auction was and how they stayed up late to get to know people better, the moment of evaluation in the auction does not seem to have been negative or of notice, as the student did not bother mentioning the auction price in the autoethnography.

As a result, if you continued to analyze the social significance of valuation, you would be leaving the frame of reference of the data that is important for the inductive qualitative content analysis. Moreover, further pursuing the interpretation of the auction would not be providing an answer to the question in focus here: "How are you organizing your daily life as a student at the beginning of the online semester 2020/21?" For the interpretation, better understanding the auctioning off, of the first-year students through the auctioning of a painting was of value as it allowed the social situation – the various roles and the process including the symbolic meaning of auctions – to be defined. The thought experiment was helpful for our interpretation and offered possible understandings of the unusual and undefined situation in the data of "auctioning off the [first-year students]."

You see that the interpretation of categorized content embedded in units of meaning is particularly challenging when the meaning of the information in the empirical material is ambiguous. Ambiguous means that the manifest content of one or more units of meaning (text excerpt = empirical context of discovery) point to implicit, latent structures that should be made explicit as biographical, emotional, normative, and social situation-related statements. As explained in

Table 7.4 Schematic Overview of a Three-Step Understanding of Units of Meaning

Original Text	Part 1: Comprehension Analysis (reformulating the manifest content in your own words)	Part 2: Descriptive Analysis (identifying the meaning in your own words):	Part 3: Discussion of Manifest and Latent Content (naming patterns, explaining and interpreting core statement(s), as well as theory-based reflection):
Section 1			
Section 2			
Section n			

Chapter 2, the aim of in-depth understanding is to gain deeper insight into the latent content of empirical data (Step 4 in Figure 7.1). Let us take a step back from the empirical material and to methodologically structure the ongoing interpretation efforts. The three-step process of understanding units of meaning defined in Section 7.4.1 can be organized and used as a procedural guide as shown in Table 7.4.

Table 7.4 suggests a procedure for trying to understand as well as for moving forward with interpretation and, at the same time, documenting the interpretation steps in a transparent way. Given its paper-like format, OpenOffice Writer and Microsoft Word may not offer enough space to document in-depth content analysis (see fourth column in Table 7.5). You could therefore use OpenOffice Calculator or Microsoft Excel, which both allow you the flexibility to set any number of table columns and rows as well as the respective column width.

Table 7.6 shows how paragraph 7/6 from Table 7.1 with the two subcategories 'student group' and 'not connected to courses' is again inserted into the scheme from Table 7.3. As you can see, the interpretations are not limited to the two subcategories but also take into account the subcategory 'face-to-face' and contribute to understanding patterns of 'social contacts' (top category) in the online semester 2020/21.

7.5 Inductive qualitative content analysis with more data and for more than one case

7.5.1 Basics of inductive qualitative content analysis

In Sections 7.2.3 (structuring analysis), 7.3.1 (gaining an overview), 7.3.2 (coding the empirical material), and 7.3.3 (inductive development of categories), the methodological process was presented using a single autoethnography as an example empirical document. However, if you are analyzing the content of two or more documents – no matter whether they are autoethnographies, group discussions, interviews, or documents on platforms of political parties – you will have to repeat the gaining an overview, coding, and inductive development

Table 7.5 Example of Paragraph 7/6 for In-Depth Understanding of Units of Meaning

Original Text	*Part 1: Comprehension Analysis (reformulating the manifest content in your own words)*	*Part 2: Descriptive Analysis (identifying the meaning in your own words):*	*Part 3: Discussion of manifest and latent content (naming patterns, explaining and interpreting core statement(s), as well as theory-based reflection):*
So I had a few more minutes before I attended another Zoom meeting organized by [student group] from [city name]. This involved auctioning off the [first-year students] as a way for them to be accepted [into the student group] – it was a lot of fun, and you got to know a lot of new people. As a result of the circumstances during the pandemic, you unfortunately don't get to know that many people in this first semester, which is entirely online, so I was pleasantly surprised that so many online meetings were planned. This makes it possible to at least make some connections. This auction went on a little longer, so I didn't get to bed until after midnight and just quickly set my alarm clock.	The student can take a few minutes to catch their breath before taking part in the student group meeting on Zoom. Part of the initiation ritual to the student group is the "auctioning off" of first-year students. The purpose of the auction is for the students to get to know each other, and it was a lot of fun for the student. The fun factor was emphasized by the concluding remark that the student didn't go to bed until after midnight. The student is pleased that (many) other students are also using the opportunity of online communication – which, in contrast to analog meetings is perceived as tedious and obstructive – to get to know new people.	Meeting new people is a high priority for the student, and the student appears to be very sociable and interested in "new people." For the student, getting to know people seems to be part of getting started at university. In addition to making contact individually, the student also takes advantage of programs and events organized by students to get to know "new people." The student puts the restrictions imposed by the Covid pandemic into perspective and emphasizes that it is still possible to get to know people online, including through games to get to know each other, and that it can even be fun. One example is the introductory game "auctioning off of first-year students," which also serves as the initiation ritual for the student group.	[The previous explanations, interpretations, and theory-based reflections on the initiation ritual with the aim of getting to know people cannot be repeated here for reasons of space.]

Table 7.6 Example of Using Numbers to Connect Differentiated Subcategories to the Top Category 'Social Contacts'

Category	*Subcategory level 1*	*Subcategory level 2*	*Subcategory level 3*
Social contacts (e.g., top category 3 due to example analysis for Topic 3)	Face-to-face (3.1)	Connected to courses (3.1.1)	Zoom (3.1.1.1) Breakout sessions (3.1.1.2) Study group (3.1.1.3)
		Not connected to courses (3.1.2)	Student group (3.1.2.1) Others (3.1.2.2)
	Messages (3.2)	Email (3.2.1)	–
		Messenger (e.g., Signal) (3.2.2)	–

of categories process once, twice, or n times depending on the number of documents to be evaluated.[3]

The n is limited, for example, when you are conducting three interviews for a bachelor's thesis or five interviews for a master's thesis. However, you must justify the n in other cases, for example, if you have 50 autoethnographies for a secondary analysis and only want to analyze a selection of them inductively and qualitatively. If you have collected the data yourself, that is, you are conducting a primary study, you must justify the n of group discussions, interviews to be conducted, or the like in the research design (for case selection, see Section 3.2.1; for qualitative sampling, see Section 6.4).

The student autoethnography used as an example would be a case for the (relatively) normal start of studies despite the problems caused by the Covid-19 pandemic. If you want to depict the variety of or even extreme cases (see Table 3.2) of the social conditions at the start of Winter Semester 2020/21, you could, if available, select the case of a student with care responsibilities who, in addition to starting their studies, has to provide childcare because the children are temporarily unable to go to daycare or school due to positive Covid-19 test results. Other good case decisions would be for an autoethnography of a student who has to partially finance their studies through a part-time job or of an older student who is able to study something they are interested in after retirement.

7.5.2 More data require a more fine-grained category system

The subcategories in Table 7.2 are a sufficient type of categorization to identify relevant text passages and analyze manifest content and patterns of daily student life. However, if the text material is more extensive than in Table 7.1, it would be best to assign the inductively developed subcategories of the top category 'social contacts' in a more analytically differentiated way. In Table 7.3,

for the top category 'social contacts', three (branch) subcategories are constructed for example purpose to code Topic 3 (getting to know new people in the digital space). In the Table 7.3 example, we use numbers as an option to identify the hierarchy according to levels of subcategories in a category system, and to avoid getting lost in a sub-subcategory, sub-sub-subcategory, etc. maze. Accordingly, a category system is also known as a category tree and should be easy to understand for you and others.

The pragmatic numbering of Topic 3 as top category 3 builds on the assumption that Topic 1 (getting started with studies) and Topic 2 (organizing the new everyday life as a student) will provide for a top category 1 (e.g., 'studying' or 'how to study') and 2 (e.g., 'organizing student life' or 'new everyday structures'). Of course, the numbering as well as the order of categories in the category system can be modified at a later stage of data processing, for example during the coding process or when you are finished with inductive category construction.

The further differentiation of subcategories means adding two or more attributes to a subcategory. In principle, this is similar to subcategories capturing specific characteristics of a top category. In Table 7.6, you can see the example of the subcategory 'Messages' (subcategory 3.2) with the subcategories 'Email' (subcategory 3.2.1) and 'Messenger' (subcategory 3.2.2). The respective category names should be easy to differentiate and not be difficult to understand for others. Choosing 'Messages' in subcategory 1 required some reflection because the name needed to be unambiguous and follow the principle of exclusiveness for category names. Text communication was not chosen because with 'email' and 'messenger' you can also send images and/or audio messages in addition to text. The use of digital and online in the category names was also excluded as both 'face-to-face' communication via 'Zoom', and the exchange of messages via 'messenger' services are also considered communication via social media.[4]

7.5.3 And the whole thing n-times from the beginning again

In Sections 7.2.3 (structuring analysis), 7.3.1 (gaining an overview), 7.3.2 (coding the empirical material), and 7.3.3 (inductive category development), the methodological process was presented using an autoethnography as an example empirical material. However, if you are analyzing two or more autoethnographies, you must repeat the steps laid out in the sections on gaining an overview, structuring analysis, and inductive category development of inductive qualitative content analysis once, twice, or n-times, depending on the n = number of documents to be analyzed.

You must, of course, have established the n of cases to be compared in the inductive qualitative content analysis when you create the research design with a view to the research question and taking into account the scope of the data. To reflect the diversity of social conditions at the commencement of studies in Winter Semester 2020/21, we could add the following from the sample of 50

autoethnographies to the example of the autoethnography of student idno05 coded here as a case for the (relatively) full-time student who's start of studies was impaired by the conditions of the Covid-19 pandemic: Autoethnographies of students who have care responsibilities such as childcare, as well as an autoethnography of students who have an avocation (see Table 8.6). This third case decision that would be suitable for a qualitative diverse sample would be an autoethnography of a student who has to partially finance their studies through a part-time job and therefore has to cope with overlapping work and study times.

You can see that through case selection new inductive categories and associated codes have to be developed (e.g., on part-time employment and children), which do not occur in the autoethnography of student idno05. Given the diversity of the three cases (full-time student, and students with care responsibilities and part-time job), it is also clear that further subcodes will be added, for example, to Topic 2 and the category "Organizing your Studies" (Section 7.3.1). As explained in the introduction (Section 7.1.1), we would, therefore, again strongly advise against coding all empirical material before all of the categories have been created on the basis of your entire empirical material.

Notes

1 Please note that 'data' and 'empirical material' are used as synonymous terms.
2 Similar analysis patterns are also found in other qualitative analysis techniques (e.g., Bohnsack et al., 2013).
3 n conventionally represents a numerically unclear number (e.g., of cases to be evaluated) and should be familiar to you from math classes at school.
4 Incidentally, WhatsApp is an abusive data monger with little data protection – the provider Signal is a data-secure alternative; you are free to make a donation to support this freeware.

8

INTRODUCTION TO QUANTITATIVE CONTENT ANALYSIS

This chapter presents an analysis technique that you can use to quantitatively analyze the content of 500 pages or more of text. This technique is suitable for larger interview studies and secondary analyzes of process-produced data (e.g., party platforms or Twitter). Using the example of a secondary analysis (like in Chapters 6 and 7, the autoethnography on "How are you organizing your daily life as a student at the beginning of the online semester 2020/21?"), the sequential analysis is explained based on the AntConc software (freeware; sociolinguistics or political linguistics) and qualitative data analysis (QDA) software. Guided by the research interest and research question(s), we explain step by step how to use search words to systematically understand the text corpus and how to analyze the information found. The semi-automated indexing of the text corpus and quantitative content analysis produces an overview categorized by the search words (keywords), which is also referred to as distant reading. In contrast to in-depth qualitative content analysis, the aim of quantitative content analysis is to record the manifest content and to use this to draw findings, make interpretations, and draw conclusions. However, quantitative content analysis should be followed by in-depth, qualitative content analysis, which is explained at the end of this chapter.

8.1 Introduction

The availability of text in digital form and, in some cases, free analysis software enables and/or encourages quick content analysis of text data and documents of all kinds. Certainly, a quick-and-dirty analysis using software at least offers more insight than just superficial and selective browsing through extensive text corpora and document collections. With (slightly) more effort, however, you can carry out systematic data analysis by combining software-supported data analysis techniques.

DOI: 10.4324/9781003496397-10
This chapter has been made available under a CC-BY-NC-ND 4.0 license.

We will show you why data processing and proper quantitative data analysis are so important using the example of word clouds. Figure 8.1 displays all of the words in the autoethnographic text corpus. This is why you can mainly see (possessive) pronouns, conjunctions, fillers, modals, and many other words that are important in communication but do not carry the sense or meaning of the text corpus. The attentive observer will find the word 'I' in a central position, which is typical for autoethnographic data.

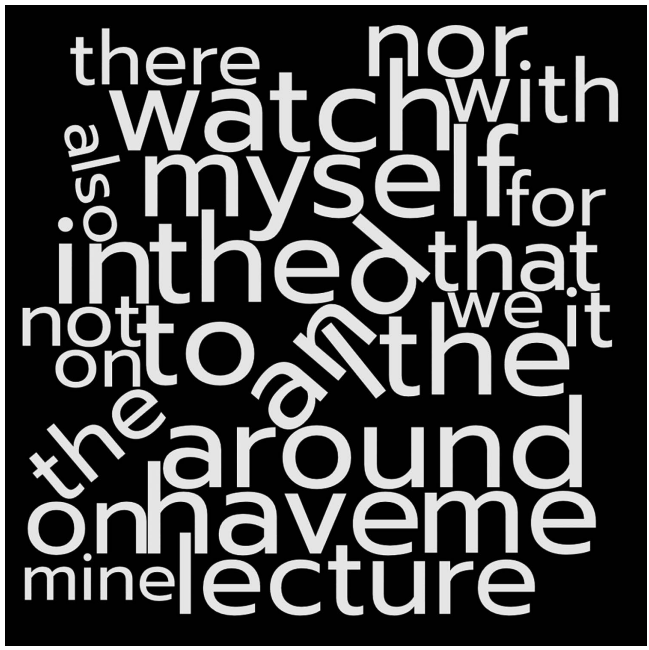
The code word cloud[1] (Figure 8.2) is organized according to the number of codes attributed to the texts by qualitative data analysis (QDA) software autocoding. In Figure 8.2, for example, the word 'moodle' (proper name of online learning platform) is written in the lower case here as all words – including proper nouns and names – are usually converted to lowercase during data pre-processing (see Section 8.3.1). In this way, we avoid the problem of incorrect typography (e.g., in the beginning of a sentence, verbs are written in the uppercase). In addition, the conversion to lower cases serves to prepare for lemmatizing and stemming, in which the various word types are reduced to the common stem (see Section 8.4.2). Otherwise, you will get several word tokens that mean the same thing but are sometimes capitalized and sometimes not. The word *token* can mean feature, sign, or occurrence.

The word clouds in Figures 8.1 and 8.2 are essentially based on quantified, that is, counted word lists, but they also show word frequency. The word clouds are based on a simple count of words in the text corpus (e.g., in AntConc). As shown selectively in Table 8.1 – the text corpus consists of about 10,000 different words and a total of about 150,000 words (about 500 A4 pages with 300 words each) – the text corpus determines which words appear at a high frequency. The word 'I' is by far the most frequent word as a result of the first-person narrative perspective typical of autoethnographies. However, you may have noticed that articles such as 'the' and 'an' occur less than half as often in the text corpus and nouns such as 'lecture' only 1,279 times. When you are directly transferring keywords into categories, you have to make sure to avoid duplicating keywords. For example, Table 8.1 contains the keyword 'topic' (Rank 8) and its plural 'topics' (Rank 18). The options for combining keywords are explained in Section 8.4.2.

8.1.1 Sociolinguistic basics of quantitative content analysis

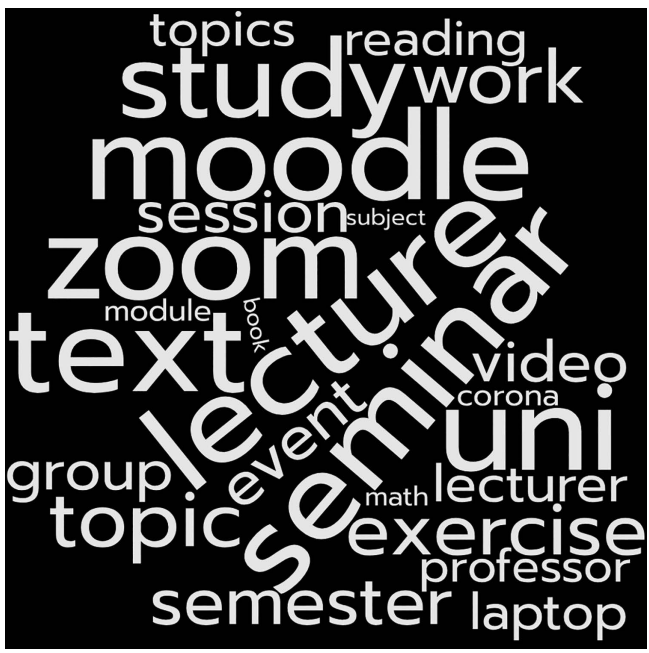
The results, interpretations, and conclusions of inductive quantitative content analysis remain at the level of manifest content. Consequently, quantitative content analysis can contribute to answering the manifest 'what' question but not the latent 'why' question of communication (see Box 8.1, Section 2.1, and Table 2.1). By systematically extracting text passages using word tokens, the information is decontextualized. Word token is a term from linguistics. In quantitative content analysis, the word *token* refers to occurrences of a word in a document or text corpus. A word as an empirical occurrence – that is, the

Figure 8.1 Word cloud of the full text of the autoethnographies.



Source: Autoethnography WS 2020/21

Figure 8.2 Word cloud of codes.



Source: Autoethnography WS 2020/21

Table 8.1 The 25 Most Frequent Words and Word Tokens in the Text Corpus

Words			*Word Tokens or Keywords*		
Rank	*Frequency*	*Token*	*Rank*	*Frequency*	*Token*
1	8551	I	1	1279	lecture
2	4875	and	2	380	seminar
3	3966	the (*die*)*	3	321	zoom
4	2754	the (*der*)*	4	284	moodle
5	2673	to	5	265	uni(versity)
6	2062	around	6	261	text
7	1990	have	7	228	study
8	1950	me	8	164	topic
9	1921	in	9	155	work
10	1898	myself	10	154	exercise
11	1625	that	11	142	semester
12	1559	the (*das*)*	12	129	event
13	1474	watch	13	126	session
14	1379	with	14	120	video
15	1329	it	15	119	group
16	1279	lecture	16	109	lecturer
17	1224	not	17	106	laptop
18	1212	on (*auf*)*	18	104	topics
19	1200	nor	19	102	reading
20	1174	there	20	96	professor
21	1172	for	21	64	covid-19
22	1158	on (*an*)*	22	63	module
23	1125	we	23	56	math
24	1042	mine	24	48	subject
25	1018	also	25	47	book

Source: Autoethnography WS 2020/21; * Please note that the English equivalent of 'the' has three different forms in German: *die* to refer to nouns with a feminine gender, *der* to nouns with a masculine gender, and *das* for nouns with a neutral gender. In German, the word 'on' can also mean 'at', 'to', or 'as' and the word *auf* can also be 'upon' or 'on top.'

keyword underlying a word token for the sociolinguistic analysis on the word level – methodologically fulfills a dual function: a targeted search for information in an extensive word dataset and for the sociolinguistic meaning attached (e.g., Coulmas, 2017; Niehr, 2014). For the information search in the word data, it is keywords or their word tokens that are used as search words for quantitative content analysis. The keyword-driven meaning attached distinguishes between the basic meaning of words as terms (linguistics: denotative meaning) and secondary meanings of terms (linguistics: connotative meaning). In linguistics, a

further distinction is made between

> "denotative and emotive components of meaning. This difference shows us that the meaning of expressions [or terms] is not limited to designating things. When we name things or circumstances, we often make an assessment at the same time. For example, certain expressions evoke specific connotations in the recipient" (Niehr, 2014, p. 67; authors translation from German).

Box 8.1 Three steps of analysis to remember

Analysis Step 1: Understanding the context of the investigated social phenomenon [(a) subject(s), (b) object(s), (c) space(s), and (d) time/periods].

Analysis Step 2: Understanding content by explaining, interpreting, and inferring.

1. Manifest content ["What" question of communication (1st level of meaning)?]
2. Latent content ["Why" question of communication (2nd level of meaning)?]

Analysis Step 3: Understanding the audience [summarize results specific to the audience, organize for presentation, and/or write up for publication].

The search guided by keywords or word tokens and the sentences or paragraphs indexed as a result make it possible for researchers to recognize structures and patterns in the data. However, quantitatively guided content analysis does not provide in-depth insight into the latent content of the individual texts in the corpus *per se* – this occurs through content analysis according to the qualitative paradigm. At this point, it should be emphasized again that there is no methodological better or worse when it comes to qualitative and quantitative content analysis, but that the research interest should guide the choice of method.

Describing your research interest is therefore also the first step in creating the research design (shown schematically in Figure 8.3). The research interest is formulated in one (or more) research questions. Explicitly writing down the research question(s) is essential in order to make the research process transparent and clear for third parties. A research question can be motivated by a research gap discovered in literature, be justified by a theoretical question, and/or help to capture a not yet investigated social phenomenon that the researcher observes.

8.1.2 Research process in a step-by-step flowchart

In a study involving the collection of primary data, meaning when the researcher collects the empirical data themselves, the research question guides the entire research process. In the secondary analysis of existing data or datasets, the

Figure 8.3 Flowchart of a quantitative content analysis that combines analysis techniques.

Step 1

Formulate the research interest as a question. Formulate theory-based hypotheses for deductive, that is, not purely data-driven research.

Step 2

Get familiar with the data: read some of the documents in the text corpus and make notes in your research journal to get a "feel" for the data and content.

Step 3

Prepare the data for the software you will use (data cleaning):
1. Convert the file format
2. Clean the documents (e.g., remove illegible symbols)

Step 4a

Conduct quantitative content analysis with AntConc:
1. Import all documents in the corpus (ideally as separate units of analysis)
2. Create a word list
3. Identify meaning words and convert them into word tokens (e.g., by lemmatizing and adding collocates)
4. Use word tokens to analyze the text corpus
5. Save intermediate results
6. Prepare word tokens as word lists
7. Keywords in context (KWIC)
8. Concordance diagrams

Step 4b

Quantitative content analysis in QDA software:
1. Import all documents (e.g., all 50 autoethnography documents as separate files)
2. Use word tokens identified in AntConc keyword search to autocode documents
3. Check results of autocoding according to search words for mistakes. Most likely you have to repeat this action several times during data processing

Secure results:
4. Category system (incl. coded frequencies for each category), and tables and figures for presentation of results (e.g., word clouds)
5. Extract relevant parts from document corpus order by the category system

Step 5

Analyze the data of the automatically coded text passages:
1. Manifest content (answer "what" question as analysis step 2a; repeat separately for each code)
2. Theory-based reflection of the results (with a deductive approach)

Step 6

Summarize the results and prepare a presentation, term paper, thesis or publication

Step 7

Securely archive the data and, if possible, prepare it for reuse in secondary data analysis.

research question takes into account the social (e.g., cultural, economic, and political) and (digital) spatial context[2] but is limited by the available data. In the case of research-produced data, the research question of the primary analysis can also be used for the secondary analysis, as described in the following example of the teaching and learning research project of an autoethnography titled "Two Weeks of Studying in Winter Semester 2020/21." The example and results generated are used to illustrate the methodological approach presented in this chapter. After a brief description of the research project and data protection requirements, the chapter structure is based on the research design, which is shown schematically in Figure 8.3.

8.2 Description of data collection and example data corpus

8.2.1 Teaching and learning research project of an autoethnography titled "Two Weeks of Studying in Winter Semester 2020/21"

The teaching and learning research project of an autoethnography titled "Two Weeks of Studying in Winter Semester 2020/21" was part of a course at a university in Germany. Given the Covid-19 restrictions (e.g., social distancing), students were not able to learn to apply research methods outside in the field. For qualitative social research in general and content analysis in particular, the social context in which the empirical data was collected is of great importance (see Section 2.2). In sociology, the term social context summarizes cultural, economic, political, etc. aspects of a context. There are two main reasons why the context of data collection is so important. First, as the reference point of content analysis, the empirical material was collected in a social context with specific spatial and temporal characteristics.

As with journal entries, you can use autoethnography to observe yourself in a methodologically guided way (Ellis et al., 2010). This type of self-observation allows you to explore and reflect on your own daily routines, (new) patterns, and rituals practiced (un)intentionally. The content of the autoethnographic self-observations was focused on how the second wave of Covid-19 – which began at the beginning of November 2020 – influenced studying and, for new students, the start of their studies at university. The research question guiding the study for the students was: "How are you organizing your daily life as a student at the beginning of the online semester 2020/21?" The task was for the students to write around 10 autoethnographic records over a period of 14 days. The students received an example text of an autoethnographic record from another research project that served as a guide, and this was supplemented by methodological literature, explanations of autoethnography in a video (available on online learning platform Moodle), and guidance from the lecturer in the course.

In terms of methodology, the following should be noted when using autoethnography as an example of quantitative content analysis. The qualitative data

Table 8.2 Characteristics of Autoethnographic Data (According to the Categories in Table 3.1)

	Research-Produced Data **Large-n Case Number Studies**
Design	
Data types	Autoethnography
Aim	Answering the research question
Operationalization	Inductive and theory-reflexive (see Section 4.3.2.3)
Data Genesis	
Sampling	Sample of students in a research methods course (self-selection by students)
Data collection	Subjects (= students) and social phenomenon (= start of Covid-19 online semester 2020/21)
Data type	Text
Archiving	
Complete	Yes
Analyzes	
Data storage	Researchers
Data analysis	Quantitative analysis method
Type of analysis	Secondary data analysis
Application	
Result	Research (and some practical) implications (e.g., guidelines for action)/conclusions

was collected with the method of autoethnography and analyzed by students, first. Here, the secondary data analysis decouples the data collection method (= journal writing by the students) from the quantitative data analysis technique as summarized in Table 8.2. In this handbook, we used one autoethnography as empirical example for the introduction to deductive (Chapter 6) and inductive qualitative content analysis (Chapter 7).

8.2.2 Data protection, anonymization, and declaration of consent

For the secondary data analysis, the students' text data cannot be used without their consent and only after, for example, names of persons and places have been anonymized (see Section 7.2.2). Information about the planned use and anonymization of the data was given to the students in an information leaflet. The information sheet is called a leaflet because it is intended to convey important information to the participants of the study in a concise manner on one or a maximum of two pages (see brief definition in Box 8.2 and Section 3.4).

The leaflet should highlight information on data protection, that is, the handling of personal data by researchers. It is essential to use easily understandable

language, even if reference is made to the legal framework for data protection. The clarity of the information is all the more important as participants in the study must explicitly give their consent in writing by signing it. This is referred to as a declaration of consent to the processing and, if applicable, disclosure of the data (see Section 3.4.1). Of the [number of students registered deleted for anonymization purposes] students of [name of course], 50 signed the declaration of consent that their autoethnography may be used for research and teaching.

Box 8.2 Brief definition of information leaflet

Information leaflet:

1. Include a brief introduction of the researcher and institution, as well as contact details (e.g., email address and telephone number).
2. Clearly state the focus of the study, but without indicating the answers expected.
3. Name the study population and explain why the persons approached should be studied and interviewed.
4. Clearly state expected time and effort; otherwise, it is likely that your data collection will be incomplete (e.g., if interviewees reserve too little time).
5. Specify how the interview data will be used, for example, for research, publication, or a report, and be sure to include information about anonymization and data protection.

The consent is contingent on the anonymization of names of persons, subjects, locations, and other identifying factors (for more information about anonymization, see Section 7.2.2). A first step toward anonymization is the assignment of identification numbers (idno for 'ID(entity)' and 'no' from English 'number'). We have chosen to use idno in this book because of its brevity, but you could use another identifier such as Student_05. For example, if you conducted three interviews, you could also choose Interview_1, Interview_2, and Interview_3 for anonymization. It would also be possible to assign other real name aliases, whereby, if relevant for your study, you should respect the biological gender in heterogeneous groups of interviewees. Otherwise use 'they' and 'them' to support anonymity.

For anonymity purposes, interventions in the original empirical text should be made uniformly and be clearly visible. In the example tables in this chapter, square brackets are used to replace names (e.g., [professor]), course in [study program 8], and marketplace in [city]). In general, common (writing-related) conventions should guide the researcher's practical decision to mark the researcher's interventions in the text (e.g., three dots for omissions). Accordingly, an [...] was chosen for deleted text. Concise alternate terms should ideally be chosen, whereby it is irrelevant in terms of content whether, for example, [city name] or [name of city] is used.

Anonymization should also be guided by content-related aspects. For example, you should omit the name of the degree program and anonymize it by using [major] and [minor] as placeholders. Journal entries for various compulsory and compulsory elective courses in the student's degree program and that the student is attending an elective course that does not belong to either the major or minor are indicated by [additional course outside of field]. The daily schedule is also relevant for a journal entry. Instead of just deleting class times to prevent them from being searched for, since the Internet doesn't forget and search engines are good, we replaced them with relatively clear times of day such as [morning], [afternoon], and [early evening].

In some cases, the text can only be anonymized by making more significant changes. However, the core of the empirical information should remain the same and not be distorted by the changes. Although you will be weighing the pros and cons of using critical information and results, please be 100% assured: This is not a conundrum! Even in cases of minor doubt, always rate data protection and ethical considerations higher than reporting the information in your results. Even if you cannot ground the most striking insight into your data, which results in a finding being less meaningful – it is not worth it. Nowadays, social media and other platforms provide so much information that a skilled data scientist can connect even small dots with little effort. Such a scandal would harm your personal credibility as well as the credibility of empirical social research, which will lower the willingness of people to participate in studies.

8.2.3 Raw data and data preparation

The 50 student autoethnographies provide content-rich empirical examples, which will be used as data to explain quantitative content analysis. Most of the students provided their autoethnographies as Word files. But although it was requested that they submit their autoethnographic recordings as a Word file, a few students sent a PDF file. As we expected, there was a problem with line breaks in the text, which is caused by outdated PDF versions when you convert the text by using the 'Save as text' option.

Manually cleaning the data might not be an elegant solution, but it is relatively time-efficient and manageable for the amount of text of 50 autoethnographies. It works well to view the original (e.g., PDF-file) and the converted electronic file next to each other on the screen and start cleaning from the bottom in the converted document (e.g., txt or Word file) as the lines are not always the same length, for example due to a line break and the end of a paragraph. For example, you can place the index finger of your left hand on the 'up-arrow' key and work alternately with the two-finger technique using the right hand with the thumb on the space bar and the little finger on the 'Delete' key in a regular rhythm.

This is monotonous work, which may be a little nicer if you listen to music; from experience, we can recommend taking regular breaks when you are

working with longer documents so that you do not get a cramp in your hand ☺. Do not create continuous text, but instead make sure to restore the paragraphs like in the original document, which usually reflect a coherent meaning.

Or a more elegant solution is to use Word or OpenOffice Writer to restore the missing line breaks. In OpenOffice Writer, simply use the 'Find and replace' function to do this. The dollar sign '$' in the 'Find' field is the command for 'Replace line break with,' whereby a [space] must then be entered in the replace field.[3] To clean up larger document corpora, it is advisable to use a Python script, and you can do an Internet search for ready-made and easy-to-use scripts that also correct incorrect characters (e.g., software application icon-v-loop).

8.2.4 Rationale for using data analysis software

Nowadays, the use of software for data analysis is standard, however, few still work with paper and stickers. Software supports data organization and makes systematic data analysis understandable (see appropriateness-transparency quality convention, Section 4.3). There are several reasons to select AntConc and QDA software for quantitative content analysis, we have been working for many years with the QDA software MAXQDA (see information on copylefts and copyrights in Section 1.5). First, both analysis programs can be used with a wide range of computer models and operating systems (which, of course, also applies to other programs). AntConc is freeware, and you can download and install AntConc free of charge at *www.laurenceanthony.net/software/antconc* for Macintosh OS X, Linux, and Windows. Second, for MAXQDA, like for other firmware (e.g., atlas.ti and NVIVO), and AntConc, you find tutorials and learning videos online. The analysis techniques presented here use a combination of AntConc and MAXQDA to help you to systematically familiarize yourself with the empirical material and the research process for quantitative content analysis.

It is important to note that these data analysis programs were originally developed for different areas of application. MAXQDA was developed for qualitative data analysis. AntConc is a program for linguistics and literary studies, which was developed for corpus linguistics, but can also be used for interdisciplinary sociolinguistic studies. As a result, the possibilities for analysis in AntConc focus on language in an extensive text corpus. AntConc is easy to use software for semi-automatically generated results, which need to be processed further for interpretation, as will be explained in Section 8.4.

8.3 Identifying search words and keywords in the corpus with AntConc

8.3.1 Basic settings in AntConc

Like other freeware, AntConc can be downloaded 'as is' – the programmer assumes no liability for any damage caused by product usage. With AntConc 'as is' means, for example, that only txt files can be imported for analysis. If you

have Word files, you can convert these to txt files in Word by using the 'Save file as ... ' option and selecting the file type 'Text only'.

AntConc offers a straightforward application interface and does not save any analyses. As a result, you have to adjust the settings, for example, for the recognition of western European languages based on Latin, by selecting them in the menu each time you restart the program. Go to 'Global Settings' 'Character Encoding' and click the 'Edit' button under 'Current Encoding' and then select 'Western Latin1 (iso-8859–1)' from the drop-down window under 'Standard Encodings'. Then click on the 'Apply' button at the bottom of the window. If you forget to configure the language settings, special language characters, for example, German *ä*, *ö*, und *ß*, will not be recognized in the text. If you use AntConc on a regular basis, you adjust the language settings (almost) automatically – and this step will then take less than five seconds.

AntConc's user interface is very straightforward and easy to navigate. The files of the text corpus to be analyzed will appear in the left frame 'Corpus Files' if you select (e.g., by holding down the 'Ctrl' key) the txt files to be imported (AINSI formatted, e.g., in the editor) from the corresponding folder on your computer by clicking 'File' 'Open File(s).' All analysis instruments (tools) can be accessed via the tabs in the main window in AntConc. There are specific settings for each tool, which you can adjust using the 'Tool Preferences'. The tools used to analyze the students' autoethnographies are explained next with helpful information about how to use them.

8.3.2 Functions to identify keywords as search words

Word tokens can be used to index the manifest content in context for a meta-analysis. As explained previously in Section 8.1.2, word tokens fulfill a double function in terms of methodology. First, word tokens are central for the assessment of sociolinguistic meaning, that is, the analysis of the manifest content after the empirical indexing of the text corpus is completed (Step 5 in Figure 8.3). Second, word tokens are used in the targeted search for information within an extensive word dataset. In the context of a quantitative search for information, keywords are therefore defined as search words. AntConc has several functions to identify the central search words for the analysis.

Word lists: Word lists ordered according to empirical occurrence offer an initial quantitative point of reference in large text corpora (see Table 8.3). The order of the word lists allows you to identify the search words for a targeted quantitative content analysis. As in Figure 8.2, it is best to create a word list that is case-insensitive, that is, one in which all words are lowercased no matter if they would normally be lower- or uppercased (this is the default setting under 'Tool Preferences' 'Word List Preferences' under 'Other options' check 'Treat all data as lowercase'). Depending on the word or composite search – composites are compound words (e.g., dataset and examination regulations) – word lists can be sorted according to decreasing or increasing frequencies and according to the beginning or end of the word (AntConc button 'Ordered by ... ').

AntConc also has the following functions to help you create a key word list based on a search word.[4]

1. **Word clusters**: In the selected text files, a list of word clusters (e.g., sorted by frequency) is created that occur in the text next to the search term. Search strings are then checked in the data and, if necessary, modified and supplemented with synonyms.
2. **Collocates**: Creating an ordered list of collocates of the search term offers an alternative or supplement to the word search with word clusters.
3. **Concordance**: AntConc creates concordance lines for **K**ey**W**ords **I**n **C**ontext (KWIC) in selected texts. For content analysis, a concordance involves the creation of an alphabetically ordered list of search words in a document according to the criterion of match.

After deleting filler words and conjunctions, the word list can be used in the four AntConc functions described previously for further data cleaning. For example, the alphabetical word list shows 77 hits for 'difficult', which is an expected and meaningful word based on our contextual knowledge of the social phenomenon of (online) semester under investigation. However, the automated word search in the concordance tool for the search term 'difficult*' includes all words beginning with 'difficult ... ' indiscriminately by adding an asterisk * and does not select according to content criteria. As a result, the concordance presents hits on what is difficult and also what students did not find very 'difficult' in their daily university life (*KWIC Hit* 78 in Table 8.3). By adjusting the 'Search Window Size' setting from 50 to 100 characters (see Figure 8.4), AntConc displays enough contextual information to give you a feel for the various contexts and to be able to classify what students found "difficult" at the beginning of the 2020/21 online semester.

8.3.3 Identifying keywords as a first step in the analysis

The example shows that you already start analyzing the data after a short time (Step 4a in the flowchart, see Figure 8.3), whereby data analysis includes getting a feel for the data. After all, you are in the middle of a quantitatively guided data sorting process. If the text corpus were more extensive, you would have to remove all of the information related to 'not difficult' in the next data cleaning step in order to include only text passages with the rating 'difficult' in the empirical analysis. This shows a limitation of AntConc, which you will encounter in various forms with freeware: AntConc cannot save the analysis (as explained previously). However, the simplicity of AntConc is helpful here and offers a flexible solution.

You can copy the concordance analysis in AntConc (as in Table 8.3) and then paste it into another program (e.g., Microsoft Excel or OpenOffice Calculator). To do this, simply click anywhere in the 'Hits' field (= hits that can be used as a reference) in the Concordance tab, hold down the 'Ctrl' or 'Apple' key while

you first select everything by pressing the 'A' key, which highlights the 'Hit column' in blue, then copy the entire content by pressing the 'C' key and paste it into the other program you have selected by pressing the 'V' key – and then finally at this point you can release the 'Ctrl' or 'Apple' key.

Table 8.3 shows one limitation that AntConc has. For example, for Hits 448 and 522, the text snippet provided for the contextualization of the search words starts with a letter, that is, in the middle of a word. By adjusting the 'Search

Table 8.3 Selected Examples of the Concordance for the **K**ey**W**ord 'difficult*' **i**n **C**ontext (KWIC)

Hit	*Sentence 1*	*KWIC*	*Sentence 2*	*File (file name anonymized)*
598	–platforms or via WhatsApp, there is quite a lot of communication and exchange, but for me it is very	difficult	to build real relationships. Meanwhile, I'm still writing with a good friend about my	Filename_ 170603.txt
498	I am studying and I answer [subject 1]. Many people seem surprised and claim that it is a very	difficult	degree program where you would hardly earn anything at the end. But I don't think that's true.	Filename_ 170243.txt
522	I had to change my diet. If this semester had been in person, it would have been	more difficult	and more time-consuming to prepare meals. I check my emails. I go on Moodle a	Filename_ 141827.txt
130	for the two courses. I am incredibly exhausted. So much so that it's even	difficult	to write this entry. I woke up this morning at five to seven and went straight	Filename_ 153929.txt
448	s the exchange is easier. I'm also relieved to hear that it is sometimes also	difficult	for the others to find the right wording and enough content. I am already very interested to see how we can	Filename_ 118438.txt
375	rewind some minutes because I didn't understand a lot of things right away and it was also very	difficult	for me to write at the same time as everything was happening so quickly. And my concentration wasn't s	Filename_ 172871.txt
78	I read a text and summarize the most important points. It was not very much	difficult	for me because I found the text to be very easy. That's why I really enjoyed the text	Filename_ 172871.txt

Source: Autoethnography WS 2020/21

Window Size' setting, an output is created according to the number of characters entered.

Be careful not to get sidetracked reading the interesting results. Although an analysis is not a race, we assume that you have a limited time frame – either one you set for yourself or a deadline. With this in mind, you should make sure to record the first insights you have in your research journal so that you do not forget them later in the research process.

8.4 Software-supported data processing

8.4.1 Software to support your keyword search

Box 8.3 QDA software

You can use commercial QDA software – for example, atlas.ti, MAXQDA, and NVIVO – to perform keyword searches and automatic coding for quantitative content analysis. See your QDA provider's website for instructions as manuals and video tutorials.

Section 8.3 explained how to identify keywords using the freeware AntConc. However, depending on the institution you attend and/or your financial resources, you may have the option of using QDA software. As explained in Box 8.3., the QDA programs discussed have many similar options for quantitative content analysis as well as special features, and they all also make manuals and video tutorials available on their websites. As we do not want to rewrite this book every time one of the QDA software providers offers new features, we will not go into further detail here.

No matter which QDA software you use, the following section will apply as it presents the basic procedures for the construction, modification, ordering, and use of keywords for quantitative content analysis. You can simplify this semi-automated approach to quantitative content analysis by making good decisions regarding the division of work between the software or machine (with its quantitative logic) and yourself (with your qualitative logic). It is also important to consider your ability to interpret data and the need to provide AI with detailed input (see also Chapter 10).

8.4.2 Construction and use of keywords: lemmas and stemming

There are at least two ways to use lemmas or stemming to capture all possible spellings. First, you can use lemmatization to determine the basic form of a word and then to map all of the possible spellings. As a word token, the lemma can be used for keyword searches, which allows you to map frequencies and identify sections of text. Basically, lemmatization involves using one term to

Table 8.4 Example of the Lemma S*tudy*

Words	*Lemma*
study	study
studying	
studies	
student	
students	

Table 8.5 Example of the Lemma *Writing*

Words	*Lemma*
write	writing
wrote	
written	
writing	
writings	
writer	
writers	
author	
co-author	
penmanship	
text production	
etc.	

assemble words with similar meanings, which can then help you to identify relevant units of meaning in larger text corpora (Table 8.4).

The advantage of lemmatization is that you can use it to capture irregular verbs and words with a similar meaning (Table 8.5).

Instead of lemmatization, the word stem can also be used for lexical searches (e.g., biblio function) and autocoding in MAXQDA, for example. Likewise, the word token captures the various characteristics of the search term and keyword via the word stem. The word stem must be empirically identified for the case being analyzed. It is often helpful to use the asterisk (i.e., *) for keyword searches using the word stem. The asterisk supplements and connects the word stem with the heterogeneous written forms. For example, the search word 'academ*' can be used to search for both 'academic' (person) and 'academics' (plural) as well as 'academia' and 'academy'. However, one problem with stemming is when similar words have different meanings. For example, the lemma 'term*' would capture the plural 'terms' but also 'terminology', 'terminator', and the like. Accordingly, conducting a keyword search with the word stem can be problematic. This means that it is essential to make sure that the stem of your search word is unique. In a

heterogeneous text corpus of newspaper articles and party platforms, for example, the short search word 'bio*' would generate a wide range of empirical occurrences such as 'biography', 'biology', 'biosphere', 'bioluminescence', 'biological weapons', etc., which you would then have to sort manually. To avoid sorting due to stemming, you can combine keywords, for example, instead of 'happ*', use 'happy' and 'happiness' as search words (see Table 8.7).

8.4.3 Transforming synonyms and evaluation terms from your keyword search into categories

When you decide on which word tokens to use for your keyword search and subsequent autocoding, you must also consider the additional aspect of how language uses synonyms. As in the case of lemmatization, search engines also find lists of synonyms. It is also a good idea to initially consider synonyms in order to avoid having to do more work later. This can be done by systematically preparing the categorization and subsequent coding of the synonyms. For the text corpus of student autoethnographies, you can take synonyms from the word lists generated with AntConc and then record them using a category you have established. As shown in Table 8.6, you should choose a generally understandable word for the category at hand. In this case, you could choose 'computer' if you want to record the device that students are using to follow lectures, read texts, and complete other tasks for university. If synonyms are used at a similar frequency in the corpus, you can also use both words (e.g., Covid-19, Covid, and Corona) for the category.

Furthermore, when you review the word lists created with AntConc, you may notice, for example, that the students have made a number of evaluations in their autoethnographies. You can summarize these using positive, negative, and neutral keywords for the keyword search (Table 8.7).

Before you decide to start with autocoding, you should definitely test the categories to check your observation. As a test, for each keyword (= subcategories), you can perform a specific search that is grouped by the evaluation types positive, negative, and neutral (= top categories). The test will simplify your data analysis as the following result for the example in Table 8.7 shows. For autocoding the documents after the keyword search, it then makes sense to group the negative search words (i.e., subcategories) under the top category 'negative evaluations'

Table 8.6 Example of Search Words Sorted by Category

eSearch Words						*Category*
laptop	Personal computer	computer	tablet	Desktop computer	ipad	computer
corona	pandemic	Covid-19				corona/ Covid-19

Source: Autoethnography WS 2020/21

Table 8.7 Evaluations

Negative	*Neutral*	*Positive*
Unfortunate*	motivati*	Happy/happiness
stress*	everyday	interest*
difficult*	routin*	excit*
annoy*	demand*	fun
bad		

Source: Autoethnography WS 2020/21

(i.e., 'unfortunat*,' 'stress*,' 'difficult', 'annoy*', and 'bad'). Furthermore, the keywords 'motivation', 'exciting', 'interest/interesting/interested', and 'fun' are used separately in order to be able to analyze nuances of manifest content in a paragraph. You can save the results, for example, as an Excel document (export function in MAXQDA) and then use it as research documentation and/or for later analysis.

8.4.4 Deselecting content based on your keyword search

A semi-automated content analysis produces a lot of findings, and not all of the findings are relevant for answering your research question or they might turn out to be case-specific. The latter was the case for the search word 'child*' (Table 8.8; created using MAXQDA). Table 8.8 contains units of meaning as well as the reference segment with the paragraph number in the corresponding document. You can see that more than half of the 23 hits are assigned to one person, who is anonymized in the results and appears as idno30. Instead of coming up with 50 anonymous names, the students were assigned an identification number in a list. It is only for two students (idno30 and idno16) that the presence of one or more children in the household involves an organizational task, albeit two different ones. A child or children are part of studying (idno08, idno21, and idno26), play pranks that distract from studying (idno34), and are the subject of reflection (idno46 and idno50). Overall, the heterogeneity of the hits clearly speaks in favor of excluding the keyword 'child*' from the analysis as neither a clear nor quantitatively usable result can be achieved. Do not forget to document the analysis in your research journal that led to your decision to deselect the keyword and possible category of 'child*'.

8.4.5 Cleaning the data and data robustness checks

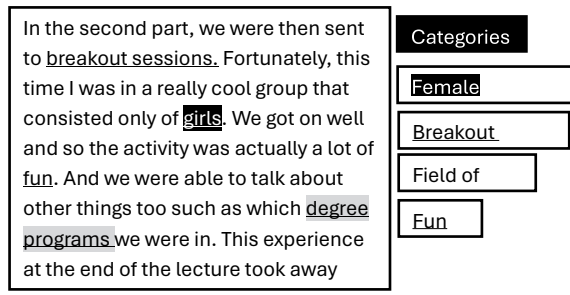
In contrast to the qualitative problem of overlooking or missing relevant content (see Chapter 7), loss of information and imprecision in data collection are assessed differently in quantitative content analysis. Analogous to fully automated quantitative content analysis (see Chapters 15–18), you have the option to use one or more of the following procedures to assess the information quality

Table 8.8 Hits for Search Word 'child*'

Content	*Document*	*Search Term*	*Seg.*
T[he] older one is a small child and [the] youngest a baby	idno30	child	1
happy to put the laptop aside for the day and spend time with my family until the children go to bed	idno30	children	1
T[he] older one is a small child and [the] youngest a baby	idno30	child	2
T[he] older one is a small child and [the] youngest a baby	idno30	child	3
After this seminar, I am exhausted and very glad that there are only three hours left before the children go to	idno30	children	3
T[he] older one is a small child and [the] youngest a baby	idno30	child	4
T[he] older one is a small child and [the] youngest a baby	idno30	child	6
T[he] older one is a small child and [the] youngest a baby	idno30	child	7
I get my children ready to go to bed at 9 pm	idno30	children	7
T[he] older one is a small child and [the] youngest a baby	idno30	child	8
After this seminar, I am exhausted and very glad that there are only three hours left before the children go to	idno30	children	8
T[he] older one is a small child and [the] youngest a baby	idno30	child	9
This morning, I stumbled across a series that I loved watching as a child	idno31	child	14
I wondered whether this analysis could also be applied to children's series and watched the episode from this point of view.	idno31	children's	14
I looked at what the series was supposed to tell and teach me as a child and succeeded	idno31	child	14
Today my day starts at 7 am because I have to take my little brother to his childcare provider before my	idno16	childcare	7
We were also shown that we encounter problems that we were previously unaware of such as disruptive children in the	idno08	children	34
She is an early childhood specialist and is currently focusing on the topic of raising children. The	idno21	children	152
However, there are most likely other parents who put a lot of pressure on their children, which is reflected in a small	idno26	children	12
It was probably the children or teenagers in town who were playing pranks	idno34	children	10
I check out the window to see if it's the children playing tricks again	idno34	children	12
This even goes so far that she moves to Hamburg with her child	idno46	child	33
Certainly, because I was exposed to severe injustice in my childhood and youth	Idno50	childhood	18

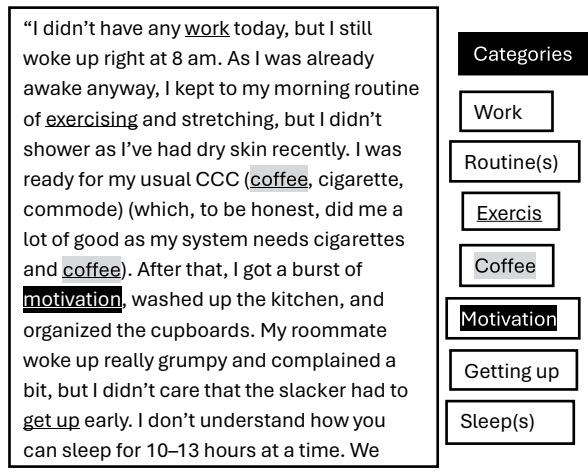
Source: Autoethnography WS 2020/21

Figure 8.4 Exemplary quote: getting to know female classmates.



Source: Autoethnography WS 2020/21

Figure 8.5 Exemplary quote: work and the error 'motivation'.

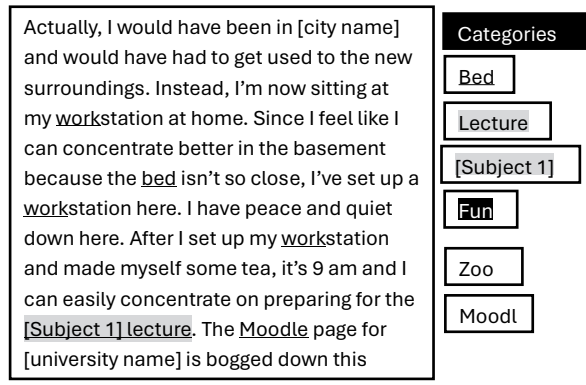


Source: Autoethnography WS 2020/21

of the text segments that have been autocoded. A first step in assessing the text segments generated by search words is to make plausible assumptions based on your everyday knowledge and comparisons. By making comparisons, for example, the use of the words 'alarm clock' and 'cell phone ringing' can be classified as analogous to the alarm clock function and be recorded in a category 'alarm clock/cell phone ringing' (= subcategory to top category 'sleep in/fall asleep'). The same procedure is recommended to translate language used in the texts into categories, for example, coding the word 'fun' as a positive evaluation and translating the word 'girls' as female classmates (Figure 8.4).

You can also use the everyday knowledge you have to detect potential errors related to the phenomenon you are analyzing. For example, it is plausible to assume that the keyword 'work(s)' has a variety of meanings. Students use 'work(s)' to mean 'working for university' or 'get started with work' as well as 'going to work' as a synonym for a (part-time) job. In the student autoethnographies,

Figure 8.6 Exemplary quote: study conditions during Covid-19 and the error 'work'.



Source: Autoethnography WS 2020/21

work was mainly used to talk about part-time jobs (Figure 8.5). Nevertheless, as expected the search word 'work*' also generated errors such as 'workstation' meaning the place to work, that is, the desk (Figure 8.6).

For quantitative content analysis of larger and bigger datasets, you must systematically record errors such as 'workstation' and a 'burst of motivation' to clean up the kitchen instead of doing something for your studies (Figure 8.6) – this will save you from having to go through the time-consuming task of checking all 5,918 coded segments (Table 8.9). Analogous to statistical tests (see Section 4.2), you should do robustness checks to determine the probability of error or standard error in the dataset, whereby depending on the data quality a probability of error of 10% or even 20% is considered acceptable with big data (Steinhardt et al., 2017). To determine the probability of error, you have to take a sample from the dataset. And there are two procedures you can use for robustness checks – ideally in combination.

1. **Random sample**: Open the Excel file you exported from the analysis software, copy the sheet with the coded segments, and save it as a new Excel file. Randomly select 100 of the 5,918 coded segments. For the 100 coded segments, check how often the keywords and 49 assigned categories produced errors. You only need to check here whether the category specified in the list is correct. If other categories are contained in the coded text extract, you should just ignore them. The starting point for searching for errors could be the hits for 'work(s)' and 'motivation' described previously. However, the targeted search for errors should not lead to an incorrect evaluation of the quality, but should produce an average value across all of the data. You must therefore also consider less error-prone or unambiguous categories such as subjects, professor, and pre-semester preparatory course. By selecting 100 segments, you can then specify the probability of error as a percentage.

But overall, this value is to be regarded as imprecise, which is why it is strongly recommended that you determine the probability of error in a systematic way.

2. **Systematic sample**: A systematic sample is one that is obtained through adherence to a standardized procedure. For example, if you take into account the 49 categories, then for 100 tests, you can check at least two segments of each category in the dataset. Considering the category frequencies in Table 8.9, one good way to do this is to check the first hit for each category and, in a predefined interval, all x-th hits for all categories in the list created by the QDA software. For clarity purposes, you should copy the corresponding text segments into a new Excel sheet so that the procedure is clear and transparent. In the column next to the text segments, you can record the results, for example, by entering a 1 for any errors you find. This will make it possible to have Excel add up the number of errors for you. If you are unsure about the significance of the result (e.g., an unexpectedly high value), it is a good idea to repeat the test with a second systematically created sample. The mean value of both samples can then be given as a percentage for the probability of error.

The value determined by a systematically generated random sample should be added as information to the results table, as in the example Table 8.9. It is then not necessary to correct all of the frequency data in Table 8.9, that is, to normalize the value determined on the basis of the probability of error.

On the other hand, you must take the probability of error into account when explaining and interpreting the manifest content. A probability of error of 20% would mean that every fifth hit would be considered incorrect. Less caution is required when interpreting results with probabilities of error of 10% and less, especially less than 5% – this is analogous to the 95% confidence interval in statistics (please search for this in a statistics textbook in your preferred language).

8.5 Software-supported quantitative content analysis

8.5.1 Organizing your categories

The number of keywords is usually not identical with the number of categories. In the example of the secondary analysis of the autoethnography, for instance, it is best to summarize the keywords ‘sleep’, ‘fall asleep’, and ‘sleep in’ as one category. Depending on your research question, at this point you can also summarize synonyms and plausible related keywords (i.e., keywords describing the same action or object) for autocoding. It is important that you document your procedure and describe it in a clear and comprehensible way.

Plausible summary in analysis software language means that you first create a top category (in MAXQDA code list, this would be the top-level code), which helps to capture certain characteristics in the text data. In order to answer

Table 8.9 Overview of the Categories and Absolute (n) and Relative Frequencies (%) of the Coded Segments (seg.) in the Documents *(doc.)*

Top Category	*Subcategory*	*N Coded Seg. (in doc.)*	*% Coded Seg. (in doc.)*	*Appeared in N Doc.*
Neutral evaluation	Motivation	76	1.28	25
Positive evaluation	Be happy/happiness	43	0.73	23
	Interest/interesting/interested	111	1.88	35
	Fun	35	0.59	19
Negative evaluation	Unfortunately, stress, annoying, bad, weak	188	3.18	39
IT/ Communication	Breakout sessions	47	0.79	19
	Computer	143	2.42	33
	Moodle	185	3.13	38
	WhatsApp	48	0.81	21
	Zoom	205	3.46	37
Reading	Book(s)	55	0.93	21
	Read (verb)	193	3.26	38
	Text (unspecific)	208	3.51	39
Sleep in/ Fall asleep	Get up	97	1.64	31
	Bed	106	1.79	29
	Sleep/Fall asleep/Sleep in (verbs)	138	2.33	34
	Alarm clock/Mobile phone ringing	90	1.52	24
Subjects of study	Subject (unspecific)	149	2.52	37
	[Subject 5]	40	0.68	10
	[Subject 4]	26	0.44	11
	[Subject 2]	20	0.34	11
	[Subject 6]	4	0.07	4
	[Subject 9]	49	0.83	11
	[Subject 7]	21	0.35	9
	[Subject 3]	74	1.25	26
	[Subject 1]	122	2.06	34
	[Subject 8]	14	0.24	4
	[Subject 10]	25	0.42	10
Studies/studying	Task(s)	227	3.84	38
	Video tutorial	203	3.43	39
	Lecturer/Teacher	120	2.03	28
	Professor	91	1.54	21
	Pre-semester preparatory course	62	1.05	22
	Seminar	177	2.99	30
	Studies/Study(ing)	251	4.24	39
	Topic	183	3.09	37
	Practice/Practical course	128	2.16	34

(*Continued*)

Table 8.9 (Continued)

Top Category	*Subcategory*	*N Coded Seg. (in doc.)*	*% Coded Seg. (in doc.)*	*Appeared in N Doc.*
	Lecture	547	9.24	41
	Knowledge (science)	136	2.30	35
	Distraction/Procrastination	15	0.25	11
	Work	385	6.51	40
	Autoethnography	143	2.42	34
	Corona/Covid-19	62	1.05	32
	Coffee	84	1.42	18
	Fellow students	118	1.99	30
	Concentration	83	1.40	29
	Routine(s)	43	0.73	17
	School	55	0.93	23
	University/[University Name]	293	4.95	40
Total		**5,918**	**100**	**–**

Source: Autoethnography WS 2020/21

the research question "How are you organizing your daily life as a student at the beginning of the online semester 2020/21?", it is plausible to assume that students evaluate the organization of their daily life as a student. Accordingly, Table 8.7 shows how the search words were assigned to the evaluation top categories of positive, neutral, or negative. This assignment was made based on very good, but not comprehensive manifest content knowledge – always bear in mind that you are conducting a *distant reading* meta-analysis of a larger, large, or very large text corpus.

When you are using categories to summarize keywords and make plausible assignments, you should be cautious. For example, after reviewing the concordance of the search word 'motiv*', we assigned it to the category of neutral evaluation as no clear assignment to positive or negative evaluation was possible via the keyword search (e.g., hits included very motivated, unmotivated, and the lecturer motivated us ...). Table 8.3 (Selected Examples of the Concordance for the Keyword 'motiv*') shows the distribution of the text passages recorded in the corpus with the search word 'motiv*'.

However, if your text corpus is more extensive and/or there is uncertainty about the hits you want to obtain with the keyword search, it is best to index the text corpus in the QDA software with each keyword = one category. In this case, you will organize the text corpus after completing the lexical search and autocoding. You must have at the very least good superficial insight into your data to be able to organize it, because based on the manifest content of the dataset, there are a number of options available to you.

Use the export function of the QDA software (e.g., 'Reports' tab in the MAXQDA menu bar) to export an Excel file. Save the exported file, ideally using the category name(s). The Excel file exported by MAXQDA has several sheets at

the bottom. In the 'Coded Segments' Excel sheet, select all of the rows (e.g., by clicking on the triangle on the left above row 1) and then select 'Custom Sort' in the menu bar under 'Sort & Filter' (funnel symbol A – Z). Please note that the 'My data has headers' box is checked if you, for example, select 'Category' via the 'Sort by' column in the drop-down menu. The selection of the category or categories to analyze first should be based on your research question, that is, be central to your analysis. Alternatively, you could filter based on the 'Segments' column to sort out any duplicate segments.

8.5.2 Quantitative results to present manifest content and for the category-based selection of in-depth analyses

For quantitative content analysis, sorting the categories and in some cases manually assigning top categories is a core result (Table 8.9). In view of the students' very different life circumstances, you should be careful of interpreting the quantitative values for the category frequencies in the dataset as equivalent to meaning. Certainly, a higher frequency indicates greater importance in the text corpus, but this is easier to show analytically with data that is more homogeneous in terms of content (e.g., an hour of a social media discussion on a particular topic). The results in Table 8.9 show categories that you would tend to expect in the context of studying.

But do bear in mind that frequent categories are not necessarily suitable for showing differences between the autobiographers – in this case the students – and the meaning of their statements. As explained in the example of the category 'children' (Table 8.8), it is often the case that the occurrence of a category allows (only) a distinction to be made between students. In this case, the distinction is between those students who have a child or children and those who do not have a child or children. Child(ren) will affect the organizing of daily life as a student at the beginning of the online semester 2020/21. However, childcare centers in Germany were not closed during the time of data collection. On the one hand, for sure, COVID-19 social distancing rules did create additional challenges in organizing childcare and studies. On the other hand, having (a) child(ren) also sets structures in daily life (e.g., time to bring them to childcare and to go to bed), which 'motivated' (subcategory to top category 'neutral evaluation') a student parent to focus on studying, whenever possible, and countered 'procrastination' (subcategory to top category 'study/studying'; e.g., rather watch videos online until four o'clock in the morning than read the text to prepare for a 'course' (idno33 pos. 2)).

Despite the fact that the corpus of all 50 student autoethnographies is analyzed in this quantitative content analysis, it is important to also briefly discuss the column on the right in Table 8.9 titled "Appeared in N Doc." In the corpus of autobiographies, the paragraphs in the text data that are relevant to answering the research question are identified using keywords. In addition to the keywords, the data consists of the paragraphs containing a keyword. These

paragraphs are extracted from their original context in the document, that is, from the respective student autoethnography. "Appeared in N Doc." points you in the direction of the relevant documents for an in-depth analysis of a category and/or subcategory. Another analysis possibility offered by "Appeared in N Doc." is drawing conclusions about the diffusion of the topic represented category in the document corpus by seeing how many documents the category occurs in. Of course, both analyses have to contribute to answering the research question "How are you organizing your daily life as a student at the beginning of the online semester 2020/21?".

However, there are various aspects to consider when selecting relevant categories for the in-depth analysis of the data. In Table 8.9, the last column never lists all 50 texts. This can be explained by the students' diverse use of language and grammar. For example, the most frequent category of 'lecture' appears in 41 of 50 text documents. Instead of using the term 'lecture' in the autoethnographies, nine students always wrote 'courses' and did not make a distinction between 'seminars' and 'lectures.' Taking note of such patterns, the differentiated coding of 'seminars' and 'lectures' chosen here may not be useful for larger text corpora, that is, when you have more data. If you analyze a text corpus with significantly more than 500 pages, the gain in knowledge is usually disproportionate to the cleaning and sorting efforts involved during data processing in part 2 of the research process.

Therefore, grouping 'seminars', 'lectures', and 'courses' in one category (e.g., called 'courses') would be an appropriate option in quantitative content analysis, which also facilitates comparisons.[5] Consequently, an equivalence is created by grouping these together in the category 'courses', which comes, of course, at the expense of a detailed analysis and interpretation. As Moretti (2000, pp. 57–58) emphasizes for the comprehension heuristic of *distant reading*, using this type of coding involves making a selection, which allows for the focused analysis of certain content and topics in the sense of less is more. The knowledge gained from the (quantitative) content analysis involves distancing yourself from the texts in the corpus, and thus from the qualitative dimension of close reading (see iteration and distancing quality convention, Section 4.3.3).

The example presented here that includes around 500 pages is neither completely distant reading nor close reading. Systematic data indexing and using analysis techniques linked to two software programs could lead to a methodological classification of *distant-to-close reading*. For example, one answer to the question "How are you organizing your daily life as a student at the beginning of the online semester 2020/21?" is that the online semester not only brings with it other types of group work (breakout sessions in online lectures; exemplary quote shown in Figure 8.4) that are in contrast to those used in analog studies before the Covid-19 pandemic. It is also plausible to interpret the disruption of the coffee – cigarette – commode ritual (Figure 8.5) at the start of the lecture period as (a) being caused by the new, possibly stressful situation for the first-year student, and that (b) the temporary caffeine-nicotine-digestion

confusion would have been similar regardless of whether it was an analog or online semester.

Nevertheless, the paragraph including the "coffee – cigarette – commode ritual" of student idno50 contains information specific in the social context of Winter Semester 2020/21: The student lives in a shared flat, which is very common for students in Germany. For example, student idno15 did not move out and set up a 'workstation' in the basement of their parents' home. They particularly emphasized in the preceding sentence: "Actually, I would have been in [city name] ..." (Figure 8.6) – and at the place of study, that is, at the university (library), also the bed would be far away during study time ("Since I feel like I can concentrate better in the basement because the bed isn't so close, I've set up a workstation here", Figure 8.6). As an exemplary quote, such a statement supports the interpretation that students are confronted with arrangements specific for a semester with Covid-19 restrictions. Accordingly, student idno50 would not have been so aware of their roommate's sleeping habits without being at home so much as they would have spent more time at the university without the Covid-19 situation ("My roommate woke up really grumpy and complained a bit, but I didn't care that the slacker had to get up early. I don't understand how you can sleep for 10–13 hours at a time" (idno50, pos. 23), Figure 8.5).

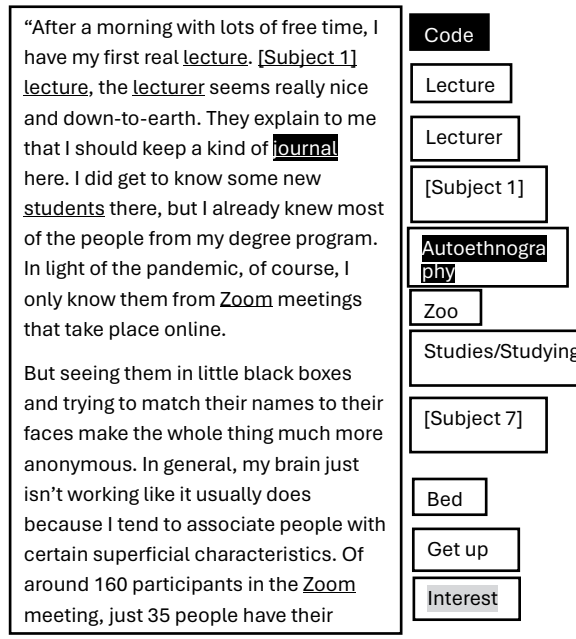
Of course, on any day without work, student idno50 could be observing the sleeping habits of their flat mate, but the autoethnography only covers two weeks. In cases where you are uncertain like this, it is best to carefully interpret or even exclude a coded text section in an in-depth analysis of a category (in the present case, top category 'sleep in/fall asleep'; Table 8.9). It is important to document this in your research journal and make your decision (e.g., for exclusion), limitations on your interpretation, and other relevant issues concerning transparency of how you got to your results (see also Section 4.3.2).

8.5.3 Exemplary quotes to illustrate patterns

As you may have noticed by now, quantitative content analysis becomes easier to understand by the use of exemplary quotes (in German: *Ankerbeispiel*). A good exemplary quote is characterized by the fact that – in comparison with other possible examples – it clearly and vividly illustrates the pattern of individual behavior, social action, a topic, or the like. Exemplary quotes can depict something *typical* or that *frequently occurs* as well as be used to illustrate *something unique* or *exceptional*.

Figure 8.7 gives an exemplary quote for how the significance of the 'bed' has changed during the online semester: The 'bed' also became a distant learning space. It is clear that in a normal semester the student would not have been able to follow (all of) their morning 'courses' during the week they kept the autoethnography from 'bed'. As you can see, not only the single paragraph captured via the keyword search and subsequent autocoding, but three paragraphs were selected for the exemplary quote. The category 'Zoom' produced hits in two consecutive paragraphs, as was evident in the list exported as an Excel file. The

Figure 8.7 Exemplary quote: adding a unit of meaning for analysis.



Source: Autoethnography WS 2020/21

third paragraph was added after looking at the relevant document as this paragraph summarizes the consequences of the Covid-19 pandemic: The routines that are established and the unintended effects the online semester can have, in this case for the girlfriend (keyword: information pollution). Accordingly, the three paragraphs form a unit of meaning (as per the definition from Miles & Huberman, 1994, p. 56).

Please be aware that an exemplary quote also needs explanation as it is not evident for others why you selected it. Carefully consider how you can add a description and interpretation about the unit of meaning so that people understand your findings. Using exemplary quotes helps to do this, but they are not enough. You must explicitly present the results as well as interpret the empirical data and draw conclusions (appropriateness-transparency quality convention, see Section 4.3).

8.5.4 Further, in-depth analysis and interpretation of results

To reiterate, the quantitative content analysis using the autoethnographies by 50 students can only provide an example to give you an idea for how to design the research process and conduct your own empirical research. Accordingly, we abstain from presenting in-depth, qualitative analysis and interpretation based on the quantitative content analysis. In Chapter 5, you can read an introduction

to how to do interpretation. Using the same data as in Chapter 8, that is, the students' autoethnographies, Chapter 6 presents a systematic deductive, theory-led approach and Chapter 7 an inductive, theory-reflexive approach to qualitative data analysis and interpretation.

For in-depth analysis, you have to be (very) familiar with your data, otherwise you just wander around your data aimlessly, and lose a lot of time and motivation. In addition, we assume that you did use QDA software for data processing and analysis and have selected exemplary quotes to justify the category system and back your interpretations of quantitative data analysis. For a term paper, doing an in-depth analysis of all categories in Table 8.9 would be too much. Accordingly, you would need to choose a top category and subcategories for in-depth analysis, for example according to which result(s) you deem to be the most relevant to answering the research question.

To conclude Chapter 8, we provide some guidance for doing the in-depth, qualitative analysis based on the example results of the quantitative content analysis of the 50 student autoethnographies. In Table 8.9, the six top categories: (1) 'negative', 'neutral', and 'positive evaluations'; (2) 'IT/communication'; (3) 'reading'; (4) 'sleep in/fall asleep'; (5) 'subjects of study'; and (6) 'studies/studying' and associated multiple subcategories provide reference points for different in-depth analyses. The six top categories can be of different use in an in-depth analysis, for example:

1. The top category 'subjects of study' can be used for selecting a specific field of study and/or comparing two or three different 'subjects' of study. The comparison would enable a close look at whether the organizing of daily life as a student was affected in different ways by the chosen field of study, for example comparing subjects with a lot of mandatory courses and those with few mandatory courses.
2. The top categories 'negative', 'neutral', and 'positive evaluations'; 'IT/communication', 'reading', and 'sleep in/fall asleep' address very specific topics for qualitative analysis. Accordingly, either top category would guide a focused in-depth analysis. For example, the top category 'reading' is basically about students pointing out or complaining about them having to read more in sociology than in other subjects.
3. However, in comparison to the top categories 'IT/communication', 'reading', and 'sleep in/fall asleep', the top category 'negative', 'neutral', and 'positive evaluations' is closely connected to other (sub)categories, which are 'evaluated'. Therefore, it is important that you keep your focus on the top category guiding your in-depth analysis and do not get lost in the data its analysis leads you to. To keep the focus, always ask yourself what does the 'evaluation' of x, y, and z contribute to answering the research question about the "organizing of daily life as a student at the beginning of the online semester 2020/21?"
4. The top category 'studies/studying' assembles a broad range of topics related to 'studying'.

Considering the research question of "organizing daily life as a student at the beginning of the online semester 2020/21?", it would be plausible to start your in-depth analysis with the top category 'studies/studying', possibly by reorganizing subcategories. For example, to analyze teaching and learning environments, you could create a subcategory 'courses', which makes a distinction between 'lecture', 'seminar', 'practice/practical course', and 'pre-semester preparatory course' as succession subcategories. Of course, the interpretation of teaching and learning environments in different types of 'courses' would have to be consider 'people involved' (i.e., 'students', 'professor', and 'lecturer/teacher').

Another focus for an in-depth analysis could be on scientific 'knowledge' acquisition and the challenge provided by 'distraction/procrastination', having to 'work', going for 'coffee' with 'fellow students' despite or by respecting 'Covid-19' social distancing rules. Not going for 'coffee', not having to go to 'work', or not giving in to 'distraction/procrastination', that is, the absence of a category in autoethnographies can provide just as much information about organizing daily life as a student. Again, as you did your quantitative content analysis to the point of in-depth analysis you will be (very) familiar with your data and can select relevant student autoethnographies for analyzing similarities and differences 'work', a 'child or children', and time to 'procrastinate' make for "organizing daily life as a student at the beginning of the online semester 2020/21."

Notes

1 Originally, the code word cloud was constructed by MAXQDA for the German data.
2 We understand (digital) spatial context to mean the physical or online constructed speaking situation in which communication takes place.
3 Conversely, a line break can be created by entering the command '\n' in the replace field.
4 Another helpful function in AntConc for quantitative content analysis is that you can create a keyword list by comparing with a reference corpus. This involves a comparison of the list of keywords that are unusually frequent or rare in the text corpora. A chi-squared test ($\chi 2$) and plausibility or conjecture function (Log Likelihood) are preset as a statistical measure to identify whether a word in the text corpus analyzed should be regarded as a keyword. This provides a basic function for testing the extent to which certain words occur independently of one other or whether the occurrence of the keywords follows an underlying structure. This logic is used in the correspondence analysis in Chapter 15. No reference corpus could be found for autoethnographies of students at the start of the 2020/21 semester. However, similar data from Twitter or party platforms that is compared to newspaper articles on current politics, for example, can be used as control groups.
5 For example, in autoethnographies written by students in natural science degree programs, the course type 'laboratory' would be expected in addition to 'lecture'.

9

DEDUCTIVE QUANTITATIVE CONTENT ANALYSIS

A bibliometric literature review

In this chapter, we will introduce you to the method bibliometric literature review (BLR). You can use this deductive quantitative content analysis method to obtain a systematic review of publications. To lay the groundwork, we will first explain what publications actually are and what the systematic bibliometric analysis of literature is based on: citations. Bibliometric analysis requires the use of databases. Therefore, we will present a selection of databases and their advantages and disadvantages. After these basics are covered, three approaches to bibliometric literature review are presented, along with the research questions that can be addressed with each approach: (1) literature review and the question "what are the key publications on a specific topic or in a research field?" (2) mapping and the question "how is a research field (on a specific topic) structured, that is, what connections can be found?" and (3) topic analysis: "what content is discussed in a research field or on a specific topic?"

9.1 Introduction

When you create a reference list for a term paper or thesis, you are required to provide a number of details. These normally include at least the author's name(s), publication date, title, and place of publication. All of these details are bibliometric data. Since there is an enormous number of publications, there is also an incredibly large amount of bibliometric data. And here we already encounter the first question. What actually is a publication? According to Stock et al. (2023, p. 40), publications are initially documents that are defined as follows: "materiality: they are physically – including the digital form – present; intentionality: there is a motivation behind them, and they carry meaning; development: they are created; and perception: they are described and perceived as documents." This definition includes all forms of textual and non-textual, and digital and non-digital resources. However, since the BLR is not about analyzing

DOI: 10.4324/9781003496397-11
This chapter has been made available under a CC-BY-NC-ND 4.0 license.

every form of document, but research publications, a more precise definition is required. A publication can first be distinguished by whether it is formally or informally published. Formally published documents are "from publishing houses, i.e., books, newspapers as well as journals, all intellectual property documents, grey literature, which are publications not provided by commercial publishers like university publications, working papers, or company newspapers" (Stock et al., 2023, p. 40). Informally published documents are web and social media documents (Stock et al., 2023). Secondly, a distinction can be made between research and non-research publications. Here, the bibliometric analysis takes a pragmatic approach[1] and specifies four criteria that define a research publication: (1) the existence of an abstract; (2) the affiliation of the authors; (3) the existence of references; (4) and the use of a citation style (Stock et al., 2023). In the following we speak of research publications if these four criteria are met and it is a formally published document.

You now know what a research publication is. The next step is to familiarize yourself with bibliometric data. Therefore, we will give you an overview of the types of bibliometric data available.

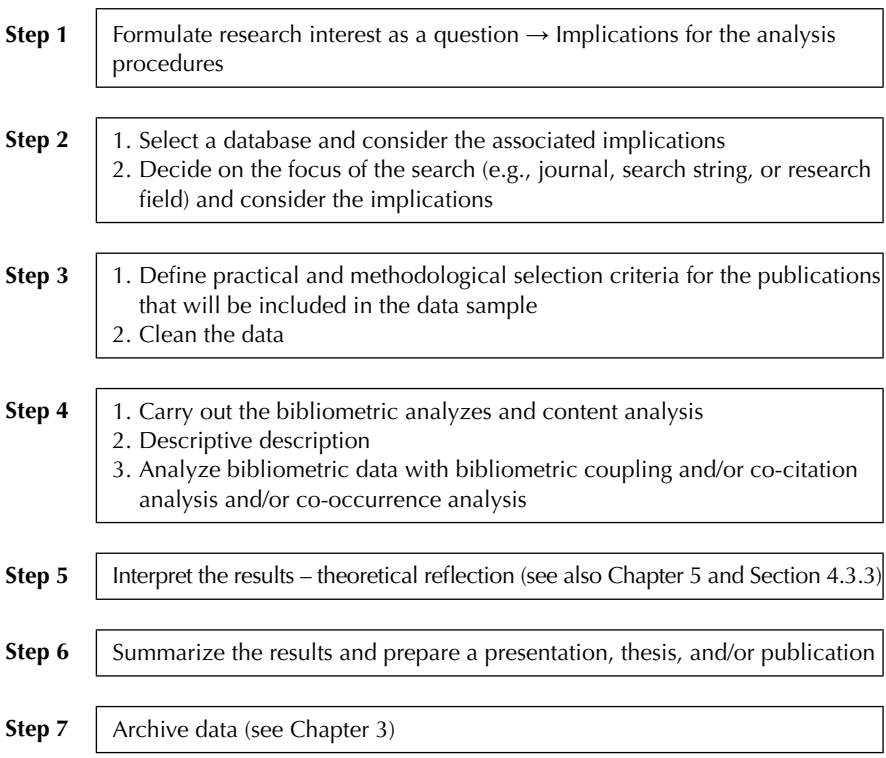
1. **Data that describes a publication**: This includes the names of the authors, date of publication, place of publication (journal name, editorial volume, place and name of the publisher, and Internet address), and unique identifiers of the publication, for example, the ISBN for books, the ISSN for journals or blogs, and the DOI for journal articles,[2] chapters in books, and Internet documents on central platforms such as Zenodo.[3]
2. **Additional data on the authors that is not always or only partially available**: This includes the affiliation, that is, the institution(s) where the authors are employed, the country or countries where the institutions are located, and information on the subjects or disciplines which the authors identify with.
3. **Data on the content of a publication**: Publications, at least articles and book chapters, usually contain an abstract (i.e., a brief summary of the content). In addition, there are often keywords that also refer to the central aspects of the content.
4. The **bibliography or reference list** is a list of publications that have been cited.
5. **Data on the citation frequency of a publication**: This refers to how often a publication has been cited in the bibliographies of other publications. In bibliometrics, this is usually referred to as citation.

Especially in aggregated form (i.e., collected and combined from multiple sources), this bibliometric data makes it possible to examine patterns and thus manifest structures both at the publication level and content level. The BLR therefore focuses on the data that is generated in and by a publication and that allows conclusions to be drawn about the communication in a discipline, in a field of research, or on a specific topic.

As this still sounds very abstract, we would like to give you a few examples of what the BLR can be used for. Imagine, for example, that you are working on your thesis and are now faced with the question of what the key publications on a particular topic are. The BLR can help you here to get an overview of a specific topic and the most relevant publications. In addition, the BLR can answer research questions such as how specific topics in a field of research are connected, how they have developed over time, and where there are overlaps or boundaries to other topics or research fields. And finally, you might want to find out what content is discussed frequently on a particular topic. The BLR is used for these and other research questions. You can conduct a BLR to get a systematic overview of the literature on a topic or research field. Or you can use the BLR as a method to explore a field of research as a subject.

But what exactly is the BLR? The BLR is based on the research steps of the systematic literature review (SLR) and also uses bibliometric analysis methods. These include co-citation analysis, bibliographic coupling, and co-occurrence (see Chapter 9.6). Don't worry, we will describe these methods in detail. But first, it's important that you understand the SLR, which is what the BLR is based on.

Figure 9.1 Flowchart of a bibliometric literature review.



The SLR is a method to create systematic and reproducible[4] literature reviews on a specific topic (Fink, 2019) in order to identify research gaps in the field (Petticrew & Roberts, 2008). It was developed to minimize researcher subjectivity in the literature review as well as in the selection and interpretation of literature. However, there can never be a completely objective selection of articles as subjectivity is already present in the choice of a research topic and search terms. The great strength of the SLR is that it makes the research process transparent. The BLR is founded on this approach as well but also includes bibliometric analysis procedures. The following steps, which are fundamental to the BLR, are summarized in Figure 9.1 and described in detail next. In the description, we will include various examples from a range of fields in order to illustrate how the research process can be structured.

Before we discuss the individual steps of the BLR in detail based on three research questions, it is important to focus on the possibilities and limitations of the databases and data that can be used. This knowledge, especially of the limitations of the available data, is crucial for understanding the methodological approach.

9.2 Weaknesses of databases and search engines

It is essential to understand the weaknesses of the specific data and databases in order to be able to interpret your bibliometric data and avoid making false statements. The BLR is very good at showing how research fields are structured and what the key publications or topics are. However, the findings are based on citation metrics and include databases that never contain all of the existing data. This means that the scope of the statements is limited. For one topic, research field, or discipline, the results are highly significant, but these cannot be transferred to another topic, research field, or discipline. As a general rule, a metric such as the frequency of citations must always be considered in the context of the respective research field and its own research culture, as doing otherwise could lead to strong distortions.

- Take, for example, citation impact, also called citation rate, which is the number of times a person is cited. The citation impact does not automatically indicate the quality or importance of a research publication. This is because of the following reasons:
- A publication can be so innovative that no one else is researching the topic and the publication is therefore not cited. If a publication is in the mainstream of a research field, it is usually cited more frequently than if it is on the edges of a research field.
- The smaller the research field, the fewer authors who can cite the publication. This is why the number of citations in small research fields is often lower.
- Research fields that are interdisciplinary often have multiple research contexts and correspondingly varying citation numbers.

- There are research fields in which mainly books and articles in edited volumes are published. However, these forms of publication are not (or only partially) included in all databases why citations are often not counted.
- If publications are written in languages other than English, the probability of citation is lower as English is the international language of research.

In addition, databases count different citations – an example will follow shortly. First, however, it must be explained what we actually mean here when we talk about databases. There are two large databases for bibliometric studies that can be used to retrieve the data described previously. The first is Web of Science (WoS) and the second is Scopus. In addition to these commercial databases, there are numerous search engines that can be used to search for publications, but which also show how many citations a publication has. The best known is certainly Google Scholar, whereas CrossRef and Semantic Scholar are less well-known. In addition, there are social networks such as ResearchGate, where researchers can save, search, and comment on publications. ResearchGate also displays the number of citations for publications. We would like to illustrate how different the citation numbers in databases, search engines, and social networks actually are using two examples in which the authors of this book were involved (as of March 17, 2025).

Jungblut, J., Vukasovic, M., & Steinhardt, I. (first published 2018). Higher education policy dynamics in turbulent times – access to higher education for refugees in Europe. *Studies in Higher Education, 45*, 1–12. https://doi.org/10.1080/03075079.2018.1525697.

- 87 citations on Google Scholar
- 69 citations on ResearchGate
- 41 citations on Semantic Scholar
- 32 citations on CrossRef
- 35 citations on Scopus
- 25 citations on WoS

Steinhardt, I., Schneijderberg, C., Götze, N., Baumann, J., & Krücken, G. (first published 2016). Mapping the quality assurance of teaching and learning in higher education: The emergence of a specialty? *Higher Education*, *74*(2), pp. 221–237. *https://doi.org/10.1007/s10734-016-0045-5*

- 140 citations on Google Scholar
- 87 citations on ResearchGate
- 74 citations on Semantic Scholar
- 48 citations on CrossRef
- 64 citation on Scopus
- 49 citations on WoS

The differences between the citations are due to the nature of the databases and their database for calculating citations, which is described in more detail later. Before we go into the respective databases and search engines, it is important to note the following: Databases and search engines are not neutral in nature. Instead, they are products that have specific interests built into them. Most databases are commercial products and serve corporations that have commercial interests, for example, to generate profits through advertising or to sell their data. However, non-commercial databases also have interests that are reflected in the data and, for example, promoted publications.

9.2.1 Google Scholar

First, it is important to note that Google Scholar is not a database. It is a search engine that compiles data by crawling websites. In this case, Google's "crawler" collects information about publications and citations on the websites where the publications are available. It does this through parsing. Parsing is the process of analyzing symbols (in this case letters and numbers) with regard to grammatical rules and document layouts such as larger font for headings. A sequence of four numbers is most likely a year, and Google Scholar therefore identifies it accordingly. This allows bibliographic information (e.g., author and title) to be recognized. However, there are no standards for the data, which means that the data is often incorrect (more on this in a moment) (see Orduna-Malea et al., 2017 for explaining the problems with Google Scholar). It is commonly known that Google Scholar searches the websites of higher education institutions and retrieves information from the databases of publishers, whom they have previously concluded contracts with for the retrieval of publication-relevant information. However, it is not known which publishers these are. In addition, a software program such as Publish or Perish (Harzing, 2019) is needed to be able to use the data from Google Scholar and, most importantly, to edit it. Publish or Perish is an open-source program that can be downloaded free of charge at *harzing.com*. The program runs on both Windows and Mac computers. You can think of the program as a tool to convert your Google Scholar search into an Excel spreadsheet that you can then work with. However, the data from Google Scholar is of very low quality (Martín-Martín et al., 2021). We would like to give you an example here so that you can understand why we currently do not advise using Google Scholar.

The following image shows a screenshot of a search using Publish or Perish. It is important to note that this search reflects exactly what you would see in a Google Scholar search. The phrase "qualitative Inhaltsanalyse" ("qualitative content analysis") was searched for in the title of publications (Category: title words). No time restrictions were made. There were a total of 708 hits. Figure 9.2 shows a small section with hits from the author Philipp Mayring. This example shows us some of the problems with the data quality of Google Scholar, which we will go through together in a moment.

We have enlarged the section to make it clear and comprehensible.

Figure 9.2 Screenshot of a search using Publish or Perish.

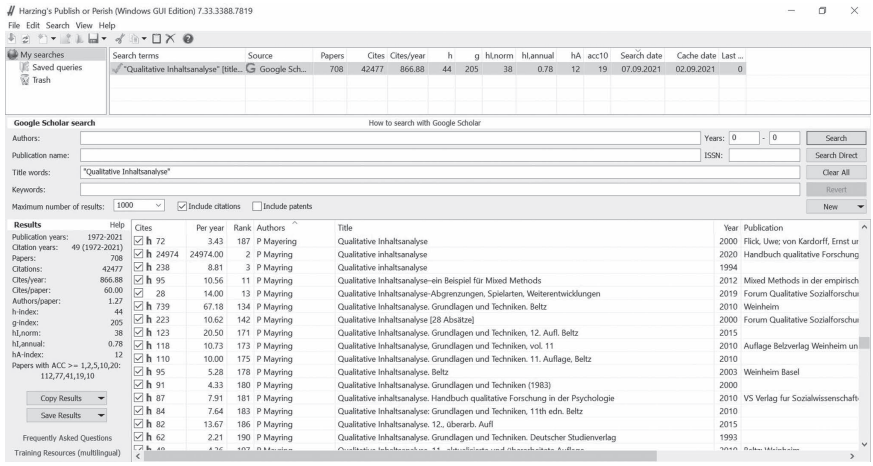
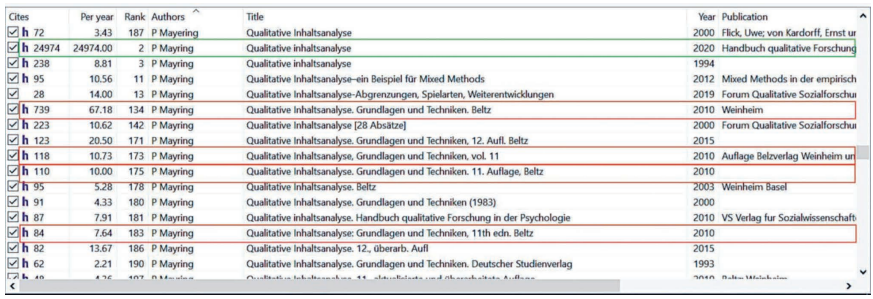


Figure 9.3 Google Scholar data error for Philipp Mayring: duplicate publications.



If you look at Figure 9.3, you will see in the second hit displayed (outlined in green) that this publication has 24,974 citations (1st column: Cites) and that these were all within one year (2nd column: Per year). You should always be suspicious about information like this! The reason why this information is given is that Google Scholar has summarized a total of 22 publications by Mayring with the title "Qualitative Content Analysis" in one hit. This, of course, does not correspond to one publication and completely distorts the result. And if you check, you will see that the number of citations given here does not correspond to the number of citations given for the 22 publications combined. It therefore remains unclear where this high number of citations comes from.

As a second example, we have outlined hits in red that are all the same publication, but that the Google Scholar crawling process has indicated as different publications (Figure 9.3). As you can see, the hits have different subtitles (Beltz,

vol. 11, 11th edition Beltz, and 11th ed. Beltz) depending on how they were listed on various home pages or in bibliographies. You could just add these hits together. But then you would have to go through the entire dataset as not only the literature references but also the author names are often incorrect. As Figure 9.4 shows, there are not only hits for Philipp Mayring as "P Mayring," but also as "M Philipp" and "M Philippe."

What we want to show with these examples is that Google Scholar is not a database, but instead a search engine and that this has an impact on the data quality. While databases systematize data entries and, most importantly, conduct quality control, this is not the case with a search engine. Bibliometric databases create metadata for publications, but this data is not used by Google Scholar. And as a result, errors often occur. First, author names are sometimes listed incorrectly (as shown in our example) (Orduna-Malea et al., 2017). Second, headings are only recognized if they differ in the layout from the rest of the text (Orduna-Malea et al., 2017). And third, errors such as the year of publication often occur because the parsers identify "any chain of 4 digits as a potential publication date, including page numbers or area codes or street addresses in author affiliation" (Orduna-Malea et al., 2017, p. 12).

Figure 9.4 Google Scholar data error for Philipp Mayring: incorrect author names.

Cites	Per year	Rank	Authors	Title	Year	Publication
0	0.00	119	M Mucundorfeanu	Qualitative Inhaltsanalyse der Zeitung Neuer Weg	2012	Journal of Media Research-Revis
0	0.00	689	M Nake	Motivationsstrategien in der Gamification Handbuchliteratur: eine qualitative Inhaltsanalyse der Handb...	2014	
3	3.00	39	M Niederberger, S ...	Die qualitative Inhaltsanalyse in den Gesundheitswissenschaften: Ergebnisse eines systematischen Revie...	2020	Forum Qualitative Sozialforschu
0	0.00	114	M Pawicki	Kuckartz, Udo (2012). Qualitative Inhaltsanalyse. Methoden, Praxis, Computerunterstützung. Weinheim ...	2014	Journal for educational research
h 102	7.29	176	M Philipp	Qualitative Inhaltsanalyse. Grundlagen und Techniken	2007	Beltz Deutscher Studien Verlag
h 80	3.81	182	M Philipp	Qualitative inhaltsanalyse	2000	Forum Qualitative Sozialforschu
4	0.22	298	M Philipp	Qualitative Inhaltsanalyse. Grundlagen und Techniken, Beltz	2003	
2	0.15	450	M Philipp	Qualitative Inhaltsanalyse. Grundlagen und Techniken. 10. Auflage, Beltz	2008	
2	0.10	472	M Philippe	Qualitative Inhaltsanalyse: Grundlagen und Techniken	2000	
3	0.33	296	M Plattner	Aufgabenkultur im Sportunterricht: eine qualitative Inhaltsanalyse der Bewegungsaufgaben von Mobile	2012	
2	0.22	448	M Plattner	Aufgabenkultur im Sportunterricht. Eine qualitative Inhaltsanalyse der Bewegungsaufgaben von Mobile...	2012	
0	0.00	139	M Reimann	Alternative Konfliktlösungsoptionen im Golfkrieg: eine qualitative Inhaltsanalyse deutscher Medien (199...	1997	
3	0.38	76	M Rogge	Die qualitative Inhaltsanalyse als Mittel zur empirischen Konkretisierung fremdsprachendidaktischer Par...	2013	Kolloquium Fremdsprachenunte
0	0.00	614	M Salih	Pressekommentare in der Medienberichterstattung über den Irakkrieg von 1991 und 2003: eine qualita...	2007	
0	0.00	696	M Schiffers	Forschungsdesign und qualitative Datenauswertung: vergleichendes Analyseraster der congruence anal...	2019	Lobbyisten am runden Tisch
23	2.88	21	M Schnell, C Schulz...	Der Patient am Lebensende: Eine Qualitative Inhaltsanalyse	2013	

We would like to add another reason here as to why an actual database that is monitored and selects metadata according to standards is preferable to Google Scholar. As part of digitalization and a system that makes it easy to publish on the Internet, there are predatory journals in academia and science that do not comply with research standards and promote pseudo-science (Grudniewicz et al., 2019). It is therefore important to emphasize once again that it is necessary to carefully select your publications and that databases should be used as they do not contain any predatory journals.

9.2.2 CrossRef

CrossRef is a non-commercial database that provides metadata (including citations). To do this, CrossRef works together with publishers and open access

repositories in so far that these provide CrossRef with the reference lists from their articles and CrossRef gives them the citations as links. Thanks to this partnership, CrossRef now has a database of over 120 million publications that have a DOI (Digital Object Identifier). The DOI is the only way to ensure clear attribution. This means that older books, for example, do not appear in the database as they usually do not have a DOI. The entire dataset can also be downloaded via an API and analyzed using Python or other programming languages (see Chapters 11–13). However, this requires advanced knowledge of Python as CrossRef does not support operators. As a result, you cannot use search phrases on CrossRef, but instead only individual words. These limitations make CrossRef difficult to use, especially for students.

9.2.3 Web of Science and Scopus

Web of Science (WoS) and Scopus are databases operated by Clarivate Analytics and Elsevier respectively. Both databases draw on data from publishers and contain peer-reviewed articles,[5] books, and conference papers. They now also include open access journals. Both databases require users to pay a fee, although they are accessible via libraries at many universities.

In terms of data, the two databases have five key weaknesses. First, the databases are primarily focused on English-language publications, which leads to an underrepresentation of other languages (and also to inequities and colonization tendencies, cf. Tennant, 2020). Second, the databases do now include books, but these are still in the minority. For example, WoS[6] lists 157,000 books (first year of coverage 2005) and Scopus[7] lists 399,000 books in its databases, while a publisher such as Springer[8] alone lists over 300,000 books on its homepage. In this respect, both databases have a major deficit when it comes to including books, which are still the central publication types in the humanities and in parts of the social sciences. Third, the databases are gatekeepers and only include the 'flagships' of journals according to their own quality criteria. WoS accepts only 10 to 12% of applications from journals who would like to be added.[9] As a result, small niche journals have little chance of being included in the index as it is primarily a matter of citation numbers. Fourth, when choosing a database, it is important to consider the type of publications they contain. WoS, for example, is the larger database in terms of age (1900 to the present), albeit with a focus on the natural and technical sciences. Scopus has a stronger focus on the social sciences, but only covers the period 1970 to the present. Fifth, WoS and Scopus only count citations that occur in publications found in their database. If an article is cited in a book that is not listed on WoS/Scopus, for example, this citation is not counted.

9.2.4 ResearchGate

In addition to search engines and databases, ResearchGate is a free though commercial social media application that allows social interaction between

authors in addition to providing publications. This requires a ResearchGate account, which can be created free of charge. It is then possible to search for publications and presentations and to save these under your profile. It is also possible to ask other ResearchGate members questions, comment on publications, and send direct messages.

ResearchGate records citations for the publications that are linked to an account. These are either found by a crawler on the web or in documents uploaded by ResearchGate members. However, ResearchGate does not make it possible to obtain this data for research purposes. And it is almost impossible to obtain the data through web scraping (Martín-Martín et al., 2021). ResearchGate is therefore not a viable alternative for conducting a bibliometric literature review.

9.3 Step 1: Formulate the research question

Every research process starts with a research question. For the BLR, it is important that you are sure about your research question before you start your research as you will use different data and databases and select the analysis methods depending on the research question.[10] The research process is illustrated here using three different research questions to illustrate the connection between the research question, the methods of analysis, and the selection of data. You can therefore read the individual steps for all three research questions or the individual steps for any one of the research questions as the steps are discussed in separate sections. The three research questions addressed are as follows:

1. **Literature review:** What are the key publications on a topic or in a field of research?
2. **Mapping:** How is the research field (on a specific topic) structured, that is, what connections can be found?
3. **Topic analysis:** What content is discussed in a research field or on a specific topic?

9.3.1 Literature review

As part of introductory courses during your studies, you have likely learned to use your library's search engine and find literature on a particular topic. This literature is then sorted according to relevance or publication date. But have you ever thought about what relevance means? Or maybe you use Google Scholar to carry out your literature search. But do you actually make it past the second page of hits? Did you know that Google doesn't reveal which algorithm is used to display the hits?

Neither of these methods are systematic and transparent ways of preparing a literature review. We will therefore present what an exemplary literature review would look like as part of the first research question. To do this, it is essential

that you clearly formulate the research question, such as: What are the central publications on the method 'qualitative content analysis'? The first step is to clarify what is actually meant by the term *central*. In bibliometrics, the centrality of a publication is measured by the frequency of its citations. But under no circumstances can you then assume that these publications are always of an especially high quality. Citation is not necessarily an expression of quality! Instead, citations refer solely to the attention (both positive and negative) that a publication receives.

9.3.2 Mapping

Different from a literature review, mapping is not only about identifying the central publications, but also about how the publications are related to each other. For this reason, mapping is also used for larger subject areas or scientific fields that are inaccessible by reading the material in its entirety. As an example, we will look at the research field of content analysis (both qualitative and quantitative) and find out which topics are discussed within the field, what characteristics they have and how they are connected.

Mapping uses two methods to illustrate these connections. The first is co-citation analysis and the second is bibliographic coupling. Co-citation analysis shows which publications have been cited together in another publication. And bibliographic coupling shows which publications have similarities based on the cited publications. Co-citation analysis allows for a look into the past of a research field as it shows which publications are central (= frequently cited) and how these cited publications relate to each other. In contrast, bibliographic coupling gives a view of the present as it shows how the publications in a research field are linked to each other.

9.3.3 Topic analysis

The third way of working with bibliometric methods is co-occurrence analysis, which analyses topics by linking words that occur together in publications. The relationship between the words is then displayed in a map. To create the map, the words that are present either in the title or in the title and abstract of a publication are analyzed. It is not possible to analyze entire publications because publishers don't make all of the content available (this is what is meant when we talk about a paywall).

To develop our research question, we will turn our focus to what we want to find out about topic analysis. In order to give you a full understanding of what is possible for a research field or topic using bibliometric methods, we will stick to our example and focus further on the research field of content analysis. In other words, we will ask the following question: Which topics related to content analysis appear in publications, and how are these topics connected to each other?

9.4 Step 2: Select a database and decide on the focus of the search

Once you have formulated your research question, the second step is to decide which database you want to use and what the focus of your search will be.

9.4.1 Literature review

In the literature review, we want to find the central publications connected with the qualitative content analysis method. For the literature review, it is important to develop your search strategy and narrow down your focus. This involves conducting both a systematic literature search and an in-depth examination of the research field. In our example, we could ask what the central publications are in a particular research context (e.g., qualitative content analysis in sociology or even more specifically in gender studies). We would then have to adjust the search strings (= search words or search phrases) to match the research context and not just search for the phrase "qualitative content analysis" but instead for coupled phrases such as "qualitative content analysis" AND "sociology." Since we want to have an overview of the publications on the method of qualitative content analysis, we will search for all publications.

In addition to the search strings, the second decision to make is whether these should only be searched for in the title or in the entire content of the publication. What is available under "entire content" varies depending on how freely accessible the publications are. If a journal article is behind a paywall, for example, then only the title, keywords, and abstract will be searched. If an article is open access and as such freely accessible, then the entire content can be searched.

For our example, we selected the search strings "qualitative content analyz*," "qualitative content analys*," and "qualitative Inhaltsanalyse" and then only searched for these in the title. Both "analyz*" and "analys*" were searched for in order to include both the British and American spellings. The asterisk in the search string means that "analyze" as well as "analysis" and "analyzes" are searched for. The reason why we only chose to search for these in the title is that we wanted to find publications that have qualitative content analysis as their central subject and those that include the method in their title would therefore be of particular interest. Since many libraries have access to WoS, which can also be used by students, we chose the WoS database for our analysis.

9.4.2 Mapping

Mapping can only be conducted with a database that connects citations to publications. This is currently done using the DOI (Digital Object Identifier). One such database is WoS, which we selected for our mapping process. This choice limits the results we will get (see Section 9.1.5) as the database only offers a specific selection of publications. Nevertheless, it is a database that is offered by more universities than Scopus, so we hope you will be able to use it.

For our research question of mapping, it is important to clarify which research field is to be considered exactly and how the hits can be narrowed down. It is essential to select the search strings in such a way that as many relevant hits (articles) as possible are included and as few hits as possible are listed that do not fit the topic (silence and noise) (Zitt & Bassecoulard, 2006). In our mapping research we want to look at the field of research that examines the content of (written) communication. This is referred to as 'content analysis', 'text analysis', and 'document analysis'. As you can see, you need to have a certain knowledge of the research field in order to map it.

We initially did a 'topic search' (TS) on WoS and searched for the search string 'content analy*'. A topic search identifies the phrase in the title, abstract, and keywords. Using an asterisk is a way to search for both "analyze" and "analysis", that is, for all variants that are possible with the word stem 'analy'. However, this search resulted in over 74,000 hits, which is far too many for mapping. It was therefore clear right after searching for the first search string that a topic search would not be the best option. We then conducted a more differentiated search, which only searched for the phrases in the title (TI). The three search strings were separated with an "OR" so that only one of them had to appear in the title. It is important to use parentheses when searching on WoS so that the phrases are searched for separately and together at the same time. Overall, the search looks like this:

((((TI=("content analy*")) OR (TI=("text analy*")) OR (TI=("document analy*"))))))

This search query resulted in 1,621 hits, which is an excellent number for mapping (see Steinhardt et al., 2017).

9.4.3 Topic analysis

To answer our research question as to which topics are connected with content analysis and how they relate to each other, we draw on the same base set of data as for the mapping (see previous section). This allows us to link the topic analysis back to the selection of publications and also to the research field. You have to remember that this step is about analyzing publications. If you are interested in developing topic areas as such, then you should instead use topic modeling (see Chapters 17 and 18) or an analysis with AntConc and qualitative data analysis software (see Chapter 8).

9.5 Step 3: Select and clean the data

9.5.1 Literature review

Having decided in Section 9.4.1 that the search terms 'qualitative content analy*' and 'qualitative Inhaltsanalyse' are what we want WoS to search for, we now need to select and cleanse the data. But what do you need to know in order

to be able to work with WoS? Well, you first need to check whether you have access to WoS via your higher education institution. To do this, just go to your university library site and find the databases. Once you've done this, search for 'Web of Science Core Collection'. WoS can only be used free of charge if your computer is logged on to the university network. Once you have access, you can find detailed step-by-step instructions on how to work with WoS on the blog sozmethode (see Box 9.1). We have decided to publish these step-by-step instructions online and not in the book because these would be too extensive and because tools change very quickly and need to be updated.

Box 9.1 How to work with Web of Science

For a step-by-step guide for using Web of Science, visit https://sozmethode.hypotheses.org/2837.

Back to our example: We searched WoS for the two phrases on September 14, 2021. We got 371 hits, a selection of the eight hits with the highest citation values is shown in Figure 9.5. We sorted the hits by 'citations: highest first', not by relevance, as is the default setting in WoS.

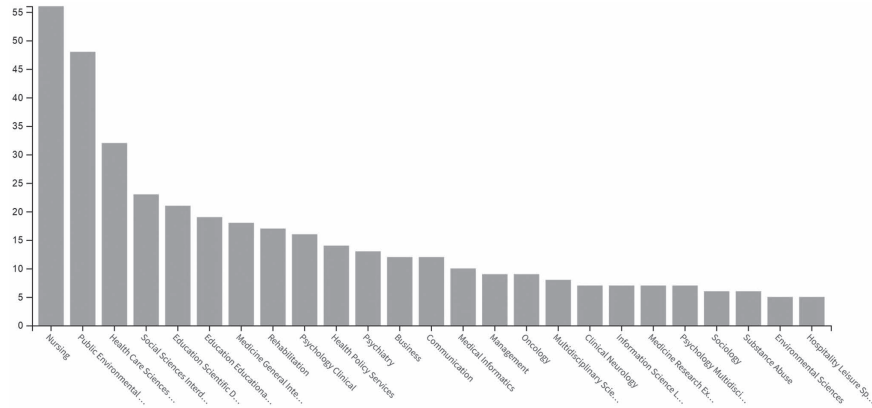
If you search WoS, it will give you a number of descriptive statistics that you can use to get an initial overview for your literature review. These include an overview of the research fields in which the publications were published. In our case, we see at Figure 9.6 that most of the publications that use the English phrase 'qualitative content analysis' were written in the field of health care. This is because there is a very long tradition of qualitative research in this field in the United States. It can therefore be assumed that these articles are primarily about the use of the method, and not about methodological discussions.

To verify this assumption, we take a closer look at the hit list (Figure 9.5). In our example, we notice firstly that the publications found all have the phrase we are looking for in the title (highlighted in yellow). Secondly, it is noticeable that there are two different types of publications, as we had already hypothesized: Those that address the method of qualitative content analysis itself and those that apply the method for research. Since we want to have an overview of the qualitative content analysis method – that is, hits that deal with the development of the method or provide an overview of the method – we have to sort out the hits that describe applying the method. In doing so, we have defined our criteria for selecting data.

Using our selection criterion, we include publications 1 to 7 from the hit list (Figure 9.5) in our sample as they describe (and perhaps develop) the method itself. But we would not choose the publication 8 from the hit list titled "What Patients Say About Their Doctors Online: A Qualitative Content Analysis" because it only uses the method. To select data, you can either go directly through the hit list on WoS and check publications that meet your selection criteria and then download the selected results as a tab-delimited file. Or you can download the entire list of results and sort it in Excel. We would recommend

Figure 9.5 Web of Science hit list of the eight most cited articles found with the search strings 'qualitative content analy*' and the German equivalent 'qualitative Inhaltsanalyse'.

Figure 9.6 Research Fields in Web of Science found with the search strings 'qualitative content analy*' and the German equivalent 'qualitative Inhaltsanalyse'.



the second option as it is not as likely that you would lose your work in Excel (e.g., in the event of Internet problems), and you will be able to retrace your selection more easily at a later date. If you cannot decide directly from the title whether a publication meets your criteria, then it is best to look at the abstract and read this short summary of the content.

Once you have downloaded the hit list, check it for possible duplicates in both the names and the titles of the publications. In our example, there was no duplication of names, but there was a duplication of titles. It turned out that both were book reviews, and since our focus is on research publications, not reviews, we removed both from our list as not relevant to our study. We also sorted out an article that was announcing a workshop and therefore was not a research article. After we reviewed and selected publications, we have 26 (out of 371) remaining in our example that meet our criterion of providing an overview or describing the further development of the qualitative content analysis method.

9.5.2 Mapping and topic analysis

If you have already read the previous Section 9.5.1, you can skip the following paragraph as it is a repetition. If not, we will give you a brief introduction to using WoS. First you need to check whether you have access to WoS via your higher education institution. To do this, just go to your university library site and find the databases. Once you've done this, search for 'Web of Science Core Collection'. WoS can only be used free of charge if your computer is logged on to the university network. Once you have access, you can find detailed step-by-step instructions on how to work with WoS on the blog sozmethode (see Box 9.2). We have decided to publish these step-by-step instructions online

and not in the book because these would be too extensive and because the tools change very quickly and need to be updated.

> Box 9.2 How to work with Web of Science
>
> For a step-by-step guide for using Web of Science, visit https://sozmethode.hypotheses.org/2837.

In our mapping example, we carried out the title search described previously (Section 9.4.2) on 2 September 2021 and received 1,621 hits. We downloaded these hits from WoS in increments of 500 and then merged the individual packages into an Excel-list. Next, we checked whether there were any duplicates in the list, which was not the case. We excluded reviews and book reviews as both formats provide summaries of other publications, which could have had a distorting effect on our analysis. As the list always shows the type of publication, it is easy to exclude certain genres. After cleaning the data, we still had 1,350 publications, which we continued to work with.

9.6 Step 4: Carry out the bibliometric analysis

9.6.1 Literature review

We now have a list of 26 publications that we will analyze further. As a first step, you can take a look at where the articles have been published and what information you can glean from the titles. In our sample 13 publications come from the field of health sciences (highlighted in grey). It is interesting to note that two publications date back to the 1980s and that the others are all from 2004 onwards. Twenty of the 26 publications were published after 2011. At least in articles in the WoS database, qualitative content analysis is a method that was primarily discussed in more recent publications.

For the literature review, we will now go into the publications and look at their content. We will look for initial evidence to answer our research question by starting with the publications that have the most citations per year. However, you must be aware that this does not mean that these publications are of a high quality, only that other people have considered the publications worth citing (for whatever reason). We start with the total number of publications from the WoS list in the column with the heading TC, which stands for Times Cited Counts. Then we divide this number by the age of the publication to obtain the average number of citations per year. Table 9.1 clearly shows that there are four publications with an average of over 150 citations per year. We start our literature review with these and read the publications.

After reading the most frequently cited articles, the next step is to categorize the articles, that is, to sort them according to criteria. In a literature review, when the most important information from the publications is summarized and sorted, categorizing is usually an inductive process (see Chapter 6). In our

Table 9.1 Sample of 26 Publications From WoS Found With the Search Strings 'Qualitative Content Analy*' and the German Equivalent 'Qualitative Inhaltsanalyse'

	Author(s)	*Publication Title*	*Place of Publication*	*Year of Publication*	*Citations per Year*
3	Lorenzer, A.	The possibility of a qualitative content analysis – a deep hermeneutic interpretation between ideology criticism and psychoanalysis	Argument	1981	0.1
3	Glassner, B. & Corzine, J.	Library research as fieldwork – a strategy for qualitative content analysis	Sociology and Social Research	1982	0.2
1	Graneheim, U. H. & Lundman, B.	Qualitative content analysis in nursing research: concepts, procedures and measures to achieve trustworthiness	Nurse Education Today	2004	487.2
1	Hsieh, H. F. & Shannon, S. E.	Three approaches to qualitative content analysis	Qualitative Health Research	2005	956.8
2	Burla, L., Knierim, B., Barth, J., Liewald, K., Duetz, M., & Abel, T.	From text to codings – Intercoder reliability assessment in qualitative content analysis	Nursing Research	2008	17.8
1	Elo, S. & Kyngas, H.	The qualitative content analysis process	Journal of Advance Nursing	2008	535.4
1	Moretti, F., van Vliet, L., Bensing, J., Deledda, G., Mazzi, M., Rimondini, M., Zimmermann, C., & Fletcher, I.	A standardized approach to qualitative content analysis of focus group discussions from different countries	Patient Education and Counseling	2011	15.1
3	Oleinik, A.	Mixing quantitative and qualitative content analysis: triangulation at work	Quality & Quantity	2011	2.4

(*Continued*)

Table 9.1 (Continued)

	Author(s)	*Publication Title*	*Place of Publication*	*Year of Publication*	*Citations per Year*
1; 3	Mayring, P.	Qualitative Content Analysis: Theoretical Background and Procedures	Approach to Qualitative Research in Mathematic Education	2015	22.3
3; 4	Bakharia, A., Bruza, P., Watters, J., Narayan, B., & Sitbon, L.	Interactive Topic modeling for aiding Qualitative Content Analysis	Proceedings of the 2016 ACM Conference on Human Information Interaction and Retrieval (CHIIR'16)	2016	1.0
1	Plumins, M., Sceulovs, D., & le-Sarkane, E.	Competitiveness definitions' and concepts qualitative content analysis	Smart and Efficient Economy: Preparation for the Future innovative Economy	2016	0.4
1	Graneheim, U. H., Lindgren, B. M., & Lundman, B.	Methodological challenges in qualitative content analysis: A discussion paper	Nurse Education Today	2017	160.5
1	Armat, M. R., Assarroudi, A., Rad, M., Sharifi, H., & Heydari, A.	Inductive and Deductive: Ambiguous Labels in Qualitative Content Analysis	Qualitative Report	2018	9.6
1	Assarroudi, A., Nabavi, F. H., Armat, M. R., Ebadi, A., & Vaismoradi, M.	Directed qualitative content analysis: the description and elaboration of its underpinning methods and data analysis process	Journal of Research in Nursing	2018	38.0
1; 3	Hall, C. M.	Quantitative and qualitative content analysis	Handbook of Research Methods for Tourism and Hospitality Management	2018	1.0
1	Ramakrishna, R. A. R.	Using intuitive judgment in qualitative content analysis: implications for research in varieties of English	International Journal of Business and Society	2018	0.0

(*Continued*)

Table 9.1 (Continued)

	Author(s)	*Publication Title*	*Place of Publication*	*Year of Publication*	*Citations per Year*
3; 4	Renz, S. M., Carrington, J. M., & Badger, T. A.	Two Strategies for Qualitative Content Analysis: An Intramethod Approach to Triangulation	Qualitative Health Research	2018	11.6
1	Faria-Schutzer, D. B., Bastos, R., Alves, V., Campos, C. J. G., Surita, F. G., & Turato, E. R.	The seven steps of the clinical-qualitative content analysis: a data-processing technique for research into clinical care settings	European Psychiatry	2019	0.0
1	Hoft, N., Heckmann, M., & Jankowicz, D.	Systematic Integration of Quantitative Measures into the Qualitative Content Analysis of Constructs	Journal of Constructivist Psychology	2019	1.0
1	Kibiswa, N. K.	Directed Qualitative Content Analysis (DQlCA): A Tool for Conflict Analysis	Qualitative Report	2019	1.5
1	Pohontsch, N. J.	Qualitative Content Analysis	Rehabilitation	2019	2.0
1	Hall, M., Wucherpfennig, F., & Rubel, J. A.	Qualitative Content Analysis	Psychotherapy Psychosomatics Medical psychology	2020	1.0
1	Lindgren, B. M., Lundman, B., & Graneheim, U. H.	Abstraction and interpretation during the qualitative content analysis process	International Journal of Nursing Studies	2020	40.0
1	Manic, Z.	Performing qualitative content analysis	Sociologija	2020	1.0
1	Selvi, A. F.	Qualitative content analysis	Routledge Handbook of Research Methods in Applied Linguistics	2020	11.0
1	de Faria-Schutzer, D. B., Surita, F. G., Alves, V. L. P., Bastos, R.A., Campos, C. J. G., & Turato, E. R.	Seven steps for qualitative treatment in health research: the Clinical-Qualitative Content Analysis	Ciencia & Saude Coletiva	2021	0.0

example, there are three categories, although there are publications that can be assigned to multiple categories.

- Table 9.1, Column 1, Number 1: Publications that summarize and structure qualitative content analysis. Special focus is placed on the coding process and specific instructions on how to carry out qualitative content analysis in its different variants.
- Table 9.1 Column 1 Number 2: Publications dealing with quality criteria for qualitative content analysis.
- Table 9.1 Column 1 Number 3: Publications that make comparisons between qualitative content analysis and other methods or use qualitative content analysis together with other methods (mixed methods design or triangulation).
- Table 9.1 Column 1 Number 4: These publications present the possibilities of computer-assisted qualitative content analysis.

9.6.2 Mapping

Now that we have our base set of data, it's time to analyze it. To create the co-citation analysis and bibliographic coupling, we use the open-source tool VosViewer (see Section 1.5 for copylefts and copyrights). You can easily download it at *www.vosviewer.com*. VosViewer allows you to feed the downloaded txt file from WoS into the program, and then it performs the analysis. As described previously, both co-citation analysis and bibliographic coupling are of interest for mapping. For both analyses, VosViewer creates maps that differ in terms of which publications are displayed on the map. A co-citation analysis allows us to take a look into the 'past'. The map shows us the publications that were cited at a very high frequency in the publications in our sample (i.e., the 1,350 publications). Accordingly, articles that have been published for a longer period of time appear more frequently than more recent articles, as they have had more time to be read and thus cited. Therefore, a normalization can be carried out when creating the map in VosViewer if this correlation is not wanted – more details later.

The bibliographic coupling map, on the other hand, maps the publications in the sample and links those that cite the same publications (i.e., the shared bibliography). This shows current developments in a research field, which can lead to a situation where no clear map can be created if the research field is not coherently connected (more on this later).

First, we will show you the co-citation map. To create this map, we fed the txt file from WoS into the VosViewer (the exact procedure can be found in the instructions online, see Box 9.3). To create a meaningful co-citation map, you need to reduce the number of publications shown. In our example, if all 1,350 publications were shown as a network, you would only see a large pile of dots and dashes. In terms of reduction, we have found in previous studies (Steinhardt et al., 2017) that reducing the number of publications displayed to around 200 via the number of citations is an effective way to obtain a meaningful map.

You can achieve this number by manipulating the threshold in VosViewer. In our example, the threshold is set to a minimum of five citations so that 218 publications are displayed on the map. Once the map has been created, you can select the normalization method that compensates for the age of the publications; we prefer the association strength method.

Box 9.3 Creating maps with VosViewer

For a step-by-step guide for using VosViewer and creating co-citation maps and bibliographic coupling maps, visit https://sozmethode.hypotheses.org/2837.

You can also change the layout so that the map either contracts or slightly expands. In VosViewer, the settings "Attraction 2" and "Repulsion 1" are set automatically; for our example, we have selected "Attraction 6" and "Repulsion 2" in order to expand the map. There are still a number of other ways to change the layout including the color, background, and size of the symbols. To find out more, you can also take a look at the VosViewer Manual: *https://www.vosviewer.com/documentation/Manual_VOSviewer_1.6.20.pdf*.

It is important to note that we do not need to analyze the content of all the texts in the corpus, but we do need to understand what the arrangement and connections between the color-coded clusters mean. To interpret the clusters and their links, it is useful to read the most central publications per cluster. Further research has shown that a very good overview of the topics in a cluster can be obtained by reading the five to ten most central publications per cluster (Steinhardt et al., 2017).

To do this, we download the map data from VosViewer and can then open the txt file we get in Excel (see detailed instructions Box 9.3). The Excel spreadsheet you now have contains publication abbreviations, such as DOIs and book names, so that you can easily find the central publications (more on this in a moment). The Excel spreadsheet also shows the cluster number and the centrality (i.e., importance) of each article using three measures. First, the centrality, measured by the number of links between the publications 'weight<links>'. Second, the strength of the links 'weight<total link strength>' and third, the frequency with which the publications were cited 'weight<citations>'. We can now sort the Excel list by cluster and centrality of the publication and use the DOI or book title to search for the central publications for each cluster. For a more detailed interpretation of the clusters, see Section 9.7.2.

For bibliographic coupling, we take the same dataset as for the co-citation analysis and use the VosViewer to create the map. As before, we also set a threshold of about 200 publications for bibliographic coupling. It only takes eleven citations to reach this threshold, whereby ultimately only 177 publications are included in the map as the other 23 publications have no links.

As Figure 9.8 shows, the bibliographic coupling map is very difficult to interpret. Despite the small number of publications selected, there are a large number of clusters (11) but no clearly distinguishable clusters. To create a better

Figure 9.7 Co-citation analysis with data from WoS.

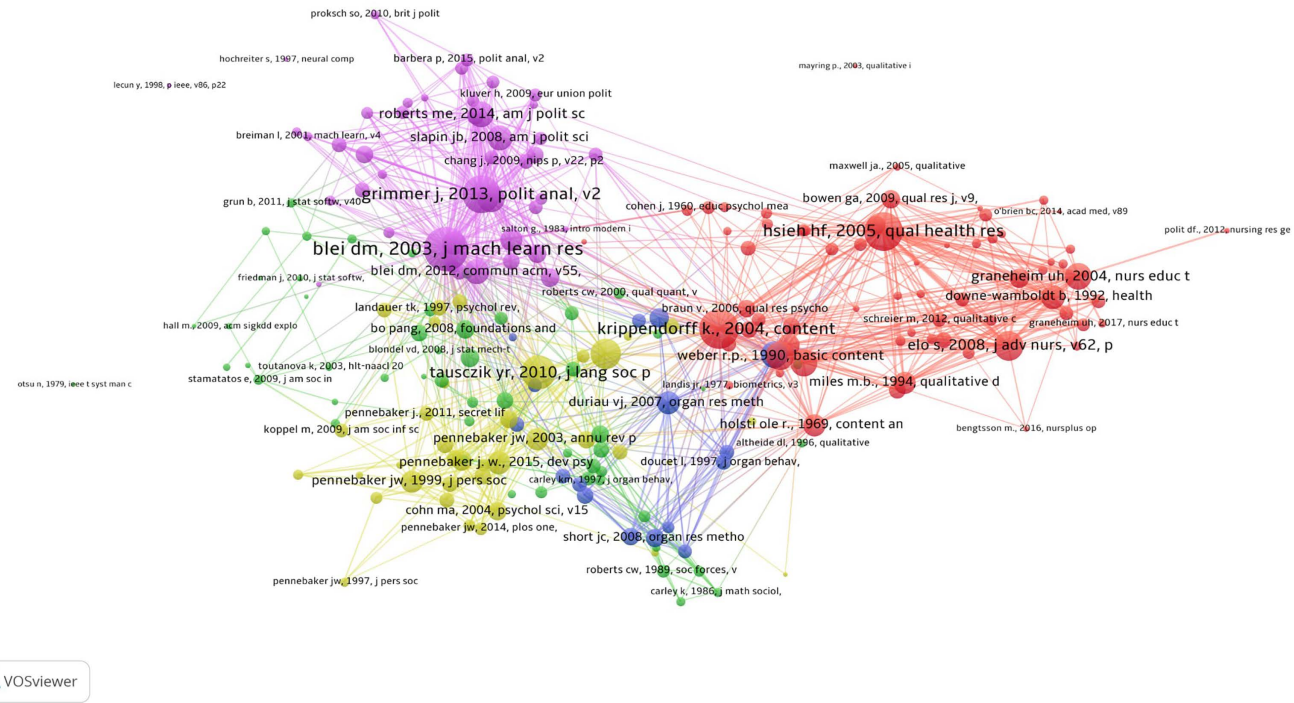
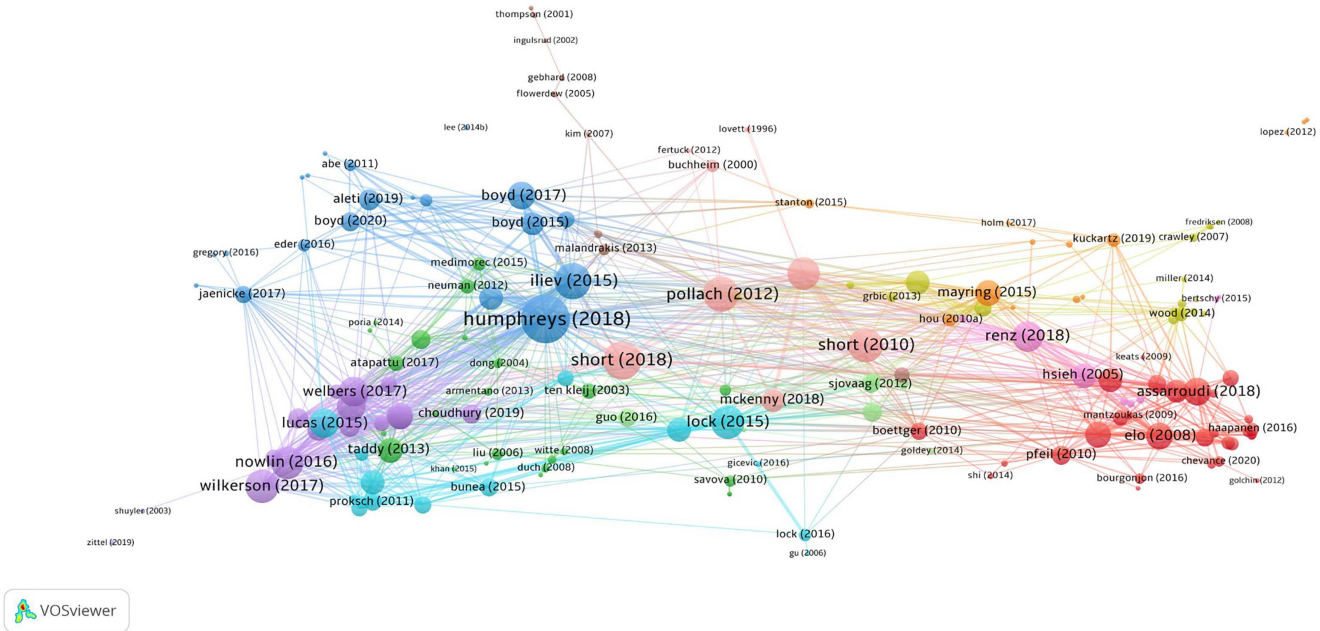


Figure 9.8 Bibliographic coupling using data from WoS.



map, we have tried all the variants with a very low threshold and a very high threshold and can tell you that the map does not change in terms of clarity.

With such a map, you can either focus on the large clusters if you want to get an overview of the research field. Or you can focus on the small, peripheral clusters if you want to discover new developments in a field of research. This is because bibliographic linkage provides a view of the present by grouping together in a cluster certain publications that have many similarities in their bibliographies, that is, cite the same publications. Since the aim of our research question is to get an overview, we focus on the large clusters, that is, the blue, red, pink, and purple clusters.

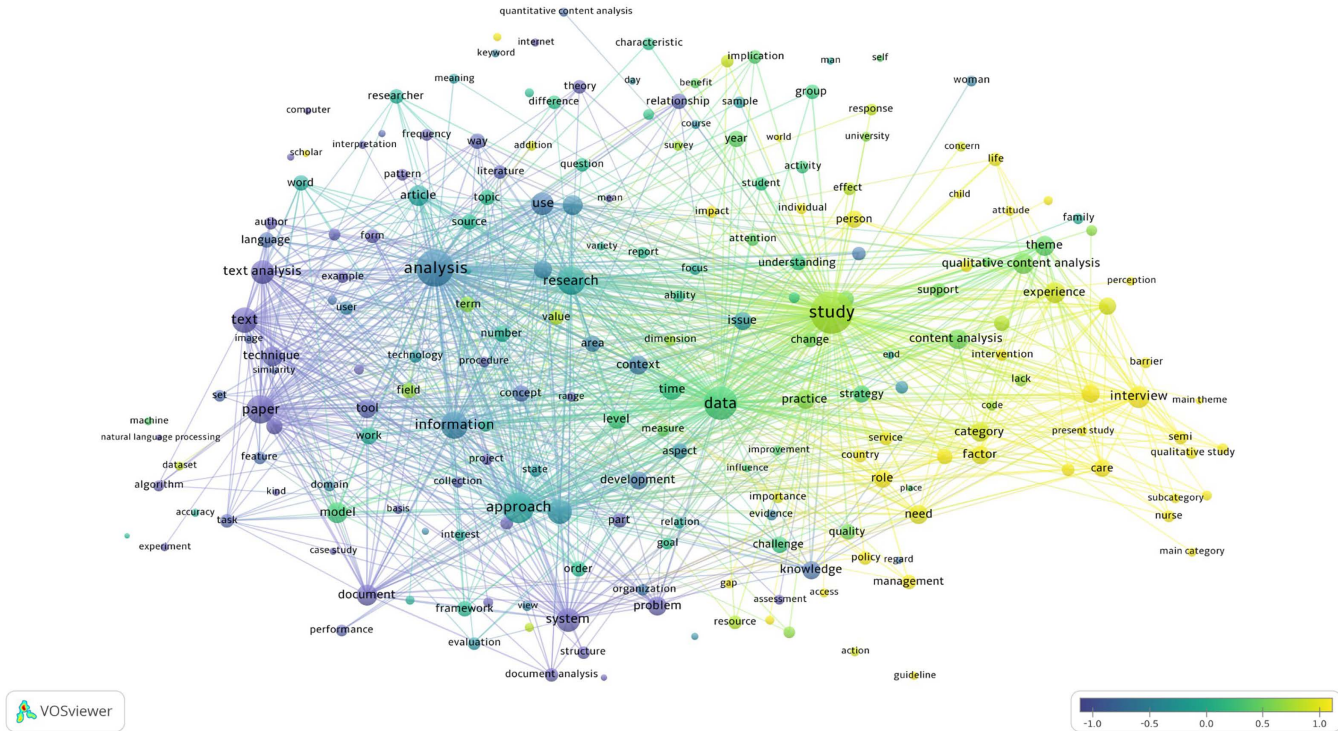
To analyze the content of the clusters, we need the publications that are grouped in each cluster. To do this, we download the map data from VosViewer and can then open the txt file we get in Excel (see detailed instructions Box 9.3). The Excel list shows 'labels', that is, the names that are also shown in the map. These are abbreviations that need to be matched with the full list of publications in the corpus (see detailed instructions Box 9.3). Once this step has been completed, the titles and abstracts of the publications can be used to gain a first insight into the content of the clusters. For a comprehensive understanding of the topics of the respective clusters, it is also necessary to read the most central 5–10 publications per cluster and identify the similarities. For a more detailed interpretation of the clusters, see Section 9.7.2.

9.6.3 Topic analysis

Now that we have our set of data, it's time to analyze it. We use the open-source tool VosViewer to perform the topic analysis (co-occurrence analysis). You can easily download it at *www.vosviewer.com*. VosViewer allows you to simply feed the downloaded txt file from WoS into the program and then carry out the analysis. The co-occurrence analysis provides an overview of the central words used in the title and/or abstract in the literature corpus. VosViewer uses the words to create a map that shows the connections between the words. The program also allows you to choose between binary counting and full counting. Binary counting specifies the number of documents in which a term occurs at least once. Full counting shows the total number of occurrences of a term in all documents. In our example analysis (see Figure 9.9) we have opted for binary counting as we are interested in the prevalence of certain terms. Full counting can easily lead to distortions if a term is used at a high frequency in a small number of publications.

The next step is to set the number of occurrences above which a term is included in the map. The number that limits the inclusion of a term on the map is called the *threshold*. In our previous studies, we have found that it works well to have a threshold that includes around 200 words on the map. And in our example, the threshold was reached at 26 occurrences so that 209 words were included in the map. For the visualization, we opted for overlay visualization as

Figure 9.9 Co-occurrence analysis with data from WoS.



the color gradient makes it easy to see the connections between the terms and at the same time shows the clusters (see Figure 9.9).

9.7 Step 5: Interpret the results

9.7.1 Literature review

In our example of literature review, there are two key findings: First, the English-language discussion of qualitative content analysis takes place primarily in the field of health sciences. Second, there are four categories of literature that focus on different and sometimes overlapping content.

1. category: the summary and structuring of qualitative content analysis. Special focus is placed on the coding process and specific instructions on how to carry out qualitative content analysis in its different variants.
2. category: texts that deal with the quality criteria of qualitative content analysis.
3. category: comparisons between (or combinations of) other methods and qualitative content analysis.
4. category: texts that focus on the possibilities of computer-assisted qualitative content analysis.

These results would make it possible for you to write a systematic overview of the literature, that is, the current state of research. This can help you to identify where there are still gaps in the research or where there is disagreement in the research and where it would be interesting to take a closer look. For example, we found that there is no book that combines qualitative and quantitative content analysis and represents an interpretative paradigm. That's why we wrote this book.

9.7.2 Mapping

Once we have created the maps, we can now answer our mapping research question "How is the research field (on a specific topic) structured, that is, what connections can be found?" In the first step, we analyze the co-citation map and the bibliographic coupling map separately from each other descriptively and interpretatively. In a second step, we then use these analyses together to provide a final answer to the research question.

The first step in interpreting the maps is to describe them. This means looking at the map and writing down anything that catches your eye. In our example, the co-citation map (see Figure 9.7) can be described as follows: First, you will notice that the circles in the purple, yellow, and red clusters are larger than those in the other clusters. Larger circles on the map indicate that the articles have been cited more frequently and that they are linked to other publications. Second, the red and purple clusters are denser, and the circles and links

are therefore more separated from the other clusters. This makes these clusters appear larger even though they are not so different in terms of the number of publications they contain. In our example, the red cluster contains 67 publications, the purple cluster 44, the green cluster 54, and the yellow cluster 37, whereas it is immediately clear that the blue cluster is significantly smaller with 16 publications. And third, you will notice that the green cluster is spread across the entire map and that it would therefore be difficult to establish its borders. At the same time, the map clearly shows that the green cluster has hardly any points of contact with the red cluster, which in turn has hardly any points of contact with the purple cluster.

The descriptive description is followed by an analysis of the 10 most central texts in order to understand what content the clusters represent. The results can be summarized as follows:

- **The red cluster:** The red cluster primarily contains publications from the discipline of health sciences. And within this discipline, we can identify two areas of concentration. First, we have qualitative content analysis, which is clustered around the central publications that we already know from the literature analysis on qualitative content analysis (Elo & Kyngäs, 2008; Graneheim & Lundman, 2004; Hsieh & Shannon, 2005). And then we have seminal works on content analysis with a connection to computer-assisted content analysis (Krippendorff, 2004) and to quantitative content analysis (Weber, 1990). This is also the reason why these two texts form the transition to the blue and yellow clusters.
- **The blue cluster:** The blue cluster contains only publications from the field of economics (Duriau et al., 2007; Morris, 1994; Short et al., 2010; Short & Palmer, 2008), which is why a separate cluster was formed here. These publications discuss quantitative content analysis as a method, including the use of automated computer-assisted analysis. The central book in the cluster (Neuendorf, 2002) provides an introduction to quantitative content analysis and is therefore close to the books in the red cluster.
- **The yellow cluster:** Like the red and blue clusters, the yellow cluster is also a disciplinary cluster, in this case for psychology. The central publications deal with questions of how automated text analysis can code words in psychologically meaningful categories (Newman et al., 2003; Pennebaker et al., 2015; Tausczik & Pennebaker, 2010). Furthermore, we notice that this cluster is dominated by the author James W. Pennebaker, who is involved in all central publications apart from the central book. The yellow cluster (Stone et al., 1966) also has one book that, like the books in the other clusters, is centrally located in the map as a whole and deals with computer-assisted content analysis.
- **The purple cluster:** The purple cluster also has a disciplinary focus: political science. The most central publications focus on how political texts can be evaluated as data using computer-assisted content analysis (Grimmer & Stewart, 2013; Laver et al., 2003). The topic modeling method is used for

this purpose, which is why the description of the method also takes up space (Blei, 2012; Blei et al., 2003; Roberts et al., 2014).

- **The green cluster:** The last cluster is the green one, which extends into the other clusters and cannot be clearly delimited. This is because it is not a disciplinary cluster, but a cluster that deals with the question of the meaning of text and words (Miller, 1995; Mohr, 1998) that can be determined using computer-assisted text analysis procedures and methods (Popping, 2000). These procedures and methods include statistical natural language processing (Manning & Schütze, 1999) and sentiment analysis (Pang & Lee, 2008).

Overall, co-citation analysis shows that the research field of content analysis has disciplinary roots and is primarily shaped by the methods of computer-assisted quantitative content analysis. Co-citation analysis makes it possible to look into the past, that is, to see which publications are referred to. In order to get an insight into the present, it is therefore always worthwhile to perform the bibliometric coupling.

The bibliographic coupling map is initially very difficult to interpret, as Figure 9.8 shows. Despite the small number of publications selected, there are a large number of clusters (11). After reading the central texts, however, there is a clear distinction between quantitative content analysis on the left, consisting of the purple, blue, turquoise, and dark green clusters, and qualitative content analysis, consisting of the red, pink, and orange clusters. In between is the light pink cluster, which includes computer-aided text analysis. The yellow, light green, and brown clusters cannot be interpreted due to their thematic breadth. The map also clearly shows that the quantitative content analysis clusters are larger, indicating that there are more publications in this area in the corpus and that they are more closely interlinked.

The purple cluster is a methodological cluster on the topic of computer-aided text analysis. The topics of the articles in the purple cluster are, for example, text analysis in R or machine learning approaches. The blue and turquoise clusters are related to the purple cluster in that they deal with computer-aided or automated text analysis from a disciplinary perspective. The turquoise cluster includes publications from political science and topics such as quantitative text analysis of lobbying and interest groups, electoral pledges, or applications to comparative politics. The blue and largest cluster combines publications from several disciplines, such as psychology, digital humanities, and consumer research. The central commonality of the cluster seems to be the analysis of narrative structures in texts, for example to uncover psychological changes in language or to analyze consumer behavior. The green cluster also deals with quantitative text analysis, but without a clear disciplinary location and using different methodological procedures. For this reason, the green cluster extends into the other quantitative clusters and shows no clear demarcation.

In the middle is the light pink cluster, which lies between quantitative and qualitative content analysis, as it contains publications that combine

both methods or reflect methodologically on how construct validation of computer-aided text analysis can be produced.

On the right, you will find the three clusters that focus on qualitative content analysis. The red cluster contains methodological introductions to qualitative content analysis and application examples from different areas, such as the analysis of video game discussion forums or the qualitative content analysis of an online survey. The pink cluster consists of publications that make or present methodological comparisons related to qualitative content analysis. For example, they present three approaches to qualitative content analysis or present triangulation possibilities. The orange cluster contains publications that provide introductions to qualitative content analysis. Since some of the authors are German, it can be assumed that the literature used is very different from the literature cited in the red and pink clusters, and therefore a separate cluster has been formed.

What do these results mean? It means that content analysis is currently not a homogeneous field of research. Rather, there is a clear separation between qualitative and quantitative content analysis. That is why we have written this book, to overcome this strong separation and to show how a combination is possible. The analysis also shows that there are trends in the use of certain methods in different disciplines. This is due to specific disciplinary questions that can be answered better with certain methods than with others (see Chapter 1). The analyses also show that there are disciplines that do not appear, which is surprising. Sociology, for example, does not appear as a discipline, even though content analysis is also used in this discipline. For a more in-depth analysis, you could go into the analysis of individual clusters. Or you could look at why certain disciplines appear more often than others. The depth of your interpretation will depend on your research question.

9.7.3 Topic analysis

The first step in interpreting the co-occurrence map is to describe it. This means looking at the map (see Figure 9.9) and writing down anything that catches your eye. The first thing we notice on the map is that there are several central points in the middle: analysis, approach, data, research, and study. This is not surprising as these are terms that appear in every publication that is based on empirical work. As a result, the smaller circles and color gradient provide us with more information. The map shows us that there is a dark blue cluster on the left-hand side that represents the analysis of text products such as papers that deal with text analysis methods and are, for example, supported by algorithms and related to linguistics. As it is connected to text analysis, document analysis can be found at the bottom left of the cluster. Document analysis is associated with terms such as ‘evaluation’, ‘organization’, ‘problem’, and ‘system’. This indicates that the analysis of documents from organizations that focus on specific problems is being examined here. There is a yellow and light green cluster on the right-hand side of the map. These clusters focus on quantitative

and qualitative content analysis. Based on terms such as 'nurse', 'care', and 'interview', we can see that the yellow cluster is in the health sciences, which we discovered as part of the co-citation analysis.

The topic analysis gives you a good overview of the terms that are central to the research field you are investigating. The color-coded map also displays the terms in clusters so that boundaries become visible.

9.8 Bibliometric literature review and artificial intelligence support

At this point, we would like to also introduce seeking support by using artificial intelligence to analyze the clusters. Once you have reached step 4 (you have created the maps and are analyzing the respective publications in the clusters), to support your analysis, it may be useful to use Large Language Models (LLMs) such as ChatGPT, Lama, or LeChat to summarize content and compare the publications (see Chapter 10 for how to use LLMs). To use AI support for content summary, you need to upload the PDF of a publication to a LLM-platform and formulate a prompt that gives precise instructions on what the LLM should do. It makes sense to use a few-shot prompt. A few-shot prompt consists of (1) the task, (2) the assignment of a role such as researcher or analyst, and (3) output indicators such as length of text and type of answer. For example, the following prompt contains the basic information to be given in a polite way to the LLM:

> Please provide a summary of the PDF. The summary should provide an overview of the central aspects of the text, such as the research question, the method used, the theory used, the disciplinary background, and the object of research. You assume the role of the sociologist. The summary should be organized around the central aspects of the text.

The LLM-produced content summaries then make it easy for you to compare the publications with each other and to search for an interpretation of why they were grouped into a topic cluster. A note of caution: the LLM summary does not relieve you from reading publications, at least (some of) the most relevant publications in a cluster, to make sure that the LLM does not make up content.

You can also directly compare multiple PDFs or just the abstracts of the publications in a cluster and ask the LLM to identify similarities in the publications. If you upload multiple PDFs of publications, keep in mind that the LLMs have a limited upload and computing capacity. Depending on the LLM you are using, it may only be possible to compare the abstracts instead of the full publication texts.

For abstract comparison, it is best to create a document in which you copy and paste the abstracts of publications. You should number abstracts consecutively so that the LLM knows when a new abstract starts. Do not forget to document in your research journal which number belongs to which publication. Before you paste the abstracts in the context window (i.e., the input mask

in which you communicate with ChatGPT, for example), you must check the maximum length of LLM text analysis, which varies by LLMs and model version. The context window specifies the length of text as the maximum number of tokens. A token is the basic unit that LLMs use to divide text into units for statistical probability calculations. A helpful rule of thumb is that one token is generally equivalent to ~4 characters of text for standard English text. For example, the version GPT 4o has a context window allowing 128,000 tokens, which corresponds to approximately 96,000 words (ca. 270 A4 pages with 350 words each).

Again, it is important to formulate a prompt that gives precise instructions to the LLM. Avoid one-shot prompts that just say 'search for similarities in the abstracts'. A good prompt for comparing abstracts might be:

> Please compare the following abstracts and focus on the similarities. The abstracts are separated by paragraph marks, and each begins with "Abstract" and a sequential number. The purpose of the comparison is to identify similarities, such as method used, theory employed, disciplinary background, or research topic. You are in the role of a sociologist. Please tell me the main similarities between the abstracts.

You can also access the most recently developed reasoning models. Simply put, reasoning suggests the ability of LLMs to draw conclusions. While conventional, non-reasoning LLMs generate text primarily on the basis of probabilities – that is, predicting which word or piece of information statistically fits best in the context – reasoning models go beyond this. They are programmed to make plausible and stringent connections between pieces of information, mimicking the steps of analytical thinking. Instead of just calculating superficial probabilities of words in context, these models perform internal structural analysis to explicitly derive plausible conclusions. When you pay for the use of such a reasoning model, you can instruct the model to access certain documents stored locally. For example, you could create a folder in which you store the publication PDFs of the ten most central texts in a cluster and instruct the reasoning LLM to summarize or compare them. The advantage of reasoning models is that you can follow how the model is proceeding in real time and intervene if an error occurs.

Reasoning or not, when working with LLMs, we recommend that you always do both summaries and comparisons of publications. If you go straight for comparisons of documents or abstracts only, you will most likely miss important aspects in the content of a publication that you could have noticed in the summaries. You should always check, if content of one or more texts are missing in the comparison. Finally, in the end, it is you who should read those texts yourself to find out why they were grouped in a cluster and interpret the key links between clustered publications.

A final note of caution: in 2025, you are moving on very thin ice when using LLMs to analyze PDFs of publications that are behind a paywall, that is, owned

by a publisher. What does that mean? For example, if you can access a PDF of an article through your university library, which is not open access, then you are not allowed by copyright law to share it with others – others include LLMs' owner companies, which is problematic under copyright law. For sure, such infringement is, at least, unethical and may have legal consequences. You should be aware of that.

Notes

1 With regard to the question of what constitutes a research publication, there is a wide-ranging debate about, for example, when it is research at all, whether only publications that have undergone a peer-review process should be included, with the peer-review process itself coming under strong criticism, or which journals can be classified as research journals and which should be considered predatory journals (for a detailed description of the debate, see Stock et al., 2023).
2 ISBN = International Standard Book Number, ISSN = International Standard Serial Number, and DOI = Digital Object Identifier.
3 Along with publishers, many Internet platforms also issue DOIs. These include non-commercial preprint servers (e.g., Zenodo and SocArXiv), where publications that have not yet been published can be made publicly accessible (open access). Preprints are manuscripts that have either not yet undergone a review process or have already been reviewed and approved for publication but have not yet been published. DOIs are also issued by commercial platforms such as ResearchGate and Academia.
4 Reproducibility, also known as replicability, means that other researchers will conduct the same results if they adhere to your research procedure (data collection, processing, and analysis).
5 Peer review is the evaluation (review) of a research article by an independent reviewer who is a peer.
6 Retrieved March 17, 2025, from https://clarivate.com/products/scientific-and-academic-research/research-discovery-and-workflow-solutions/webofscience-platform/web-of-science-core-collection/book-citation-index/
7 Retrieved March 17, 2025, from https://www.elsevier.com/products/scopus/content
8 Retrieved March 17, 2025, from https://www.springernature.com/gp/products/books
9 Retrieved July 26, 2021, from https://clarivate.libguides.com/webofscienceplatform/coverage.
10 In this respect, the BLR differs from the research styles found in ethnography or grounded theory, where the phenomenon is in the foreground and the research question can be changed and adapted during the research process.

10

ARTIFICIAL INTELLIGENCE AND LARGE LANGUAGE MODEL-POWERED CHATBOTS TO SUPPORT QUALITATIVE CONTENT ANALYSIS

In this chapter we provide an overview of how large language models and large language model-powered chatbots can be used for qualitative content analysis. We explain what large language models are and how they relate to artificial intelligence and chatbots. And to help you really understand how they can be used to analyze qualitative data, we present a literature review (see Chapter 9) that gives you an overview of the February 2025 state of research on the use of large language model-powered chatbots. Then, we will explain how large language models work and provide practical examples of how they can be integrated into your research. Finally, we will provide guidance on the decisions you need to make about the use of large language models.

10.1 Why we are cautious about using AI for qualitative content analysis

When we finished writing this chapter in February 2025, ChatGPT had been available to everyone for about two years. All of you most likely used ChatGPT (or another chatbot) at least once. And you are probably amazed at the speed that ChatGPT can answer questions, summarize articles, draft outlines, and write texts. In connection with ChatGPT, terms such as *artificial intelligence* (AI), *chatbot*, and *large language models* (LLMs) have since become buzzwords. Accordingly, it is helpful to understand just what these three terms mean and how they relate to each other. First, *AI* refers to systems that can mimic human thought and action, for example, by processing and analyzing data. *LLMs* are a specific type of AI that is trained to understand and generate human language. They use huge amounts of data to analyze texts and then produce (seemingly) relevant answers or content. LLMs are based on artificial neural networks and most notably on an artificial neural network called transformer architecture. These models are trained with large amounts of text data to statistically

DOI: 10.4324/9781003496397-12
This chapter has been made available under a CC-BY-NC-ND 4.0 license.

calculate which sequences of words, sentences, and paragraphs typically fit together (Hicks et al., 2024).

However, LLMs do not 'understand' content in the true sense of the word. Their strength comes from their ability to recognize patterns from training data and apply these patterns to new contexts. As a result, LLMs can quickly generate text and perform simple context-dependent tasks such as answering questions and summarizing and translating texts. But they reach their limits when tasks require logical reasoning, causality, or complex interpretations. LLMs can give answers that sound plausible but are not necessarily correct in terms of content or logic – a phenomenon often referred to as 'hallucination'.

Chatbots are practical applications of LLMs that communicate with people in natural language, for example, in customer service or educational environments. The connection is therefore that chatbots are based on the technology of LLMs and LLMs rely on AI. As a result, it is possible for ChatGPT (and other generative LLMs) to have a human-like dialog with us. So when you ask ChatGPT a question or ask it to produce a text, it analyzes the request in real time and generates a response based on the language patterns it has learned. ChatGPT (as well as other LLMs) uses several sources as training data including books, online forums, social media, news articles, and in the ChatGPT 4-version the Internet. It therefore processes massive amounts of data to recognize patterns.

ChatGPT, developed by the company OpenAI, is an example of a chatbot based on a GPT language model. GPT stands for 'Generative Pre-trained Transformer'. Other known chatbots based on LLMs include Claude by Antrophic, Copilot by Microsoft, Gemini by Google, Le Chat by Mistral, Grok by xAI, and DeepSeek by DeepSeek. Along with these large models, there are also local LLMs, but these do not yet have the quality of the large models (for more information on LLMs, see Hicks et al., 2024).

Box 10.1 Connection of AI, LLM, and chatbot

Artificial Intelligence (AI): AI focuses on building systems that can perform tasks requiring human-like intelligence, such as understanding language.

Large Language Models (LLMs): These are AI models trained on vast amounts of text to process and generate human-like language. They are a key tool in language-related AI tasks.

Chatbots like ChatGPT: They use LLMs to simulate conversations. ChatGPT, for example, understands text inputs, generates responses, and helps with questions or tasks using its underlying AI model.

In the following, we will use the term 'chatbots' exclusively when referring to specific chatbots, such as ChatGPT, and when discussing the use of their interfaces. If we are referring to the underlying models that power these chatbots, we will use the term 'LLMs' (large language models).

This very brief description of what an LLM is leads us directly to the question of what limitations LLMs have for qualitative content analysis. There are five aspects that lead us to conclude that LLMs should be used with caution for supporting qualitative content analysis:

1. **Bias**: In the case of commercial LLMs (such as ChatGPT), we do not know or only have a rough idea about which data was used to train the LLMs. An exception are open source LLMs such as the German LLM Blablador, where we do know which training data was used. But if you do not know which training data is used, then you cannot assess which training data-based distortions, miscalculations, or even gender, racial, or other biases are informing the outputs of an LLM. For example, if the training data contains sexist or racist language, the model may produce text that reinforces these biases. Conversely, the developers of LLMs censor content (see aspect 2) to eliminate bias in the form of racism, sexism, or adultism from their programs. However, if you are conducting research on precisely these topics and the data you are analyzing contains this type of content, the LLMs will not be accurate. In addition, most of the training data originates from the so-called Global North (i.e., Europe, North America, etc.), which means that a worldview with specific norms and values is reproduced. If, for example, you need to analyze data from the Global South, you have to check for biases in your analysis.
2. **Filtering and censorship**: The developers have built filters into all of the LLMs that can currently be used as chatbots. This means that they simply do not answer questions on certain topics. For example, you have to trick DeepSeek to get answers about the Tiananmen Square protests in 1989 or critical views on the Chinese Communist Party. None of the LLMs (except maybe Grok by xAI) will directly answer questions such as how to build a bomb or questions about topics such as drug use, hate propagation, sexual practices, or fantasies of violence. As a result of these filters and censorship, you should not analyze (very) sensitive topics with the LLMs because qualitative data analysis would generate heavily distorted results (see bias).
3. **Limited contextual understanding**: LLMs work on the basis of manifest pattern recognition, which means that they are limited in their ability to consider the social, spatial, and temporal context as well as the underlying, latent meaning of the text being processed. Since context is crucial in qualitative research, this means that the analysis is likely to be deficient.
4. **Hallucinations (or bullshit)**: LLMs are based on artificial neural networks and, therefore, do not possess human judgment. Nevertheless, LLMs are programmed in such a way that they must give answers, regardless of whether they may be correct or not. This means that LLMs may end up inventing things like references to publications, court rulings, and events that never happened. Hicks et al. (2024, p. 39) even go so far as to call the answers ChatGPT and other LLMs give as bullshit: "Because these programs cannot themselves be concerned with truth, and because they are

designed to produce text that looks truth-apt without any actual concern for truth, it seems appropriate to call their outputs bullshit." The hallucinations are a major problem for qualitative content analysis, especially when LLMs propose categories or identify patterns that are not derived from the data analyzed but are grounded in the training of the LLM. How hallucinations can be reduced is explained in Section 10.2.

5. **Ethics**: Please do never ever enter sensitive, personal data or copyright protected data in the Chatbot interface. Data entered into ChatGPT and other chatbots is secured by the owner companies and can be used by companies as further training data: You trade information for the free use of a chatbot. Even if you pay for a data secure chatbot package, when processing your request on the company's server in real time, your data is still collected on an aggregated level by the company (please see the general terms and conditions). For future chatbot responses this means that information you provided will be processed and then contributes to answers provided to other users. You should therefore never enter sensitive data – such as patent specifications or interviews with criminals or people in the public eye – into ChatGPT or other Chatbots. For example, this includes entering anonymized or pseudonymized interview transcripts or other sensitive documents (see Section 7.2.2). If you upload interview data to a chatbot, there is a possibility that individuals could be identified, as LLMs have been trained on social media data. Likewise, for legal reasons, data that is protected by copyright or licensed should not be entered into the chatbot interface either. For example, according to the European General Data Protection Regulation 2016/679,[1] it is against the law to use LLM-support for qualitative data analysis (e.g., of interviews) unless the respondents have given their explicit consent to the use of LLMs (see Section 3.4).

10.1.1 Some initial positive and negative insights into using LLMs for qualitative content analysis

LLMs are being tested in qualitative data analysis software (QDA programs). For example, the QDA programs Atlas.ti, MAXQDA, and NVIVO have integrated access to the LLMs from OpenAI (= GPT model that is also used in ChatGPT) or CoPilot via Amazon Web Services (AWS). According to the company's marketing, the aim of AI assistance programs is to make research processes simpler, faster, and therefore more cost-effective. This follows market logic (keyword efficiency) and not research logic.

Qualitative researchers who have conducted studies using ChatGPT and other chatbots have reported time savings when analyzing large amounts of data (Barany et al., 2024; Gao et al., 2023; Hamilton et al., 2023; Lixandru, 2024; Perkins & Roe, 2024; Turobov et al., 2024). The focus of the studies to date (as of February 2025) has been on thematic analysis, which is a method used to identify, analyze, and interpret patterns following themes in qualitative data (Braun & Clarke, 2006). In addition to thematic analysis, we found studies

using LLMs integrated in the methodology of grounded theory (Sinha et al., 2024) and documentary method (Schäffer & Lieder, 2023), as well as articles on the AI-assisted collaborative qualitative analysis (Gao et al., 2023), and the development of a visualization tool to make analyses involving LLMs transparent (Gauthier & Wallace, 2022).[2]

Common to these studies is the question of whether and how LLMs can be used for the analysis of qualitative data. Most studies conclude that LLMs can be used as AI assistants: "The comparative analysis in the current study suggests that AI should be seen as a complement to human analysis, rather than a replacement" (Hamilton et al., 2023). However, other researchers reject the use of LLMs in qualitative research: "In our view, LLM tools are still very far from processes of logical-deductive reasoning, creation, understanding subjective aspects, and emotions, which differentiate man from machines" (Carius & Teixeira, 2024).

Using AI assistants can help researchers to amend perspectives, patterns, or contexts of a phenomenon that may not be identified by a human researcher working on their own (De Paoli, 2023; Gauthier & Wallace, 2022; Goyanes et al., 2024; Hamilton et al., 2023; Perkins & Roe, 2024; Turobov et al., 2024; Wachinger et al., 2024; Zhang et al., 2023). Gao et al. (2023) refer to LLMs in this role as mediators. Based on these positive views, it seems that researchers using LLMs for their analyses may be able to understand the data more quickly. Just like when researchers discuss their ideas with colleagues, researchers could use LLMs to generate more ideas for data analysis and answer questions grounded in the data that would not have been asked without LLMs. Researchers can also use LLMs to help group codes into clusters (Lixandru, 2024; Nguyen-Trung, 2024). It is even assumed that LLMs could help to reduce human bias in the interpretation of data, which may be present because of the researcher's theoretical lens or sociocultural context (Jalali & Akhavan, 2024). Perkins and Roe (2024) see easy and free access to tools such as ChatGPT as a means to democratize the research process. However, the more advanced LLMs are not available free of charge (anymore), which seems to make this argument obsolete. In addition, the research process could only be democratized if all users are (a) skilled in qualitative content analysis and (b) possess competencies in dealing with the LLMs, which, at large, is not the case yet.

Alongside the opportunities offered by LLM use, most cited studies also highlight the uncertainties and problems associated with the use of LLMs in the analysis of qualitative data. First and foremost, studies show that LLMs cannot take over or even replace human interpretations and analyses as the analyses conducted by LLMs are generally superficial. Furthermore, LLMs cannot detect latent content in text, do not take sufficient account of the context, and are not very diverse, that is, since their output is based on probability, they do not 'think outside of the box' (Barany et al., 2024; Gao et al., 2023; Goyanes et al., 2024; Hitch, 2023; Turobov et al., 2024). These findings mean that researchers using LLM support should always keep a critical distance to the output produced by LLMs and that they should make data grounded assessments as to which LLM

produced results they consider plausible for further analysis and interpretation (Mesec, 2023). For example, Gao et al. (2023) point on the risk that the rapid production of results may lead researchers to simply accept and not sufficiently scrutinize the analyses and results. Finally, researchers must always be aware that LLMs are generative AI and as such never generate the exact same results, order of categories, topics, or reproduce other previously requested output.

In addition to LLMs producing challenging outputs, the input can be a source of error, beginning with a human sending a request to an LLM. Accordingly, researchers working with LLMs should first learn to design meaningful prompts, a skill set known as prompt engineering (Chubb, 2023). Prompting refers to the input that you enter as a question or request in ChatGPT or another chatbot. Current studies show that when analyzing qualitative data, the results generated will differ based on the type and number of consecutive prompts (Kristensen-McLachlan et al., 2023; Mesec, 2023; Nguyen-Trung, 2024; Xiao et al., 2023). Since very specific content and context knowledge informs researcher's well-thought-out prompting, Goyanes et al. (2024) raise the question whether ChatGPT or another chatbot is actually helpful or whether it is not straightforward for researchers to carry out the qualitative content analysis themselves. Such a question becomes even more pertinent given the fact that ChatGPT, for example, does not always follow the instructions given in the prompt (Nguyen-Trung, 2024).

Although more an issue in quantitative than in qualitative data analysis (see Section 4.2), some researchers consider it as problematic that it is not possible to replicate or reproduce LLM generated results (Jalali & Akhavan, 2024; Kristensen-McLachlan et al., 2023). On the one hand, reproduction of results fail because LLMs are constantly evolving, and developers rarely openly communicate which changes have been made for an updated version of an LLM. On the other hand, complying with the appropriateness-transparency quality convention (see Section 4.3.2), researchers should always document their use of LLMs (e.g., in the appendix of their publication), for example how LLM support was used at which stage of the research process and which decisions they took to carry on with data processing, analysis, and interpretation (see also Gauthier & Wallace, 2022). The documentation-need applies, in particular, to the prompts used by researchers (De Paoli, 2023; Turobov et al., 2024). If you are using LLMs, it can be helpful to incorporate methods of visualization such as process mining and idea threads as these are intended to make the research process and the decisions made therein transparent (e.g., see Figures 14.1 and 15.2; see also Lopez-Fierro & Nguyen, 2024).

10.1.2 Our assessment of using LLMs for qualitative content analysis

The studies published until February 2025 have all been conducted by researchers who have experience working with qualitative methods and were, therefore, able to assess the quality of the categories or category systems, codings,

summaries, etc. produced by LLMs. In conclusion, we feel obliged to recommend to you (a) to do your first qualitative content analysis without LLM support, and (b) cannot fully advocate for using LLMs for qualitative content analysis in an introductory methods textbook. LLMs do not take quality conventions into account, and researchers are responsible for securing and documenting the quality of their analyses (see Chapter 4).

All of the cited studies emphasize that humans have to verify the analyses and results generated by LLMs as the LLMs can produce categories or do coding with bias, often hallucinate, and in some cases may not even be working with the data inputted. Furthermore, the relevant ethical issues for LLM use in (qualitative) data analysis have not yet been sufficiently clarified. In particular, data protection and de-anonymization are major problems, which is why the studies published have always worked with openly accessible material (e.g., interviews) that did not contain any sensitive content (e.g., on sexism, racism, crimes, or health). The studies emphasize that researchers should meet the following criteria:

1. Be very familiar with the data.
2. Have a good understanding of the social (including cultural, economic, political, etc.), spatial, and temporal context(s) of the data.
3. Have sound methodological skills in order to be able to understand the output produced by LLMs.

The three criteria ensure that (future) researchers are prepared for the challenges of assessing the content and implications of LLM generated outputs. This means checking all LLM outputs, especially results, carefully, and not to believe them easily without considering the context or to give in to the temptation to accept the results quickly to be done with the content analysis. In other words, you face the danger of conducting superficial research with LLMs – and the temptation to take shortcuts in the research process is great, especially when the deadline for submitting your paper is approaching. Nevertheless, you can use LLM support to do methodologically sound in-depth qualitative content analyses that include the context of the data as we will outline in the following section.

10.1.3 A note for beginners: first, learn how to do qualitative content analysis and then second how to use LLMs to support your qualitative content analysis

We decided not to include a chapter in this book on how to use ChatGPT or LLMs included in the commercial QDA programs such as Atlas.ti, MAXQDA, or Nvivo, but rather to give you an understanding of the basic capabilities of LLMs. LLMs and chatbots belong to a rapidly evolving field and any manual would quickly become outdated. What will not become outdated, however, is the basic epistemological understanding (i.e., the basis on which scientific discoveries are made) of LLM support in qualitative content analysis. Accordingly,

we would like to encourage you to do your research without the help of LLMs and to learn the basic methods of qualitative content analysis and only then turn to the help of LLMs. Bluntly, without knowing what you are doing, you also do not know what you need help with or what help might mean. Or as Hitch (2023, p. 10) put it:

> "Qualitative research is often challenging, and novice researchers could potentially see AI as a 'short cut' that reduces the effort required to engage with reflexive [content] analysis. However, as clearly demonstrated in the worked example, AI can augment but not adequately replace human researchers".

This is the reason why we have placed this chapter on AI, chatbots, and LLMs at the end of Part 2 of this book. This signals that you should first master the basics of inductive or deductive qualitative content analysis, understand quality conventions and the importance of interpretation in the research process. Then, second, LLM support may become beneficial for your qualitative content analysis. Knowing that available tools will be used, in the following section we would like to present a way of dealing with LLMs that should help you to use them as support in the research process without taking methodological shortcuts. To do this, it is necessary to clarify the functions of LLMs.

10.2 The function of LLMs and generative artificial intelligence and what this means for qualitative content analysis

We will start by describing what LLMs powered chatbots such as ChatGPT cannot do so that you will understand that using LLMs in a naive manner is not at all helpful for your qualitative research. For qualitative content analysis, an LLM can be a support for you, but an LLM simply cannot replace you in an entire research process and 'spit out' a finished qualitative content analysis at the end. In other words, telling a chatbot something like "Please analyze the following data material using inductive qualitative content analysis according to Schneijderberg et al." will not deliver any meaningful results – although ChatGPT could access this textbook because it is published open access. First, it is unclear whether the chatbot (depending on which one you would use) can access the Internet and would make use of (Chapter 7 in) our book. In addition, our textbook contains introductions to various methods of qualitative and quantitative content analysis, therefore ChatGPT will very likely link multiple qualitative content analysis methods or be mixing qualitative and quantitative content analysis. This is because LLMs calculate probabilities for what a meaningful answer could be based on the training data (which, as described previously, is not clear). The chatbot generated outputs will make sense according to the LLM's quantitative probability calculations but, most likely, not for your qualitative analysis.

Second, there is no guarantee that the chatbot output is fact-based (e.g., grounded in the empirical material) and correct (Kim & Thorne, 2024). The

outputs produced by LLMs can be distorted (i.e., have a bias), depending on which data LLMs are trained on and how models were censored (Shetty et al., 2024). Some LLMs (e.g., the GPT-model of OpenAI) have been trained by human coders who mark polarizing, radical, distorting, and discriminatory content as undesirable. Others, for example, xAI's LLM Grok draws on data from the social media platform X. This is problematic because, like TikTok, X uses algorithms that prioritize and, therefore, promote content featuring emotional, polarizing, and narrative content, and often push fact-based discussions into the background. Freedom of expression is a fundamental right, but it also has limits in the case that it causes psychological and emotional harm to people. Moreover, the intentionally almost uncensored LLM powering the chatbot Grok has been largely trained on X social media posts without human intervention, which can further affect its output. These examples highlight why it is most likely that the output will differ depending on which LLMs you use.

10.2.1 Prompt engineering: precision, length, creativity, temperature

As described in the review of the literature in Section 10.1, what you can do is to use LLMs for supporting specific tasks in the process of qualitative content analysis. For example, you can use LLMs to recognize patterns in the data – pattern names could then be used as categories. Or you can ask LLMs to summarize text data and then use these summaries to help you understand the data and familiarize yourself with the material.

The process of crafting prompts to get the right output (e.g., a summary) is called prompt engineering. Prompt engineering has now become a market sector in its own. If you search on YouTube (or the Internet), you will find many helpful tutorials on prompting. As LLMs are evolving so quickly, prompting will also evolve and change. The following instructions are therefore a temporal snapshot for prompting in the beginning of the year 2025, but they will show you why you should not just go ahead, and prompt based on your gut instinct when doing a scientific analysis.

Prompting in support of qualitative content analysis should be systematic and depend on several parameters. Before you start with prompting you have to define several parameters when using LLMs. The first parameter is the maximum length of text *max length*, which allows you to control how long the chatbot response output should be. This is specified using what are called *tokens*. A token is the basic unit that is used by the LLMs to divide the text into units for the statistical calculation of probabilities. Tokens also indicate how long a text is, which is important, for example, to be able to calculate the price if you are using OpenAI's API (application programming interface) interface for GPT models (see box 10.2). Tokens can correspond to the number of words but do not have to. If you want to calculate how many tokens a text has, you can use OpenAI's tokenizer.[3] OpenAI provides the following guideline with regard to tokenization: "A helpful rule of thumb is that one token generally corresponds to ~4 characters of

text for common English text. This translates to roughly ¾ of a word (so 100 tokens ~= 75 words)." By using the max length parameter, you can ensure that the responses generated are not too long (and often redundant), and you can also keep costs in check – both yours and the (energy) costs of the server processing your and others requests that consume a lot of power.

Box 10.2 API

An API (application programming interface) makes it possible for applications to communicate with each other. The API is not the database or even the server; it is the code that regulates the entry points for the server and enables communication. This speeds up and simplifies data exchange between systems many times over.

The second parameter is called *temperature*. You can use this to set the LLM's 'creativity'. If you select a very low temperature toward zero, LLMs will select the output token based on the highest probability. Increasing the temperature will decrease statistical probabilities of output, which generates more random, varied, or creative outputs. For example, you could select a very low temperature for summarizing your text material, so that the summary is as close as possible to the original text (see Section 10.2.3 on RAGs) and a higher temperature for creating categories to allow for more creative distance to or abstraction of the text content.

Box 10.3 Recommendation for LLM settings

Low temperature and high top p: The low temperature results in answers that are usually logical and consistent, but the high top p includes more diverse vocabulary and ideas.

High temperature and low top p: the answers make sense on their own but as a whole seem disconnected or less logical. The high temperature allows for more variation in sentence building, while the low top p limits word choices to the most likely ones.

The third parameter is *top p*, which is also known as nucleus sampling. The top-p value determines how many possible tokens should be taken into account. This means that with a high top-p value, more possible words are included, even if the statistical probability of fit decreases. A high top-p value results in a text output that is more diverse in its choice of words.

10.2.2 A prompt is not a prompt: Zero- and few-shot prompt engineering

In addition to output variations depending on the options for LLM settings and depending on which LLM you use, the training data preconditions differences

in terms of norms, values, ethics, and language mode computed by LLMs (Lee et al., 2024).

> "We can see, for example, that ChatGPT, has the tendency to structure text-based data clearly in terms of content, but still too schematically for in-depth qualitative analysis. Claude tends to make detailed and theoretically sound interpretations. And Gemini often introduces surprising and sometimes unconventional perspectives".
>
> *(Krähnke et al., 2025, own translation)*

These chatbot characteristics can be reinforced or minimized by prompts, which leads us to address the issue of the different ways to write prompts. First, prompts can be roughly divided into two categories.

1. Zero-shot prompts: These are prompts that lead directly to an answer such as "Explain why the sky is blue." Depending on how the LLM has been trained and how you have set the parameters described previously, you will get a well-founded scientific or creative answer.
2. Few-shot prompts: In addition to an instruction (e.g., write a summary) or asking a question, the chatbot can be given information to learn from for solving the task. Such information could be (1) definitions and examples; (2) the assignment of a role such as researcher, student, or analyst; and (3) output indicators such as the length and type of answer (e.g., essay or introduction for a term paper). It is important to note that the performance of LLMs varies depending on where the relevant information is placed in a prompt:

   > "Language model performance is highest when relevant information occurs at the very beginning (primacy bias) or end of its input context (recency bias), and performance significantly degrades when models must access and use information in the middle of their input context".
   >
   > *(Liu et al., 2024, p. 158)*

As you can imagine, a few-shot prompt can become a very large task, depending on what you want from the LLM. It can therefore be useful to split large tasks into several subtasks and then combine the LLM outputs. For example, you could first ask for a part of an interview transcript to be summarized (e.g., response to one question of the interview manual) and repeat this prompt for all remaining parts of the interview, then ask for core or key content (i.e., similar to human interpretations) in each of the summaries. Next have the key content of interview parts compared, and finally, ask for a continuous text explaining the results of the few-shot prompted task sequence.

Just like the length of the prompts, also the input text length can be a challenge for the LLM. With most LLMs, the text sections to be analyzed are limited to a certain length. The context window of an LLM indicates the maximum

number of tokens that a prompt can have. It also indicates the extent to which the LLM can 'remember' the previous conversation. In the past, this was often a major limitation when asking the LLM to give detailed examples or uploading texts for summary and analysis. For example. the version of OpenAI's GPT 4o has a context window of 128,000 tokens. This corresponds to approximately 96,000 words. As a comparison, this chapter has about 11,700 words. Even though you comply with the maximum text input limit of an LLM you should consider splitting the text input, as some LLMs will only use certain parts of it because they cannot deal with texts that are very long.

10.2.3 Addressing hallucinations with RAG (retrieval augmented generation)

Chatbot applications such as ChatGPT and Gemini also allow you to upload PDFs or enter links to online documents to be analyzed. However, tests done using LLMs show that they may not analyze the content of texts provided, but instead LLMs make up answers, that is, they hallucinate. Hallucinations mainly occur when a request in the prompt cannot be fulfilled, but most LLMs are programmed to always give a 'positive' answer. To avoid hallucinations, you can state in the prompt that no output should be given if no suitable answer can be derived from the text(s) provided.

To avoid the problem of chatbot hallucinations, it is possible to use a RAG system. RAG stands for retrieval augmented generation. Retrieval mechanisms ensure that responses are grounded in factual, data external to an LLM rather than solely relying on the probabilistic 'guessing' behavior of LLMs. This means when using a RAG system, you can define that the output generation exclusively accesses documents you provide (e.g., interview transcripts). The LLM solely processes the provided information in documents and generates the output, which is a coherent and contextually relevant text, category or summary based on the specified sources.

This form of prompting (i.e., with RAG) is used in the AI tools incorporated into QDA-software such as Atlas.ti, MAXQDA, Nvivo, and QInsights,[4] as well as the open-source tool QualCoder,[5] which all import the reference documents for analysis. RAG is the only way to link text passages with categories, for example during coding your empirical material. If you want to build a RAG for yourself, however, you will need some knowledge of Python or another programming language (see Chapter 12). Recently, rather small LLMs are being developed that can be used locally on your own computer. This allows you to work in a way that is both ethical and compliant with privacy regulations, for example analyzing interviews containing sensitive and/or personal information.

Without RAG the connection between categories and coded text passages is volatile. Prompted to code, the LLM makes suggestions (usually one or two text passages), but it cannot do any comprehensive coding – which an LLM is simply not designed to do. Moreover, for example, if you use ChatGPT for

coding (without RAG), you need to remember that LLMs work sequentially and therefore do not remember which text passages have already been coded. As a result, you will not get a suitable answer to the question of how often category xy occurs in your document.

10.2.4 Addressing hallucinations with reasoning

The problem of hallucinations is now also being tackled by the further development of LLMs, through what is called 'reasoning'. Put very simply, 'reasoning' means that LLMs can be trained to draw conclusions. In comparison to conventional LLMs generating output texts primarily based on probabilities – predicting which word or type of information statistically fits the context best – those that can 'reason' have capabilities that go beyond this. 'Reasoning' LLMs are optimized to create logical or causal links between pieces of information, and in the process, they mimic the steps of analytical thinking. Instead of just calculating probabilities of words to deal with the task at hand, 'reasoning' LLMs perform internal structural analyses to explicitly derive logical conclusions. This is often done using additional mechanisms or techniques such as the following:

1. Chain-of-thought (CoT): This is a method in which the model is trained to explicitly generate its logical step-by-step explanation or 'reasoning' process before providing the answer. This is similar to the human approach, in which we carry out and document the intermediate steps during problem-solving. CoT 'reasoning' increases the accuracy of tasks such as doing mathematical calculations and logic problem-solving since researchers can see errors in the output documentation of intermediate steps and then intervene, if necessary.
2. Few-shot and zero-shot 'reasoning': 'Reasoning' models are trained in such a way that they can make the correct conclusions for certain tasks even without previous examples (zero-shot) or with only a few examples (few-shot). This mechanism extends the scope of LLMs to tasks that they have not been pretrained for.

The major difference between more pedestrian LLMs and 'reasoning' models can be highlighted for the task of analyzing political debates. While an LLM could extract the most frequent arguments or topics of a discussion, a 'reasoning' model would be able to trace argumentative strands, check their validity, and establish causal relationships between the positions presented by politicians.

Despite the potential of 'reasoning'-based LLMs, there are limitations here as well. Unlike humans, these models do not use true 'logical intelligence'; they merely simulate logical processes based on patterns in their training data. But 'reasoning' models make the 'thinking' of LLMs explicit and can also indicate sources they have worked with to draw their conclusions. However, new

challenges arise here as the 'reasoning' is mainly based on Internet sources that are freely accessible to, for example, avoid copyright infringement. This means that in addition to the bias contained in the training data, the use of Internet sources adds a second bias that has not yet been sufficiently researched (Wieczorek et al., 2025).

10.2.5 Prompt summary

Before you enthusiastically start creating prompts and trying out LLM-powered chatbots, here are three important points that we would like to now reiterate for your consideration:

1. First, for qualitative content analysis, only use empirical material that you are permitted to use according to data protection regulations. These regulations will differ depending on which part of the world you are in. For example, in Europe, you can only use material that does not contain personal information. If the data does contain personal information (e.g., from the person you interviewed), you will need to obtain a consent form that expressly authorizes analysis supported by LLMs (see Section 3.4.1). The data protection laws that should be respected vary greatly between countries and continents, and you should therefore search carefully and find out which apply to you. If you are unsure, either ask your supervisor or contact the data protection officer at your institution. For example, in 2025, it is a legal grey zone and ethically controversial whether social forum posts or tweets that contain personal information, even if they are publicly accessible, can be analyzed by using LLMs.
2. Second, copyright law should also be observed. With many LLMs, it is now possible to upload PDFs and work with these. However, if these are publications that are not freely accessible (i.e., not open access), then this is a violation of copyright law. Legally enforced or not, not respecting copyright law is unethical.
3. This picks up on the third point – it is important to think about ethical behavior when you are using LLMs (see also Section 3.4). What information entered into a context window of an LLM can you ethically justify sharing with the big tech companies? How can you be 100% sure that information about your interviewees will remain anonymized and not be publicly available online? For considering ethical limits, just think about what information about your mother or your own research you do want to share with the big tech companies. Already some might be enough because they have big data to match information. If you are unsure how to proceed correctly in terms of research ethics, contact the ethics officer at your university or research institute. You can also turn to ethics guidelines issued by associations in your discipline that will give you a good overview on what is considered ethical research behavior (see Section 3.4.3).

10.3 Interpretative paradigm and LLMs

In addition to the ethical and technical challenges for using AI for qualitative content analysis, using LLM-powered chatbots touches upon epistemological considerations. Epistemology is the study of how knowledge is created and how it can be justified. As we stated in the introduction to this handbook (Section 1.4), we introduce methods of content analysis according to the interpretive paradigm. The term 'paradigm' was coined by Thomas Kuhn (1962/1976, p. 186) in his book *The Structure of Scientific Revolutions*. For Kuhn, the term paradigm encompasses the theoretical assumptions and accepted methodological approaches to problem-solving shared by a scientific community. In comparison to the paradigmatic need of understanding and interpreting the social, for example, positivists, and researchers from the "thought collectives" (*Denkkollektive*) (Fleck, 1935/2019) of critical rationalism or critical theory have a particularly different view on the study of social reality.

But what do thought collectives, and their epistemology have to do with LLM support of (qualitative) content analysis? A lot, because a thought collective's epistemology guides how you as a scientist pursue research and generate knowledge. Accordingly, the question arises as to whether LLMs can 'understand' or 'know' what they are 'talking' about or even whether they could have or develop a 'consciousness' and can 'rationally explain' social reality. From the previous technical explanations, you have already realized that LLMs do always present 'positive' answers, even hallucinate findings, references, etc.; and LLMs do not convey facts based on their training data, for example, they do not 'know' what the latest findings on a research topic are, but instead LLMs give the most likely answers that are based on statistical probability calculations. Their answers sound quite solid because LLMs have been trained to give answers that sound as if they were generated by a living and breathing human. This is why some humans tend to anthropomorphize chatbots and assume that what their LLM has given as an answer is well thought through (e.g., considering facts).

According to the interpretative paradigm, we assume that scientific knowledge is obtained through the interpretation of data that has been collected and analyzed based on methodological considerations – and that knowledge just does not appear on its own from data or that any finding can be declared a social fact. Before a finding can become knowledge extracted from data, the interpretation should be abstracted with the support of theory (see Sections 4.3.2.3 and 5.3.3, and Chapter 6). Theory guides a deductive research process or should be used in an inductive approach to reflect on empirical findings from a theoretical perspective.

10.3.1 The interpretative paradigm and a reflective approach to using LLMs

The interpretative paradigm aims for understanding and subsequently explaining social phenomena (see also Chapter 5). Central to the interpretative paradigm is

the reconstruction of patterns found in actions and social processes. The interpretative paradigm has its roots in the hermeneutic tradition and *verstehende* (understanding) sociology, as established by Max Weber (1981) in the beginning of the 20th century and was further developed by scholars of phenomenology, symbolic interactionism, ethnomethodology, and hermeneutics. According to the hermeneutic tradition, truth only arises in the interplay between understanding and interpretation (Gadamer, 2010). Hermeneutics originally comes from biblical exegesis and refers to the interpretation of text through abduction,[6] that is, asking questions of the material and developing various possible interpretations to get closer to what message or meaning a text might contain (see also Chapter 5).

In the interpretative paradigm, society and social reality are subjectively constructed and interpreted through cultural knowledge and attribution of meanings by social actors (Schütz, 1962) and not seen as objective social facts independent of human interpretation. Epistemologically, the interpretative paradigm is based on a constructivist approach to knowledge, which assumes that knowledge is always a (re)construction of the social world. Researchers not only investigate external actions but also attempt to understand the subjective motives and meanings that social actors attribute to their actions (Weber, 1981).

For the interpretative paradigm, truth is not regarded as an objective, corresponding relationship between a statement and reality (as in positivism or critical rationalism[7]). Instead, truth is considered to be dependent on the social context – it is shaped by personal experiences, life circumstances, a community, societies, and cultures, which differ depending on the social space and may evolve over time. Accordingly, there is no one truth; rather, different realities can only be understood intersubjectively. In this sense, truth is not absolute, but results from the degree of understanding that a researcher achieves about the subjective perception of social reality by observation or interviews. Accordingly, the interpretative paradigm also assumes that knowledge is never objective and neutral knowledge. Constructed knowledge varies between people, depending on the social, spatial, and temporal context. This also means that people do continually interpret and understand the situations, interactions, and relationships to be able to act and react, for example considering existing norms and different role models.

In the interpretative paradigm, researchers see themselves as part of social reality, which is why it is not assumed that an objective analysis of the data can take place or that there is one correct interpretation of the data. Instead, it is essential to understand that you as a researcher come with your own contexts, biases, and assumptions, and accordingly, researchers have to reflect on these during data analysis. In concrete terms, this means questioning why an interpretation was made by you and whether there could be other or additional interpretations of data. Such reflexivity about your own position as well as the interpretation of data could be supported by theory.

10.3.2 Epistemology of the interpretative paradigm and a reflective approach to using LLMs

Based on the brief introduction to the interpretative paradigm, we can now address the epistemology of LLM support in qualitative content analysis. As described previously, LLMs construct or generate answers based on the patterns in their training data, may hallucinate (i.e., produce plausible-sounding positive but factually incorrect answers), and depending on the LLM model have various biases and filters that censor information processing. LLMs generate a type of inferred 'knowledge' that is based on statistical probabilities and is dependent on human input via prompt engineering, including decision of using a RAG or not and on parameter settings of maximum length, temperature and top p (see Section 10.2). Accordingly, LLM 'knowledge' processing and generation are different from human knowledge and understanding.

Nevertheless, the LLM outputs can be understood as representations of a specifically constructed, training-data and filter-dependent reality. Similar to researchers interpreting various human subjects constructed realities and bodies of knowledge, researchers following the interpretative paradigm can analyze LLM generated or (re)constructed patterns of the social world.[8] Since LLMs were trained on data from the social world, for example data provided by people on the Internet, it can be assumed that the outputs reflect the construction of the social world based on a big variety of subjective understandings of social reality, situation, actions, etc. Accordingly, LLM outputs can be interpreted according to the interpretative paradigm, and it is the task of the researcher, you, to reflexively address the variations of constructed knowledge and do your best to understand and interpret LLM outputs considering the social (including cultural, economic, political, etc.), spatial, and temporal contexts. Of course, being reflexive starts with the documentation of prompt engineering, contextualization of text input, etc. in your research journal.

However, interpreting LLM outputs you should always be critically aware and reflexive about most, and definitely, commercial LLMs being a black box – we do not know exactly which training data was used to train them or what type of outputs were not permitted (censorship). For example, your interpretation should reflect that most of the training data originates from the Global North, which, of course, contains the corresponding ethical, social, and cultural norms.

10.3.3 LLMs as a sparring partner for interpretation in qualitative content analysis

In the previous section, we established that LLM output can generally be analyzed within the framework of the interpretative paradigm. For you doing qualitative content analysis, this means that LLMs are an interpretative assistant, that is, a sparring partner for your own interpretation. Like with interpretations offered,

for example, by colleagues in an interpretation group or fellow students in a seminar, then you need to analyze LLM outputs and not adopt them one-to-one. Assisting your interpretation of data, LLM outputs can give you a broader view of your own material. But it is then up to you to assess and reconstruct the LLM output and decide whether it is plausible, relevant, and worth reflecting on, or useless or even discriminatory (e.g., racist). So then you may use or discard LLM output as is appropriate. Complying with the appropriateness-transparency convention (see Section 4.3.2), your use of LLM outputs requires documentation of and justification for the decisions you made in dealing with LLM outputs.

So far, we have always written about LLM output and have avoided answering the question: can LLMs interpret at all? Interpretation involves analyzing data in a structured, methodologically guided way to gain deeper insight and attribute meaning to data (see Chapter 5). Without interpretation, you are merely summarizing your data and repeating what was said in a text in another way. And this is precisely why you should never comprehend LLM output as an interpretation. LLMs cannot interpret; instead, they recognize patterns in (a big amount of) texts based on statistical probabilities, are able to model highly complex statistical relationships between words and phrases, and generate answers based on this. LLMs can summarize texts, identify similarities, extrapolate topics, and highlight stylistic or semantic features in text or a text corpus.

Despite these achievements, an LLM does not 'understand' texts in the human sense, let alone in the sense of the interpretative paradigm as it lacks subjectivity, its own horizon of experience, and reflexivity. An LLM can neither take into account contextual nor cultural ambiguity expressed in data; it simply lacks the ability to socially construct meaning. Although an LLM can analyze and classify words in a text according to their statistical correlation, the process of attributing meaning remains a profoundly human activity that is embedded in individual and collective experience. Interpretations recognize the ambiguity of a text and its intentional embedding in various discourse contexts, read between the lines and reconstruct latent structures of meaning, and reflect the positioning of the researcher and the sociocultural circumstances in order to, for example, uncover hidden power structures.

In summary, we can say that despite their limitations, LLMs can be used for content analysis according to the interpretative paradigm as tools that support researchers in the process of interpretation. LLMs can generate statistically probable hypotheses, offer alternative readings, and analyze texts for patterns. However, the actual work of attributing meaning and questioning LLM output remains in the hands of human interpreters, whose subjectivity, experience, and critical reflection are indispensable for the sociocultural construction of meaning.

10.4 QDA programs and LLMs

As discussed at the beginning of this chapter, the main QDA programs – Atlas.ti, MAXQDA, and Nvivo – have APIs (application programming interface) to large LLMs (e.g., on Amazon Web Services and by Open AI) and offer AI assistants

that can help with the analysis of qualitative data. There are also a number of newly developed programs, including QInsights, QualCoder, and DokuMet. What all of the named QDA programs have in common is that they are designed to analyze the data based only on the user's own empirical material (e.g., files and interviews). Doing content analysis with QDA program integrated LLMs makes a big difference to using LLM-powered chatbots, for example ChatGPT. So if you use an LLM integrated into a QDA program, you will have fewer hallucinations in the outputs, and you will be provided with references to documents so that you can look at the respective text passages. Before using these AI assistants of QDA programs, you should check which LLMs the QDA program and respective LLM access provider so that you understand the limitations (e.g., filters) and possible bias of the outputs. If you are writing a term paper or a bachelor, master or PhD thesis, it is best to get an agreement about the use of QDA program AI assistants in advance from your lecturer or supervisor(s). Prior to using QDA program AI assistants, it is important to also clarify whether you are allowed to use an LLM to analyze your data at all. Depending on the data protection guidelines of the country you are living in, you may need to obtain informed consent (e.g., the explicit consent of the interviewees). There might also be ethical or copyright issues that need to be addressed in advance, for example, if you want to analyze newspaper articles.

10.4.1 Inductive text analysis using AI assistants

From our point of view, since the first generative LLMs appeared on the market, three possible uses for qualitative research have developed that can be applied to qualitative content analysis. QDA program AI assistants can be used in inductive content analysis, for support of category development, the coding process, writing summaries, and in support of your interpretation. In communication with a QDA program integrated AI assistant you can formulate questions and get outputs that relate only to the documents that you have uploaded to the QDA programs. You can talk to the AI assistant about your material and can analyze the chats you have with the AI assistant. This makes sense, for example, if you want to identify patterns in your data on a certain topic or simply check whether a topic is (sufficiently) relevant in your data to be considered for further analysis (e.g., development of a category).

However, this approach does not relieve you of the task of being familiar with the content of your documents, a precondition for working closely with your data. You are still dealing with LLMs that cannot interpret or analytically deduce by considering social, spatial and temporal contexts what is in your data. And latent topics and correlations are hardly recognized – if at all – by LLMs, which means that QDA program AI assistant's output will remain at the level of manifest content. Nevertheless, the chats between you and the responding AI assistant can, for example, still help you to get initial ideas as to what categories you could develop inductively, or they can give you an overview by producing summaries and coding your data based on your developed categories. The

advantage of using the AI assistance integrated into QDA programs is that the outputs contain references to relevant sections of the analyzed documents so that you can check whether the text passages, actually, correspond to the questions you asked and to your underlying assumptions.

You can also ask AI assistants like any other LLMs to summarize or paraphrase texts. As we described in Chapter 7 on inductive qualitative content analysis, it makes sense to summarize the central topics found in your documents to obtain an overview of the data. The AI assistant can help you with this, but you will need to have worked closely with the data beforehand to be able to check the plausibility of LLM output, first. Second, without knowing your data well, you will not be able to assess the quality of the summary produced by the AI assistant. Third, considering LLM generated summaries, it is important to remember that the AI assistants' output (i.e., the summary) will only provide an initial overview of possible patterns in your empirical material. Analyzing the output, you can then derive actual categories for pattern tracing and pattern interpretation from this in the next step.

When using AI assistants, there is also a technical limitation as the LLMs used can only process a certain number of tokens (see Section 10.2). MAXQDA (version 24) states, for example,

> "The chat supports documents up to 100,000 characters long, approximately 25 [A4] pages, depending on formatting. However, character limits may be lower for languages such as Japanese due to shorter text lengths. MAXQDA will notify you if your selected document exceeds the chat's capacity".[9]

In addition to the chat function, the QDA software included AI assistants can also explain words or technical terms that you may not know or translate text data, for example MAXQDA has a built-in automatic translation function. However, you should also check the automatic translations since language, especially in interviews, not only conveys manifest content, but also contains context specific and latent meanings that could be lost if the translation is too superficial.

General rule of thumb 1 for QDA program integrated AI assistant use for qualitative content analysis: The better you know the content of your empirical material, the more specific questions you can ask the AI assistant about the content of the material. It is then more likely that you will gain new insights into your data and be able to further develop your own research. With or without AI assistance, only when you know exactly what your data is about and understand its contextual embedding that you will not get lost in the empirical material. And this is important because the chat protocols and summaries will provide you with additional material that you will have to assess, analyze, and interpret. Chatting may give you some insight into your data as you get a look at it from a different perspective, but AI assistance does not make the interpretation process any easier, faster, or more efficient.

10.4.2 How LLMs can provide support for the coding process

In addition to supporting inductive text analysis, the QDA programs also offer the option of developing categories and subcategories and performing automatic coding. To recall: A category is a unit to identify, classify, and sort actions, actors, artifacts, phenomena, etc. in qualitative content analysis. A category should be distinct, have a clear meaning, and not overlap with other categories. To code means to attribute a category to a segment of text, for example a sentence, several sentences, or a paragraph. Coding is the activity of creating order in document(s) by selecting units of meaning (i.e., coherent content in a sequence) to answer the research question. For example, in MAXQDA, the LLM-powered AI assistant suggests various categories that could be adopted based on pattern recognition. A distinction is made in MAXQDA between specific categories that cover smaller aspects of the material, categories that refer to more general aspects of the material, and 'interpretative' categories.

Once you have created the main categories of a category system, the QDA programs can use these main categories and generate suggestions for subcategories, that is, turning main categories into top categories with associated subcategories. When asking the AI assistant for category suggestions, it is important to keep in mind that LLMs are not neutral but are instead 'normatively shaped' through their respective training data, which can lead to biases and training data induced overemphasis or ignorance of certain topics. Especially when you are doing research on sensitive topics such as violence, drugs, or hate, as a result of censorship LLMs may not generate related categories.

Another AI assistant feature is automatic coding, which is based on word recognition. This feature already was part of QDA programs before the integration of LLMs, but now semantic correlations amor words are recognized as well. Automatic coding can therefore be used once you have inductively developed a category system or in deductive content analysis – but only for the first 'quick' run-through your empirical material. This is because LLMs cannot recognize latent structures of meaning and will therefore not recognize text passages that need to be assigned to a category interpretatively (see Section 5.3.1). For automatic coding, in addition to the category system, you need coding rules and unit of meaning examples from your data so that the LLM can learn which text passages might fit into a top category or subcategory. For example, as stated on the MAXQDA website, their AI assistant can be helpful in specific, complementary ways for your coding process:

> "Whether coding a document from scratch or reviewing a fully coded project, AI Assist can act as a second set of eyes, helping you cross-check your work, identify any segments you might have missed, and even highlight differences in interpretation".[10]

In most QDA programs, the automatic coding works only for one category and one document at a time. Kai Dröge (2023) recommends a different approach

using the AI assistant in his open-source tool QualCoder AI: Horizontal coding. Horizontal coding describes an iterative process in which qualitative data is analyzed along certain predefined or emergent topics. The aim is to identify patterns, relationships, and commonalities of a topic in the data (e.g., interviews or documents). Instead of analyzing the data vertically – on a case-by-case basis (e.g., separately for each interview) – the data AI assistance horizontally compares the topics or categories. According to Dröge (2023), horizontal coding is particularly helpful for discovering initial categories or topics, which can then be the starting point for researchers to carry out a detailed analysis and interpretation for category development. The researcher ultimately remains responsible for critically interpreting the data, contextualizing the results, and not mechanically following the AI assistant's suggestions.

General rule of thumb 2 for QDA program integrated AI assistant use for qualitative content analysis: AI assistant support with coding or inductively developing categories can be helpful for gaining an initial understanding of your data – for inductive as well as horizontal approaches to qualitative content analysis. However, you should always consider that LLMs cannot interpret and do not recognize latent patterns and could miscalculate probabilities of manifest meanings. As a result, there is the risk that the actual attributions of meaning made in documents by the participants during an interview will not be adequately captured by an AI assistant. As the researcher, you are therefore always responsible for interpretation, creating categories, and appropriate coding (see Section 4.3.2.2).

10.4.3 LLMs as a source for interpretation support

As we have shown, LLMs are not able to do interpretation, as their output remains at the manifest level of text meaning. Taking the term 'machine learning' seriously, to rise beyond the manifest text analysis level, LLMs have to be taught what human interpretation is and how it is structured. This is in much the same way that you have to learn the difference between a descriptive representation of data (e.g., in a summary) and an interpretation of data.

For LLM learning, more sophisticated prompting is required (Krähnke et al., 2025; Lieder & Schäffer, 2024; Schäffer & Lieder, 2023). In their articles, the authors describe how the prompts were formulated and give examples. For the QDA software DokuMetAI Schäffer and Lieder created, they fine-tuned an LLM with many examples of human interpretation according to the research style of the documentary methodology. This allowed the LLM to learn how human interpretations are structured, in particular, what the style and patterns of these type of interpretations were. The authors conclude that researchers can use AI outputs to help develop their own interpretations of data and to get input that might make them reconsider their own interpretations. Accordingly, the benefit is not to have the interpretation automated, but to have a kind of sparring partner that supports you to rethink and possibly improve your own interpretations. This can be useful if you work on your own and have to tackle interpreting the

data on your own because you are not working in a group that you could discuss your interpretations with.

Depending on which LLM you use, you will get different feedback on your interpretations (depending on the LLMs' training approach and settings). Krähnke et al. (2025) propose this to be a hybrid 'interpretation' approach. They used the three chatbots ChatGPT, Gemini, and Claude (with their underlying LLM) in support of iterative interpretation loops. The authors found that ChatGPT provides structured but often superficial and simplified outputs, Claude offers theoretically sound, detailed outputs, and Gemini often provides creative, unconventional perspectives on their interpretations. Krähnke et al. (2025) do not consider the variance in the outputs as a disadvantage due to incoherence but recommend to deliberately use LLM multi-perspectivity and heuristic variance to support your interpretation process.

Krähnke et al. (2025) define three phases for a hybrid 'interpretation' process: initiative 'interpretation', elaborative 'interpretation', and synthesizing 'interpretation'. In initiative 'interpretation', a start prompt is used that contains the chatbot role assignment, work instructions, and the text material to be examined (unfortunately, the prompt presented in the appendix of the article is only available in German but can be easily translated with AI). The text material is supposed to be limited to a text sequence that is intended to be 'interpreted,' the entire data material would be too unspecific for the chatbot. In hybrid 'interpretation,' the sequences are interpreted one after the other. The start prompt is entered into the first chatbot and the output with the first proposed 'interpretation' is analyzed by researchers, who then do formulate the (intermediary) interpretation that they can continue to work with. In the second phase, the elaborative 'interpretation', the (intermediary) interpretation result is to be processed by the second chatbot, and then the output 'interpretation' is presented to the third chatbot, of course, always based on a suitable prompt. The researchers then analyze the chatbots 'interpretation' suggestion, possibly revise their (intermediary) interpretation result, and feed the output back into the first, second, and third chatbot until no further 'interpretation' suggestions on the first batch of text material are generated by the three chatbots.

The third phase of the hybrid 'interpretation' process, the so-called synthesizing interpretation can begin, once all of the relevant sequences of the text material that the researcher(s) want to interpret have been processed and interpretations were finalized by the researcher(s). In the third phase of hybrid 'interpretation', all interpretations are systematically combined and analytically condensed. The aim of the synthesizing phase is to create an overall interpretation that researchers can validate with explicit references to the data (in sociolect: empirically grounded).

General rule of thumb 3 for QDA program integrated AI assistant use for qualitative content analysis: You will need to become proficient in prompt engineering to get LLMs to support your interpretation process. And it is important to note that the output on your interpretations provided by an LLM are dependent on the respective LLM as these are trained differently. According

to the interpretive paradigm, the LLM outputs should be considered as various subjective outputs, which you should interpret in connection with your research question, your knowledge about social, spatial, and temporal contexts, and your own positioning in the research process. You should therefore never simply adopt the outputs by LLMs as 'interpretations' but rather see them as suggestions that can be used in your own interpretation process.

10.5 Conclusion

LLMs, chatbots, QDA programs, and likely other specialized content analysis tools will continue to develop quickly. This is at least the way it looks at the moment, in February 2025, judging by the speed at which the big tech companies are releasing new LLMs and applications based on these that possess an API. Furthermore, it is currently unclear whether it will become cheaper to train, well, fine-tune LLMs, as, for example, DeepSeek has shown. Depending on how the LLM market will develop, either a few tech companies will dominate the market, or/and many small, specialized companies will emerge, whose business concepts are based on specifically trained LLMs. Since everything is changing so quickly, we cannot make a recommendation as to which LLM or QDA program is best for you to support your qualitative content analysis. But we would like to give you some guidance on how to decide based on the aspects discussed in this chapter:

1. **Data**: First look at your data. Which data protection regulations do you have to comply with in the country you are conducting the research (see Section 3.4.2)? And which ethical principles do you have to adhere to in your research (see Section 3.4.3)? Then compare these requirements with data protection information provided by the companies that offer LLMs. And yes, it is necessary to familiarize yourself with this legal information so that you do not make any mistakes when using LLMs to support your research. If you are unsure, seek advice from the data protection officer and/or ethical approval board at your institution. In Europe, for example, it is necessary for interviewees to give their written consent for you to use LLMs for data analysis. If you have not yet collected any data, you can formulate the informed consent accordingly but need to be aware of the risk that potential interviewees, document providers, etc. will reject giving information. Even if consent has been given, you have to decide on a case-by-case basis whether it is ethically justifiable to use LLMs for qualitative content analysis. For example, is there a risk of de-anonymization even though you have guaranteed your interviewees anonymity? Or are you working with sensitive data that should not fall into the hands of tech companies and the (authoritarian) state, for example? Then it is better not to use LLMs because depending on the location of the servers used by the companies, country governments do have access to the data even without you knowing about it.

2. **Purpose**: If you have data that you can analyze using LLMs, the next question to ask yourself is what you want to achieve. In addition to using a QDA program, either with or without integrated AI assistance, you can choose whether to get support by no, one or more of the available chatbots in your research process. For example, if you want to use multiple LLMs to support your interpretation process (see Section 10.4.3), to date, you have to work outside the QDA program as you can only use the AI assistant integrated LLM. If you want to use multiple LLMs for hybrid 'interpretation', you will need to learn the skill of prompt engineering (see Section 10.2) so that you get the chatbot to do what you are aiming for. If you want to have an AI assistant at your side to help you summarize or develop categories, then it makes more sense to use a QDA program as this ensures that the LLM refers to your data and hallucinates less. Again, using a QDA program for systematic qualitative content analysis you should take a close look at the data storage and protection information here as well.
3. **Usefulness**: Depending on the type of research you are conducting, you should first ask yourself whether it is helpful to use LLMs at all. For example, if you only have a few interviews and are doing a deductive content analysis (see Chapter 6), it may be quicker and more efficient to code the interviews on your own than seeking AI assistance support. For deductive content analysis, you would already have developed a category system theory or studies before you start with data processing and analysis. Teaching the LLM which theoretical lens to use and then working with the AI assistant will take time and energy, and the results may not be better in the end – you will have to check the plausibility of outputs, do the interpretation, detect biases, etc. If you pursue an inductive approach to qualitative content analysis and are having difficulties constructing categories, then using an AI assistant can be helpful to overcome the mental block and get started with the process. However, make sure to not simply take LLM's suggestions as 'truth' or 'correct' and therefore always scrutinize LLM suggestions during your construction of categories. You should also consider which epistemology you follow and whether the use of LLMs is paradigmatically acceptable, for example yes for the interpretative paradigm and no for positivism and critical rationalism (see Section 10.3).
4. **Values (and biases)**: LLMs and AI assistants integrated in QDA programs are not neutral tools but software technologies that have been pretrained on data with specific other purposes, including bias and values, and that most LMMs are subject to human censorship. As we have shown in Section 10.2.2, LLMs therefore differ from each other in terms of content input and output characteristics. You can take advantage of this if you want to get a diverse output as described for LLMs supporting your interpretation (see Section 10.4.3). In addition, you could make the mistake and choose the 'wrong' LLM for your qualitative content analysis. For example, if you want to analyze data on political discourse in democracies, DeepSeek would not be your first choice since democratic values, freedom of

opinion, and political discourse tend to be censored in this (China-based) LLM. Moreover, you should consider using a 'reasoning' model for analyzing political discourse (see Section 10.2.4). This means that you need to become familiar with the LLMs available to select the most suitable one for your data and research purpose.

Once you have the answers to the four aspects and have addressed all the associated questions discussed here, you can make an informed decision as to whether you want to conduct a qualitative content analysis supported by one or more LLMs. If so, you can then decide which LLM to choose. Finally, we would like to emphasize once again that using an LLM or a QDA program does not mean that you no longer need to learn the qualitative (or quantitative) method of content analysis and have to be very familiar with your data and its social, spatial, and temporal contexts.

Notes

1 See: retrieved May 2, 2025, from https://eur-lex.europa.eu/eli/reg/2016/679/oj.
2 The same does not yet apply to qualitative content analysis as no analyses of qualitative content analysis using LLMs are available to date. But Kuckartz and Rädiker (2022) have published guidelines for using LLMs in MAXQDA for qualitative content analysis.
3 Retrieved May 2, 2025, from https://platform.openai.com/tokenizer.
4 Retrieved May 2, 2025, from https://www.qinsights.ai/.
5 Retrieved May 2, 2025, from https://qualcoder.wordpress.com/how-to-use/.
6 Similarly to induction, *abduction* is defined as an epistemological attitude of discovery (Peirce, 1934; Reichertz, 2013). Beyond the argumentation for a specific abductive analysis as an approach to discovery, it is hardly possible to distinguish between abductive and inductive conclusions in data analysis (Frankfurt, 1958).
7 Positivism is a paradigm of scientific theory that aims to explain social and natural phenomena through relationships that can be observed and are subject to natural laws. It is characterized by methodological rigor and empirical verifiability as well as the assumption that knowledge can be obtained from observations that can be experienced by the senses. Truth is understood as the correspondence of statements with observable reality and is verified by empirical methods such as experiments or measurements. Positivism assumes that an objective reality exists independently of human perception and that this can be fully recognized. As a form of critique, Karl Popper (1935/2002) developed critical rationalism, which addresses the problem of induction: deriving universal statements of law from a finite number of observations cannot be logically justified. As emphasized in positivism, Popper sees scientific progress not in verification but rather in falsification (i.e., deliberately refuting hypotheses). For Popper, scientific knowledge is always provisional and arises from the interplay of theory and criticism, not from the mere accumulation of facts. Popper proposes the hypothetical deductive method, in which hypotheses are formulated, and verifiable predictions are derived and tested empirically. Falsified hypotheses are replaced by new ones, whereby science merely approaches the truth. In contrast to positivism, critical rationalism emphasizes the limitations of scientific theories: these are perspectives on reality, not complete representations. As in positivism, an objective reality is nevertheless assumed, and this reality can be increasingly better understood by correcting errors.

8 In comparison to the interpretative paradigm, in the positivist paradigm and in critical rationalism the variation of outputs produced must be regarded as methodologically inadmissible. Accordingly, LLM support in content analysis is incompatible with positivism and critical rationalism – although LLMs seem to function according to a bunded (statistical) rationality – because of their respective methodological standards.
9 Retrieved May 2, 2025, from https://www.maxqda.com/help-mx24/ai-assist/ai-chat-with-one-document.
10 Retrieved May 2, 2025, from https://www.maxqda.com/help-mx24/ai-assist/single-document-ai-coding.

Taylor & Francis
Taylor & Francis Group
http://taylorandfrancis.com

PART 3

Practical guide to automated big data content analysis

Taylor & Francis
Taylor & Francis Group
http://taylorandfrancis.com

11

AUTOMATED CONTENT ANALYSIS

Basic concepts and useful tips prior to data collection and data analysis

11.1 Introduction

Methods of **automated content analysis** surged over the last two decades (Bashiri & Naderi, 2024; Churchill & Singh, 2022) and were successfully applied in the following domains:

- Parliamentary debates (Fuhse et al., 2020), or political debates (Müller-Hansen et al., 2021; Ylä-Anttila et al., 2022)
- Newspaper articles (Bleich et al., 2021; Corvo & De Caro, 2020; DiMaggio et al., 2013)
- Scientific publications (Benz et al., 2024; Daenekindt & Huisman, 2020; Grothe-Hammer & Kohl, 2020; Rossier et al., 2022; Schwemmer & Wieczorek, 2020; Wieczorek et al., 2021; Wieczorek et al., 2023b)
- But also fiction, poetry, or song lyrics (Czedik-Eysenberg et al., 2024; Kozlowski et al., 2019)

These methods of automated content analysis root in natural language processing and machine learning and increasingly rely on deep learning algorithms. **Automated** means that the individual aspects of data collection, data preparation, data cleaning, data analysis, and model evaluation (1) are carried out without human intervention, (2) according to previously defined rules which are (3) applied to the computer to complete different tasks. Procedures subsumed under the label of automated content analysis differ (a) in terms of the degree to which human expertise is incorporated into the analyses and (b) the steps at which this expertise is injected into a workflow.

Natural language processing (NLP) is a branch of computer science and artificial intelligence, which is used to understand and process natural language texts (Chowdhary & Chowdhary, 2020). Processing, in this sense, means

DOI: 10.4324/9781003496397-14
This chapter has been made available under a CC-BY-NC-ND 4.0 license.

that text documents are prepared in such a way that our computers can automatically calculate different properties via statistical methods, extract different kinds of information, for example latent topics present in a text corpus (see Chapters 15, 17, and 18), or assign properties to texts, for example sentiments (Chapter 16). **Natural languages**, such as the English language, are spoken by humans. In contrast, there also exist **formal languages**, which are used in mathematics, computer science, linguistics, as well as logics. They are defined as "a set of sequences, or strings over some finite vocabulary" (Jäger & Rogers, 2012, S. 1956). As such, a formal language is a collection of mathematical and algorithmic tools "with computing instructions and its own grammar" (Jäger & Rogers, 2012, p. 1956).

The programming language Python, which we are going to use, is an example of a formal language. It uses a set of instructions (commands, functions, and methods) that are handed to the computer, which then translates these instructions into calculations (e.g., count words per text, extract topics, assign sentiment values), and provides a readable output. The tasks of NLP include text and speech processing, morphological analyses (lemmatization, stemming, part of speech tagging, named entity recognition), syntactic analyses (e.g., parsing, named entity recognition), translation, sentiment analysis, and discourse analysis (= topic modeling). Against this backdrop, the methods of automated content analysis we will employ rely on the recognition of meaning and patterns present in natural language.

Attempts to analyze texts automatically can be traced back to the 1950s (Deng & Liu, 2018, S. 2–18). During this first phase, computer scientists tried to use fixed sets of rules, mainly to translate texts (e.g., from Russian to English). The basis for this was Noam Chomsky's (2009) universal grammar, as well as the assumption that key elements of language are represented in the human brain and are thus linked to the innate ability of humans to understand language and meaning.

This 'rule-guided phase' lasted until the 1980s, when it was replaced by an empiricist paradigm which favored the application of statistical methods and models to (a) analyze properties of human language and (b) to extract patterns from texts, which are organized in text corpora (Murphy, 2012). Afterwards, the same algorithms that were trained to identify patterns in one text corpus were used to find similarities in other text corpora and to predict properties of texts in the latter. One famous example is the spam filter, which tries to classify each incoming e-mail either as spam or regular e-mail based on the properties of the text (e.g., phrases, words), its title, and the sender.

From the 1990s onwards, multi-layered neural networks appear and evolved into deep learning methods in the 2000s and 2010. Deep learning methods are increasingly used to recognize patterns in large amounts of text data, extract features, or classify the content (Young et al., 2018). **Deep learning** models simulate the structure of the human brain (so-called **artificial neural networks**)[1] in order to learn patterns from data (Schmidhuber, 2015). They process inputs using a multi-layered, "neuronal" structure whose connections can change in

each calculation step and translate the input into an output (e.g., the classification of a text into a category). As we will see in Sections 16.3 and 16.4 in regards to sentiment analysis, and Section 18 regarding topic modeling, these algorithms are able to contextualize given content, learn text representations, and to map these representations onto an output (topics, sentiments).

Machine learning is a field of computer science in which novel algorithms and statistical methods are developed. These algorithms and methods, in turn, mimic the learning behavior of humans. These algorithms should ideally perform tasks without explicitly being instructed how to fulfill them. Furthermore, these algorithms yield the ability to adapt to novel information (Molina & Garip, 2019). They are used in pattern recognition (e.g., detection of topics in texts, facial recognition, detection of tumors in medicine, differentiation between regular emails and spam) and apply statistical calculations to recognize these patterns in yet unknown data to predict outcomes such as behavior or decisions (regression or classification problems). A **pattern** "describes aspects of a phenomenon in a recognizable, yet abstract form" (Grisold et al., 2024, S. 6). We will encounter different types of machine learning in more detail in Section 11.2.

You may ask why we should care about such computational, automated methods stemming mainly from the computer sciences. One of the main reasons is the sheer vastness and complexity of **unstructured text data** available in the world wide web, newspapers, or libraries, as well as its interlinkage with different types of data such as video and audio data (e.g., multimodal analysis, see O'Halloran et al., 2018). Machine learning algorithms linked to automated content analysis allow us to make sense of the enormous amount of data in a concise manner using a comparatively small amount of time.

For example, social media platforms such as X (formerly twitter), TikTok, YouTube, or Facebook not only create enormous amounts of data each second but also provide users (individuals as well as organizations) with the ability to directly interact (e.g., via comment functions), to create or link content using hypertext and hashtags (Albu & Etter, 2016). Keep in mind that in 2024 alone, 149 zettabytes worth of data are produced in addition to the 118 zettabytes produced in 2023 and 94 in 2022 (Sukumar et al., 2024). One zettabyte is equivalent to 10^{21} bytes, and the current book approximates to 20 megabytes, or $20*10^6$ bytes. Consequently, one zettabyte is equivalent to 50 trillion times this book.

In other words, we enter the domain of **big data analysis**.[2] We as researchers simply cannot comprehend such vast, interlinked, and potentially multimodal data in its complexity. Even if we try to, we neither can read nor identify all the relevant texts on our own, let alone perform a qualitative in-depth analysis. Similar to theoretical sampling in grounded theory, we may rather focus on a small number of texts which we identify as interesting cases after using machine learning algorithms in a first step (Carlsen & Ralund, 2022), before we perform an in-depth analysis of the texts and refine and differentiate the extracted patterns in a second step (Nelson, 2020).

We can use automated content analysis in combination with **digital trace data** (see below) to study social interaction or human behavior in real time. We can furthermore use digital trace data for theorizing (Grisold et al., 2024; Power, 2022), to generate research questions (Kallio et al., 2022) or to analyze how social phenomena unfold over time online. Digital trace data comprises of the following:

1. Social media data, such as social interactions (comments, likes, shares), and information dissemination patterns,
2. Web browsing data, such as website interactions,
3. Data collected in applications on mobile phones, such as in-app behavior, time spent, timestamps (when apps were used), and
4. Sensor data, such as GPS location data, blood pressure, heart rate, or sleep statistics collected by smart watches.

Social media data not only stems from social media platforms (e.g., Instagram), but also commercial platforms such as Airbnb or care sharing platforms, review portals such as the Internet Movie Database (IMDB), wikis (Wikipedia), or blogs. In the box 'Data sources used for research' you will find some of the most common data sources of online communication used in previous studies for automated quantitative text analyses.

Textbox 11.1 Data sources used in research

Social Media

"Social media are Internet-based channels that allow users to opportunistically interact and selectively self-present, either in real-time or asynchronously, with both broad and narrow audiences who derive value from user-generated content and the perception of interaction with others" (Carr & Hayes, 2015, p. 50).

Blogs

Online diaries that are run by individuals or operated by individuals or a small number of people. They are used to publish either personal experiences or topic-related contributions (posts).

Microblogs

Online platforms on which many users publish short messages with just a few characters.

Commercial platforms

This includes search engines, video platforms such as YouTube, trading platforms such as Amazon, brokerage platforms such as Airbnb, and crowdfunding platforms such as Kickstarter are subsumed.

Review portals

These are databases and websites on which users write ratings and reviews of films, music, books, and other media and save them.

Wikis

Websites on which information is provided and at the same time can be collaboratively added to or changed.

Forums

Websites on which discussions are held on specific topics (e.g., text analysis methods) or subtopics.

Internet archives

Websites or databases which preserve digital content, such as versions of websites, books, audio files, or other historical records.

Overall, automated content analysis combines qualitative and quantitative content analysis and, while more aligning to the latter, allows us to analyze the latent and manifest content of texts and communications. As a reminder: the manifest content is linked to the question of *what* is communicated directly in the text, while the latent content focuses on *why* some content is communicated. The analysis of latent meaning thus aims to capture the implicit meanings of communications (see Chapter 2 in this book), which usually must be uncovered by us scholars. As we focus on automated content analysis, the first of the four types of digital trace data is the most important in the following chapters. Nonetheless, social media platforms (Carr & Hayes, 2015) are not the only sources of vast amounts of (interconnected) texts ready for analysis.

We can identify **manifest content** in automated inductive-quantitative text analyses, for example, by counting words or phrases. Given the assumption, that people choose certain phrases to talk about different topics, we may be able to infer **latent meaning** from the manifest patterns manifest in, such as the distributions of co-occurring word in a corpus (= set of documents) (Blei, 2012; Chang et al., 2009). If, for example, stable topics that convey latent meaning emerge, these still must be interpreted using techniques of qualitative content analysis, which, in turn, enables us to understand and interpret the texts as well as the appearance of topics across documents in our text corpus. Using mathematical methods, we may not only be able to identify these patterns, but may also be able to find them in novel, previously unseen documents. Together with qualitative interpretation, these methods are beneficial for the in-depth analysis of large corpora.

11.2 Types of machine learning

As we will employ different machine learning algorithms for the purpose of automated content analysis in the forthcoming sections, we must get familiar with different families of these algorithms. We do so to get an impression of and understand the ideas underlying the algorithms we apply during

automated content analysis. The families of machine learning algorithms we will encounter include **unsupervised**, **supervised**, **semi-supervised,** and **reinforcement machine learning algorithms**. Their main differences are (1) the share of labeled data available for model training, (2) the task of pattern recognition, prediction, or outcome optimization. We broadly speak of **labeled data**, when an observation, in our case a text (document, paragraph, sentence), yields a class (e.g., topic), or value (e.g., sentiment polarity score) annotated by humans. **Training** means the process of pattern detection and the adjustment of the detection process by the machine learning algorithms as novel, or previously held-out data is presented. We will describe the algorithm underlying this process in Section 11.3. Distinct from labeled data, we may also speak of **pseudo-labeled** data when a machine learning algorithm assigns a label (topic) or value (sentiment) with high certainty of being correct. The labels are then reintegrated into the training data to refine the machine learning models (Arazo et al., 2020; Doersch & Zisserman, 2017). We will further distinguish between **statistical methods** (e.g., dimensionality reduction approaches, such as geometric data analysis), shallow **neural networks**, and **deep neural networks**.

Keep in mind that, for each method, it is important to define separate **parameters** and **hyperparameters** for model training and analysis, as well as to document the reasons for this definition. Please keep in mind that every decision we make – especially at the early steps of our model training – can and will cascade throughout the subsequent steps of the workflow. Accordingly, they may have negative impacts on the results and further reduce both reproducibility and replicability. In regards to the model training, a **parameter** is a value that is learned throughout model training. You may have encountered parameters, for example, in linear regression. In this context, the parameters provide information on the association between the input variables and the target variable (e.g., the impact of socioeconomic status and migration background on educational success). In contrast, a **hyperparameter** is a parameter which governs the learning process of the algorithm, that is, how the model behaves when confronted with novel data, how fast it adapts to errors, or how errors, such as misclassifications, are treated. For many machine learning-tasks, fine-tuning the hyperparameters to minimize errors and to avoid overfitting is extremely important. While the former might lead to underfitting (and thus many misclassifications), the latter substantially decreases the generalizability of findings due to the fact that the trained model is perfectly trained to detect the specific patterns (and random noise) that characterize the training set.

Unsupervised machine learning algorithms recognize patterns in digital data without human intervention using previously defined formulas. Examples of this are cluster analysis methods, for example K-Means, dimension reduction methods, such as principal component analysis or multiple correspondence analysis (Blasius & Greenacre, 2014; Hjellbrekke, 2019; Le Roux & Rouanet, 2010), or topic modeling methods, including structural topic models (Roberts

et al., 2014), Latent Dirichlet Allocation (Blei et al., 2003), or BERTopic (Grootendorst, 2022). Here, the task is to reduce the complexity of the data and to subdivide the resulting mathematical space so that we can easily grasp the latent content present in the text corpus and make sense of the meaning conveyed in the respective texts.

The aim of **supervised learning algorithms** is to predict output values (dependent variables) using previously annotated data with known characteristics, the so-called **ground truth**, and a series of input variables (independent variables). To do so, we use a completely labeled dataset and split this dataset into a training set, validation set, and a test set.

As the name suggests, a **training set** is the part of a dataset (e.g., 80% of the documents) that is used to fit a statistical model on the data. The fitting is evaluated using metrics which either quantify how adequate the model covers the classes or underlying values, or how erroneous the predictions of the model are. Usually, the model is fitted multiple times. Each of the fitting attempts is called **epoch**. We will introduce metrics for the quantification of errors of our models in Section 11.3.

The **validation set** is a portion of the training set (e.g., 20%) that is used to evaluate how well the model generalizes to unseen data. Unlike the test set, it is used iteratively during training to guide the tuning of hyperparameters. The model's predictions – and errors – on the validation set help assess its performance, but they do not update the learned parameters. Instead, they provide insight into whether hyperparameters should be adjusted to improve the ability of the algorithm to generalize to unknown data. At the same time, the validation set – or rather the predictions on the validation set, are used to avoid overfitting.

Overfitting means that our machine learning algorithm learned patterns from the training data too well, which greatly reduces its ability to generalize from the training data. As a result, the model may perform exceptionally on the training data but cannot make accurate predictions on unseen data. Finally, the **test set** consists of unseen data that was not used during (a) the training process and (b) hyperparameter tuning. The test set therefore provides an evaluation of the ability of the model to generalize its predictions to new data. By comparing the model's predictions with the labels, we can assess its performance using appropriate metrics, such as accuracy (for classification), or R^2 (to measure explained variance). A model trained which performs well on the test set indicates that it may be useful for automated content analysis or sentiment analysis of other corpora.

Semi-supervised learning algorithms are hybrids between supervised and unsupervised learning. In this case, our dataset includes only a small number of labeled observations (e.g., positive, neutral, or negative sentiments) and an abundance of unlabeled observations. Such algorithms then either try to transfer the labels of the unlabeled observations by some condition, or try to infer the labels of the unlabeled observations. According to Duarte and Berton (2023, p. 9419–9431), there are many different, elaborated approaches to semi-supervised learning. However, we will focus on cluster-then-label

approaches, self-training approaches, co-training approaches, and transfer-learning approaches, as these resonate with the topic modeling approaches and sentiment analysis approaches used in the following sections.

Finally, let us take a look at **reinforcement learning** (RL). Different from supervised and unsupervised learning approaches, reinforcement learning does not seek to extract patterns or structures from the data, or to predict predefined outcomes. Instead, an RL algorithm has a general goal and learns to achieve it by optimizing a **reward function** (see Icarte et al., 2022). Within this framework, the algorithm is referred to as an **agent**, as it autonomously interacts with the **environment**. An environment can be static or dynamic. In case of the latter, it changes its properties over time (e.g., party preferences of voters, issue salience) and forces the agent to adapt repeatedly. To do so, it uses **policies**, which equate to strategies for decision-making. An agent is successful within this environment, if it maximizes the reward over time – more precisely: across multiple learning steps (Sutton, 2018). On abstract level, a reward is feedback on how well an agent performs within an environment. Often, rewards are delayed, meaning that the consequences of an action only play out in later learning steps.

Take, for example, a political party (agent) which seeks to increase its share of votes in upcoming elections (rewards). To do so, it implements a strategy (policy) to increase the percentage of received votes. The voters (with their preferences) and competing parties are the environment. The strategy (e.g., campaign messages) may have immediate effects, while others have an impact on voter sentiment over multiple elections. In terms of reinforced learning, our party seeks to maximize the long-term effects, meaning that it seeks to optimize its strategy to get the highest share of votes not only for the next, but for all future elections.

11.3 Relevant concepts in machine learning and deep learning

After providing you with an overview on the different types of machine learning, it is now the time to introduce relevant machine learning termini in the context of shallow and deep **artificial neural networks**. We do so to provide you with the means to understand the fundamental building blocks of the BERT model and similar artificial neural networks that are currently employed for automated content analysis and denoted as "state of the art" (Jim et al., 2024; Minaee et al., 2021; Suissa et al., 2022). An artificial neural network consists of different **layers** of **artificial neurons**, which are connected by ties between the layers. A **shallow neural network** is a network with three layers, one **input layer**, one **hidden layer**, and one **output layer**. A **deep neural network** consists of one input layer, multiple hidden layers, and one output layer. The ties connecting the layers could be unidirectional, meaning that they pass a signal from the input layer across the hidden layers to the output layer (feed forward). In other cases, these ties could link artificial neurons within a layer or even loop back to lower layers. These ties are so-called **recurrent** ties. There are different types

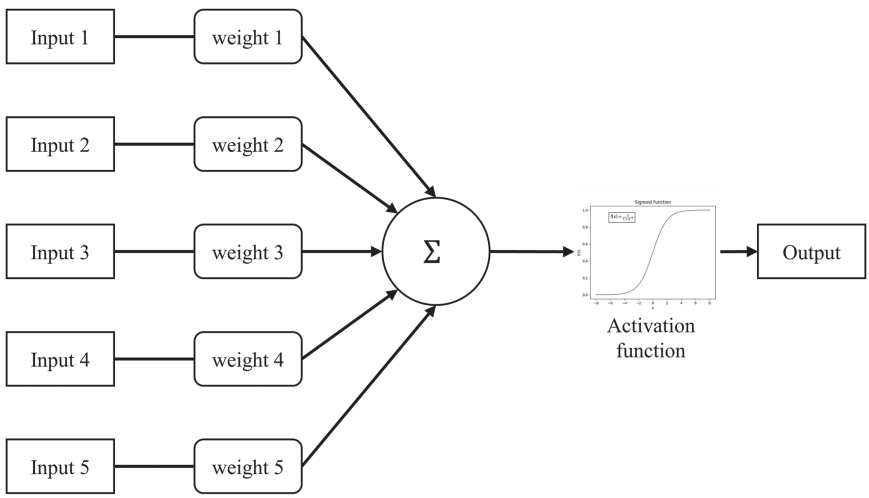
of deep neural networks,[3] of which we will introduce the Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2018) in more detail.

Against this backdrop, let us begin with the basic building block of a neural network: the **artificial neuron**. An artificial neuron, or **perceptron** (Rosenblatt, 1957), calculates the weighted sum of an input (e.g., the co-occurrence of words), activates and produces an output (e.g., a "topic") if the combined input is strong enough. Figure 11.1 depicts the perceptron, with the inputs on the left (e.g., words by their position in a text), weights/coefficients (= numbers that represent the "strength" of a word), the summation of all the input weights in the middle, an activation function, and the output on the right.

But when and how do these artificial neurons know if they should activate and process a signal? Like neurons in the brain, each neuron activates if a signal is strong enough. Whether a signal is considered strong enough by the artificial neuron is governed by an **activation function**. Here, a threshold value (e.g., 0) of the sum of all signals must be exceeded to send a (weighted) output, which will then be the input of the next layer.[4] A neural network without activation functions "is only a linear regression model", albeit with millions of parameters, whereas "the activation function gives the network the capacity to learn more complex tasks" (Mercioni & Holban, 2020, p. 141), such as the identification of sentiments or topics based on words, phrases, dependencies, syntactic and semantic features.

Artificial neural networks consist of an input layer, hidden layers, and one output layer. As these layers are stacked upon each other, we speak of the input layer and the first hidden layers as lower layers, and layers that follow as upper layers. An **input layer** is the layer where the data, in our case the texts as well as its metadata, is introduced to the network. The number of artificial neurons is

Figure 11.1 Stylized depiction of a perceptron with a sigmoid activation function.



equal the number of features (e.g., words, phrases) we pass to the network. **Hidden layers** are situated between the input and output layer and contain a certain number of nodes (e.g., 128, 256, 512) which are connected to some or all artificial neurons of the previous (= lower) layer and the next (= upper) layer. Their task is to detect more or less abstract patterns present in the data. The number of hidden layers used to build the architecture of an artificial neural network usually depends the difficulty of the task to solve (e.g., regression analysis versus topic modeling), the degree of abstraction and potential interlinkage between the input variables (Kumar & Garg, 2018).

By stacking hidden layers, an artificial neural network learns to recognize progressively more abstract patterns, such as word co-occurrence, semantics, grammar, word dependencies, abstract phrases, and nonlocal aspects spread across long distances in the text (Galassi et al., 2020). There are also different types of hidden layers. **Dense layers** are layers in which every artificial neuron of a given layer is connected to every neuron in the next lower layer and the upper layer. A **regularization layer** adds a penalty, for example drops a number of neurons to simplify the model and to prevent overfitting (Srivastava et al., 2014). A **recurrent layer** includes a sort of memory of previous inputs, which is why this type of layer is used to analyze sequences, such as words in a sentence or audio data (e.g., songs) (Yu et al., 2019).[5] The **output layer** is the final layer of an artificial neural network, which translates the calculations done in the hidden layers into an output, such as a sentiment class, or topic distributions. Figure 11.2 depicts a (not so) deep neural network, consisting of one input layer, three fully connected hidden layers which are responsible for pattern recognition (words, their surroundings, dependencies, semantics etc.), and one final layer which classifies the given sentences, in our case with a sentiment.

So now that we know how neural networks are structured, that each neuron has an activation function, and that there are different layers present in a (deep) neural network, how exactly does such a network learn? Keep in mind that learning in an artificial neural network is formalized as minimizing the deviance between the "true" values (e.g., annotated by humans) and the values predicted by our artificial neural network.

Machine Learning algorithms seek to predict classes, for example sentiments, or phenomena as accurately as possible. The errors it makes are also called *cost* and can be calculated using different formulae (see Wang et al., 2020 for a comprehensive overview). If a statistical measure is used to quantify this error, the measure is called **loss function**. Overall, it tries to minimize the loss function, and the speed it adapts to the errors made is called **learning rate**. We will encounter some in Section 16.4 when we train deep learning algorithms to identify sentiments in our texts.

One, if not the most important algorithm that enables machine learning algorithms to "learn" is called **Stochastic Gradient Descent**, SGD[6] hereafter (Bottou, 2010). In general, it tries to minimize the cost/errors of the predictions, as indicated in Table 11.1 and checks, after each step, if there are less errors, and if so, the extent of the **rate of change** (= gradient) in errors (see f.e. Zaheer & Shaziya,

Figure 11.2 Depiction of a three-layer deep, fully connected artificial neural network (with ReLu activation function).

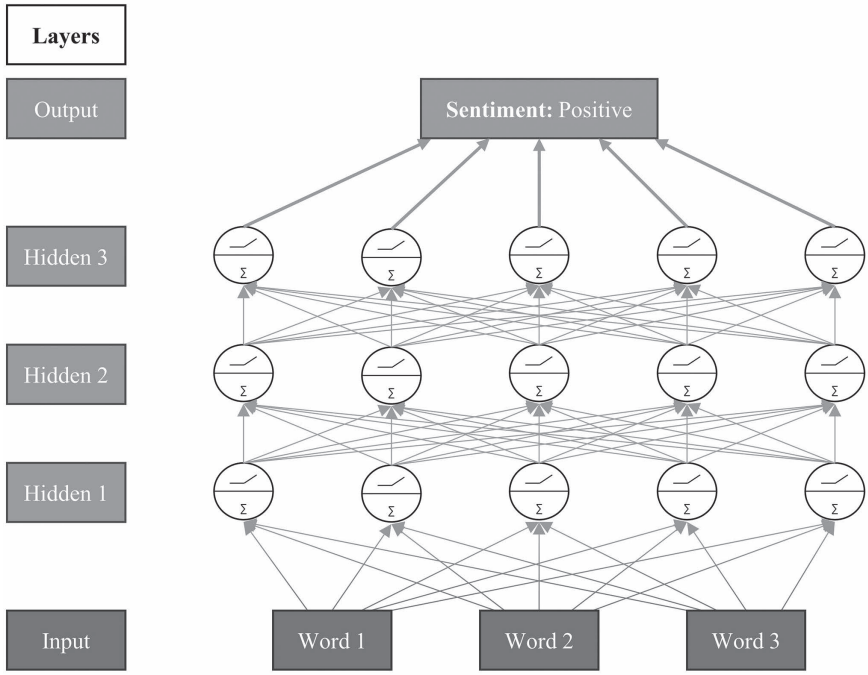
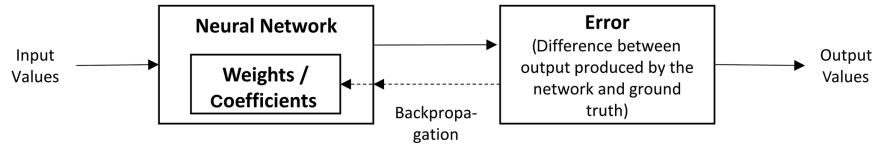


Table 11.1 Simplified, Exemplary Calculations of the First Steps of Stochastic Gradient Descend (Errors are colored grey)

	Sentiment $_{step1}$	*Predicted sentiment* $_{step1}$	*Error* $_{step1}$	*Sentiment* $_{step2}$	*Predicted sentiment* $_{step2}$	*Erros* $_{step2}$	*Change*
1	Positive	Positive	No	Positive	Positive	No	0
2	Negative	Negative	No	Negative	Negative	No	0
3	Neutral	Positive	Yes	Neutral	Neutral	No	1
4	Positive	Neutral	Yes	Positive	Neutral	Yes	0
5	Positive	Positive	No	Positive	Positive	No	0
6	Negative	Neutral	Yes	Negative	Neutral	Yes	0
7	Negative	Neutral	Yes	Negative	Negative	No	1
8	Neutral	Negative	Yes	Neutral	Negative	Yes	0
∑			5			3	2

2019). After this step, the machine learning algorithm changes the parameters, also called **weights**. SGD continues to do so until it cannot find a solution which reduces the deviation between the predicted and given values of the target variable (e.g., sentiments, topics) any further. The "stochastic" part of SGD indicates

Figure 11.3 Schematic representation of the backpropagation algorithm (according to Werbos, 1990, S. 1552 slightly modified).



that we need a sample and not the entirety of our data to update our parameters and to minimize the error/cost. This is important, as this speeds up the calculation enormously and even allows us to train our machine learning algorithms in the first place, especially if we have abundant and complex data, such as texts, to classify. In this case, the model picks a sample, or **batch**, of 32, 64, 128 texts for classification, calculates the error, updates the parameters/weights, draws the next batch, calculates the error and continues until all batches were used to calculate the error. Afterwards, this process is repeated multiple times (e.g., 100 times) until no further improvement regarding the prediction is seen. Albeit SGD is a well-known and widely used algorithm, there are many other, widely used variants which are more suitable for automated topic modeling, such as the adaptive moment estimation (ADAM, see Kingma, 2014).

SGD and its variants apply the **backpropagation algorithm** (stylized in Figure 11.3) (Rumelhart et al., 1986). This algorithm works as follows. First, it predicts an output, for example sentiment class, and preserves all weights of the connections between the layers of an artificial neural network. Second, it calculates the loss function and, by doing so, measures the prediction error. Third, it calculates the extent to which each connection across the layers contributed to the error. Fourth, it sends the error made in a training step/epoch (e.g., the difference between assigned and predicted sentiment of a text) back from the output layer throughout the hidden layers to the input layer. Finally, the gradient of the error is calculated individually for each artificial neuron and, subsequently, the algorithm changes the weight of each connection between the neurons to minimize the overall error.

Setting the **learning rate** adequately is an extremely relevant task in machine learning. Learning rates can either be too small or too large. A small learning rate might cause the SGD algorithm to take a long time to settle at a local minimum, meaning that the solution (e.g., the overlap or deviance between predicted and current sentiments) may not be the best solution, even though there might be better solutions out there. For example, this is the case if we can correctly predict positive text sentiments but may often classify neutral sentiments as positive ones. Against this backdrop, let us illustrate the SGD algorithm so that we can better understand how it works.

Imagine yourself on a mountain ridge. Imagine further, that it is afternoon and it will soon be getting dark, and on the horizon, you see clouds moving your direction. It is the first time that you got up the mountain ridge, thus you

are not too familiar with the terrain, its slopes, peaks, valleys, and vales. Yet you want to descend towards the valley with the lowest altitude where you see villages or towns from afar. Preferably, you want to descend and lower your altitude as much as possible, as you know that there is a nice village somewhere down there. You can either decide to take small steps and take longer to get downhill (as you do not know the terrain too well), or you can decide to descend faster, for example run or take larger steps. In case of the former, you might find a town to stay, which is at a higher altitude, but you do not want to ascend potential hilly terrain (upwards). In case of the latter, you may reach a village, but decide to hasten towards the next potential town or village situated at a lower altitude. This may or may not work and you might end up at a higher altitude amidst a thunderstorm at night (usually, we do not want to end up at a mountain peak in a thunderstorm). Furthermore, with each step that you take, you learn more about the terrain, the slope, etc., and you might decide to change direction, or stay at a location that suits you well (e.g., a small village to weather the thunderstorm in a relatively low altitude). In this example, the altitude is the error/cost, the speed of your steps are the learning rate, the villages in low, but not the lowest, altitudes are called local minima, and the terrain is an abstract representation of all combinations of errors and weights based, in our case, on our texts and the machine learning algorithm employed for automated content analysis.

Regarding deep learning algorithms used for classification (especially sentiment analysis in Section 16.5), we encounter four evaluation metrics. These metrics quantify how well our deep neural networks classify the data and help us identify misclassified cases. In the context of sentiment classification, this means that we evaluate how well our model predicts positive, neutral, and negative sentiment classes across the texts. To understand their calculation, we first need to introduce the confusion matrix (Table 11.2).

A confusion matrix compares the predicted classes (y-axis) with the actual class label (x-axis), which are usually are human-annotated. From this matrix, we can derive four canonical evaluation metrics:

- **Precision** measures the proportion of correctly predicted positive cases out of all cases predicted as positive: $Precision = \frac{True\,Positives}{True\,Positives + False\,Positives}$.
- **Recall** is the proportion of correctly predicted positive cases out of all actual positive cases: $Recall = \frac{True\,Positives}{True\,Positives + False\,Negatives}$.
- **Accuracy** measures the proportion of correctly classified cases out of all observations: $Accuracy = \frac{True\,Positives + True\,Negatives}{Total\,number\,of\,Observations}$.
- Finally, the **F1-Score** is the harmonic mean of precision and recall: $F1 = \frac{2 * Precision * Recall}{Precision + Recall}$. It is used to balance both measures, for example in cases when our classifier should be sensitive to false positives and false negatives.

Table 11.2 Example of a Confusion Matrix

		Ground Truth	
		Positive	**Negative**
Predicted	**Positive**	True Positive	False Positive
	Negative	False Negative	True Negative

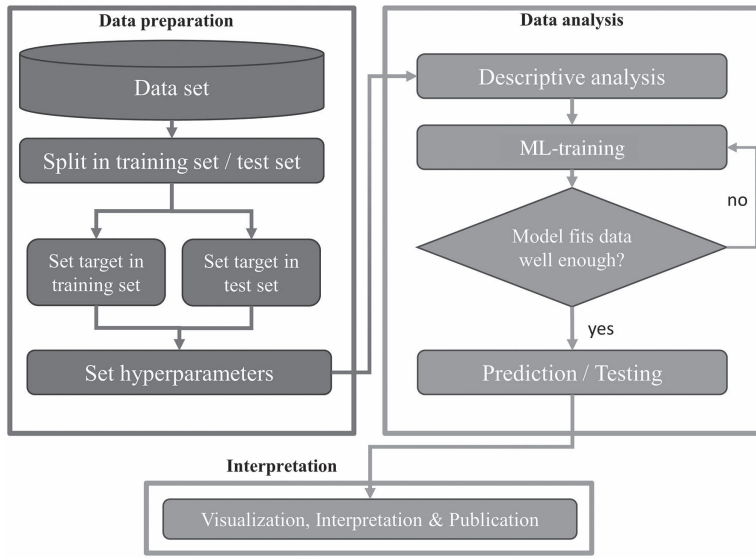
11.4 Prepare your workflow

Similar to research projects in other domains, we must plan the individual steps of our automated content analysis to minimize effort and errors in every step taken in the course of our research endeavor. As we are working with Python, this means that we not only have to plan what data to use, how we prepare the data, what methods we will use, which models (e.g., Latent Dirichlet Allocation, BERT) to use, but also how to interpret the output. It also includes that we must plan ahead the programming steps to take and how each individual programming step relates to the tasks we seek to accomplish (e.g., data cleaning, descriptive analysis, model training etc.). In other words, we must prepare a **workflow**, which links the programming steps to our research endeavor. The main idea of a workflow is to separate the research process – in our case: related to topic modeling and sentiment analysis as forms of automated content analysis – into discrete chunks. Each chunk is a task, which can be further subdivided into subtasks. Each task or subtask is ideally linked to a single programming step and a few lines of code at best. Usually, if a workflow is established and tested, it is scalable (= can be adapted to different data of different sizes) and may be applied to many tasks with only minor tweaks.

Against this backdrop, we define a workflow as a precise, schematically visualized description of a process that takes place in several interdependent and sometimes parallel steps (Goble et al., 2020), in which individual tasks are linked to specific programming and calculation steps (= implementation) and brought into a chronological sequence (Deelman et al., 2018, S. 168). A workflow serves to ensure the traceability, reproducibility, and replicability of the research, programming and calculation processes, their transferability to other problems, error reduction, and increased efficiency of the entire work and documentation process. Despite all planning, a workflow is usually not implemented in a single, error-free, linear process. Instead, it is an iterative process in which individual code blocks are written, the commands and specified functions are tested, errors are corrected, models, architectures, and analyses are tried out and discarded if necessary.

A simplified, exemplary machine learning workflow is depicted in Figure 11.4. As you notice, it is divided into three main tasks, namely data preparation (left upper side of the figure), data analysis (upper right side of the figure), and interpretation (at the bottom of the figure). Each task is subdivided

Figure 11.4 Idealized workflow with sequential, parallel, and recursive steps.



into subtasks. Beginning with the data preparation task, we see a cylindrical shape on top. This shape represents stored data, in our case a structured data frame containing the text corpus, metadata, and a target variable, such as a text topic or text sentiment. The arrow below tells us that we will use some command to split our data into a training set and test set. The branching arrow below indicates that we will do the next step, namely naming or setting the target variable, in parallel for the training set and the test set. Afterwards, the arrows indicate that we will continue sequentially with setting the hyperparameters, which will govern the training and learning procedure.

The next task "data analysis" (right hand side of the figure) encapsulates the steps as subtasks which must be taken to train our model and evaluate its accuracy on the test set. In a first step (on top), an inspection of the input- and output variables is conducted similar to the descriptive analysis outlined in Section 14.5. We then initiate the machine learning process (e.g., a pretrained BERT model for sentiment analysis, see Sections 16.3 and 16.4) and train the model. The rhombus below is a logical inquiry, asking whether the model is trained sufficiently. If not, the recursive, branching arrow on the right indicates that the training should continue until the model fits the data well enough. If the model converges and the data fits well enough, we use the trained model on the test set and evaluate how well it predicts the target variable in the test set (e.g., the sentiment, topics of texts or documents).

Lastly, we see the task "interpretation" (at the bottom of the figure). Here, visualization, interpretation, and publication tasks are merged into a single subtask. We did this since these three are linked in the sense that the visualization of the

models, the interpretation of the output (e.g., topics), and model improvement are intertwined. Finally, we need the interpretations of our automated content analysis to substantiate the output and visuals. As you may have noticed, this is a circular, hermeneutic task that resembles the way in which we conduct qualitative text analysis. Nonetheless, we will make use of workflows in the sections to come to structure our programming process and to guide you through the steps and commands necessary to conduct automated content analysis.

11.5 Literate programming

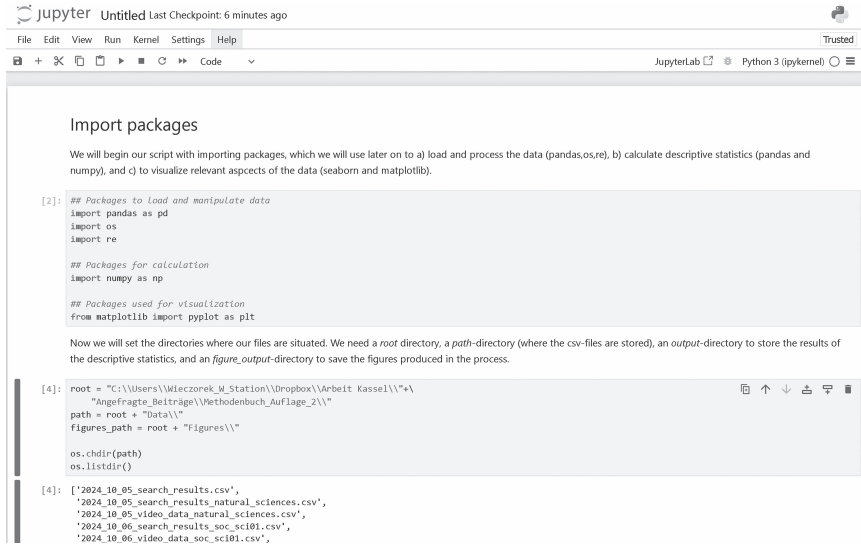
Now, let us spend some time thinking about how we can integrate the programming process using Python into a theoretically driven framework that is also compatible with qualitative approaches, such as grounded theory (Glaser & Strauss, 2017; Strauss & Corbin, 1997), or thick description (Geertz, 1973). We must keep in mind that our decisions in regards to the machine learning part of automated content analysis may greatly alter the output our models produce, what is eligible (or not) for interpretation, and ultimately, how we deal with social phenomena linked to social communication. This is why we need to integrate workflows, as outlined in Section 11.4 and link the discrete (sub-)tasks and the programming into our research process in a structured, comprehensible manner.

For this to succeed, we need a programming paradigm that (1) is compatible with the workflow of the social sciences, (2) structures the program code in such a way that other researchers – and thus potential co-interpreters of the topics – can understand it, and which (3) practically integrates the documentation firmly into the workflow. All points are fulfilled in the "literate programming" paradigm developed by Knuth (1984), which emphasizes the human readability of program code.[7] On the one hand, this is achieved through extensive documentation of the program code (weaving), in which the initial thoughts of the researcher that led to this step, the objective of the programmed code snippet, as well as the commands and command options of the code itself are executed. The second, parallel step is the concrete implementation of the program code (tangling). In this way, we can record the ideas and thoughts of our workflow – and thus also our handling of digital data. "Literate programming underlies interactive platforms such as R-Markdown and Jupyter Notebooks in Python" (Kery et al., 2018), making it easier for social scientists to implement. Figure 11.5 shows a screenshot of a Jupyter Notebook, which encapsulates both weaving and tangling of literate programming.

11.6 Working with data

In order to carry out an automated, quantitative content analysis, we need to collect text data. But how do we obtain this data? What data types are suitable

Figure 11.5 Screenshot of a Jupyter notebook incorporating literate programming.



to organize, analyze, and store the data? And finally, how is this data structured, and what are differences in the types of structure?

11.6.1 How to obtain text data

There are basically two ways to obtain text data for an automated quantitative content analysis. Either we draw on existing text corpora, or we create a **text corpus** ourselves by manually or automatically downloading texts and associated contextual information (so-called **metadata** such as year and place of publication). A text corpus is a collection of texts that follows a specific structure, for example a structured data table with clearly labeled column names, one of which contains the texts. In order to apply procedures of automated content analysis, correspondingly large text corpora are required. These comprise a few hundred texts in the case of geometric data analysis, semantic network analysis, and pretrained machine learning models, but can also contain many millions, even billions of texts if we wish to train a deep learning algorithm from scratch. The accuracy of the methods used, the number of topics found, and the complexity of the patterns associated with topics or sentiments depend on the number and the length of the texts available (Murshed et al., 2023; Sbalchiero & Eder, 2020).

Some examples of openly available text corpora include the Google Ngram Corpus and the Corpus of Contemporary American English. The Google Ngram Corpus contains words and word strings (so-called N-Grams, where n stands for the number of concatenated words). It is based on digitized books, which

can also be viewed on Google Books. This data corpus contains the words and word strings from 6% of all books that have ever been published (Lin et al., 2012). The Corpus of Contemporary American English most recently comprised more than 400 million words and several hundred thousand texts (Davies, 2010). It is based on transcripts of TV and radio programs, fiction and poetry, newspapers and scientific publications.

There are generally four options for accessing such text corpora or generating them yourself:

- **Repositories**: A repository (content repository) is a data management system. Repositories are large databases in which prepared and annotated data records are stored. One of the largest is the Harvard Dataverse, in which tens of thousands of datasets with assigned DOIs are stored for replication purposes. Users can search for specific text corpora in the search fields. One repository curated in Germany, for example, is the GESIS Data Archive for the social sciences.
- **Application Programming Interfaces (APIs)**: APIs are interfaces between computer programs and databases that make it possible to download large amounts of data by means of targeted requests. Examples of this are the Amazon API, YouTube API, TikTok API, Instagram API, or Reddit API. To be able to make the respective requests, you must first register as a user with the services to which the API is linked. After registering, you will be sent or assigned a user key (access token, authentication) by email or in the respective service. This can be used to make data requests on the site itself or in programming environments (such as Python). If the retrieved data is available in the database, it is stored temporarily and must then be saved in separate files or a shared data format. Finally, APIs are programming interfaces that researchers can use to make specific queries to databases that return (more or less detailed) information.

Textbox 11.2 Key concepts for data collection and data processing

Corpus

A structured collection of texts and words.

Repository

Data management system with an extensive variety of datasets for secondary analysis.

Web scraping

Automatic download of information from websites.

Advanced Programming Interface (API)

Programming interface containing a set of rules and protocols that enable the communication between programs and with databases.

- **Manual input (e.g., transcription):** This is only possible with considerable time expenditure and is therefore not considered further as data access in this chapter.
- **Web scraping**: Web scraping automatically downloads publicly accessible content from previously specified websites. However, we do not opt to use this method due to legal, ethical, and privacy concerns (Gold & Latonero, 2017; Krotov et al., 2020).

11.6.2 Open and propriety data formats

There are different file formats that vary in terms of their openness and longevity that can be used to store text data as well as its metadata.

According to Charalabidis et al. (2018), **open data** includes all digital data that can be accessed freely and free of charge in online archives and repositories and is available in a machine-readable, non-proprietary file format. **Machine readable** means that the data is structured (see Section 11.6.3) in a way that allows programs to parse, analyze, and use the data without human intervention. However, open data can also be subject to access restrictions, for example if personal rights are affected or copyright is potentially infringed. Open data formats are data types that can be used without legal or technical restrictions. **Proprietary data** formats are data formats that are protected by copyright and can only be opened using manufacturer-specific software. Table 11.3 provides an overview of both types of data formats and their areas of application.

Longevity means that data and all associated, documented work steps, file types, program code, statistical evaluations, trained models, and metadata can be retrieved for as long as possible. Usually, these files are stored in a repository that assigns a persistent ID (**PID**) and a digital object identifier (**DOI**). Without a PID/DOI, digital research data is usually no longer retrievable after a few years, for example because websites linked to research projects on which the data is stored are no longer accessible (Federer, 2022; Vines et al., 2014).

Table 11.3 Examples of Open and Propriety Data Formats

	Open data format	*Propriety data format*
Table	csv; tsv (Comma/Tabulator Separated)	xls (Microsoft Excel Spreadsheets)
Text document	odf (Open Office)	doc (Microsoft Word Dokument)
Program code	py (Python), R (R File)	dta (Stata), sav (SPSS)
Pictures	png (portable network graphics); svg (scalable vector graphics)	psp (Photoshop), bmp (Microsoft Windows bitmap image file)

11.6.3 Structured, semi-structured, and unstructured data

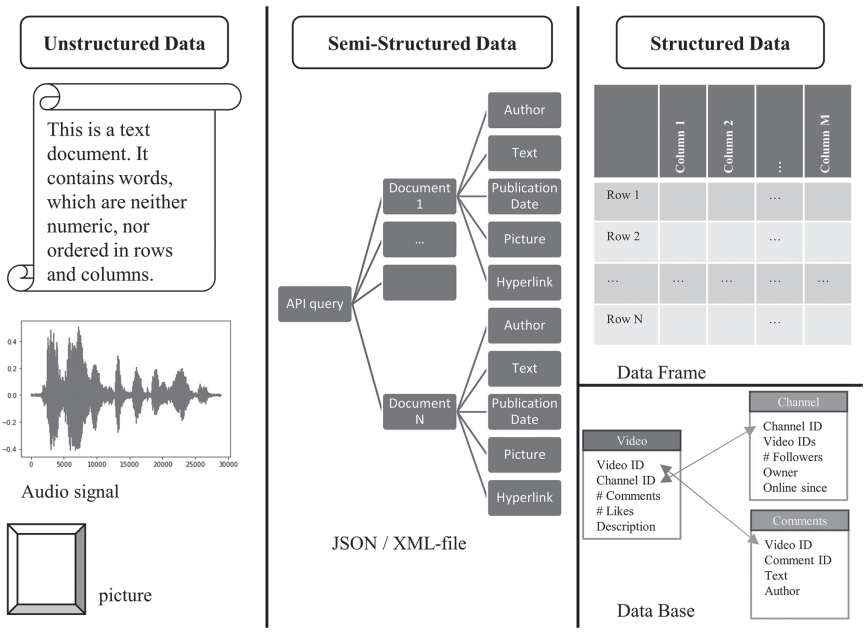
The data we encounter or produce throughout our research endeavor can be organized in structured, semi-structured, and unstructured form.

Structured data is arranged in a clear manner, for example in tables or relational databases based on tabular structures. Tables consist of entries or observations (rows) and variables (columns, e.g., identifier/number of the observation, age, income, gender of the respondent). Databases can contain several tables with identifiers (so-called primary and foreign keys). These keys make it possible to merge several tables.

Unstructured data, on the other hand, is not available in any such predefined structured format. Typical examples are free texts, such as answers to open questions or transcripts of various types of interviews, for example guided interviews. Consequently, we must convert unstructured data into a structured data format to perform automated content analysis.

Semi-structured data is positioned between these two extremes. They structure data hierarchically, for example with the help of tags, without specifying a strict relational structure as in databases. XML or JSON data are examples of semi-structured data. Figure 11.6 depicts the data types with examples separately.[8]

Figure 11.6 Different types of data. The depicted audio signal on the left originates from https://www.tensorflow.org/io/tutorials/audio (last retrieved November 11th 2024).



11.7 Where to find help

Since all steps of our workflow, including data collection, data preparation, data analysis, model training, visualization, and (partly) interpretation are carried out in Python, we also need to know where we can find help online and ask questions in case of doubt. This is even more important as this text is only an introduction to a small part of data analysis and programming in Python. We can get help directly in the Python programming environment, but you may also find help on online platforms, most notably on GitHub and Stack Overflow.

GitHub is a collaborative online platform where users can upload, document, and modify program code. A package is a collection of commands and functions that you can use to perform statistical calculations, process your data or create graphics. The core component of GitHub is shared version control of the program code. This means that every change in a code is documented, and different versions (on which different programmers are also working) can exist in parallel (Cosentino et al., 2017). This not only allows you to understand how the packages are structured and which programming code forms the basis for these packages, but also provides examples and assistance to facilitate your independent work with Python. This also applies to the packages that are used in the following chapters.

StackOverflow is a forum in which questions about programming languages (Python, R, C#, C++, JavaScript), packages (e.g., Pandas for data processing in Python) and statistical methods are discussed. Occurring problems and errors are uploaded with sample code and its potential, erroneous output. The answers to the questions and problems also contain code examples that can be used to resolve the original problem (e.g., the dataset cannot be loaded correctly, data cleansing does not work).

This is even more important, as we will encounter errors during programming, some of which are more obvious, some of which are less so. In order not to be discouraged from pursuing automated content analysis, the ability to understand errors, and to ask for help are most important. Even if this may be frustrating at first, keep in mind that it is okay to make mistakes, and that mistakes facilitate the learning procedure. Even deep neural networks make mistakes, and if they are modeled after our brains with the ability to correct errors, why shouldn't we be able to learn from making mistakes and improve our ability to conduct research?

Notes

1 Despite their widespread use from the 2010s onwards, the first artificial neural networks that sought to imitate the human brain, were proposed in 1943 by McCulloch and Pitts (1943).

2 According to Kitchin and McArdle (2016), big data has four dimensions: (1) volume = data volume is too large to be analyzed with conventional computers; (2) variety (of data types); (3) velocity = high rate of increase in data volume, coupled with increasing

complexity of the internal reference structure; (4) veracity = high variability of data quality.

3 Such as convolutional neural networks, recurrent neural networks, adversarial neural networks, long short-term memory (LSTM) networks, or transformers.

4 There are plethora of activation functions, but according to Mercioni and Holban (2020) and Rasamoelina et al. (2020), the sigmoid, arctan, ReLu, PreLu, leakyReLu, Elu, SeLu, and softmax are the most common functions. We will introduce the ReLu, sigmoid, and softmax when we train our artificial neural networks. The sigmoid is shaped like an S, whereas others, like ReLu, are piecewise-linear, meaning that its function contains at least two sections with different 'lines'.

5 There more layers, such as convolutional, pooling, and flatten layers. These, however, are employed in convolutional neural networks which are used to find patterns or recognize objects in images.

6 For the interested: The formula of each learning step is: $\Theta^{\text{next step}} = \Theta - \eta \Delta f(\theta)$, in this notation Θ (Theta) is a parameter, η (Eta) is the learning rate, $\Delta f(\theta)$ is the gradient, or: the rate of change of an error function in regards to the parameter.

7 There exist many different programming paradigms, for example object oriented programming, declarative programming, functional programming, procedural programming, or pipeline programming to name a few. We do not suggest that other approaches are worse in our opinion. We are only of the opinion that literate programming is most aligned to the way in which social scientists structure and publish the findings of qualitative inquiries.

8 We also retrieved and downloaded comments and video-metadata using the YouTube-API. The YouTube-API uses a JSON-format, which is structured like a tree.

12
GETTING STARTED WITH PYTHON

12.1 Installing and setting up Python and Anaconda

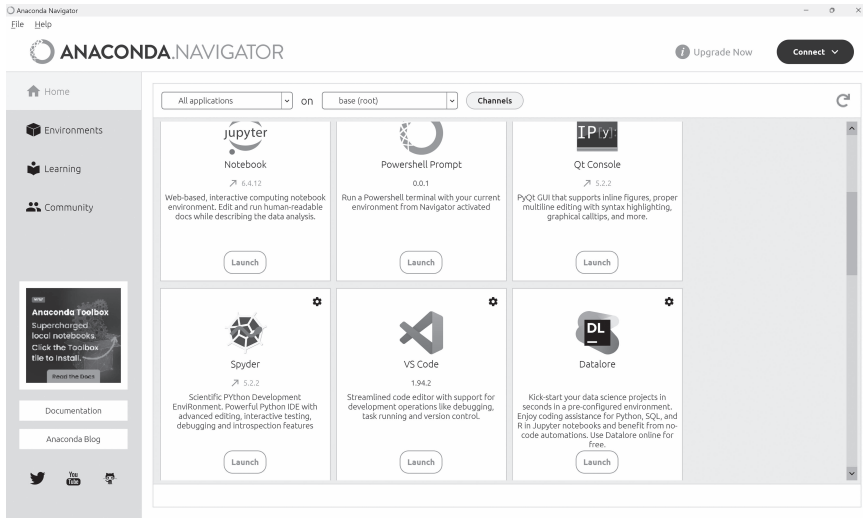
Now it is time to get more into the technical details and to acquire the knowledge and skills needed to carry out automated content analysis in **Python**.[1] The first step to take is to install a software distribution, and a development environment with the ability to run Python. In the former case, we opt for **Anaconda**; in the latter case, we prefer Spyder. Anaconda is used to create encapsulated **working environments** and for **package management**. We can use Anaconda for free for non-commercial purposes, such as research or education. A **package** is a collection of task-specific, predefined functions, methods, classes, and variables. A **function** is a block of code, which only runs if explicitly called. A **method** is a function, which enables us to directly change a variable. A **class** is a sort of blueprint, which is used to create objects in programming. As it is associated with object-oriented programming,[2] we refrain from using it for now. Lastly, a **variable** is used to store data, for example an imported excel spreadsheet, a csv-table, a text document, a list, or a number (among others). We will get back to these concepts in more detail in Section 12.4. An example of a (very important) package is **Pandas** (McKinney, 2011), which contains functions and methods that enable us to load, organize, analyze, change, and save structured data tables.

Spyder provides us with an interactive interface, including a programming editor, a variable manager, a console for direct programming and plotting, and tools for code analysis. It is short for **S**cientific **P**ython **D**evelopment **E**nvi**r**onment, especially suited for data analysis, including automated content analysis. We will get back to its structure after we guide you through the installation of Anaconda.

Now it is time to download the Anaconda distribution. To do so, you have to visit https://www.anaconda.com/download in your Internet browser and

DOI: 10.4324/9781003496397-15
This chapter has been made available under a CC-BY-NC-ND 4.0 license.

Figure 12.1 Anaconda download page with different installation options suitable for your personal computer or notebook.

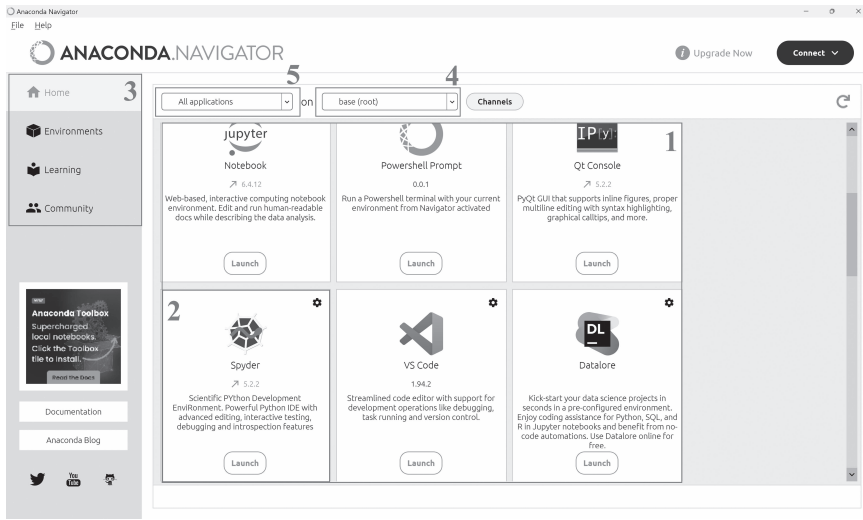


provide your email address to get a download link. After you click on the link sent in the email, you will be guided to the download page. As seen in Figure 12.1, you can choose the Anaconda version suitable for your operating system, either Windows, macOS, or Linux. After you download a suitable version, use the downloaded installer to install Anaconda.[3]

After installing Anaconda, you can either use the symbol placed on your desktop, use the program search window, or use a symbol pinned on your taskbar (usually at the bottom of your screen) to start the **Anaconda Navigator**. The Anaconda Navigator is a **graphic user interface** (GUI), which allows us to adjust all settings, applications, and content to our preferences. The first time Anaconda is initiated, the GUI looks like Figure 12.2. We begin with **the application tiles** (1). These show you all applications (including Spyder), which are either installed (indicated by the button 'Launch') or not installed (indicated by the button 'Install'). Each application is situated in an own application tile (2). The screenshot in Figure 12.2 might be different from your view due to the fact that the author of this section has already installed a number of applications.

On the left-hand side, we see the **Navigator pages** (3). Currently, we are at the Home page, where all apps are lined up and either ready to get launched or installed. Next, you see the Environments page. This is where all our working environments are listed, typically one per programming project. Here, we will also find the buttons and options which allow us to use or create our own programming environment. Furthermore, you will find different documentations and tutorials on the Learning page and links to community pages, weblogs, and help forums on the Community page.

Figure 12.2 Anaconda Navigator menu after the first initialization.

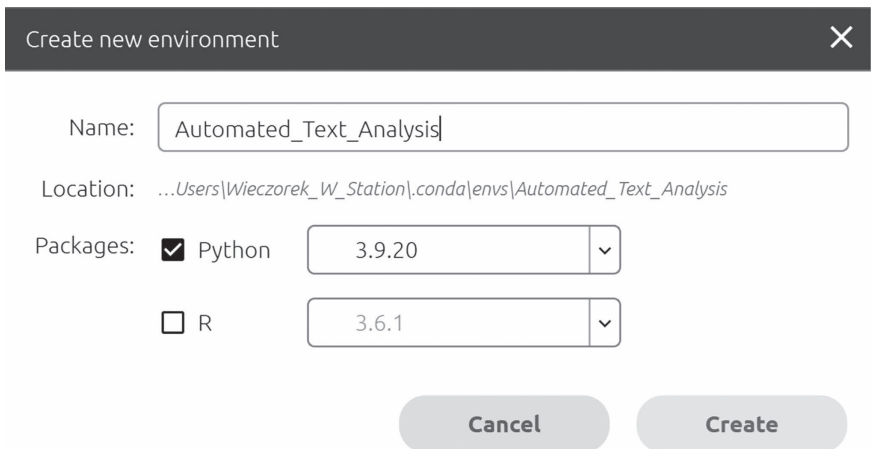


Just above the application tiles on the Home page, you see the **Environment menu** (4). This is a drop-down menu with the text 'base (root)'. This is where we can change our working environment, given that we created our own environment. Keep in mind, that the 'base (root)' directory is our most important working space, from which we can also install packages (see Section 12.2). However, as these packages sometimes have conflicting dependencies – meaning that they are built upon different versions of the same packages – you can efficiently render your base environment unusable and cause Spyder to crash each time upon launching. This is why we strongly recommend to set up a new working environment each time you start a new programming project. The second one is that you are able to keep track of the packages needed to run your code without errors and to reproduce your findings in the future. On the left side of (4), we see another drop-down menu (5), which allows us to select which applications tiles (1) are displayed. The options provided in the drop-down menu (5) include 'All applications', 'Installed applications', 'Not installed applications', and 'Updatable applications'. This alters the view insofar as you gain an overview of what applications you can update, use, or install.

Let us now switch to the Environments page using the navigator pane (3) seen in Figure 12.2 and set up our own working environment. This environment will contain all the packages necessary to conduct automated content analysis or, in our case, geometric data analysis, network analysis, topic modeling, and sentiment analysis. After switching to the Environments page, as seen in Figure 12.3, we will encounter a slightly different layout compared to the Home page. In the middle, we see the **environment list and environment search box** (1). Your view may differ insofar as the author of this section has already set up a number of different environments for previous research and programming projects.

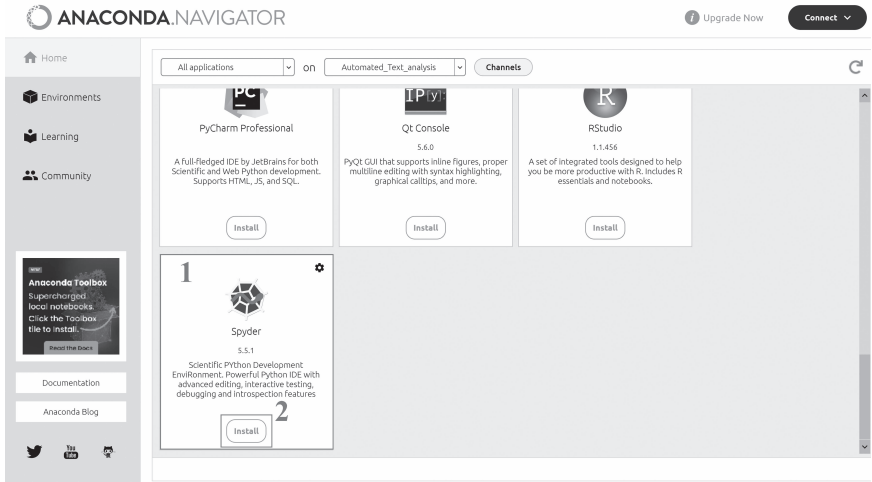
Figure 12.3 Navigating the Anaconda Navigator environment and creating a new programming environment.

Figure 12.4 The 'Create new environment' window.



At the bottom, we see **buttons for managing our environments** (2). The first button, 'Create', allows us to open a dialogue window, as seen in Figure 12.4, in which you may set the name of your working environment, whether you want to install sPython or R, and which version you wish to install. The second button, 'Clone', enables you to create an exact copy of the current working directory.[4] The next button, 'Import', allows you to load an environment created by another person, may it be a fellow student, an advisor,

Figure 12.5 Installation of Spyder after we established our environment.



or a programmer. The 'Backup' button enables you to store a copy of your working environment to the Anaconda Cloud, given that you have created an Anaconda account. And lastly, the 'Remove' button deletes the current environment. This is sometimes necessary, since you must install and download Spyder, the packages needed to conduct the (automated text) analysis you wish to accomplish, and sometimes the models (e.g., Deep Neural Networks) separately for each environment.

Many of these packages (such as SpaCy or TensorFlow), accompanying models (e.g., BERT), and text corpora are huge, and if space on your hard drive is limited, it is sometimes necessary to delete environments to create enough space on your hard drive to conduct automated content analysis. On the right side of Figure 12.3, you see a **list of packages** (3). In the figure at hand, we see the installed packages, a short description per package, as well as the version of each package. On top, we see the **Package Display dropdown menu** (4). It allows us to select if the Anaconda Navigator should display installed packages (Installed), not (yet) installed packages (Not Installed), packages which can be updated (Updatable), packages that are selected to install/uninstall (Selected), as well as all currently available packages (All).

To set up our new environment, we press the 'Create' button (2) in Figure 12.3. A new dialogue window, depicted in Figure 12.4, opens. Here, we enter the name of our new environment in the first dialogue box (here: Automated_Text_Analysis). Afterwards, Anaconda asks us if the new environment should use Python or R as programming languages. We tick the box next to Python and then select the most recent Python version. While writing this book, the newest Python version was 3.9.20. We then press the 'Create' button. And wait until Anaconda has finished creating a new environment.

After installing Anaconda and setting up the environment, all that remains for now is installing Spyder. To do so, we head back to the 'Home' page and scroll through the application tiles until we find the Spyder tile (1). We then press the Install button (2) to start the installation process. After a few moments, we should be able to launch Spyder. This concludes our first step of setting up a suitable working environment for automated content analysis.

12.2 Installing the packages necessary for automated content analysis

Let us now continue and install the packages that contain the functions (the definition is provided in Section 12.4) necessary to conduct automated content analysis. There are basically three ways to install or update packages. Firstly via the Environments page, secondly by using the 'terminal', and finally by using so-called 'magic-commands' in the Spyder console. We will focus on the former two for package management and version control.

Let us start with installing our packages using the Environments page in the Anaconda Navigator. Here, we have to make sure that we selected the correct environment from the environment list, in our case 'Automated_Text_Analysis' (1) as depicted in Figure 12.6. We then use the Package Display dropdown menu (2) and select the 'Not installed' option to see a list of packages ready to be installed. To find the right ones amidst the hundreds of packages available, we use the search bar on the top right corner (3) and type the name of the package we wish to install. For demonstration purposes, we typed 'pytorch', which is a library for conducting deep learning in Python. Now, we select the package

Figure 12.6 Installing packages using the Anaconda Navigator.

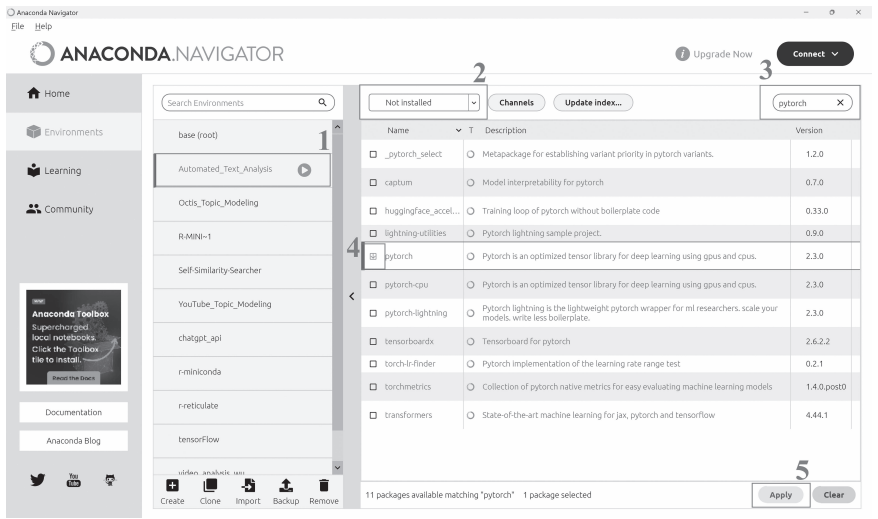
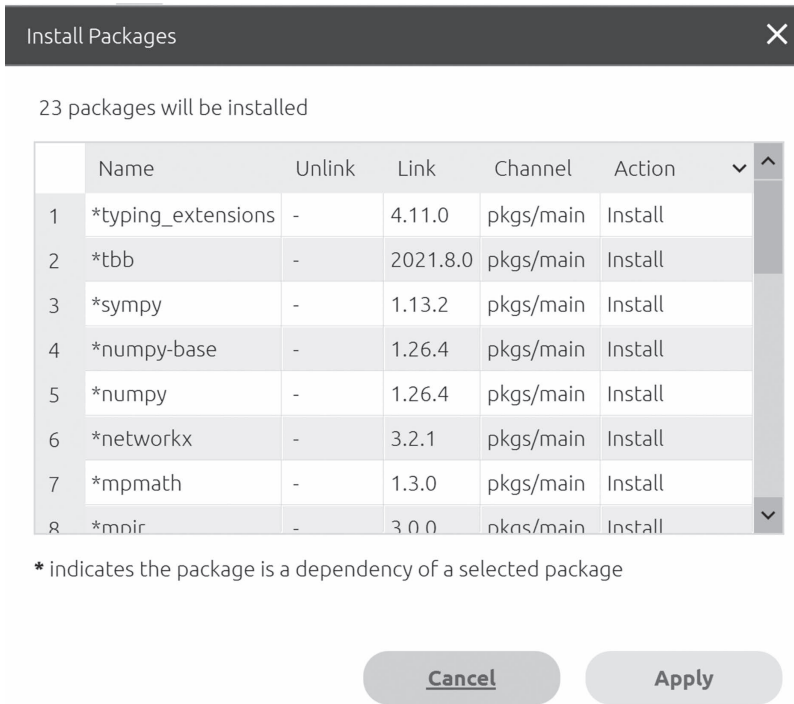


Figure 12.7 Exemplary package installation dialogue.



we wish to install by ticking the box left from the package (4) and then press the apply button (5) in the lower right corner.

Afterwards, a new dialogue box opens which lists all packages that will be installed along pytorch (Figure 12.7). These packages are the dependencies, on which the packages are built that we want to install. We now need to press the 'Apply' button and proceed with installing the next packages, including all needed to conduct automated content analysis. These packages are listed in Table 12.2.

The second option to install the packages is using the terminal, also called 'Shell'. As you see in Figure 12.8, we press the button with the triangle (1) which resembles a Play button in a media player, and choose the option 'Open terminal' (2) from the box. Afterwards, the terminal initiates and opens a dialogue box (see Figure 12.9). As you see in the command line in the Terminal, we are going to use pip as installation device. The pip package is Python's package installer and is also useful as a package management tool as it checks the version of the packages to be installed as well as their dependencies. It also tries to automatically solve dependency issues and warns us if such issues exist. As we see in Figure 12.10, these are colored red after we install a package. The basic

Figure 12.8 Opening the terminal of our selected working environment.

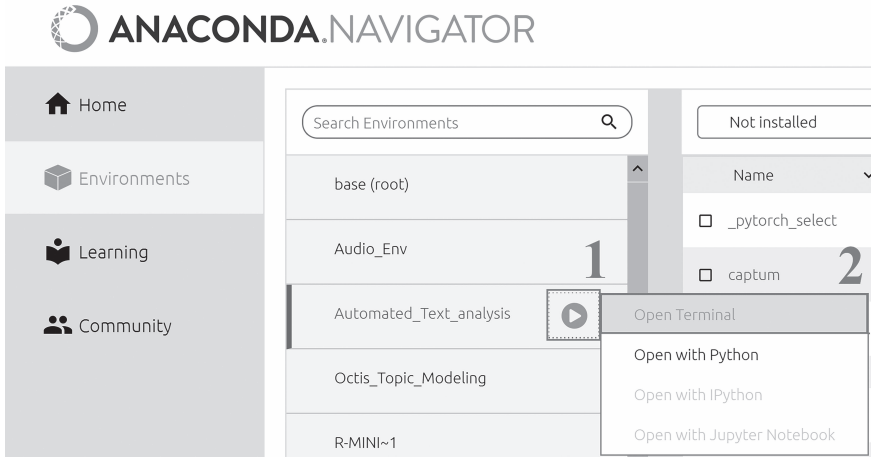
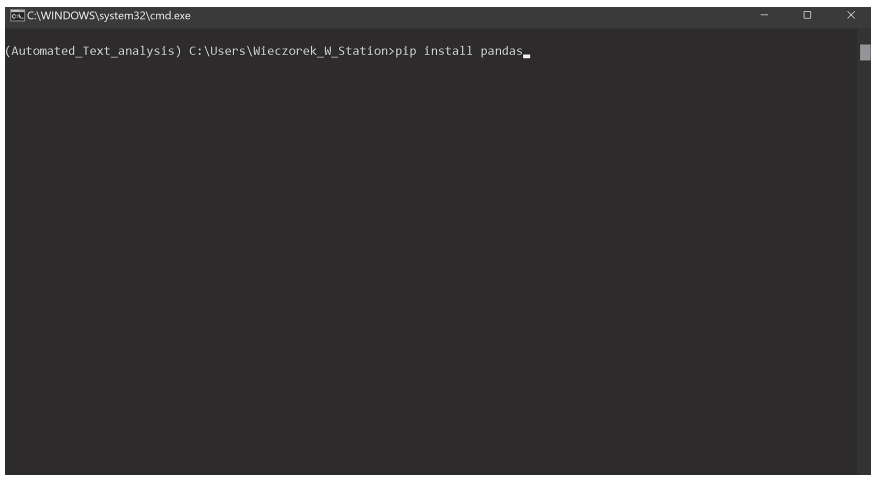


Figure 12.9 Command line with the pip command to install the package Pandas.

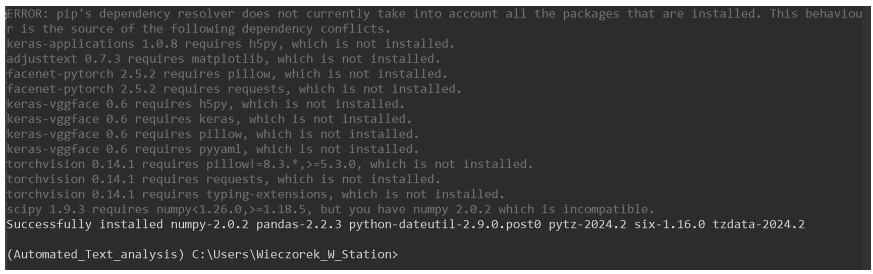


command to install a package is `pip install [PACKAGENAME]`. However, you have to exchange `[PACKAGENAME]` by an actual package name. In our case, we install `pandas`, which is a package for opening, editing, and analyzing structured tables. After doing so, we just press the Enter button and wait for the package to be installed. However, there are also commands which enable you to install specific versions of a package, to update packages, to uninstall packages, or to check the version of the packages in your working directory. Table 12.1 provides you with an overview of the pip commands the author of

Table 12.1 Overview of Pip Commands for Package Management

Pip Command	*Description*
`pip install [packagename]`	Installs a package
`pip install [packagename]==version`	Installs a specific version of a package
`pip uninstall [packagename]`	Removes a package from the working space
`pip install -upgrade [packagename]`	Upgrades a package to the newest version
`pip list`	Lists all packages currently installed in the working space along with their version

Figure 12.10 Output and warnings after Pandas is successfully installed.



this chapter most often uses. A comprehensive list is available using the following link: https://pip.pypa.io/en/stable/cli/.

Let us now look at the packages we need to install to conduct automated content analysis listed in Table 12.2. We also provide the version numbers of the packages we used in this book. To reproduce the analyses in the following chapters, you can, for example, use the second method described previously and type, for example, `pip install spacy==3.7.6` to install `spaCy` in the correct version to your workspace. In order to ensure reproducibility of your analyses and that others will install the correct packages (including dependencies), we must at least provide a list of packages used along with the versions used. Better yet, we may write a requirements.txt. We will come back to this after we introduced the packages.

We begin with introducing the `os`, and `re` packages, which are both Python standard libraries and do not need to be installed separately. The `os` package provides functions for communication between your Python environment and your **o**perating **s**ystem (hence the abbreviation `os`). This includes changing, creating, and deleting files or file directories on your hard drive. The `re` package is for working with **r**egular **e**xpressions. A **regular expression** is a sequence of characters that defines a search pattern. We will use the `re`-package in conjunction with `os` for file management, and to search for, modify, and exclude certain patterns in the text which we do not wish to include in our analysis (e.g., hyperlinks).

Table 12.2 Packages Used to Conduct Automated Content Analysis (With Version)

Package name	*Version*
`os`	3.13.2
`re`	2.2.1
`Pandas`	2.2.3
`Datasets`	3.0.1
`Openpyxl`	3.1.5
`numpy`	1.26.4
`sklearn`	1.5.2
`prince`	0.13.1
`Networkx`	3.3
`python-louvain`	0.16
`spacy`	3.7.6
`Langdetect`	1.09
`genism`	4.3.
`BERTopic`	0.16.3
`vadersentiment`	3.3.2
`Transformers`	4.44.2
`Tensorflow`	2.17.0
`TQDM`	4.64.1
`matplotlib`	3.9.2
`seaborn`	0.13.2

Now, we will focus on the packages that need to be installed either by using the Anaconda environment page or the `pip` commands in the terminal. We begin with installing `Pandas`, which derives from **pan**el **data (McKinney, 2011, S. 1).** It is a package suitable for handling data frames. We will use it to load, transform, analyze, and save both text corpora and metadata. The `openpyxl` is a small package that enables us to read and write Microsoft Excel files. Although we discourage using Excel files, there is a chance that we encounter such files and need to import them into our working space.

`numpy` (Oliphant, 2006), short for **Num**erical **Py**thon, is a package that specializes on matrix algebra and statistics. Next in line is `Scikit-learn` (Pedregosa et al., 2011), a powerful library designed for scientific computing and machine learning.

Let us now turn to the packages for the relational analysis of – relatively small – text corpora. First is `Prince`, a package that enables us to conduct geometric data analysis, including multiple correspondence analysis and multiple factor analysis. These methods enable us to simultaneously include text features (tokens or n-grams), extracted topics, sentiment scores, and metadata into our analysis. Usually, this allows to paint a more complete picture of the social dimensions underlying the analyzed texts. We will make use of this package in Section 15.2. Furthermore, we will use `NetworkX` (Hagberg et al., 2008) to

translate our text corpus into a network graph to calculate centrality measures (e.g., how important and connected certain words are throughout the texts) and to visualize semantic networks. The `python-louvain` package enables us to detect 'communities' in our network graph. A community is a part of the network, in which the nodes (in our case, words), are much more interconnected while having only a few connections to other parts of the network. In our case, we will interpret a community as a topic, which arises from words that are used together systematically throughout the text corpus.

The package `spaCy` (Honnibal et al., 2020) is specialized in natural language processing. It has its own pipelines (= a pretrained language model, a number of consecutively called functions that produce an output) for (a) text preprocessing and cleaning, (b) named entity recognition (e.g., if you wish to identify a location, organization, or person in a text), (c) sentence detection, and (d) semantic analysis. This package is extremely helpful to clean your text corpus, as we will see throughout the following sections. The `langdetect` is a small package which helps us to identify the language of a document. This package is handy if we wish to keep texts written in a certain language, for example English, and to drop texts written in different languages. The `gensim` (Řehůřek & Sojka, 2011) is a package which contains different topic modeling algorithms, including Latent Dirichlet Allocation (LDA) (Blei et al., 2003), Word2Vec (Mikolov et al., 2013), Doc2Vec (Le & Mikolov, 2014), or Latent Semantic Indexing (Deerwester et al., 1990). Besides these algorithms, `gensim` provides us with evaluation metrics which helps us to decide on which topic models to choose for further, qualitative interpretation. We will use `gensim` extensively in Section 17, where we will conduct topic modeling. `BERTopic` (Grootendorst, 2022), which we will use in Section 18, combines topic modeling with deep learning models (Transformers). We will use `BERTopic` to extract context sensitive topics and to make use of different pretrained large language models to assist us in interpreting the extracted topics.

Leaving the domain of topic modeling and entering the domain of sentiment analysis, we will first encounter the `VADERsentiment` package, where VADER is short for **v**alence **a**ware **d**ictionary and s**e**ntiment **r**easoner (Hutto & Gilbert, 2014). It follows a dictionary- and rule-based approach to assign sentiments to texts. This means that `VADER` uses a sentiment lexicon, in which different words are assigned sentiment scores. Furthermore, it contains rules that specify how to modify the sentiment score of words listed in our sentiment lexicon if other types of words, so-called negators and amplifiers are present in a text. Finally, it calculates the probability that a given text yields either a positive, negative, or neutral sentiment. The `Transformers` (Wolf et al., 2020) package contains an interface to many pretrained deep neural networks, including large language models. In our case, we will use **b**idirectional **e**ncoder **r**epresentations from **t**ransformers (or, in short, BERT) models to detect and predict sentiments in our text corpus. Complementarily, we will use `Tensorflow` to train and fine-tune artificial neural networks.[5] Fine-tuning means that we will use a pretrained deep learning model and tell it to refocus on certain aspects of our text so

that it will be able to specifically find and assign positive, negative, or neutral sentiments in our text corpus. However, it is also possible to use these models to detect different categories and phenomena, such as populism (Erhard et al., 2024). The `TQDM` package enables us to display a progress bar, which is helpful to check if the Python code is executed. Lastly, we will employ matplotlib and seaborn for data visualization. A list of all packages and the specific versions used throughout the following sections is provided in Table 12.2.

12.3 The Spyder interface

Let us now take a quick look at the Spyder interface depicted in Figure 12.11. The color scheme you see in this figure is called "Spyder Dark" and can be selected in the Preferences (7) → Appearance Menu → Syntax Highlighting Theme. We begin with the **Editor** (1). The editor is an interactive text editor which enables us to write Python code and execute either selected code lines, sections of the code, or the whole code. The code written in the editor can be stored onto or loaded from your local hard drive. We will write all the code in all subsequent chapters in the editor and run it from here. Among others, the shortcuts to run codes are listed in Table 12.3.

On the bottom right, we will find the **IPython console** (2) containing the console panel itself and a "History" panel. The console runs the **kernel**, which interprets our commands and processes them in a form that allows Python to do the calculations associated with the code. The console also allows us to write and execute code directly, indicated by a bright, green `In []`:-statement. The number situated within the brackets shows the number of the current

Figure 12.11 The Spyder interface.

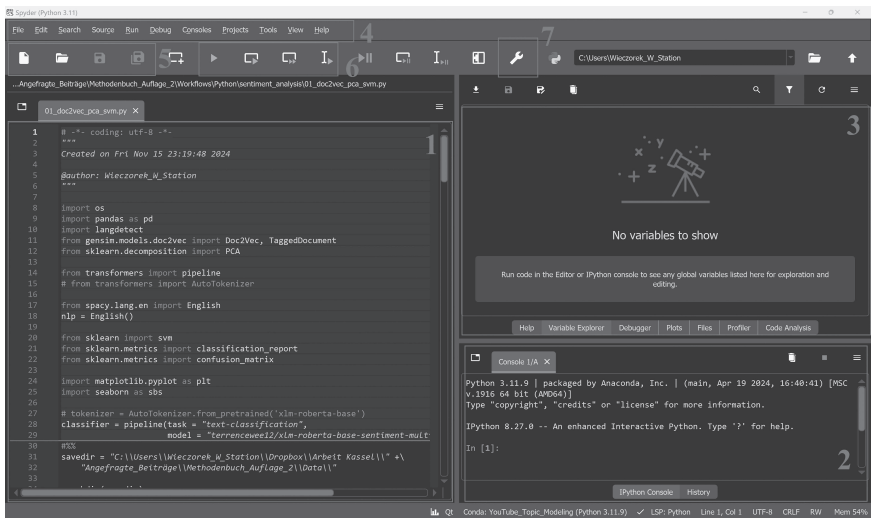


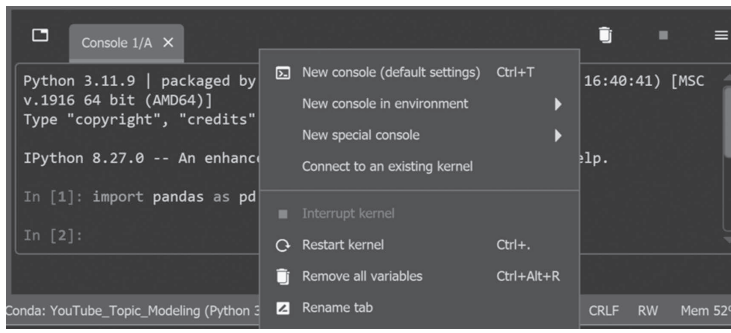
Table 12.3 Selected, Useful Shortcuts for Working With Spyder

Shortcut	*Meaning*
CTRL + N	New editor window
CTRL + W	Close editor window
CTRL + S	Save the code written in the editor
CTRL + O	Open a previously saved file
CTRL + A	Select all
CTRL + F	Search text
CTRL + R	Replace text
CTRL + C	Copy
CTRL + X	Cut
CTRL + V	Paste
CTRL + Z	Undo
CTRL + Shift + Z	Redo
Tabulator	Indentation
Shift + Tabulator	Remove indentation
CTRL + 1	Comment
CTRL + 4	Block comment (to highlight outcommented parts of the code/ section introductions)
CTRL + 5	Remove block comment
CTRL + I	Opens a help page containing information on the highlighted command in the console or editor
CTRL + .	Restart Kernel
Tab	Autofill functions and methods
CTRL + T	Open new console
CTRL + ALT + R	Remove all variables
F1	Show documentation
F5	Run the whole code
F9	Run selection or current line
CTRL + Enter	Run a section of the code, usually subdivided by `#%%`

command, so it enables us to keep track of how many commands we have used so far. After execution, it prints the output, which is indicated by a red `Out []`:-statement. The output may contain novel variables, calculations, but also errors and warnings. Such errors and warnings provide hints what went wrong and how to possibly fix it (see also Section 11.7 where to look for help if you cannot solve the problem by yourself). The **History** panel, in contrast, stores the most recently executed functions and lines of code.

The IPython console panel also allows us to open new consoles by right clicking on the empty panel on top and choosing `New Console (default settings)` from the menu (see Figure 12.12). This enables us to run different Python scripts, which is useful, if we have a fully functional script (e.g., that enables us to download data via an API) and if we wish to write, test, and improve another script (e.g., data preparation, descriptive analysis of the text corpus) simultaneously. If we have multiple environments with, for example,

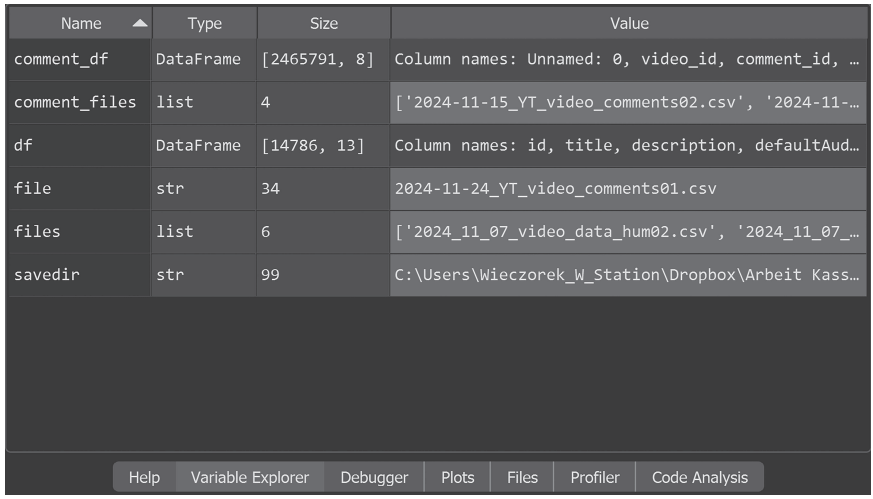
Figure 12.12 Console menu which opens after right clicking on top of the console window.



incompatible package versions (this could happen, e.g., if we want to use different types of deep learning packages or topic modeling packages at once), but we need to execute calculation steps nonetheless, we may choose the second option, `New console in Environment`, and start a new Kernel using a different Environment. The third option allows us to open special consoles, for example a Cython console, which bridges between Python and the programming language C. The next option, `Connect to an existing kernel`, allows us to run code on an external Kernel, for example on a server. This is useful if your institution provides you with access to a server to calculate large amounts of data. The `Interrupt kernel` option forces Python to stop the code that is currently executed. It stops, for example, long-running processes, infinite loops (we will come back to this concept in Section 12.4.8), cancels code initiated unintentionally or stops erroneous code from being executed. The next option, `Restart kernel`, allows us to clear the session, remove all variables, and start the calculations anew. This is useful if our code crashed due to code errors or memory errors – for example when you train a deep neural network with billions of artificial neurons on a large text corpus comprised of millions of long texts. The option `Remove all variables` enables us to clean the working space insofar, as we delete all variables, such as data frames, lists, numbers, arrays, or texts (see Section 12.4.2 for an overview and explanation of different data types) and to clean the memory of our computer. Finally, the `Rename`-option enables us to change the name of the current console.

On the top right of our screen, we see the **utility panels** (3), including the 'Help', 'Variable Explorer', 'Debugger', 'Plots', 'Files', 'Profiler', and 'Code Analysis' panel. As the name suggests, the 'Help' panel shows the documentation of a function or method and provides information on different data types. You can initialize the help either by right clicking on the function and choosing Go to Documentation from the menu, or using the shortcut CTRL + I, as listed in the shortcuts in Table 12.3. Usually, it explains not only a function or method itself but also the options for adjusting the function. Furthermore, examples are

Figure 12.13 Variable explorer with defined variables.



provided to explain how the function works and usually what output to expect. Next to it is the `Variable Explorer`. As seen in Figure 12.13, it contains information on all variables defined within the current working space. This includes the name of the variables, the data type, the size, for example the number of observations (rows) and variables (columns) of a data frame or matrix, and the values, namely information on the shape and type of data stored in the respective variables. Next is the `Debugger`, which allows us to view and correct errors we included in our code, whereas the `Plots` panel stores all figures we created using `Matplotlib` and `Seaborn` unless defined otherwise in the settings (7). The `Files` panel contains a browser, which allows us to navigate through the folders of our hard drive, change the working directory, and to save, copy, load, and remove data stored on the hard drive. The `Profiler` runs every function of your code and calculates the time needed for each function to be executed. This allows us to identify parts of our code that increase the computing time and thus could be optimized. Finally, the `Code Analysis` panel allows us to check the code for inconsistencies, errors, and possible violations of coding standards. This allows us to make our code more readable and less prone to errors.

On top of Figure 12.11, we see the **Menu Bar** (4). As the name suggests, the menu bar contains menus with different functions. Many of the functions stored in these menus can also be called using the shortcuts in Table 12.3. The first menu on the farthest left of the menu bar is the **File menu**. It allows us to create a new code file, to open an existing code file from our hard drive, to save or close the current or all open code windows (seen on top of the editor), to print the currently opened code, to restart or to close Spyder. Next is the **Edit** menu. Here, we find options that enable us to undo the last coding steps consecutively, to redo changes in the code, to cut, copy, and paste lines of code, to

select all the code, to include comments in the code or block comments (e.g., to highlight important code or notes), or to indent or unindent code, which is relevant if we write loops or conditions (`if`, `elif`, `else`). This is relevant as Python uses indentation to identify the lines of code that should be executed within a flow control sequence (see Section 12.4.7), namely if a condition is met or which functions should be called in succession within a loop. Next is the **Search** menu. It provides us with functions to search and replace text/lines of code. The **Source** menu allows us to adjust the editor view to our needs. For example, it enables us to scroll past the lines of written code, which is useful if we write many lines of code. We can also tell Spyder to show warnings, blank spaces, indentations, or underline errors and warnings in the editor. These adjustments are designed to help us structure our code and to reduce errors.

Next in line is the **Run** menu. It provides us with functions to run a single line of code, code blocks (separated by **#%%**), or the whole code. These are also listed in the tool bar (5). Right of the Run menu is the **Debug** menu, which contains functions that help us to find and eliminate errors in our code. The **Console** menu includes the same options for opening and closing new consoles we introduced earlier in regards to the IPython console (2). The **Projects** menu enables us to create, change, and delete new programming projects. It allows us to automatically save and restore the files we opened, provides version control functionalities, and makes it easier to use version control systems like Git. In the **Tools** menu, we will find functions to customize the preferences (such as appearance, Python interpreter), or to manage the remote connections, if you wish to run your code on an external server. Furthermore, we can customize the appearance of Spyder in the **View** menu, for example split the editor (1), to change the overall layout, or to hide panes such as the Debugger or Variable Explorer in (3). Lastly, the **Help** menu provides access to a comprehensive Spyder documentation, a list of all shortcuts, tutorials, and the possibility to report errors.

Finally, we see the **tool bar** situated between the editor and the menu bar. Here, we focus on the buttons to **load and save the Python code** (5), to **run the code** (6), and to open the **Preferences menu** (7). The **Save** buttons (5), explained from left to right, enable us to create a new file (page with folded edge), to load a preexisting Python code file (folder), to save the current code file (floppy disc), and to save all currently open Python code files (two stacked floppy discs). All buttons with a lime green symbol resembling the "play" button in a media player (6) are used to run different parts of the code. The first, which most closely resembles a "play" button is called **Run file** and used to run the entire code at once. We will use this button or the according shortcut to test if our code runs in its entirety. The second, **Run current cell**, executes all functions that are encapsulated in a cell. Again, a cell is a part of your code, usually a few up to a few dozen lines of code, which is encapsulated by **#%%**. The encapsulation should be chosen to reflect distinct portions of the code. The next one is **Run current cell and go to next one.** This is an extension to the last function, which runs the lines of code contained by a cell in a first step. In the second step, the first row of the subsequent cell is selected in Python, and the display in

the editor is adjusted so that the new cell is visible. The last one with the vertical line and the Play button, called **Run Selection or Current Line,** allows us, as the name suggests, to run either a single line or a selection of code.

12.4 Introducing relevant Python concepts

Now, it is time to introduce the Python building blocks and relevant concepts needed to conduct automated content analysis successfully. We will describe these building blocks and provide code examples in the following subsections and start with a quick introduction into variables, how to assign values to variables, and how to name variables in a concise manner. We then continue with introducing different data types, before we introduce operators, data structures, and the distinction between functions, methods, and variables. Afterwards, we introduce flow control sequences, including conditional statements as well as loops, before we introduce list comprehensions. We then proceed with regular expressions and close with a summary of what we learned so far, and we also provide an overview of the possible applications of these concepts in the context of automated content analysis.

12.4.1 Variables and naming conventions

On an abstract level, a **variable** stores a value under an assigned name during a working session. We assign values with equal marks `=` (see Section 12.4.3). A variable can store anything if we have enough random access memory (RAM) at hand. It can store single numbers, a sentence, texts, a text corpus, a data matrix (with columns and rows), even trained machine learning models. But beware, a variable is always overwritten when new values are assigned, as it is "a unit of storage that can be modified during program execution" (Butterfield & Ngondi, 2016, p. 584). For example, if we assign `text = "This is a sentence"`, and afterwards `text = "This is another sentence"`, the latter sentence will be stored instead of the former in the variable `text`. Furthermore, variables are case sensitive, meaning that a variable named `text` will be different from a variable named `Text` or `TEXT`. Code box 12.1 exemplifies how we assign values to variables.

```
text = "This is a sentence."
Text = "This variable is not the same as text!"
TEXT = "This is also a sentence, but a different one, as Python
is case sensitive."
Variable = 1
List_Variable = ["This is a list", "it contains a sequence of
elements", 1, ("a", "tuple")]
```

Code box 12.1: Variables and variable assignment

Despite the example depicted in Code box 12.1, we advise you to follow a naming convention for assigning names to variables. We do so, as employing a naming convention to improve code readability helps to understand our code in the long run, reduces the chances of accidentally overwriting variables and, by doing so, causing errors and frustration. For example, programming experts use variable names that are descriptive (= explain the content of the variable) but that are not too long (Wang et al., 2021). Besides using descriptive and short variable names, we advise you to choose variable names that are not too similar (e.g., `text_corpus1` and `text_corpus2`), as this might cause confusion during the coding process and may cause errors to occur during your automated content analysis. Furthermore, we advise you to use either the snake case naming convention or the camel case naming convention. A **snake case** separates each word of a variable by an underscore `_`, for example `variable_name`, `Variable_name`, or `VARIABLE_NAME`. As you see, you can uppercase or lowercase either the first character, the first character of each word, or every character as long as the variables you named follow the same pattern. A **camel case** is a naming convention in which either each word begins with an uppercase letter, or the first word of a variable name begins with a lowercase letter, and the following with uppercase letters. Examples of variable names, along with **comments** introduced by `#`, are listed in Code box 12.2.

```
# Snake case
Text_corpus
text_corpus
TEXT_CORPUS
# Camel case
TextCorpus
textCorpus
```

Code box 12.2: Examples of the snake case and camel case variable naming conventions

12.4.2 Data types

We will encounter different data types while working in Python, which can be stored in variables and behave differently when we use functions and methods (see Section 12.4.5). A **data type** is **a representation of elements** (e.g., numbers, characters), which is **linked to a behavior**, such as what happens if we add the values of two elements, with the **goal of abstraction** (Parnas et al., 1976, p. 150). In the case of numbers, addition simply means that numbers are added up, whereas character strings are glued together.

We start with **primitive data types**, such as numbers, a series of characters, information on whether a statement is wrong or right, or if a variable points to any data at all. These data types consist of only a single element. Afterwards,

we will proceed with data structures which store multiple elements including primitive data types or data structures. You can envision the difference between primitive data types and data structures (see Section 12.4.4) like the distinction between a T-shirt and a closet. You can store a T-shirt in a closet. Like a number, the T-shirt is a representation of a class (T-shirts), but a single, specific unit that can be stored in a manifestation of a larger structure (the specific closet, which is one of abundant closets in existence on our – still existing – planet).

Let's get back to primitive data types. A first category of primitive data types are **numeric types**. These include different numbers, such as **integers** (= a number with no decimal or fractional part), **floating point numbers** (contain fractions), and **imaginary numbers** (with a real and imaginary part). We will skip the last ones as they are typically used in the natural sciences to describe waves and oscillations (e.g., in audio-feature analysis or fluid dynamics).

A second category are **string literals**. String literals are a sequence of characters delimited by quotation marks, such as `"book"` or `"Don't look up!".` These quotation marks are required to distinguish strings from variables. At the same time, there are some characters with a special meaning within Python, such as the quotation marks. To tell Python that such special characters like " are part of a string, we must introduce them with a backslash \, or **escape sequence**. These include single and double quotation marks, the backslash, which is itself reserved for the escape sequence, the `\n` to indicate a new line, or the `\t` to indicate a horizontal tabulation. Table 12.4 provides an overview on these escape sequences. More will be introduced when we dig into regular expressions, which are used to create search patterns, for example if we want to look for certain words or character sequences in our texts.

Furthermore, Python also provides us with a **None** type. A None type simply represents the absence of a value. At last, there are the two Boolean types **True** and **False**. These indicate if a previous statement aligns with a condition (e.g., the statement "this is a book" is `True`, while "this is a whale" is `False` in regards to the book at hand). We will introduce logical operators to test whether a statement is true or false in the following Section 12.4.3. Code box 12.3 provides an overview of the primitive data types we encountered so far.

Table 12.4 Overview of Escape-Sequences Applicable on String Literals With Explanation

String with escape-sequence	*Meaning*
\'	Single quotation mark
\"	Double quotation mark
\\	Backslash
\n	New line
\t	Tabulator (useful for indentation)

```
1 # integer
3.14 # floating point number
"This is a string"
"This is a string with \n at least one \t escape sequence \\!".
None # None type
True # Boolean True
False # Boolean False
```

Code box 12.3: Overview on different data types

12.4.3 Operators

Operators are **special symbols that perform specific tasks** in Python. We distinguish between arithmetic operators, assignment operators, comparison operators, logical operators, membership operators, and bitwise operators. **Arithmetic operators** are used primarily to perform operations with numbers, such as adding, subtracting, etc. A special case is the addition (+) operator. In case of a string variable, the addition (+) operator can be used to chain strings together. An overview and examples of arithmetic operators are provided in Table 12.5.

Assignment operators (Table 12.6) are used to assign values to variables. The following table provides an overview, a brief explanation, the operators themselves, and the output our console will provide if we pass the variable `number = 5`. We can also use arithmetic operators in combination with the equals mark to simultaneously perform calculations and assignment. For example if we previously assigned `number = 5`, and type and execute `number += 3`, the value stored in variable `number` will be 8.

Comparison operators (Table 12.7) compare, as the name suggests, two input values. They test if a condition for a comparison is met, for example if a first input has a higher value than the second. It outputs `True` if the condition

Table 12.5 Arithmetic Operators With Examples

Operator	*Meaning*	*Example*	*Output*
+	Addition	`5 + 3`	`8`
		`"a" + "B"`	`'aB'`
-	Subtraction	`5 - 3`	`2`
*	Multiplication	`5 * 3`	`15`
/	Division	`5 / 3`	`1.6666666666666667`
**	Exponentiation	`5 ** 3`	`125`
//	Floor division	`5 // 3`	`1`
%	Remainder (modulus)	`5 % 3`	`2`

Table 12.6 Assignment Operators With Examples

Operator	*Meaning*	*Example*	*Output*
`=`	Assign	`number = 5`	`5`
`+=`	Add and assign	`number += 3`	`8`
`-=`	Subtract and assign	`number -= 3`	`2`
`*=`	Multiply and assign	`number *= 3`	`15`
`/=`	Divide and assign	`number /= 3`	`1.6666666666666667`
`**=`	Exponentiate and assign	`number ** 3`	`125`
`//=`	Floor divide and assign	`Number // 3`	`1`
`%=`	Remainder and assign	`Number %= 3`	`2`

Table 12.7 Comparison Operators With Examples

Operator	*Meaning*	*Example*	*Output*
`!=`	Not equal	`5 != 3`	`True`
`<`	Less than	`5 < 3`	`False`
`<=`	Less equal	`5 <= 3`	`False`
`==`	Exactly equal	`5 == 3`	`False`
`>=`	Greater equal	`5 >= 3`	`True`
`>`	Greater than	`5 > 3`	`True`

Table 12.8 Logical Operators With Examples

Operator	*Meaning*	*Example*	*Output*
and	Checks, if two statements are true	5 < 10 and 5 > 0	True
or	Checks, if at least one of two statements are true	5 < 10 or 5 < 0	True
not	Negates a statement	not 5 < 10 and 5 > 0	False

for the comparison is met, and `False` otherwise. Comparisons not only work for numerals but also for other types of input, such as strings.

Logical operators (Table 12.8) chain two comparisons together.[6] By doing so, they test if both conditions are met. Even if a single logical operator chains two statements together, for example if a first variable and a second variable is greater than a value, we may add more logical operators to chain even more conditions. This could be used, for example, to select documents in a corpus by several conditions (e.g., if the language of the document is English and if the document was published from January 2024 onwards), or to select promising tokens for further analysis (e.g., if a token is a noun and appears at least 100 times throughout the documents).

Membership operators (Table 12.9) test if an input value is part of a data structure (see Section 12.4.4) or not. We will encounter these operators, for

Table 12.9 Membership Operators With Examples

Operator	*Meaning*	*Example*	*Output*
`in`	Checks, if an input is part of a data structure	`"verb" in ["adjective","verb", "noun"]`	`True`
`not in`	Checks, if an input is not part of a data structure	`"verb" in ["adjective","propn", "noun"]`	`False`

example, when we select words by part of speech (e.g., noun, proper noun, verb, adjective). Note that this operation also works with numerals and when we check whether several elements from one data structure are located in a second one (e.g., if some words situated in a first text are also present in a second text).

12.4.4 Data structures

A **data structure stores and organizes data** like the closet in our previous example, which stores different types of clothing and instances thereof. In our case, we can store either data types (see Section 12.4.1) or other data structures within a data structure (i.e., a text corpus which is a list that comprises of lists of tokens). We distinguish between mutable and immutable data structures. **Mutable** means that these data structures can be changed after they are assigned to a variable. Other data structures are **immutable**, meaning that after they are assigned, they cannot change. We can further distinguish between **sequential types**, **mapping types**, and **set types.** Furthermore, we will introduce **matrix types**, **series**, and **data frames**.

We will begin with **immutable sequential data structures**. Strictly speaking, we already introduced an immutable sequential data type previously, namely **strings**. The values can only be reassigned to a variable using methods `like lower()`, which will produce a new string with only lowercase characters.

A **range** represents a sequence of numbers. These can either be integers or floating-point numbers. The range usually counts from a lower threshold to an upper threshold −1, in a sequence of predefined steps (e.g., in steps of 1, 5, 10). This may sound counterintuitive at first. But keep in mind that Python is a programming language with zero-based indexing. Zero-based indexing means that counting starts at `0` instead of `1`. For example, if we type `range(5)`, and use a list comprehension (see Section 12.4.9), we will get the following output: `[0, 1, 2, 3, 4]`. So if we want to count from 1 to 5 with a step size of one, we would type `range(1,6)`, or `range(1,6,2)` if we want to count from 1 to 5 with a step size of two.

Tuples are another immutable, sequential data structure. In Python, a tuple is called by using round brackets (). They are used, when stored data should not be changed during the programming process, and the order of the sequence is important. For example, we will encounter tuples when we introduce you to

semantic network analysis in Section 15.2. In this case, tuples are stored information on the linkage between two tokens or concepts (such as noun phrases), like the times the tokens etc. co-occur or the direction of linkage between concepts (e.g., *this book is good* could be summarized as (*this book*, *good*), with the object as first element and the attribute of the object as second element within the tuple).

Let us turn to the **mutable sequential data structures**, namely arrays and lists. An **array** is a homogeneous collection of values with a shared variable name, an index number indicating the position of a value within a variable and ranks of values (Ascher et al., 2001, S. 16–20). Arrays are optimized for numerical computation and allow us to use arithmetic operators on all elements of an array (e.g., integers, floating point numbers) simultaneously. This will save us a lot of time as we do not need to iterate through every single value stored in an array. Furthermore, arrays can be one-dimensional (like a vector containing numbers), two-dimensional (data-frame like), or multidimensional (e.g., containing the color-values of pixels inside a picture). We will introduce two-dimensional arrays for feature extraction and feature selection in Sections 13.3.1 and 13.4 respectively.

Lists, on the other hand, are dynamic collections of different data types. Lists can simultaneously contain elements of multiple data types at once, for example numerals, strings, and none-types. Lists can be seen as stacks, and as with stacks, we can add or remove elements, we can sort, or we can count the different types of elements present. You can envision a list as a tower of Lego bricks which are stacked upon each other (and sorted, for example, by color). Another advantage is that we can also use list comprehensions (see Section 12.4.9) to modify its content by preset conditions (e.g., keep a token if it is a noun, or keep a text if it contains more than five words). Lists use square brackets `[]` to store data. Due to its flexible and dynamic nature, we will encounter lists on multiple occasions from now on, including storing tokens of a document or storing tokenized documents in a text corpus.

The next data structure is a **dictionary**. A dictionary **maps** values to a key and is used for fast and efficient data storage. An example of a dictionary is the Contacts app on our mobile phone. Here, we can store name (key) – phone number (value)-pairs. We could store the same information in a list or a tuple within a list. However, it makes the lookup for the number much easier (and less computationally burdensome for Python, meaning that it takes less time) if we just have to look for a name and get the phone number. We could also store more information, for example the email address of a contact, the place of residence, postal code, a picture of the contact, among others. All of this information could then be retrieved using the key (the name of a person) of a dictionary. Curly brackets `{}` denote a dictionary.

Next are sets. A **set** stores the unique values of a data structure. For example, the set of the tuple (1,3,4,3) is {1,3,4}. A set is mutable, meaning that values can be added, removed, or modified. There is also an immutable type of set, called **frozenset**, meaning that we cannot change this type of set if it

is once assigned to a variable. In both cases, we can use methods to check for disjunctions, intersections, or differences between two sets, or if one set is a subset of a second set (for methods and functions, see Section 12.4.5). These methods allow us, for example, to compare the words used in different documents and, by doing so, could be used to calculate the similarity of documents.

A **matrix** is a two-dimensional array which stores a single data type. They are a special case of arrays (also stored in `NumPy`) and allow us to apply the same methods to them. Finally, a **Pandas data frame** is a more familiar structure for us social scientists. Data frames contain named columns (variables) and observations (rows). Different from a matrix, it can store different data types between and within columns; for example a column containing documents, another column with the year of publication of the respective document, another column with the number of sold copies of a document, author name, gender of author, publishing company, etc. Furthermore, Pandas also provides us with **Pandas Series** objects. It is a one-dimensional labeled array. Different from the array introduced earlier, it is able to store different data types.

12.4.5 Functions and methods

When we use Python for automated content analysis, we will encounter functions, methods, and variables during programming.

A **function** contains statements, which only run when called. Functions allow us to organize our code, as they encapsulate code that we otherwise would have to write repeatedly. We can either **use built-in functions**, which are already defined and ready for use without the need to import specific packages, **define functions** once and can reuse them, or **import functions from packages**. One of these built-in functions is the `print()`-function. As seen in Code box 12.4, the `print()`-function requires us to pass a string as input and displays the text seen in Output 12.1. The brackets () behind the function indicate a **function call**. It activates the function preceding the brackets.

```
print("Hello! Welcome to Python!")
```

Code box 12.4: The print()-function with function call

```
Hello! Welcome to Python!
```

Output 12.1: The output produced by the print()-function from Code box 1

```
# We now define a function
def hello(your_name):
    print("Hello " + your_name + "! Welcome to Python!")

hello("reader") # this is a function call
```

Code box 12.5: Example of a user-defined function including function call

```
Hello reader! Welcome to Python!
```

Output 12.2: Output printed by our user-defined hello()-function

Unlike built-in functions, Code box 12.5 shows a **user-defined function**. It starts with a comment describing the function in the first line of code. A **comment** is a line of code which is not executed and used solely for clarification. The function `hello()` introduced in the second line of code is defined using `def`. This function takes a character string as input, which we named `your_name`. Inside the function, the `print()`-function combines the input with a greeting: `"Hello " + your_name + "! Welcome to Python!"`. Indentation is crucial as it signals which statement belongs to the function.[7] After definition, we can call the `hello()` function anytime we want and pass a character string to generate a personalized greeting.

A function can not only print output but also perform calculations and return results using the `return()` function. The `return()` stops the function's execution and, as the name tells, returns the value specified in the brackets. Code box 12.6 provides an example with its output shown in Output 12.3. In contrast to the former function, we use string formatting in the third line of code. It is introduced with an f and allows us to embed variables directly into a string by using curly brackets `{}`. In our case, we use `print(f"Returning {a} + {b} = {a_b}")` to highlight the addition of the values of `a` and `b` passed to the defined function `a_plus_b`.

```
def a_plus_b(a,b):
    a_b = a + b
    print(f"Returning {a} + {b} = {a_b}")
    return(a_b)
print(a_plus_b)
```

Code box 12.6: Example of a user-defined function which returns an added value.

```
Returning 10 + 11 = 21
21
```

Output 12.3: Output printed and returned by our user-defined a_plus_b()-function.

12.4.6 Importing packages

As mentioned in Section 12.2, many functions and methods for automated content analysis are stored in packages written by other programmers. This saves us from the burden to write all functions and methods from scratch. A **package** contains previously defined functions, classes, and variables. A package is like a library – but instead of books, it stores reusable code. Packages can include **modules**, such as `matplotlib` which includes the modules `pyplot`, `cm` (colormaps), or `axes` among others. To use packages or modules, we can use the `import` function and specify the following:

1. The package name
2. An alias (= short name)
3. If needed, specific functions to import

By default, importing a package (e.g., `import pandas`) allows us to access all functions and methods stored in the imported package. For example, if we want to call the function `read_csv`, which opens a csv-file (comma separated value table), by typing `pandas.read_csv()`. However, we can also `import pandas as pd`, which enable us to call the same function by typing the alias `pd` and call the function afterwards, for example `pd.read_csv()`. Additionally, we can import specific functions. For example, to be able to call the `read_csv()` function stored in `pandas` directly, we use the phrase `from pandas import read_csv`. We can also use an alias for the imported function in the same way we did earlier for the package. In case of the Pandas `read_csv()` function, we may type and run `from pandas import read_csv as csv`. We can also import multiple functions. To do so, we separate the functions to be imported with a `,` for example `from numpy import array, linspace`.

Modules can also be imported similarly, for example `from matplotlib import pyplot as plt`. Finally, to import all functions from a package, we may use `from [package] import *`. Here, the asterisk tells Python to import all functions stored in a package. However, we advise you not to import all functions from a package or module as this may result in naming conflicts and makes your program code much more prone to errors. Code box 12.7 summarizes the different ways to import packages, subpackages, and functions.

```
import pandas # import pandas in default mode
import pandas as pd # import pandas with the alias pd

from pandas import read_csv # import the read_csv function stored
in the pandas package
from numpy import array, linspace # import the array and linspace
functions from numpy
from pandas import read_csv as csv # import the read_csv function
under the alias csv

from matplotlib import pyplot as plt # import all functions
stored in the subpackage pyplot
from nltk import * # import all functions stored in the nltk
package
```

Code box 12.7: Different ways to import packages

12.4.7 Flow control I: conditional statements

Let us now turn to flow control statements. **Flow control statements** determine which blocks of code are executed and in what order. One type of flow control statements are **conditional statements**. The key conditional statements in Python are `if`, `elif`, and `else`:

- The `if` statement checks whether a condition is met and executed the corresponding, indented block of code.
- The `elif` statement, short for else if, allows us to define additional conditions if the first condition is not met. We can chain multiple `elif` statements.
- Finally, the `else` statement provides a fallback if none of the other conditions defined in the `if` and `elif` statements are met.

Code box 12.8 exemplifies the application of the `if`, `elif`, and `else` statements to test if a sentiment value is negative, positive, or neutral. The result of this flow control sequence is shown in Output 12.4. We will use similar statements for assigning sentiment values to sentiment categories in Section 16.2.

```
sentiment_value = 0.99
if sentiment_value < -0.95:
    print(f"Sentiment with value {sentiment_value} is negative.")
elif sentiment_value > 0.95:
    print(f"Sentiment with value {sentiment_value} is positive.")
else:
    print(f"Sentiment with value {sentiment_value} is neutral.")
```

Code box 12.8: Example of if, elif, and else statements

```
Sentiment with value 0.99 is positive.
```

Output 12.4: Output provided by the if, elif, and else statement in Code box 8

12.4.8 Flow control II: loops

The next flow control structures for data preparation and automated content analyses are **for** and **while** loops. A `for` loop iterates through a sequential data structure, such as a list or a tuple, starting with the first element continuing to the last element. For example, we can use a `for` loop to iterate search for specific words or names or to map sentiment categories onto the sentiment values assigned to each text.

As shown in Code box 12.9, for loops require an **iterator**. An iterator holds an element from the data structure and allows functions situated within a loop to be performed on each element passed to the loop (=data types or other data structures). An analogy is a person that intends to fold T-shirts stored in a closet. The closet is the data structure, the T-shirts are the elements, and the folding task equates to the loop.

```
numbers = [1,2,3,4,5]
for number in numbers:
    print(number)
```

Code box 12.9: Exemplary application of a for loop

In contrast, `while` loops execute a block of code as long as a previously specified condition is met. This makes them useful if the number of iterations is unknown. Each iteration checks the condition, and if met, the loop continues until the condition is no longer met.

For example, in Code box 10, we use a `while` loop to count from one to ten. We initialize `number_count` and assign a value of `1` in the first line of code. We

then initialize and continue the loop until `number_count` reaches `10`. We use the `print()`-function in the loop to print the number in each iteration. Finally, we use the assignment operator `+=` to add `1` to the variable `number_count`. Using the previous T-shirt analogy, we could instruct the person to fold the shirts until they've folded all shirts in ten drawers.

```
number_count = 1
while number_count <= 10:
    print(number_count)
    number_count += 1
```

Code box 12.10: Exemplary use of a while loop

12.4.9 List comprehensions

List comprehensions provide a quick way to create or modify lists. They apply functions or methods on elements stored in a data structure. In our case, they are useful for tasks like part of speech-tagging, lemmatization, or to iterate through properties assigned to document (e.g., sentiment values, values for topic assignment).

Code box 12.11 illustrates different types of list comprehensions, such as `[value for value in variable_list]`, which constructs a new list from all elements stored in `variable_list` (first line of Output 12.5). In the following line, we tell Python to keep every element of the `variable_list` but multiply each element by two. In the third example, we conditionally assign a `1` if a `value` in `variable_list` is smaller than zero, and `0` otherwise. In the final example, we keep all elements in the `variable_list` that are also included in a second list `y` and drop all elements that are not part of `y`. As you may have noticed, the main difference between the third and the fourth example is that if we want to apply changes to the elements in a list, we use the condition at the beginning of the list comprehension, but if we want to filter elements from the list, we use conditional statements at the end of the list comprehension. We can also chain conditions for filtering elements from the list using logical operators (`and`, `or`, `not`). We will make use of such chained filtering conditions for data preparations in Sections 13.4. and 15.1.1.

```
variable_list = [1, -1, 0, 2, -2]
y = [1, 2]
print([value for value in variable_list])
print([value*2 for value in variable_list])
print([1 if value < 0 else 0 for value in variable_list])
print([value for value in variable_list if value in y])
```

Code box 12.11: Different examples of list comprehension

```
[1, -1, 0, 2, -2]
[2, -2, 0, 4, -4]
[0, 1, 0, 0, 1]
[1, 2]
```

Output 12.5: Different outputs produced by the list comprehension seen in Code

12.4.10 Regular expressions

The last important, fundamental concept that we will apply in Python throughout the following chapters are regular expressions. Regular expressions (abbreviated: regex) are powerful tools for string searching and string manipulation. A regex contains a sequence of strings and metacharacters which define a search pattern. We will use Python's `re` library to perform searches, substituting strings, or splitting strings into list of strings. Table 12.10 lists common regex metacharacters, and Table 12.11 summarizes regex functions we will use for data preparation and automated content analysis.

Table 12.10 Overview of Common Metacharacters and Regular Expressions

Regular Expression	*Explanation*
`.`	Matches all characters, numbers, and white spaces
`^`	Matches a pattern at the beginning of a string
`$`	Matches a pattern at the end of a string
`*`	Matches a character that repeats at least zero times
`+`	Matches a character that repeats at least one time
`?`	Matches a character either one or zero times
`{l,u}`	Matches at least `l` repetition (lower threshold) and at most `u` repetitions (upper threshold)
`()`	Matches a group of strings
`[]`	Matches a set of characters, e.g., [a-z]
`|`	Either or (bitwise or)
`\w`	Matches any alphanumeric character (characters and numbers)
`\W`	Matches all non-alphanumeric characters
`\d`	Matches all decimal digits
`\D`	Matches all non-digit characters
`\s`	Matches any whitespace character
`\S`	Matches all non-whitespace characters
`\t`	Tabulator
`\n`	New line
`\A`	Matches only the start of a multiline string
`\Z`	Matches only the end of a multiline string
`\b`	Word boundary

Table 12.11 Overview of the re Functions Used for Pattern Detection, Pattern Substitution, and String Splitting

Regex Function	*Explanation*
`re.compile(pattern)`	Converts a string pattern into a regex object.
`re.search(pattern, string)`	Searches the string for a pattern and returns information on where the pattern was found first in the string.
`re.match(pattern, string)`	Checks if the pattern matches at the beginning of a string.
`re.findall(pattern, string)`	Returns a list of substrings that were detected using the string patterns.
`re.finditer(pattern, string)`	Returns an iterator object that yields all substrings that were detected using the string patterns.
`re.sub(pattern, substitute, string)`	Substitutes a pattern in a character string by a substitute pattern.
`re.split(pattern, string)`	Splits the string into a list of strings at places where the pattern is found.

```
import re
string = "This is a small text for testing regular expressions"
print(re.search("text", string))
print(re.search("sma\w+", string))
print(re.search("^\w+", string))
print(re.search("\w+$", string))
print(re.findall("[t|s]\w+", string))
print(re.findall("(te\w+|sm\w+)", string))
```

Code box 12.12: Example of search patterns using regular expressions

```
<re.Match object; span=(16, 20), match='text'>
<re.Match object; span=(10, 15), match='small'>
<re.Match object; span=(0, 4), match='This'>
<re.Match object; span=(41, 52), match='expressions'>
['small', 'text', 'testing', 'ssions']
['small', 'text', 'testing']
```

Output 12.6: Output provided by the regular expression search patterns

We will show you a few regex examples in Code box 12.12, and the corresponding output of the examples in Output 12.6. In the following, we will explain the code and the examples line by line.

In the first line, we import the `re` package and assign a character string to a variable we call `string` in the second line. From line 3 onwards, we perform different regex searches. The first `re.search()` function covers a simple pattern, namely '`text`'. As it does not include metacharacters, it will only find the first instance of text within the string defined in the second line of code. The second pattern '`sma\w+`' searches specifically for `sma` and at least one character afterwards (indicated by the metacharacters `\w+`). Consequently, it will match 'small' in the string defined in the second line of code. The third regex pattern matches the first word in the string (`^`), as we tell Python to look for at least one character afterwards (`\w+`). The next regular expression searches for multiple characters at the end of the string, as indicated by "`\w+$`". By using the `re.findall()` function call, the next two lines of code produce a list of substrings. The `re.findall("[t|s]\w+",string)` function returns all elements of the string defined in the second line, that start with either `s` or `t` (indicated by the bitwise or: `|`), and checks if at least one character follows. The last function call captures a sequence that either started with `te` or `sm` followed by at least one further character.

Now let us turn to output produced by these function calls. In case of the `re.search()` function calls, you will notice that the output is a regex match object. This object contains information on the position in the character string (`span = (start,end)`), as well as the text found at the given position. We can extract the information on the content of the found pattern with the `group()` method, the starting position with the `start()` method, the end position with the `end()` method, and the span with the `span()` method. In contrast, the `re.findall()` function returns a list of all character sequences that match the provided pattern.

12.5 Summary and concluding remarks

Now let us pause for a moment and recap what we have learned in this section. We covered how to install Python and Anaconda. We also covered package management related to automated content analysis, as well as navigating the Spyder interface. These are essential prerequisites for programming in Python. We then introduced fundamental concepts such as variables, data types, operators, data structures, functions, methods, conditional statements, loops, list comprehensions, and regular expressions. Taken together, this may seem overwhelming at first. But as you will see from now on, these concepts will be repeatedly used as building blocks for our Python code, which enables us to preprocess, analyze, interpret, and visualize texts automatically. Mastering these practices will not only enable you to conduct automated content analysis on your own, but to analyze your data beyond the scope of this book.

Notes

1 Python was named after the British comedy series *Monty Python's Flying Circus*. Retrieved November 14, 2024, from https://docs.python.org/3/faq/general.html#why-is-it-called-python.
2 There are other introductions into object-oriented programming, for example Phillips (2018).
3 The authors of this book use Windows 11 as operating system, which is why, in the following, we can only show the Windows version of Anaconda. We used Anaconda version 2.4.0.
4 This is useful if you want to install packages that are important for different steps of your programming procedure or for different forms of automated content analysis have conflicting dependencies.
5 A tensor tensors are multidimensional arrays of numbers that represent complex data. For example, a vector is a 1-dimensional tensor, data matrix a 2-dimensional tensor, and a stack of data matrices a 3-dimensional tensor.
6 There are also bitwise operators present in Python, some of which we will encounter when introducing regular expressions. They differ insofar as they treat the input as bits. We will need those when we use regular expressions, or regex, to search for patterns in our text. It will provide either 1 (true) or 0 (false) as output. There is the bitwise and &, the bitwise or |, the bitwise not ~, and the bitwise xor ^ (=disjunctive or). Furthermore, there are the << and >> operators available. They return a value with the bits shifted to the left or to the right, which equates to exponentiation. For example, 2 << 2 will output 4, 2 << 3 will output 16, whereas 16 >> 2 will output 4. This is a substantially faster but low-level type of computation compared to exponentiation. Low-level means that these operations interact directly with the physical components of a computer, such as the central processing unit (CPU) or the random access memory (RAM).
7 We will see a similar behavior when we encounter flow control sequences such as conditional statements in Section 12.4.7. and loops in Section 12.4.8.

13
DATA PREPROCESSING

13.1 Introduction

As outlined in Section 11.6.3, we work mainly with unstructured data in automated content analysis. This implies that our documents must be preprocessed before we can apply different methods of automated content analysis, including relational methods (GDA, network analysis), sentiment analysis, or topic modeling algorithms. **Preprocessing** is defined as the transformation of unstructured data into a machine-readable data format, and the removal of errors from machine-readable data. Usually, preprocessing leads to (a) an improvement of model performance (e.g., they improve the accuracy; Camacho-Collados & Pilehvar, 2018) and (b) reduction of bias (Chai, 2023; Zimmermann et al., 2024).[1] Data preprocessing includes the removal of typos, mixed data types in data columns, duplicates, or mislabeled data (Chu et al., 2016).[2]

Bias in machine learning is defined as decisions made by an algorithm that "are skewed toward a particular group of people" (Mehrabi et al., 2021, S. 2) and might be caused by decisions prior to data generation, including decisions on (a) what data to collect, (b) how this data was labeled during the training procedures (Chakraborty et al., 2021, S. 429), or (c) how the used algorithm assigns these labels after training (Mehrabi et al., 2021). In the case of the former two, we can avoid biases by carefully designing our study, that is by constructing a balanced, representative text corpus,[3] as well as deciding how to deal with ambiguity (Beck et al., 2020, S. 67f.). We can deal with such ambiguities by carefully selecting cases – for example texts stemming from different genres – and by applying preprocessing techniques after data was collected. According to Jim et al. (2024, S. 13),

DOI: 10.4324/9781003496397-16
This chapter has been made available under a CC-BY-NC-ND 4.0 license.

we can distinguish between ten commonly applied methods for data preprocessing:

1) **Data cleaning** = identification and correction of errors or inaccuracies.
2) **Tokenization** = process of splitting a text into discrete entities such as single words (unigrams), phrases consisting of two or three words (bi-/trigrams), sentences, or paragraphs.
3) **Feature extraction** = transformation of raw data (e.g., text data) into numerical features, usually stored in a structured data format.
4) **Feature selection** = select variables (also called 'features') to reduce the dimensionality of the data to either theoretically relevant variables or most important features (e.g., with the highest association to the dependent variable/target variable or the threshold for removing a token from the text corpus).
5) **Normalization** = Adjustment of numerical data into a common range.
6) **Standardization** = Transformation of numerical data into variables with a mean of zero and a standard deviation of 1.
7) **Noise Elimination** = Removal of random fluctuation in the data or irrelevant data (e.g., observations unrelated to our research question).
8) **Data augmentation** = enhancing the size and diversity of a dataset by applying transformation to the data, such as replacing elements (e.g., tokens within a text document), mirroring, or omission of rows or columns. Data augmentation is especially relevant in case of deep learning, as the deep neural network needs abundant training data to work properly (Shorten et al., 2021).
9) **Handling missing data** = managing and filling in missing values in a dataset (e.g., via multiple imputation procedures).
10) **Stemming** = reduction of words to their base form by removing suffixes.

These procedures are neither mutually exclusive as they may be combined sequentially nor without overlap (e.g., missing data handling overlaps partly with data cleaning). Furthermore, "text preprocessing is not a one size fits all-process" (Chai, 2023, S. 510), as some methods of automated content analysis require (or omit) different preprocessing steps.[4] Keep also in mind that preprocessing (either manually or automatically) may not catch all errors. This is why we need to manually check samples of the generated output after each step and correct errors as soon as possible, as errors usually entail other errors, rendering your dataset unusable.

As these preprocessing steps are not mutually exclusive, we will structure the remainder of this chapter as follows. We begin with data cleaning procedures and focus on the removal of duplicates and missing values, as well as the unification of data types (Section 13.2). We then continue with tokenization (Section 13.3) and proceed with introducing feature extraction techniques on corpus-, document-, and token level (Section 13.4). Afterwards, we will introduce feature selection techniques (Section 13.5) and close with a summary and outlook on how to apply these techniques for automated content analysis (Section 13.6).

Table 13.1 Example of a Messy, Uncleaned Dataset

Nr	*Title*	*Text*	*Publication date*	*Source*	*Problem*
1	Title1	„some text“	22.06.2024	Source1	Duplicate
2	Title1	„some text“	22.06.2024	Source1	Duplicate
3	Title1	„some text““	22.06.2024	Source1	Duplicate
4	Title2	EMPTY	23.06.2024		Omitted information
5	EMPTY	„some other text“	EMPTY	Source2	Omitted information
6	Title1	EMPTY	22.06.2024	Source1	Omitted information, potentially a duplicate
7	Title6	„something about cats“	23.06.2024	Source1	Unrelated to research question

13.2 Data cleaning

Data cleaning is the process of removing errors from data. As such, it is a crucial step in every machine learning workflow, including automated content analysis. If you do not clean your data thoroughly, you will unavoidably introduce bias into your analysis. For example, if you collect 1,000 identical texts and keep these in your text corpus, then you will obviously extract one or many topics that will just reflect these duplicates. In regards to GDA, you may find the strongest opposition (= topic dimension) between these 1,000 texts and every other text, given that these 1,000 texts represent a significant portion of the corpus.

Data cleaning involves **the removal of duplicates**, **missing data**, or the **unification of mixed data types** in data columns, as each column should represent a single variable suitable for your analysis. While the first introduces bias in your dataset, the latter two could prevent your algorithms from running, as they usually expect a certain data type as input (e.g., a list of tokens, an array). Finally, we must try to **detect outliers** or **unintentionally collected data** which does not relate to your research interest/research question. Table 13.1 provides an example of a messy dataset in dire need for cleaning.

```
import pandas as pd
data = {"tokens": ["text", "something else", "more of the same",
"text", None, "text"],
    "numbers" : [None, 0, 1, 0, 42, 0],
    "other_numbers" : [100, None, 100, 1337, -1400, 1337]}
df = pd.DataFrame(data)
print(df)
```

Code box 13.1: Creating a Pandas data frame from a dictionary with empty cells and duplications in the columns

```
              tokens  numbers  other_numbers
0               text      NaN          100.0
1     something else      0.0            NaN
2   more of the same      1.0          100.0
3               text      0.0         1337.0
4               None     42.0        -1400.0
5               text      0.0         1337.0
```

Output 13.1: Overview of the Pandas data frame established in Code box 13.1

So how can we achieve this? Let us first prepare a small toy dataset. As seen in Code box 13.1, we begin with importing pandas into our workspace, assigning a dictionary (see Section 12.4.4) with key and value-pairs (=names and values) to a variable. Finally, we call the `DataFrame()`-function and assign the new data frame to a variable we will call `df` (short for data frame). Our artificial data frame will include three variables and six observations. In our small test case, we name the variables `tokens` (a string variable), `numbers`, and `other_numbers` (both including floating point numbers). We will leave some cells empty (carrying a `None`-type, as indicated by `None` in case of strings, and `NaN` in case of floating-point numbers), integrate some duplications in the same columns, and a duplication of the fourth and sixth entry (index numbers 3 and 5 in Output 13.1).

We can use two simple methods provided by pandas to remove duplicates and missing data. These are `drop_duplicatates()` and `dropna()`. Both share the `subset =` option, meaning that we can also pass either the name of a single column or a list of column names which serves as the basis for dropping rows with duplicate entries, or removing rows if entries are omitted. By default, these methods will keep the first of the rows with identical entries.

Now let us take a closer look at the `drop_duplicates()` method. In its most basic form, this method removes all identical rows but one, namely the first entry in our data frame unless defined otherwise. If we want to remove only duplicates conditional upon values within a specific column or a set of columns, we must again specify this column (or multiple columns) using the `subset =` option and either a string (for a single variable) or a list of strings (for multiple variables). As seen in Table 13.2 (with variable names `Content`, `Title`, and `AuthorName`) on the left, the `drop_duplicates()` identifies rows with index numbers 2 and 3 as identical and consequently removes the third row. When we specify `drop_duplicates(subset = "Content")`, then Python will recognize that the content of the entries in with index numbers 2 and 3, as well as the content of rows 1 and 4 are identical. Consequently, it will remove the rows with index numbers 3 and 4 by default. If we are only interested in the text itself for the purpose of model training, we might drop duplicate the texts but might keep the information on who issued these texts separately. Examples

Table 13.2 Difference Between the drop_duplicates()-method Without and With Specified Subset Option

index	*Content*	*Title*	*AuthorName*	*index*	*Content*	*Title*	*AuthorName*
1	TEXT2	TITLE1	AUTHOR1	1	TEXT2	TITLE1	AUTHOR1
2	TEXT	TITLE1	AUTHOR1	2	TEXT	TITLE1	AUTHOR1
3	TEXT	TITLE1	AUTHOR1	3	TEXT	TITLE1	AUTHOR1
4	TEXT2	TITLE2	AUTHOR2	4	TEXT2	TITLE2	AUTHOR2
`df.drup_duplicates()`				`df.drop_duplicates(subset = "Content")`			

Table 13.3 Difference Between the dropna()-method Without and With Specified Subset Option

index	*Content*	*Title*	*AuthorName*	*index*	*Content*	*Title*	*AuthorName*
1				1			
2		X		2		X	
3	X			3	X		
4				4			
`df.dropna()`				`df.dropna(subset = "Title")`			

include comment in an online forum, quotations, or song lyrics issued by an artist and covered by different artists.

Next, let us show you how to remove rows with omitted data. In its most basic form, the `dropna()` method returns a Pandas data frame without rows where *any* cell is empty. However, we may only want to omit missing values located in columns relevant for our investigation. In our exemplary case, we might want to remove all missing values from the Title column. We can do this with the `subset =` option and provide it with a single string, if we wish to remove only missing values for a single column/variable, or a list of strings when we want to remove missing values from a selection of columns/variables. If we want to override our current pandas-data frame, we may also wish to include the `inplace = True` – option. Otherwise, we must assign the data frame to a new variable to keep the changes in our working space. The former is more memory-efficient, while the latter enables us to keep track of the modifications and changes performed on the data. Table 13.3 highlights the difference between the `dropna()` method without and with a given subset option (filtered by the column with name Title).

Let us now look how these methods with and without `subset`-option work in practice (Code box 13.2). We will also produce an output by enclosing both variants using `print()`-functions, and take a look at the produced output. In the first part of the output, we see that only the row with index number 5 was dropped, as it is completely identical with the fifth row (index number = `4`). In the second part of the output, only three out of the six rows were kept, as the

other three (rows `0`, `1`, and `4`) yielded at least one missing value. The third output removed the rows with index numbers 3 and 5, as the value of the `numbers` variable was identical with row `1` in the former case, and with row `4` in the latter case. The last dropped rows with index number `0` and `4`, as there were missing values either in the tokens or the `numbers` variable.

```
# without subset option
print(df.drop_duplicates())
print(df.dropna())
# with subset option
print(df.drop_duplicates(subset = "numbers"))
print(df.dropna(subset = ["numbers", "tokens"]))
```

Code box 13.2: Differences in the pandas drop_duplicates() and drop_na() methods with and without options

```
             tokens  numbers other_numbers
0              text      NaN         100.0
1    something else      0.0           NaN
2  more of the same      1.0         100.0
3              text      0.0        1337.0
4              None     42.0       -1400.0

             tokens  numbers other_numbers
2  more of the same     1.0          100.0
3              text     0.0         1337.0
5              text     0.0         1337.0

             tokens  numbers other_numbers
0              text      NaN         100.0
1    something else      0.0           NaN
2  more of the same      1.0         100.0
4              None     42.0       -1400.0

              tokens numbers other_numbers
1     something else     0.0           NaN
2   more of the same     1.0         100.0
3               text     0.0        1337.0
5               text     0.0        1337.0
```

Output 13.2: Variants of a data frame with removed duplicates and removed rows with missing values (with and without subset-option)

13.3 Tokenization

Some of the algorithms used to conduct automated content analysis, such as geometric data analysis, and topic modeling via Latent Dirichlet Allocation (Section 17) require us to provide discrete text elements (such as words) instead of whole text-strings to work properly. This procedure of splitting the texts within a corpus into discrete elements is called **tokenization**. Tokens include words, bigrams (= sequences of two words), n-grams (= sequences of words of N length, with N equals a number of words) or noun phrases (= adjectives followed by nouns).

In the following, we will employ `spaCy` to tokenize text, but keep in mind that other packages, such as `nltk`, also provide functions for tokenization and text preparation. As seen in Code box 13.3, we begin with importing the spaCy-package. In the next line, we use the `spacy.load()`-function to import the `en_core_web_sm` language model[5] and assign this model to a variable we call `nlp` (short for natural language processing). Spacy provides us with different pretrained language models, which must be downloaded separately. So if you not already downloaded the model, you must download it via the terminal (see Section 12.2). When you open the terminal, you need to type 'python -m spacy download en_core_web_sm' and press Enter afterwards. This process is similar but not identical to downloading packages using `pip`. The notable differences are (1) we need the `Python` shell to (2) open a module (synonym for package) with `-m`, in our case `spacy`, and (3) `download` the language model named `en_core_web_sm.`

Once downloaded and imported into our working space, the spacy-model will automatically tokenize a passed text-document and convert each word to a token by default and extract features on token level. To do so, we assign a document (as string) to a variable named `text`, which will then be tokenized using `nlp(text)` as seen in the fifth line of code in Code box 13.3 we assign the text to a variable named `tokenized_text`. Finally, use `print(type(tokenized_text),tokenized_text))` to check the data type assigned to the new variable, and its content. As indicated in Output 13.3 with the `spacy.tokens.doc.Doc`-object, which indicates that our text is now tokenized and contains spacy tokens. Strictly speaking, this step is not necessary. Nevertheless, we recommend to print the results produced by a function or a message that indicates at which step of the workflow Python currently is throughout the workflow to check if the functions and methods have been carried out successfully.

```
import spacy
nlp = spacy.load("en_core_web_sm")
text = "We will use spacy to tokenize our text and to extract only
nouns, adjectives, proper nouns, and verbs from the tokenized
text."
```

Code box 13.3: Application of spacy to tokenize text data

```
tokenized_text = nlp(text) # We tokenize the text using our
NLP model
print(type(tokenized_text), tokenized_text)
```

Code box 13.3: (Continued)

```
<class 'spacy.tokens.doc.Doc'> We will use spacy to tokenize our
text and to extract only nouns, adjectives, proper nouns, and
verbs from the tokenized text.
```

Output 13.3: Output of the tokenization process provided by print() functions

13.4 Feature extraction

After tokenization, we continue with extracting the features we decide to keep for automated content analysis. A **feature** "is a measure that describes relevant and discriminative information about a data object" (Theng & Bhoyar, 2024, S. 1575), for example the times a token appears in every text document in our text corpus. **Feature extraction** – or feature engineering – "is the process of preparing, transforming, constructing, and filtering features to optimize the performance of a data analysis task" (Wang et al., 2022, S. 2). Its goal is to capture the essential information from the original features and represent the information in a lower-dimensional feature space (= data matrix with much less columns compared to the data matrix used as input). Feature extraction aims to reduce calculation time and to keep only informative dimensions (Jia et al., 2022, S. 2664). This includes the conversion of the text corpus into a bag-of-words format, the application of weighting schemes (such as TF-IDF), or dimensionality reduction techniques with the aim of mapping the complexity of our data onto a lower dimensional space (e.g., via principal component analysis or topic modeling techniques). Strictly speaking, topic modeling and sentiment analysis are also feature extraction techniques. However, we focus on feature extraction procedures with the aim of text preparation. In the following, we will show you how you can extract features on corpus level, on document level and on the level of individual tokens.

13.4.1 Feature extraction on corpus level

On corpus level, we can extract either an unweighted or weighted document-term matrix or count frequencies of different tokens throughout our text corpus. Both types of document-term matrices convert the corpus into a bag-of-words format and, by doing so, stripping the tokens from their position within the respective texts (Blei et al., 2003)[6]. We will make use of this in Sections 15 and 17 in the case of relational and generative topic modeling approaches to extract topics from the text corpus.

We will begin by extracting an unweighted document-term matrix using a dialogue sequence from "Don't look up" found in Chapter 5. This sequence includes 15 lines of dialogue, which we arrange in tuples of two. The first element of this tuple stores the name of the character which speaks the dialogue line. The line of dialogue is stored in the second element of each tuple. As usual, we start with importing the relevant packages: the `text` module from `sklearn.feature_extraction` with the alias `tx`, and `pandas` as `pd` (see the first two lines of Code box 13.4). Next, we define the column names using the list assigned to the `columns` variable. We then create the Pandas data frame `df` from a list of tuples named `dialogue`. Such a list of tuple informs pandas that the rows (tuples) are nested within a larger data structure (list).

```
import pandas as pd
from sklearn.feature_extraction import text as tx

columns = ["character", "text"]
dialogue = [
("President Orlean", "The point is you can't go around telling
people there's a 100% chance they will die. You know, that's just
nuts. And we should get our own scientists on this. No offense,
but you're just two people who walked in here with ... "),
#[...]
("Jason", "I'm the fucking Chief of Staff. Boy with the dragon
tattoo. So, I'm doing fine.")]

df = pd.DataFrame(dialogue, columns = columns)
```

Code box 13.4: Importing packages for bag-of-words and tf-idf representation and creating a pandas DataFrame from a list of tuples

Using this data frame, we create an unweighted document-term matrix with the `CountVectorizer()` function (Code box 13.5). The `CountVectorizer()` initializes a pipeline where multiple functions are used in succession for preprocessing purposes. To initialize this pipeline, we assign the `CountVectorizer()` to a variable before applying it to our text corpus stored in the `df.text` column. If we do not define the functions within `CountVectorizer()`, it will only convert input to lowercase without performing feature selection tasks. The functions we apply are as follows:

- `min_df` = Minimum cut-off value. It removes tokens appearing in fewer texts than this value.
- `max_df` = Maximum cut-off value. It removes overly common tokens that appear in too many texts.

- `lowercase` = Automatically convert tokens to lower case (default is `True`).
- `stop_word` = List of stop words (e.g., common tokens, punctuation) to remove.
- `vocabulary` = Predefined list of relevant tokens. Only the tokens passed to as dictionary are counted.
- `max_features` = Maximum number of features to extract (e.g., the 1.000 most frequent tokens).

After assigning the `CountVectorizer()` to `count_vectorizer` (see the first line of Code box 13.5), we will create an unweighted document-term matrix using the `fit()`, `transform()`, `A` (for array) and `get_feature_names_out` methods. The `fit()` method learns the vocabulary used in our corpus, and we store its result in a new variable `dtm` (short for document-term matrix). The document-term matrix itself is established using the `transform()` method. This results in a sparse matrix where only cells representing the tokens found in the texts are filled. The `A` method converts the sparse matrix into a numpy array and fills all empty cells with zeros (see Output 13.4), which enables us to perform further calculations on the document-term matrix. We assign this matrix to the variable `dtm_matrix`, which will be used as input to construct a Pandas data frame. As the array lacks column names (= the counted tokens), we apply the `get_feature_names_out`() method on `dtm` to extract the token names. We pass these extracted token names to the columns of the Pandas `DataFrame()` function as `columns` and assign the new data frame to `dtm_data_frame`. Applying these methods results in the data frame seen in Output 13.4.

```
count_vectorizer = tx.CountVectorizer()
dtm = count_vectorizer.fit(df.text)
dtm_matrix = dtm.transform(df.text)
dtm_data_frame = pd.DataFrame(dtm_matrix.A,
                              columns=dtm.get_feature_names_out())
```

Code box 13.5: Creating a document-term matrix using sklearn's CountVectorizer function

After we constructed our document-term matrix, we utilized two consecutive Pandas methods to count token frequencies and sort them in descending order. The `sum()` method counts the appearances of tokens across the dialogue-lines, while `sort_values()` arranges the data in ascending order, starting from the lowest value. Using the `sum()` method on the Pandas data frame returns a `Pandas series`. To sort token frequencies in descending order, we use the sort_`values()` method and add the option `ascending = False`. This can be achieved in a single line of code (see Code box 13.6). We store this sorted Pandas series object in the variable `sorted_bow_data_frame`. As seen in Output 13.5, this allows us to identify the tokens that appeared most times and least times in our

```
   100  admission after all am an . . . who will with years yes you
0    1          0     0   0  0  0 . . .   1    1    1     0   0   3
1    0          0     0   0  0  0 . . .   0    0    0     1   0   0
2    0          0     0   0  0  0 . . .   0    0    0     0   0   0
3    0          0     0   0  1  0 . . .   0    0    0     0   0   1
4    0          0     0   0  0  1 . . .   0    0    0     0   0   0
5    0          0     0   0  0  0 . . .   0    0    0     0   1   0
6    0          1     0   0  0  0 . . .   0    0    0     0   0   2
7    0          0     0   0  0  0 . . .   0    0    0     0   0   0
8    0          0     0   0  0  0 . . .   0    0    0     0   0   0
9    0          0     0   0  0  0 . . .   0    1    0     0   0   1
10   0          0     0   0  0  0 . . .   0    0    0     0   0   0
11   0          0     1   1  1  0 . . .   0    0    0     0   0   2
12   0          0     0   0  0  0 . . .   1    0    0     0   0   0
13   0          0     0   0  0  0 . . .   1    0    0     0   0   2
14   0          0     0   0  0  0 . . .   0    0    1     0   0   0
```

Output 13.4: Pandas data representing a document frequency matrix with named columns. Raw frequencies

corpus. As you may have noticed, common pronouns, determiners, and conjunctions like 'you', 'and', 'the', 'is', 'this' appear most often. Since they provide little contextual information, they are usually classified as stop words and should be removed prior to employing generative topic models or geometric data analysis.

```
sorted_dtm_data_frame=dtm_data_frame.sum().sort_values(ascending
= False)
```

Code box 13.6: Counting and arranging token frequencies in our dialogue sequence

```
you       11
and        7
the        7
          ..
fine       1
fifteen    1
ms         1
Length: 126, dtype: int64
```

Output 13.5: Token counts in descending number of occurrences in our data corpus

```
tfidf = tx.TfidfVectorizer()
tfidf_dtm = tfidf.fit(df.text)
tfidf_dtm_matrix = tfidf_dtm.transform(df.text)
tfidf_dtm_data_frame = pd.DataFrame(tfidf_dfm_matrix.A,
                          columns = tfidf.get_feature_names_out())
```

Code box 13.7: Creating a weighted document-term matrix using sklearn's TfidfVectorizer function

```
        100 admission    after . . .    years      yes      you
0  0.159846   0.00000 0.000000 . . . 0.000000 0.000000 0.284455
1  0.000000   0.00000 0.000000 . . . 0.194382 0.000000 0.000000
2  0.000000   0.00000 0.000000 . . . 0.000000 0.000000 0.000000
3  0.000000   0.00000 0.000000 . . . 0.000000 0.000000 0.244711
4  0.000000   0.00000 0.000000 . . . 0.000000 0.000000 0.000000
5  0.000000   0.00000 0.000000 . . . 0.000000 0.527011 0.000000
6  0.000000   0.23957 0.000000 . . . 0.000000 0.000000 0.284219
7  0.000000   0.00000 0.000000 . . . 0.000000 0.000000 0.000000
8  0.000000   0.00000 0.000000 . . . 0.000000 0.000000 0.000000
9  0.000000   0.00000 0.000000 . . . 0.000000 0.000000 0.142246
10 0.000000   0.00000 0.000000 . . . 0.000000 0.000000 0.000000
11 0.000000   0.00000 0.217246 . . . 0.000000 0.000000 0.257734
12 0.000000   0.00000 0.000000 . . . 0.000000 0.000000 0.000000
13 0.000000   0.00000 0.000000 . . . 0.000000 0.000000 0.442881
14 0.000000   0.00000 0.000000 . . . 0.000000 0.000000 0.000000

[15 rows x 126 columns]
```

Output 13.6: Pandas data frame representing a document frequency matrix with tf-idf weighting

Similar to the `CountVectorizer()` pipeline, the `TfidfVectorizer()` pipeline allows us to create a weighted document-term matrix.[7] As the name of the pipeline suggests, this function implements the **TF-IDF** (term frequency-inverse document frequency) **weighting scheme**. This scheme assigns higher values to tokens that appear frequently in a limited number of documents but are less common across the corpus. The assumption is that tokens with high tf-idf values are more informative and characterize specific documents rather than being widely used. While tf-idf is not inherently designed for topic modeling, it can serve as preprocessing step for some topic modeling approaches, for example latent semantic indexing (LSI) (see Deerwester et al., 1990; Fuhse et al., 2020).

As you see in Code box 13.7, we use the standard functions of the `TfidfVectorizer()` pipeline and assign it to `tfidf`. We then fit `tfidf` on the text corpus stored in `df.text`. Similar to the `CountVectorizer()`, we then use the `transform()` method to establish the weighted document-term matrix. Finally, we pass the document-term matrix as input to the Pandas `DataFrame()` function, and the extracted vocabulary as column names. The resulting weighted document-term matrix is depicted in Output 13.6.

13.4.2 Feature extraction on document level

Let us continue with feature extraction on the document level. To illustrate relevant functions and methods, we will use the following dialogue line transcribed from the movie *Don't Look Up* introduced in Chapter 5.3:

> Look, I ... I was confused and outraged as well that they turned the mission [to destroy the comet with nukes] around too. But they've [BASH] got Gary Talcamont out of Stanford and Lisa Inez from Princeton on their team. I mean, this is cutting edge next generation technology.

We apply methods provided by `spacy` on this line of dialogue using:

- the `lang_` method to detect the most probable langue using the `en_core_web_sm`, language model.
- the `noun_chunks` method to identify noun phrases.
- `ents` for named entity recognition (NER).

```
import spacy
nlp = spacy.load("en_core_web_sm")

dlu = nlp("Look, I ... I was confused and outraged as well that
they turned the mission [to destroy the comet with nukes] around
too. But they've [BASH] got Gary Talcamont out of Stanford and
Lisa Inez from Princeton on their team. I mean, this is cutting
edge next generation technology.")

print(f"detected language: {dlu.lang_}")
print(dlu.noun_chunks, [x for x in dlu.noun_chunks])
print(dlu.ents)
```

Code box 13.8: Language detection, extraction of noun chunks, and named entity recognition with spacy

```
detected language: en
<generator object at 0x00000158849D30A0> [I, I, they, the mis-
sion, the comet, nukes, they, BASH, Gary Talcamont, Stanford,
Lisa Inez, Princeton, their team, I, this, edge, next generation
technology]
(Gary Talcamont, Stanford, Lisa Inez, Princeton)
```

Output 13.7: Detected language, extracted noun chunks, and named entities

Let us now exemplify how to use these three spacy methods on a text document. Code box 13.8 and Output 13.7 demonstrate their usage. In the first two lines of code, we import spaCy and load the English language model to `nlp`. In the next line, we tokenize the document and assign it to the variable `dlu`. We then use the spacy methods described previously on the tokenized document.

Detecting noun phrases is useful when extracting sequences of adjectives followed by nouns or proper nouns. For example, in sociology journal articles, noun phrases such as 'qualitative analysis', 'regression analysis', 'field theory' or 'applying post-structuralist theory' indicate relevant theoretical concepts. Treating those as single tokens helps with the extraction of coherent and meaningful topics. Named entity recognition (NER) links topics or sentiments to persons or organizations. By doing so, it enables us to construct co-appearance networks of named entities or link different actors to topics.

13.4.3 Feature extraction on token level

Beyond methods for feature extraction on document-level, `spacy` provides methods for analyzing individual tokens. These include the following:

- *Lemmatization*, which reduce words to their base form (e.g., running, run, ran, run).
- *Part-of-speech tagging*, which identifies a token's grammatical role (noun, verb, adjective among others).
- *Stop word detection*, which flags common words that may be considered too uninformative for analysis.
- *Morphological analysis*, which examines grammatical features (e.g., tense, case, number)
- *Dependency parsing*, which identifies relationships between words in a sentence.

For example, we want to lemmatize all tokens, remove stop words, and keep only certain part-of-speech tags when we conduct a Latent Dirichlet Allocation (see Chapter 17). We do so to reduce the complexity situated within our text

corpus and, by doing so, increase the chances to extract comprehensive and more clear-cut topics.

To explore token-level methods, let us analyze the first token from our text (Code box 13.9) and print the results of the spacy methods applied for further examination (Output 13.8). As you see, we can extract the token string with the `text` method, the lemma using the `lemma_` method, or the part-of-speech tag using `pos_`. Furthermore, we can analyze the morphology using the `morph` method, and the token dependencies using `dep_`, `head.text`, and `children`.

```
token = tokenized_text[0] # Select the first token from the
tokenized text

print(f"token: {token.text},\n",
      f"lemma: {token.lemma_},\n",
      f"part-of-speech: {token.pos_}\n",
      f"is_stop word: {token.is_stop}\n",
      f"morphology: {token.morph}\n",
      f"dependency: {token.dep_}\n",
      f"dependent from: {token.head.text}\n",
      f"children: {[x for x in token.children]}"
      )
```

Code box 13.9: Overview of spacy methods used for feature extraction on token level

```
token: We,
lemma: we,
part-of-speech: PRON
is_stop word: True
morphology: Case=Nom|Number=Plur|Person=1|PronType=Prs
dependency: nsubj
dependent from: use
children: []
```

Output 13.8: Exemplary output of extracted token features

As you see, the first token `We` is *identified* as:

- we (`lemma`)
- pronoun (`PRON`)
- stop word (`True`)
- nominative plural, first person, and personal pronoun (`Case = Nom`, `Number = Plur`, `Person = 1`, `PronType = Prs`)

- nominal subject as dependency relation (`Dependency: nsubj`)
- *use* as head word (= word that "we" is dependent upon)
- No words dependent on "we" (`children: []`)

To extract token-level features across an entire text, we iterate through all tokens using a list comprehension. Alternatively, we can use a for loop achieve the same. Both approaches are outlined in Code box 13.10.

```
for TOKEN in tokenized_text:
    return(TOKEN)
[TOKEN for TOKEN in tokenized_text]
```

Code box 13.10: Application of a for-loop and a list comprehension to iterate through a tokenized text

However, we advise you to use the list comprehension, as it returns a list of elements instead of printing each token of the variable `tokenized_text`. We can apply each of the methods listed in Code box 13.9 in the list comprehension and, by doing so, extract one or many features of every token stored in the `tokenized_text` variable.

Let us extract a single feature, namely all token lemmas from the text using the `lemma_` method. As you see in Code box 13.11, we can simply use the method on the iterator. This will produce the output seen in Output 13.9.

```
[TOKEN.lemma_ for TOKEN in tokenized_text]
```

Code box 13.11: Example of a list comprehension used to lemmatize all tokens in a tokenized text

Alternatively, we can extract and store multiple token-level features stored in `tokenized_text` simultaneously in a structured format, for example a Pandas data frame. To do so, we can nest a tuple which extracts all token-level features inside a list comprehension.[8] As seen in Code box 13.12 we extract the token string, lemma, part of speech, and employ stop word detection using `(TOKEN.text, TOKEN.lemma_, TOKEN.pos_, TOKEN.is_stop).` For demonstration purposes, we assign this list of tuples to `tokens_list` (Output 10).

```
['we',
 'will',
 'use',
 'spacy',
 'to',
 'tokenize',
 'our',
 'text',
 '\n',
 'and',
 'to',
 'extract',
 'only',
 'noun',
 ',',
 'adjective',
 ',',
 'proper',
 'noun',
 ',',
 '\n',
 'and',
 'verb',
 'from',
 'the',
 'tokenized',
 'text',
 '.']
```

Output 13.9: List of lemmatized tokens

```
tokens_list = [(TOKEN.text, TOKEN.lemma_, TOKEN.pos_, TOKEN.
is_stop) for \
 TOKEN in tokenized_text]
print(tokens_list)
```

Code box 13.12: Application of a list comprehension for the extraction of multiple token features

```
[('We', 'we', 'PRON', True),
 ('will', 'will', 'AUX', True),
 ('use', 'use', 'VERB', False),
 ('spacy', 'spacy', 'NOUN', False),
 ('to', 'to', 'PART', True),
 ('tokenize', 'tokenize', 'VERB', False),
 ('our', 'our', 'PRON', True),
 ('text', 'text', 'NOUN', False),
 ('\n', '\n', 'SPACE', False),
 ('and', 'and', 'CCONJ', True),
 ('to', 'to', 'PART', True),
 ('extract', 'extract', 'VERB', False),
 ('only', 'only', 'ADJ', True),
 ('nouns', 'noun', 'NOUN', False),
 (',', ',', 'PUNCT', False),
 ('adjectives', 'adjective', 'NOUN', False),
 (',', ',', 'PUNCT', False),
 ('proper', 'proper', 'ADJ', False),
 ('nouns', 'noun', 'NOUN', False),
 (',', ',', 'PUNCT', False),
 ('\n', '\n', 'SPACE', False),
 ('and', 'and', 'CCONJ', True),
 ('verbs', 'verb', 'NOUN', False),
 ('from', 'from', 'ADP', True),
 ('the', 'the', 'DET', True),
 ('tokenized', 'tokenized', 'ADJ', False),
 ('text', 'text', 'NOUN', False),
 ('.', '.', 'PUNCT', False)]
```

Output 13.10: List of tuples containing the extracted token level features

Lastly, we assign this list of tuples to a Pandas data frame (Code box 13.13 and Output 13.11). Doing so facilitates further analysis, such as filtering parts of speech in the forthcoming section.

```
column_names = ["token", "lemma", "part_of_speech", "is_stop
word"]
tokens_df = pd.DataFrame(tokens_list,
                         columns = column_names)
```

Code box 13.13: Establishing a Pandas data frame with column names based on the list of tuples created in Code box 13.12

```
          token      lemma part_of_speech is_stop word
0            We         we           PRON         True
1          will       will            AUX         True
2           use        use           VERB        False
3         spacy      spacy           NOUN        False
. . .
23   tokenized  tokenized            ADJ        False
24        text       text           NOUN        False
25         . .        . .          PUNCT        False
```

Output 13.11: Pandas data frame established using the code in Code box 13.13

13.4.4 Overview on the methods used for feature extraction on document and text level

In the last, three subsections, we introduced methods and functions to extract text features on corpus level, document level, and token level. The following Table 13.4 provides a quick, albeit non-exhaustive, overview of the methods and functions provided by `sklearn` and `spacy`, which will come handy in the following chapters.

13.5 Feature selection

Feature selection is the process of identifying and selecting the most relevant features for investigating a given phenomenon (e.g., extracting and interpreting topics from a large text corpus). There are three main approaches to feature selection: filtering methods, wrapper methods, and embedded methods. Filtering methods are applied prior to model training (e.g., prior to the application of topic modeling algorithms). Wrapper methods assess the relevance of an input variable (e.g., a token) in predicting a target variable (sentiment or topics). Finally, embedded methods incorporate feature selection within the training process itself (Pintas et al., 2021, S. 6154).[9] For now, we will focus on filtering techniques. But why should we use feature selection in the first place, and what are the reasons to do so?

Feature selection can either be guided by theoretical considerations or by data-driven considerations. In regards of the former, features are kept by their assumed association with theoretical concepts, for example which feature or dimension represents different orders of worth, forms of power, or forms of capital (e.g., social capital, cultural capital, symbolic capital). Regarding the latter, feature selection aims to:

1. Reduce the computational complexity, and decreasing model training time.
2. Improve prediction accuracy, such as enhancing sentiment or topics detection in previously unknown documents.
3. Increase model interpretability.
4. Enhance its stability by improving reproducibility across datasets (Theng & Bhoyar, 2024, S. 1576).

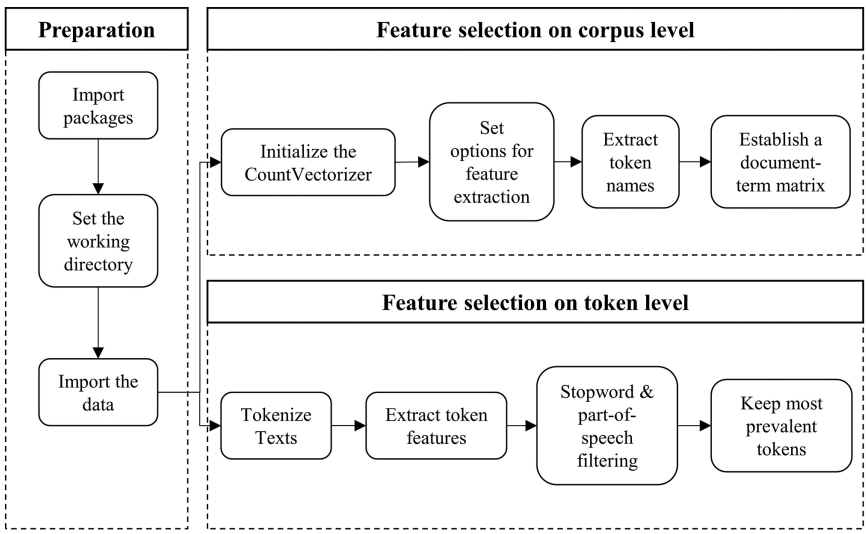
Table 13.4 Overview of Document- and Token-Level Methods (Not Exhaustive)

Method	*Description*	*Level*	*Package*
`CountVectorizer()`	Initiates the pipeline to create an unweighted document-term matrix.	corpus	sklearn
`TfidfVectorizer()`	Initiates the pipeline to create a weighted document-term matrix.	corpus	sklearn
`fit()`	Extracts the vocabulary from a text corpus. Can be used by both.	corpus	sklearn
`transform()`	Transforms the text corpus into a weighted or unweighted document-term matrix.	corpus	sklearn
`get_feature_names_out()`	Outputs the vocabulary in the correct order.	corpus	sklearn
`dtm_matrix.A`	Outputs the document-term matrix as a two-dimensional array. Can be used as input to establish a Pandas DataFrame.	corpus	sklearn
`document.noun_chunks`	Provides a generator object, that contains all noun chunks in a given document.	Document	spacy
`document.ents`	Provides a Generator object containing named entities (e.g., person names, organization names).	Document	spacy
`document.similarity()`	Checks the similarity between two tokenized text documents. Needs a second tokenized document to be passed to the `similarity()`-method.	Document	spacy
`document.text`	Outputs the raw text string.	Document	spacy
`document.lang_`	Most probable language of the document.	Document	spacy
`document.sentiment`	Outputs a positivity/negativity score.	Document	spacy
`token.text`	The raw text as character.	Token	spacy
`token.lemma_`	Extracts the lemmas of the tokens.	Token	spacy
`token.pos_`	The part of speech of the respective token.	Token	spacy
`token.is_stop`	Checks if the token is a stop word.	Token	spacy
`token.morph`	Displays morphological information of the token.	Token	spacy
`token.children`	Creates a generator yielding the tokens dependent upon the token (= lower hierarchical position).	Token	spacy
`token.dep_`	Shows the dependency relation to a token.	Token	spacy
`token.head.text`	Shows the string of the token of which our token is dependent upon (= higher hierarchical position).	Token	spacy

Examples of filtering methods include stop word removal (= eliminating frequently occurring, non-distinctive tokens), or filtering tokens based on parts-of-speech tags (e.g., selecting only nouns, verbs, or adjectives). This is a standard procedure applied prior to topic modeling with generative models (Chai, 2023). Removing stop words increases the informativity of our text corpus, making it easier to distinguish between topics. For example, retaining tokens that appear in at least 25 documents reduces the sparsity of the data. Reducing sparsity also prevents the extraction of topics which describe only a single or a few documents. Feature selection by part-of-speech filtering can be useful for analyzing domain-specific terms, such as sociological theory or sociological methods (see, for example, Wieczorek et al., 2023a).

However, not all automated content analysis algorithms require feature selection as a preprocessing step. In fact, removing features – such as punctuation or specific token types – also removes information that could be useful for model training. This is particularly relevant for deep learning models, such as Transformers, which leverage all available textual information. Nevertheless, we will demonstrate how to perform feature selection on token level using either `sklearn` or `spacy`. While we prune the corpus to the 1,000 most frequently used token lemmas in the corpus using `sklearn`, whereas we will use `spacy` for stop word removal, tokens selection by their part-of-speech tags, and to prune to corpus to the 1,000 most used token lemmas. Figure 13.1 provides a schematic workflow of the techniques used in both filtering procedures.

Figure 13.1 Workflow depicting the feature selection procedures on corpus level and on token level.



To begin, we import the usual packages, including `os`, `pandas` (as `pd`), `text` (as `tx`) from `sklearn.feature_extraction`, and `spacy`. Furthermore, we load the `en_core_web_sm` language model into our workspace using the `spacy.load()` function and assign this model to a variable named `nlp` (similar to Section 13.3). These steps are listed in Code box 13.14.

```
import os
import pandas as pd
from sklearn.feature_extraction import text as tx
import spacy
nlp = spacy.load("en_core_web_sm")
```

Code box 13.14: Importing the necessary packages to perform feature selection on corpus level and token level

Next (see Code box 13.15), we define the root directory and data path (`data_path`), change the working directory with the `os.chdir()` function, and import the data using `pd.read_csv()`. We use the option `sep=";"` and tell Python to that the cells present in the spreadsheet named `2024-11-24_YT_video_comments01.csv` are separated by semicolons. We assign the Pandas data frame to `df` (short for data frame) and drop missing values based on the `textOriginal` column using the `dropna()` method. The `textOriginal` column stores the all documents, in our case comments placed under YouTube social science-videos. The `inplace = True` overwrites `df`. Finally, we use the `sample()` method and draw a sample of `n = 10000` documents from `df`. The `random_state` option sets the starting point of a random process needed to draw the sample. We do so to increase the reproducibility of our code, meaning that every time we run these lines of code on the same computer, we will sample the same 10,000 comments.

```
root = "C:\\Users\\[YOUR_PATH]\\"
data_path = root + "Data\\"
os.chdir(data_path)
df = pd.read_csv('2024-11-24_YT_video_comments01.csv', sep = ";")
df.dropna(subset= "textOriginal", inplace = True)
df = df.sample(n = 10000,
               random_state= 42)
```

Code box 13.15: Initializing packages and loading the dataset for feature selection

We will now filter tokens that appear in at least five documents and retain the 1,000 most common tokens. As you see in the first line of Code box 13.16, we use `sklearn`'s `CountVectorizer` to do so and pass the options `min_df = 5`, and `max_features = 1000`. The former sets the threshold for the minimum number of text documents in which a token must appear to be kept in the data matrix to five, whereas the latter will restrict the number of lemmatized tokens included in the resulting sparse data matrix to 1000. The next three steps we take to establish a document-term matrix are identical to Code box 13.5 in Section 13.4.1, which is why we refrain from explaining them in detail.

```
count_vectorizer = tx.CountVectorizer(min_df = 5,
max_features = 1000)
dtm = count_vectorizer.fit(df.textOriginal)
dtm_matrix = dtm.transform(df.textOriginal).A
dtm_df = pd.DataFrame(dtm_matrix, columns = dtm.get_feature_
names_out())
```

Code box 13.16: Feature selection using sklearn's CountVectorizer with subsequent generation of a data frame containing the document-term matrix

```
list_of_texts = []
for text, comment_id in zip(df.textOriginal, df.comment_id):
    tokenized_text = nlp(text)
    text_list = [(t.text.lower(), t. lemma_, t.pos_, t.is_stop,
    comment_id) for t in nlp(tokenized_text)]
    list_of_texts.append(text_list)
```

Code box 13.17: Extracting token features with spacy, including comment IDs for potential metadata integration

We will now proceed with feature selection on token level with `spaCy`. With the exception of the `for` loop flow control sequence, the code enclosed in Code box 13.17 is similar to the code applied in Code box 13.11, Code box 13.12, and Code box 13.13. The for loop iterates through every document (`text`) and also the comment IDs (`comment_id`). Within the for loop, we employ a list comprehension to extract features on token level as tuples, including the token string itself (`text.lower()`), the token lemma (`lemma_`), the part-of-speech tags (`pos_`), whether it is a stop word or not (`is_stop`). Additionally we also pass the identifier (`comment_id`) of the comment the respective token belongs to the tuple. The identifier allows us to append additional document metadata on our

corpus after feature selection. In the final line of our loop, we use the `append` method to glue the tuple containing the extracted features to the list of texts.

```
list_of_texts_flat = [item for tokenized_text in list_of_texts \
                for item in tokenized_text]

print(len(list_of_texts_flat), len(list_of_texts))
```

Code box 13.18: Process to flat the list of lists into a single-leveled list

```
174519 10000
```

Output 13.12: Comparison of the elements of the unnested list (left) and the list of lists (right)

Since this results in a nested list structure, we flatten it to create a Pandas data frame. We will use a list comprehension (Code box 13.18) to flatten the list of lists this. However, different to a simple list comprehension, we must

1) Iterate through each list, and
2) Iterate through each item (tuple with extracted token features) situated in the respective list.

The first part of the list comprehension in Code box 13.18 (`item for tokenized_text in list_of_texts`) iterates through the lists contained in the `list_of_texts` variable. The second part (`for item in tokenized_text`) corresponds to (2) and iterates through each tuple containing the token features. We assign this flattened list to `list_of_texts_flat`. If we now run the code and compare the number of entries of the flattened list and the list of lists using the `print()` and `len()` statements, we see that the former contains 174,519 elements (= tuples of token features), and the latter 10,000 (=lists) (see Output 13.12).

Now, we will establish a Pandas data frame similar to the one in Code box 13.13 in Section 13.4.3, but with the notable differences, that (1) our data frame contains the tokens of 10,000 documents instead of one and (2) that it also contains a column with the comment ID. As you see Code box 13.19, we assign a `DataFrame()` to the variable we call `text_dataframe` with our `list_of_texts_flat` as input, and provide the column names with the `columns = ["token", "lemma", "pos", "is_stop", "comment_id"]` option.

```
text_dataframe = pd.DataFrame(list_of_texts_flat,
             columns = ["token", "lemma", "pos", "is_stop",
             "comment_id"])
```

Code box 13.19: Conversion of the unnested list into a Pandas data frame

This, however, leaves us with a list of lists, which contain the extracted features per token of each text. If we wish to create a Pandas data frame to perform further feature selection, we must flatten the list of lists first.

Let us now check the unique set of part of speech tags extracted from our corpus. To extract this unique set, we simply use the `set()` function on the `pos` column in `text_dataframe`. To check this set, we use the `print()` function and enclose `set(text_dataframe.pos)` to print the resulting set of part of speech tags (Code box 13.20). As we see in Output 13.13, we extracted, for example, interjections (`INTJ`), punctuation (`PUNCT`), proper nouns (`PROPN`), adverbs (`ADV`) among others. A full list of these universal part of speech tags (sometimes abbreviated UPOS) can be found using the following link https://universaldependencies.org/u/pos/.

```
print(set(text_dataframe.pos))
```

Code box 13.20: Extraction of the part of speech tags stored in the pos-variable of the Pandas data frame named text_dataframe

```
{'INTJ', 'PUNCT', 'PROPN', 'ADV', 'ADJ', 'NOUN', 'SYM', 'AUX',
'PRON', 'PART', 'VERB', 'X', 'SCONJ', 'DET', 'CCONJ', 'ADP',
'SPACE', 'NUM'}
```

Output 13.13: Unique values for the extracted part of speech tags

In the following, we will select only proper nouns (`PROPN`), nouns (`NOUN`), adjectives (`ADJ`), and verbs (`VERB`) and filter stop from our data frame. To set one or many criteria for feature selection in a Pandas data frame, we need to select the column names by using square brackets, in our case `text_dataframe[]`. Here, we can add logical queries, each of which is separated by `()`, or in case of a single logical query, without the round brackets. Between these queries, we can make use of bitwise operators to chain the queries, namely `&` (bitwise AND), `|` (bitwise OR), `^` (disjunctive OR), potentially followed by `~` (negation, e.g., `&~` equates to "and not"). For example, if we want to filter out all stop words in

our `text_dataframe`, we need to type `text_dataframe[text_dataframe.is_stop == False]`.

Remember, that the logical operator `==` tests if a given value on the left side of the equation marks is equal to what is displayed on the right-hand side. This logical statement produces a Pandas series object containing as many Boolean data types (`True` or `False`) as there are rows in the data frame. These Boolean data types not only indicate if the condition was met or not, but indicate if a row of a Pandas data frame should be kept or not when used as input in a logical query (see Code box 13.21 and Output 13.14):

```
print(text_dataframe.is_stop == False)
```

Code box 13.21: Exemplary Logical Queue for Filtering Out Stop Words

```
0          True
1         False
2         False
3          True
4         False
174514     True
174515     True
174516     True
174517     True
174518     True
Name: is_stop, Length: 174519, dtype: bool
```

Output 13.14: Result of the filtering process enclosed in the logical process outlined in Code box 13.21

Testing whether a part-of-speech tag is a noun, a verb, a proper noun, or an adjective is a little bit trickier. We could either test, if a part of speech-tag equals a `PROPN`, or a `NOUN`, or a `VERB`, or an `ADJ`. This would introduce four logical queries, all separated by a bitwise OR operator. A better – and more easy – way to do so is to define a list (named `pos_to_keep` in the first line of Code box 13.22) containing these four parts of speech tags and to check, if the part of speech-tag stored in the `text_dataframe.pos` column is an element of the list. Pandas provides us with the `isin()` method to do exactly this. So the logical query we need to define to filter out all rows without the part of speech-tags included in the list is `text_dataframe.pos.isin(pos_to_keep)`. So (1) to check if a token is a stop word and (2) to filter out all tokens but those with the defined part of speech tags,

we chain both logical queries with a bitwise and operator as seen in lines 2–3 in and assign it to a new variable, we call `filtered_text_dataframe`. To check if the feature selection process was successful, we finally compare the number of rows of both data frames by using two chained `print(len())` functions. As you see in Output 13.15, the data frame contained 174,519 rows prior to filtering and now contains 73,235 rows after filtering.

```
pos_to_keep = ["NOUN", "VERB", "PROPN", "ADJ"]
filtered_text_dataframe   =   text_dataframe[(text_dataframe.is_
stop == False) & \ (text_dataframe.pos.isin(pos_to_keep))]
print(len(text_dataframe),len(filtered_text_dataframe))
```

Code box 13.22: Setting the part of speech-tags to keep and procedure to filter stop words and to keep only the tokens with part of speech listed in the pos_to_keep variable

```
174519 73235
```

Output 13.15: Number of tokens prior to filtering the part of speech tags and after filtering the part of speech tags

Lastly, we select the 1,000 most common token lemmas from the `filtered_text_dataframe` variable similar to the `max_tokens` options as shown in the `CountVectorizer()` in the previous section. To do so, we will chain methods provided by the pandas package to accomplish the following: (1) count the appearances of each token lemma throughout the corpus; (2) sort these lemmas in descending order, and (3) keep the 1,000 most common tokens. As you see in Code box 13.23, we achieve the (1) first step by applying the `groupby(by = "lemma")` method, which causes pandas top group the observations by the unique appearances of the token lemmas listed in the lemma column. We then count the appearances of each lemma using the `size()` method. Different to the `sum()` method, which adds values in the defined columns or over the whole data frame, `size()` simply counts the appearances of the elements in either a single, given column, or the combination of the elements if more than one column is given. To achieve (2) we use the `sort_values()` method and set `ascending = False` to sort the data by descending number of appearances of the token lemmas. To keep (3) the first 1000 token lemmas, we use the `head()` method with the `n = 1000` option. Without this option, pandas would keep the first five rows of a data frame. We could also extract the last rows of our data frame by using the `tail()` method. Finally, to extract the names of these

lemmas, which is stored in the index by the grouping process, we can use the `index` method provided by pandas. We assign the extracted index to a variable we call `top1000_tokens` and use the `print()` function afterwards to output the 1,000 most tokens (see Output 13.16).

```
top1000_tokens = filtered_text_dataframe.\
    groupby(by = "lemma").\
    size().\
    sort_values(ascending = False).\
    head(n = 1000).\
    index

print(top1000_tokens)
```

Code box 13.23: Selecting the 1,000 most frequent token lemmas

```
Index(['❤', 'video', 'good', 'love', 'know', 'thank', 'sir',
'time', 'think',
       '😂',
       . . .
       'chat', 'enemy', 'request', 'terrible', 'industry',
'rha', 'city',
       'worry', 'hun', 'biology'],
      dtype='object', name='lemma', length=1000)
```

Output 13.16: The 1,000 most common token lemmas extracted from our text corpus

13.6 Summary and concluding remarks

In this chapter, we showed you how to perform feature extraction and filter selection procedures on corpus level, document level, and token level to prepare text data for automated content analysis. However, keep in mind that the vast majority of NLP research focuses on English, so these preprocessing steps may not be universally applicable. Consider also that some algorithms, such as BERTopic for topic modeling or many sentiment analysis algorithms, do not require extensive feature selection or extraction. In fact, deep neural network architectures such as BERTopic or RoBERTa, benefit from retaining all available linguistic information (including semantic and morphological information). In contrast, methods such as LDA or relational approaches to topic modeling depend heavily on preprocessing. Nonetheless, with the tools introduced in this

section, you are now well-equipped to explore and prepare text corpora and apply algorithms for automated content analysis.

Notes

1 This is extremely important when we train supervised machine learning models, especially artificial neural networks. These models are extremely sensitive to class imbalances and wrongly labeled data, which are much more likely to appear in uncleaned data.
2 Be aware that a significant share of the time you will spend on topic modeling, sentiment analysis, or GDA will be allocated to data preprocessing and cleaning procedures.
3 A corpus is balanced, "when the size of the subsamples, i.e., the samples of different genres, registers, or varieties, is proportional to the size of the subgroups which the corpus aims to represent" (Beck et al., 2020, S. 67).
4 For example, Latent Dirichlet Allocation requires us to tokenize the text and remove stop words, whereas BERTopic advises us not to do so.
5 The "en_core_web_sm" is trained on telephone conversations, newswire messages, newsgroups, broadcast news, broadcast conversations, weblogs, and religious texts and covers English, Mandarin Chinese, Arabic, and Chinese texts (Weischedel et al., 2013), the constituent-to-dependency conversation, and the WordNet database (Fellbaum, 2005).
6 In case of generative topic models like the LDA, ignoring grammar and syntax has proven to be efficient in discovering coherent topics.
7 It shares all the basic functions and methods, and we also must assign it to a variable, but yields some extra options, like which measures to be applied for the normalization of the document-term matrix.
8 More precisely, we could apply any method or function on the token, as long it accepts the spacy token-class as input.
9 Some regression techniques, such as LASSO (least absolute shrinkage and selection operator) regression are examples of wrapper methods. They aim is to filter out features with little association with the target variable and, by doing so, reduce the complexity of the model used to predict the target variable.

14

INTRODUCING AND EXPLORING A DATASET STATISTICALLY

14.1 Introduction

This chapter has three main objectives. First, we introduce the dataset used for the remainder of this book, namely for topic modeling and sentiment analysis. Second, we demonstrate how to explore the dataset as well as calculate descriptive statistics. Third, we will show you how to visualize the descriptive statistics in a way that is appealing to readers of your papers or thesis, so they can easily understand the statistics. All of these will reappear from now on in regards to different topic modeling approaches.

We will use the **pandas** library (McKinney, 2011) extensively for data exploration, while the **pyplot**-interface of **matplotlib**[1] and **seaborn**[2] are used for data visualization. More specifically, pandas is used to modify the dataset and to calculate descriptive statistics. The **pyplot**-interface and **seaborn** allow us to create clear-cut visualizations.

The dataset we will use in the remainder of this book consists of comments posted under YouTube videos on social science–related topics. It further includes metadata, such as the video IDs, comment IDs, a timestamp (= when the comment was posted), how many likes it accumulated, and how many times the specific comment was commented. It is part of a larger dataset covering comments and metadata on a wide variety of subjects, including sociology, economics, anthropology, psychology, but also chemistry, physics, engineering, and computer science. For demonstration purposes (e.g., group comparisons), we focus on a subset of videos and comments related to sociology, anthropology, economics, ethnology, and psychology. In sum, this dataset includes 4,809 videos uploaded between February 2007 and November 2024, with 117,131 comments posted between December 2010 and November 2024.

The dataset was collected between November 5th and November 9th using the official YouTube API.[3] We drew samples based on search queries spanning a

DOI: 10.4324/9781003496397-17
This chapter has been made available under a CC-BY-NC-ND 4.0 license.

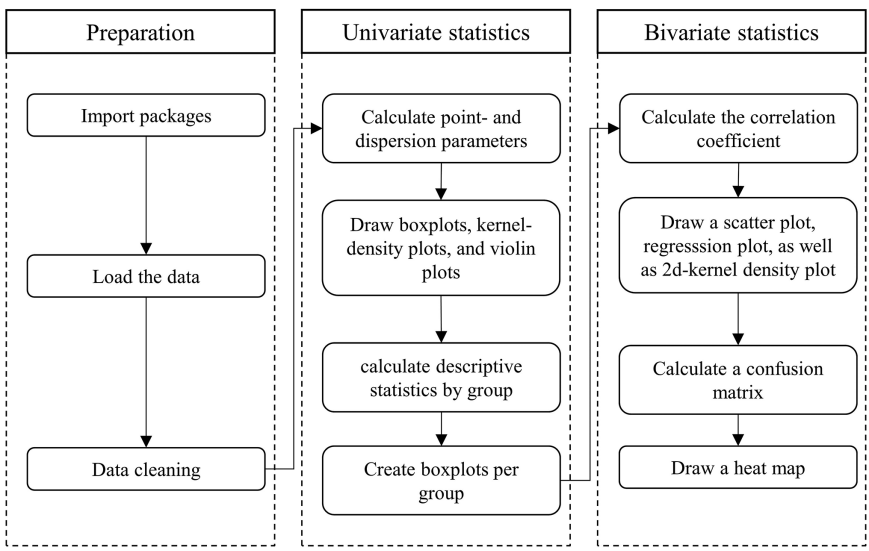
wide range of disciplines[4] and stored the results as comma separated value files (CSV). The video metadata was downloaded separately by querying the video IDs. The comment dataset included comment ID, linked video ID, number of replies, number of likes, and publication date. The video metadata dataset includes video ID, video title, video description (where available), view counts, like count, number of comments, tags, search term used to retrieve the video, and if available, YouTube topic ids, Wikipedia links, topic categories, and language spoken in the video. The video dataset and comment dataset were merged by video ID.

14.2 Setting up our workflow

We now outline a workflow that will guide us through our coding procedure like the one in Section 11.4. In the following, we will analyze a subset of video metadata from social science disciplines.[5] Figure 14.1 depicts the workflow, which contains of three distinct phases:

1) Data Preparation (left), including importing packages, loading the dataset, and performing data cleaning or data preprocessing steps.
2) Univariate Statistics (middle), including the calculation of point- and dispersion parameters to characterize single variables. Furthermore, we demonstrate how to visualize distributions using boxplots, kernel density plots, and violin plots. We also show you how to calculate these measures groupwise.
3) Bivariate Statistics (right), including the calculation of the correlation coefficient and 2-dimensional data visualization techniques, such as scatterplots, regression plots, kernel density plots, and heatmaps.

Figure 14.1 Workflow of the descriptive analysis of video metadata.



14.3 Importing and exploring the dataset

Let us begin with the data preparation task and import relevant packages, setting folder paths, changing the working directory to read the CSV data, and to explore the shape and structure of the data.

14.3.1 Importing the packages and navigating through the folders

We import pandas (commands callable with `pd`), `os`, `re`, `seaborn` (callable with `sbs`), `pyplot` from `matplotlib`, and `numpy` as follows (Code box 14.1):

```
import pandas as pd
import os
import re
import seaborn as sbs
from matplotlib import pyplot as plt
import numpy as np
```

Code box 14.1: Importing the relevant packages and modules to explore the YouTube video metadata

Remember that we used the `os` library to change our working directory and to create folders. We will use the `re` library later to reformat the video upload date (*year-month-day-hour-second* into *year-month*). Finally, `numpy` is used for numerical calculations.

We then proceed with defining the directory names and navigate through the folders to load the CSV file (Code box 14.2):

```
root = "C:\\Users\\[YOUR_ROOT_DIRECTORY]" # change this directory
path = root + "Data\\"
figures_path = root + "Figures\\"

os.chdir(path)
pd.read_csv("social_sciences_videos_with_metadata.csv", sep = ";")
```

Code box 14.2: Setting and changing the working directory, as well as importing the csv file containing the video metadata

We begin by defining the root directory. This is where all the data, figures, outputs, and the workflows are stored. After defining the directories, we call `os.chdir([PATH])` to navigate to the folder on the hard drive where our data is stored. Finally, we read the CSV data table using `pd.read_csv()`. This function

Figure 14.2 Variable explorer with imported Pandas data frame object.

usually requires us to pass the name of the file to be loaded. Nevertheless, we can pass arguments to specify it to our needs. In our case, we pass the `sep = ";"` and define that our cells are separated by a semicolon instead of commas. Executing these lines, the code loads the dataset into the Python workspace, where it becomes visible in the variable explorer as depicted in Figure 14.2.

14.3.2 Exploring the data structure

Let us now apply `pandas` methods to examine the structure of the dataset `df`. The `shape()` method reveals the number of its dimensions. In our case the number of rows and columns (=variables). Furthermore, the `size` method shows the total number of cells present in the current data frame, whereas the `axes` method provides list the index and column names (Code box 14.3):

```
df.shape
df.size
df.axes
```

Code box 14.3: Pandas methods to explore the shape and the size of a Pandas data frame

```
(6543, 17)
111231
[RangeIndex(start=0, stop=6543, step=1),
  Index(['index',    'id',    'title',    'published_year_month',
  'description',
         'defaultAudioLanguage',   'publishedAt',   'channelId',
         'channelTitle',
         'viewCount', 'likeCount', 'commentCount', 'topicCatego-
         ries', 'topicIds',
         'tags', 'videoId', 'search_term'],
         dtype='object')]
```

Output 14.1: Information on the data frame extracted by using the pandas shape, size, and axes methods

As Output 14.1 suggests, our dataset consists of 6,543 rows (= videos), 17 columns (variables), containing 111,231 cells. Understanding the structure of the dataset is crucial when applying `tensorflow` models or when working with `numpy` arrays, or `tensorflow` tensors to specify the input shape to construct artificial neural networks and to train them.

We will now proceed with data cleaning prior to exploring the data. We remove missing values and drop duplicates. For demonstration purposes, we focus on the following variables:

- `viewCount` = number of views per video.
- `likeCount` = number of likes per video.
- `commentCount` = number of comments per video.
- `search_term` = search term (=discipline) for grouping purposes.

To filter incomplete rows, we call Pandas' `dropna()` method, and pass a list containing the four variables the `subset =` option, and replace `df` with `inplace = True` (Code box 14.4):

```
df.dropna(subset = ["viewCount", "likeCount", "commentCount",
          "search_term"],
          inplace = True)
```

Code box 14.4: Pandas dropna method with removal of observations by a subset of columns

14.4 Univariate statistics and suitable graphs

We now proceed with the middle part of the workflow depicted in Figure 14.1, calculating univariate descriptive statistics for the `viewCount`, `likeCount`, and `commentCount` columns.

14.4.1 Levels of measurement

Each of these columns contains a **metric variable**. But what exactly do we mean by 'metric variables'? This term is linked to **levels of measurement**. Based on measurement levels, variables can be categorized as nominal, ordinal, or metric, with metric variables further divided into interval and ratio scales:

- **Nominal variables** denote membership in groups or classes (e.g., religion, assigned sex, or part of speech in regards to tokens). There is no inherent ranking among classes, for example one part of speech is worth more than another.

- **Ordinal variables** introduce a rank order, such as grades in school (from A to F) or sentiments (negative, neutral, positive). Although we can rank the categories, distances between them are not meaningful.
- **Metric variables** allow continuous values (e.g., 1, 1.1, 1.0000001, and so forth), which carry meaningful distances between values. They can be further divided into the following:
  - **Interval variables**, that do not have a true zero point (e.g., normalized values such as degrees centigrade or IQ).
  - **Ratio variables**, which have a true, natural zero point (e.g., zero comments posted under a YouTube video). For example, a text containing 200 words is twice as long as a text with 100 words.

Understanding these differences helps us to choose appropriate statistical measures to analyze the distribution and concentration of the data.

14.4.2 Point and dispersion estimates

Regardless of the differences in measurement levels, **univariate statistics** comprise of the following:

- **Point estimates**, which describe a location in a distribution (= central tendency), including the mean (average of all observed values), median (50% percentile), 25% and 75% percentiles, minimum, and maximum. A percentile is a value, which a given share of our data points does not exceed. In our case, if the median of the number of comments per video is 100, then 50% of all videos were commented 100 times or less.
- **Dispersion estimates** describe the spread of values around a central point (e.g., the mean). For example, the standard deviation (square root of the variance) is a commonly used measure of variation.

Let us now compute descriptive statistics for our three metric variables (Code box 14.5):

```
descriptives_metric_variables = df[["viewCount", "likeCount",
            "commentCount"]].\
            describe().\
            round(2)
```

Code box 14.5: Extracting point and dispersion measures with pandas

To do so, we first use the Pandas indexing operator (`df[]`), to select the relevant columns (`df["viewCount", "likeCount", "commentCount"]`). We then call the `describe()` method on these three selected columns to calculate

the point- and dispersion estimates. The backslash (\) in the code splits the command across multiple lines of code, helping us to keep the code readable and to improve the chances to debug our code. Finally, to round our values to the second decimal place, we chain the `round(2)` method.

If we run our code and inspect the corresponding output (Output 14.2), we may notice the somewhat cryptic scientific notation (e.g., `8.068432e+05`). The `e+` -values represent exponentiation, meaning that a mean of `8.068432e+05` views per video is equivalent to $8.068432*10^5$, or 806,843.2 views on average. Python automatically formats very large or very small numbers this way unless defined otherwise.

```
        viewCount        likeCount    commentCount
count   6.543000e+03       6543.00          6543.0
mean    8.068432e+05      25153.77           774.0
std     4.048049e+06     139657.53          2374.9
min     0.000000e+00          0.00             0.0
25%     4.139000e+03         81.00             3.0
50%     4.139900e+04       1101.00            52.0
75%     3.649580e+05      10236.00           460.0
max     2.357312e+08    7137028.00         60677.0
```

Output 14.2: Summary of views, likes, and comments per YouTube video using point and dispersion measures from Code box 14.5

14.4.3 Boxplots, kernel density plots, and violin plots

Corresponding to the second box in the middle of the workflow depicted in Figure 14.1, we now introduce three types of plots to visualize and explore the data further. These include boxplots, density plots, and violin plots.[6]

Boxplots visualize the dispersion of data using five values, namely the minimum, first quartile (25%), median (50%), third quartile (75%), and maximum (McGill et al., 1978). A box is drawn that captures the range between the first and third quartile (interquartile range, or IQR) with a line indicating the median. Furthermore, so-called *whiskers* extend the box to the maximum or minimum values or, in our case, values that are within 1.5 times the interquartile range. Data points beyond the whiskers are considered as *outliers* and are depicted as dots unless defined otherwise.

Kernel density plots illustrate the distribution of observations (such as the number of views, comments) across values (Rosenblatt, 1956). Such plots help to identify underlying distributions of the data, such as unimodal (= single peak), bimodal (= two peaks), or multimodal (= multiple peaks). The latter two

Table 14.1 Example of a Data Structure for Seaborn to Visualize Multiple Variables

Variable	*Value (e.g., to plot on x-axis)*
viewCount	849464.00
...	
viewCount	85861.00
likeCount	94.00
...	
likeCount	11200.00
...	
commentCount	63.00

might indicate an underlying group structure and, consequently, the need to divide a metric variable into two or more discrete categories or groups to catch the bi- or multimodal structure of the data. Since many machine learning algorithms and statistical methods assume specific distributions of data, understanding these distributions is necessary to select a fitting method and to prevent the algorithms from creating nonsense or biased results.

Violin plots combine box plots and the density plots. They retain the point and dispersion estimates from a box plot but vary the box width to indicate density (= local concentration of observations at specific values), helping us to identify potential groups or clusters (Hintze & Nelson, 1998).

In the following, we will generate boxplots and kernel density plots using `seaborn` and `matplotlib.pyplot`. To do so, we need to pass our dataset in a long-format, meaning that the variable names, group names, and group names are stacked. A suitable data structure for plotting a figure with multiple variables using `seaborn` is shown in Table 14.1.

As you may notice, each of the three metric variables, which were stored as independent columns in `df` (called *wide format*) are stacked in a single column (here: variable), and their corresponding values are stacked in a different column (here: value). This format is called a *long format*. This data structure is needed to visualize multiple variables at once. To transform the data into this format, we use the Pandas' selecting procedure and call the `melt()` method without further options (Code box 14.6):

```
descriptives_long = df[["viewCount", "likeCount",
"commentCount"]].melt()
```

Code box 14.6: Changing the data structure from wide to long format to allow seaborn to draw figures containing multiple variables at once

```
               variable          value
0             viewCount       849464.0
1             viewCount          971.0
2             viewCount       308116.0
3             viewCount        60506.0
4             viewCount        85861.0
               . . .            . . .
19624      commentCount         1391.0
19625      commentCount          791.0
19626      commentCount         3082.0
19627      commentCount        16644.0
19628 commentCount 16644.0
[19629 rows x 2 columns]
```

Output 14.3: Number of views, likes, and comments posted under YouTube social science videos stored in a long format

The data structure shown in Output 14.3 ensures that `seaborn` can effectively generate meaningful plots with multiple variables.

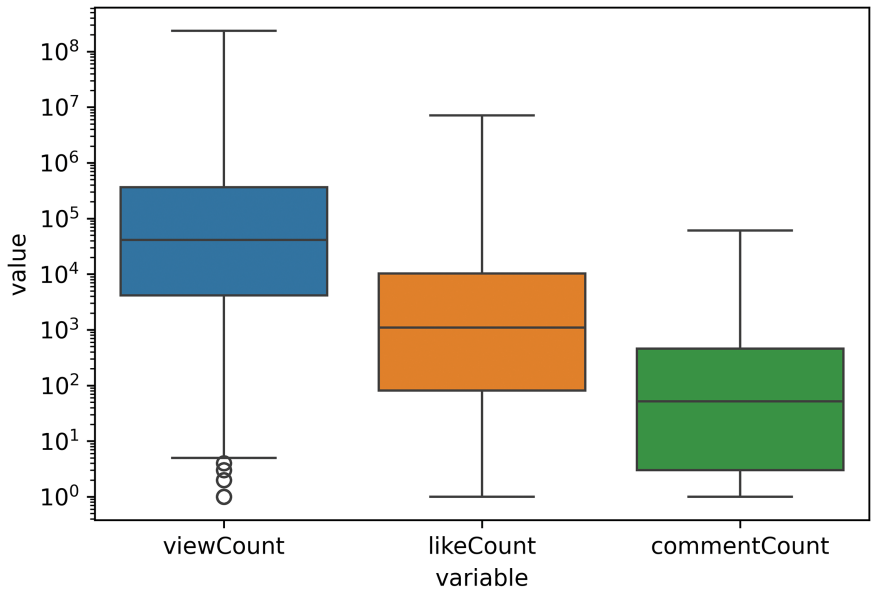
We now proceed with drawing the plots and begin with boxplots. To establish a boxplot, like in Figure 14.3, we call `sbs.boxplot()` as seen in Code box 14.7:

```
sbs.boxplot(data = descriptives_long,
            x = "variable",
            y = "value",
            hue = "variable",
            log_scale = True)
plt.savefig("Boxplot_metric_variables.png",
            dpi = 600)
plt.close()
```

Code box 14.7: Using the seaborn boxplot() function to draw a boxplot

As you may have noticed, we pass our `descriptives_long` dataset defined in Code box 14.6 to the `data =` option. This enables Seaborn to navigate through the data frame and to specify the x- and y-axes of the figure. The `x =`

Figure 14.3 Boxplots of the three metric variables, with log-transformed y-scale.



"`variable`" argument maps the `variable` column (1 = `viewCount`, 2 = `likeCount`, 3 = `commentCount`) on the x-axis. Furthermore, the `y =` "`value`" argument defines the input passed to the y-axis of the boxplot. The hue parameter ensures that each group is colored differently. This enhances the readability of our figure. If categorical variables, stored as strings, are detected, Seaborn automatically assigns a discrete color scale (e.g., red, blue, and green). In turn, if numeric values are detected, Seaborn applies a continuous color scale (e.g., from dark blue to light blue).

Setting the `log_scale = True` transforms the y-axis values using the natural logarithm. This helps to standardize the scale of variables with hugely different valuers (see Output 14.2). Finally, the `plt.savefig()` stores the figure as an image file. Here, we specify the image resolution with `dpi =` 600 (600 dots per inch), which equates to a high resolution. We then close the figure canvas using the `plt.close()`-command.

We will now use the `sbs.kdeplot()` function to create the density plot seen in Figure 14.4. But before we can do so, we must transform the scale as demonstrated in Code box 14.8 first and insert the newly scaled variable into the Pandas data frame.

We do so, as unlike a boxplot, the density plots do not allow for zero values when the logarithmic scale is applied. As log (0) = –∞, we call `descriptives_long.value.add(1)`[7] to add a value of one, and enclose this method with

```
descriptives_long.insert(2,
                         "log_value_plus_1",
                         np.log10(descriptives_long.value.add(1)))
```

Code box 14.8: Adding a new column to the descriptives_long data frame using the pandas insert() method

`np.log10()` to create a log-scale with a base of 10 (e.g., transforming 1000 to 3, 10000 to 4 etc.). The newly generated variable is added as new column to `descriptives_long` with the `insert()` method. The `insert()` method comprises of the following:

1) An **insertion index**, which is an integer that signals where to add the new column.
2) The **column name** of the column to add.
3) An array, series, or list of **values** added as column.

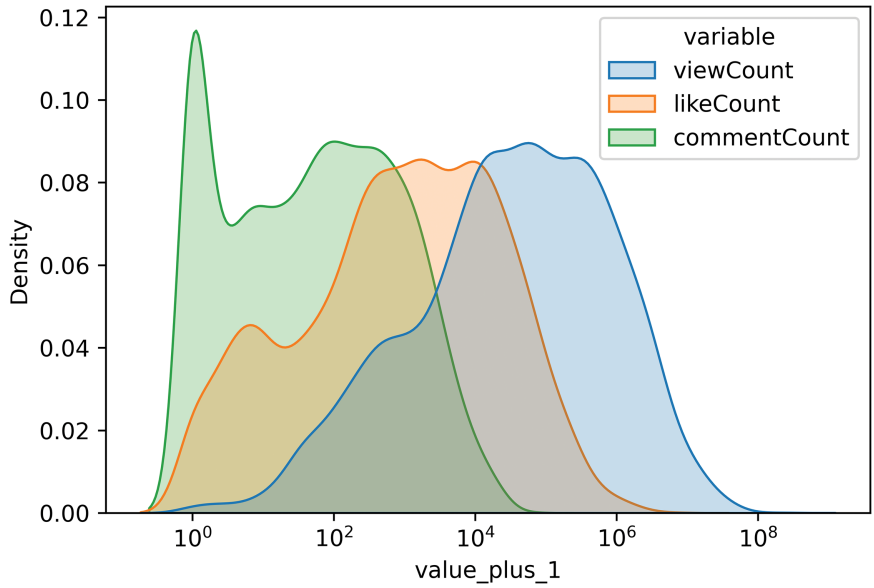
Next, we call the `sbs.kdeplot()` function to create the kernel density plot (Code box 14.9):

```
sbs.kdeplot(data = descriptives_long,
            x = "log_value_plus_1",
            hue = "variable"
            fill = "variable")
plt.savefig("Density_plot_metric_variables.png",
            dpi = 600)
plt.close()
```

Code box 14.9: Creating and saving a kernel density plot

In this case, `x="log_value_plus_1"` maps the log-transformed values to the x-axis, while omitting `y =` ensures a one-dimensional density plot. We will pass a second variable in Section 14.5.3 to create a two-dimensional kernel density plot. The `hue` option differentiates the density curves by variable, whereas the `fill = "variable"` ensures that the area under the curve is filled

Figure 14.4 Kernel density plot of the number of views, likes, and comments per YouTube video.



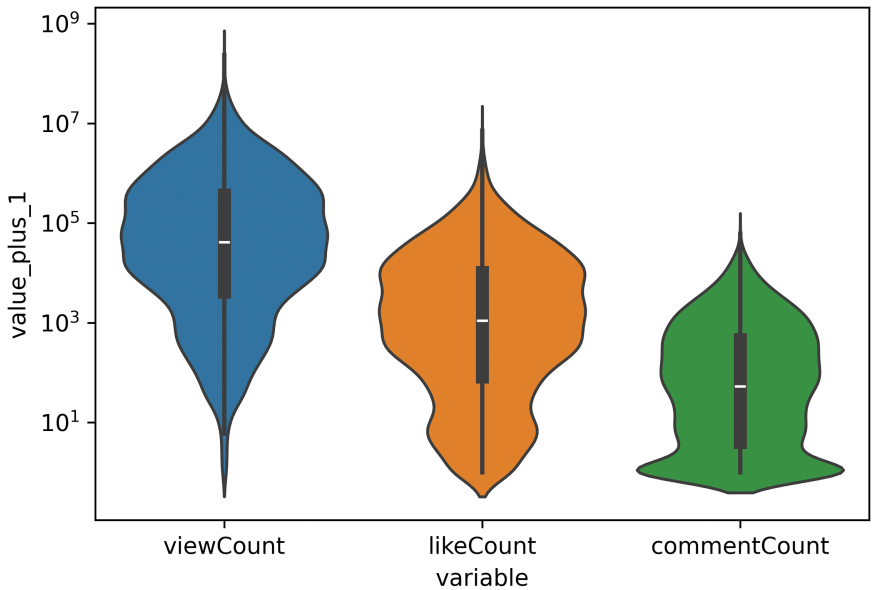
with different colors. We finally save and close the figure by calling `plt.savefig()` and `plt.close()`.

Finally, the `sbs.violinplot()` function call initiates the violin plot in Figure 14.5. As demonstrated in Code box 14.10, the `violinplot()` function is called similarly to `sbs.kdeplot()`.

```
sbs.violinplot(data = descriptives_long,
               x = "variable",
               y = "log_value_plus_1",
               hue = "variable")
plt.savefig("violin_plot_metric_variables.png",
               dpi = 600)
plt.close()
```

Code box 14.10: Establishing and saving the violin plot seen in Figure 14.5 using functions provided by seaborn and pyplot

Figure 14.5 Violin plot of the number of views, likes, and comments per YouTube video.



14.4.4 Drawing boxplots for grouped data

But what if a variable includes so many groups to investigate that a single graph may become overcrowded and incomprehensible? In such cases, we can use `sbs.FacetGrid()`, which allows us to create multiple subplots. These subplots are arranged systematically, either by row or by column. However, we must prepare the data a little bit more for the `sbs.FacetGrid()` to work properly (Code box 14.11):

```
languages_discipline = df.groupby(by = ["defaultAudioLanguage",
                "search_term"])\
                .size()\
                .reset_index(name = "count")\
                .sort_values(by="count", ascending = False)

languages_discipline = languages_discipline[languages_discipline
["count"] >= 20]
```

Code box 14.11: Data preprocessing before plotting with seaborn's FacetGrid()

The Pandas `groupby()` method aggregates the dataset by language and discipline. The chained `size()` method counts the number of occurrences (= videos) by language (`defaultAudioLanguage`) and discipline retrieved via the YouTube API search (`search_term`). Next, the `reset_index()` method converts the index to columns, whereas the `name = "count"` option defines the name of the column where the values are stored. If we do not pass a name, then this new column is automatically numbered `0`. The data frame is finally sorted in descending order, as indicated by the method call `sort_values(by="count", ascending = False)`. In our case, we will draw a bar plot by discipline as seen in Figure 14.6. However, we will have to preprocess the data a little bit to create bar plots by discipline. Finally, we use a Pandas selecting procedure and keep only categories (language by discipline) which occur at least 20 times in our grouped data frame.

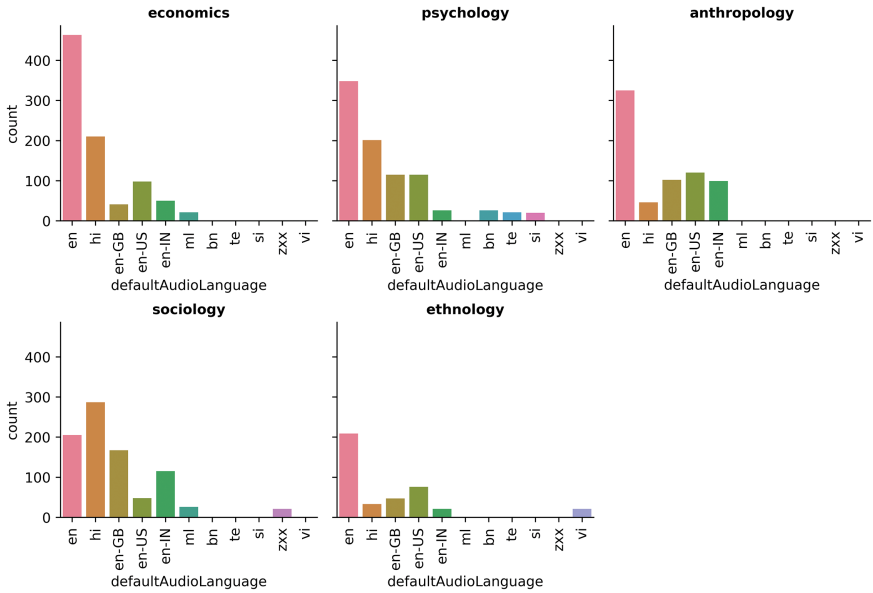
Let us now draw the bar plots depicted in Figure 14.6 using the following code (Code box 14.12):

```
graph = sbs.FacetGrid(data = languages_discipline,
                      hue = "defaultAudioLanguage",
                      col = "search_term",
                      col_wrap = 3)
graph.map(sbs.barplot,
          "defaultAudioLanguage",
          "count")
graph.set_titles("{col_name}", fontweight = "bold")
[plt.setp(ax.get_xticklabels(), rotation=90) for ax in graph.
axes]

graph.tight_layout()
```

Code box 14.12: Code used to draw a bar chart with the language used in the video for each discipline contained in the dataset

We call Seaborn's `FacetGrid()` function to initiate a multi-plot figure and assign this to the `graph` variable. Here, we pass the grouped data frame to `data = languages_discipline`, color the bar charts by language spoken in the videos with `hue = "defaultAudioLanguage"`, and pass `col = "search_term"` to define that the bar charts should be plotted column-wise (from left to right), while `col_wrap = 3` defines how many bar charts should be plotted per row. The following `map()` method applies the sbs.barplot() function onto every subplot, while "`defaultAudioLanguage`" defines the values of the x-axis, and "`count`" the values plotted on the y-axis.

Figure 14.6 Bar plot displaying the number of YouTube videos per language and academic discipline.



The subsequent `graph.set_titles("{col_name}", fontweight = "bold")` formats the title of each subgraph. The "`{col_name}`" ensures that only the values addressed by the `col =` option are displayed. By default, seaborn would print "variable name = variable value", in our case, for example, `search_name = sociology`. The option `fontweight = "bold"` sets the font style to bold. The following list comprehension ensures that the x-axis labels are rotated by 90 degrees. The `ax.get_xticklabels()` function extracts the names of the x-tick labels, whereas the `plt.setp()` function changes the properties of a plot object. Finally, we fit the figure to the width and height of the figure canvas with `tight_layout()`.

14.5 Bivariate statistics and suitable graphs

In this section, we focus on calculating and visualizing **bivariate statistics for interval- and ratio-scaled variables**, specifically **correlation coefficients**, and suitable graphics to depict the association between two variables. This corresponds to the right-hand box in the workflow depicted in Figure 14.1.

14.5.1 Calculating the correlation coefficient

A correlation coefficient quantifies the association between two variables, ranging from −1 to 1. A value of −1 indicates a perfect negative association between two variables (the higher the values of variable 1, the lower the values

of variable 2), while a value of +1 implies a perfect positive association, and 0 equals no association.

As demonstrated in Code box 14.13, we calculate the correlation coefficient using the pandas `corr()`-method. In the first step, we apply a logarithmic transformation with base 10 (`np.log10()`) to the three metric variables after adding 1 via the `add()` method. Like in Code box 14.8, we do so to avoid issues with $\log(0) = -\infty$ and data loss. By default, Pandas does calculate Pearson's R ("`pearson`"), but Spearman's rank correlation ("`spearman`") and Kendall's tau ("`kendall`") are also available. The latter two are useful for ordinal variables, as they are based on ordered ranks. Since `viewCount`, `commentCount,` and `likeCount` are metric variables, we apply Pearson's R (`method =` "`pearson`"). Finally, we chain `round(4)` to round the correlation coefficient to four decimal places. This returns a pairwise correlation matrix similar to Figure 14.6.

```
log_data = np.log10(df[["viewCount", "commentCount",
                        "likeCount"]].add(1))
log_data.rename(columns = {"viewCount" : "log_viewCount",
                        "commentCount" : "log_commentCount",
                        "likeCount": "log_likeCount"},
                inplace = True)
correlations = log_data.corr(method = "pearson").round(4)
```

Code box 14.13: Code used to calculate Pearson's correlation coefficient on the logarithmized number of views, comments, and likes of the YouTube videos under observation

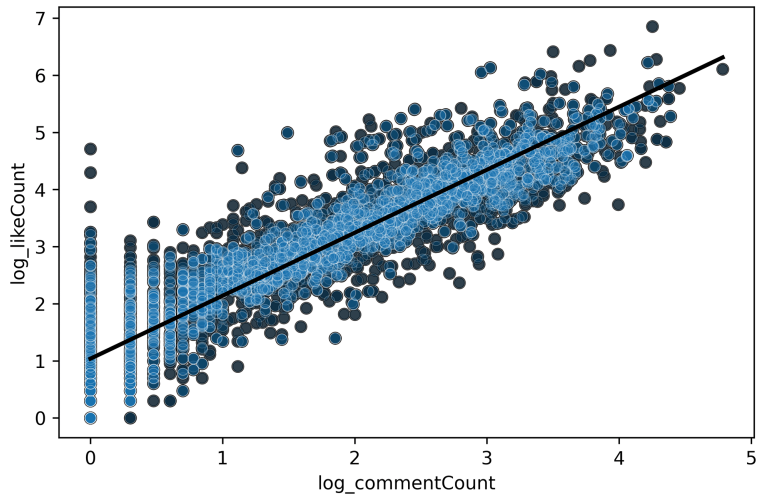
```
                log_viewCount  log_commentCount  log_likeCount
log_viewCount          1.0000            0.8908         0.9659
log_commentCount       0.8908            1.0000         0.9213
log_likeCount          0.9659            0.9213         1.0000
```

Output 14.4: Pairwise correlation matrix of the logarithmized number of views, comments, and likes of the observed YouTube videos

14.5.2 Scatterplots and regression plots

We will now visualize the relationship between the number of comments and the number of likes per video. To do so, we will combine a scatter plot and a regression plot. To unify both plots in a single figure, we call `matplotlib.subplots()` and combine seaborn's `scatterplot()` and regression **plot** `regplot()` functions. This allows us to draw multiple subplots within the same figure canvas as depicted in Figure 14.7.

Figure 14.7 Scatterplot and regression plot highlighting the association between the logarithmized number of comments and logarithmized number of likes per YouTube video under observation.



To achieve this, we use the code shown in Code box 14.14:

```
fig,ax = plt.subplots()
sbs.regplot(data = log_data,
            x = "log_commentCount",
            y = "log_likeCount",
            ax = ax,
            color = "black")
sbs.scatterplot(data = log_data,
            x = "log_commentCount",
            y = "log_likeCount",
            ax = ax,
            color = "steelblue",
            alpha = 0.25)

fig.tight_layout()
fig.savefig("scatterplot_and_regression_plot.png", dpi = 600)
plt.close()
```

Code box 14.14: Syntax used to create a scatterplot and regression plot highlighting the association between the logarithmized number of comments and logarithmized number of likes per YouTube video under observation

In the first line, we initiate the figure (fig) and the axes object (ax) using `plt.subplots()`. We then initiate Seaborn's `regplot()` and `scatterplot()` functions and plot the log-transformed number of comments and likes onto the figure canvas. Both functions use a similar structure and need us to pass the following options:

1) The pandas dataset (`data =` log_data).
2) The variable to be plotted on the x-axis.
3) The variable to be plotted on the y-axis.
4) The same axes (ax) to unify both plots.

We furthermore change the colormaps using the color of the points and the regression line with the color-option and increase the transparency of the points in the scatterplot by alpha = 0.25. We finally fit the figure to the figure canvas with `fig.tight_layout()` before saving and closing the figure with `fig.savefig()` and `plt.close() respectively`.

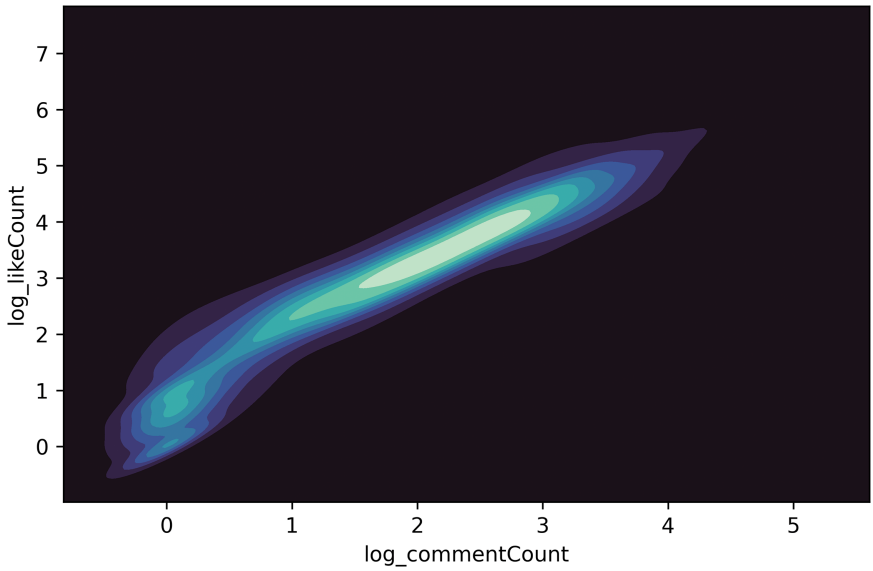
14.5.3 The kernel density plot for two-dimensional data

In a next step, we can visualize the density of the points depicted in the Seaborn scatterplot() using a two-dimensional kernel density estimate plot, or `kdeplot()`. This is the same function introduced in Section 14.4 to draw Figure 14.4. The following code (Code box 14.15) is used to generate Figure 14.8:

```
fig,ax = plt.subplots()
sbs.kdeplot(data = df,
            x = "commentCount_1",
            y = "likeCount_1",
            cmap = "mako",
            thresh = 0,
            levels = 10,
            fill = True,
            ax = ax)
fig.tight_layout()
fig.savefig("comments_likes_density.png", dpi = 600)
plt.close()
```

Code box 14.15: Syntax to create a two-dimensional kernel density plot highlighting the association between the logarithmized number of comments and logarithmized number of likes per YouTube video under observation

Figure 14.8 Two-dimensional KDE plot illustrating the association between logarithmized comments and likes.

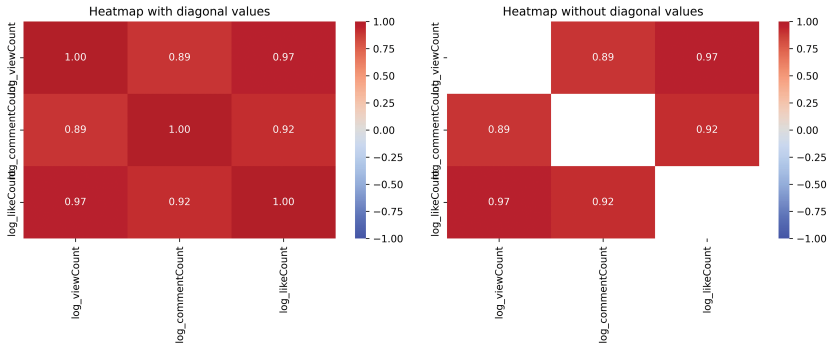


Despite using the same function, we witness a few differences:

1) We added a second variable to `y =`. This allows us to transform the density curves to a sort of map, where brighter colors indicate more dense regions in the two-dimensional space.
2) We pass a colormap to the `cmap =` option.[8] This allows us to distinguish between regions with lower and higher density.
3) The `thresh = 0` option allows us to estimate the density of empty regions.
4) The `fill =` True option fills the regions with colors, otherwise only contour lines would be plotted.
5) The `levels =` option defines the number of density levels to be plotted.

The advantage of such a two-dimensional kernel density plot is that it highlights patterns present in the data, especially if there are many thousands or even millions of observations. Furthermore, we can use such plots to identify groups or outliers in our data. For example, kernel density plots can be combined with relational methods (Baier, 2017, p. 110–112), or in regards to the topic spaces generated by deep neural networks, such as BERTopic (see Angelov & Inkpen, 2024) in Section 18. This allows to identify regions with outlier topics (e.g., topics generated if text documents contain words in different languages), or to identify regions crowded with (potentially) very similar topics.

Figure 14.9 Heatmap of the pairwise correlations between the number of views, comments, and likes of the YouTube videos under observations with and without diagonal.



14.5.4 Drawing a heatmap

Finally, we will show you how to draw heatmaps. A **heatmap** visualizes the association between multiple variables. In our case, we use it to visualize the correlation coefficients between the number of views, number of comments, and likes of a YouTube video with social science content. Since correlation values range from −1 to +1, we usually choose a diverging colormap (e.g., Matplotlib's `coolwarm`) to visualize the associations. In a diverging colormap, negative values appear blue, positive values red, and values near zero as white. Figure 14.9 encapsulates two heatmaps, one with a diagonal (left), and one without a diagonal (right). The former includes self-correlations, whereas the latter emphasizes meaningful pairwise correlations. We will use heatmaps in Section 16.4.2 to visualize the extent to which our deep learning model (DistilBERT) was able to predict sentiments correctly.

```
fig,(ax1, ax2) = plt.subplots(1,2)
fig.set_size_inches(12,5)
## plot heatmap with diagonal values
sbs.heatmap(correlations,
        annot = True,
        fmt = ".2f",
        cmap = "coolwarm",
        vmin = -1,
        vmax = 1,
        ax = ax1)
ax1.set_title("Heatmap with diagonal values")
```

Code box 14.16: Initialization of the figure canvas in which the heatmaps shown in Figure 14.9 were stored. Furthermore, the code used to create the left of the two heatmaps is displayed

Let us now reproduce Figure 14.9 step by step. To do so, we begin with creating the first heatmap on the left (Code box 14.16):

The Pyplot `subplots(1,2)` function in the first line of code initiates a figure canvas (`fig`) and two subplots, which are stored in the tuple `(ax1, ax2)`. The first integer denotes the number of rows, the second the number of columns to be plotted. In the following, we draw the heatmap with diagonals on the left subplot (`ax1`). To do so, we call `sbs.heatmap()` and assign the following parameters to the function:

- `data = correlations` the dataset used to establish the heatmap.
- `Annot = True` tells Python to annotate each cell (here: with Pearson's correlation coefficient).
- The `fmt = ".2f"` option adjusts the format displayed per cell, in this case the first two decimals.
- Next, the cmap = "`coolwarm`" defines the colormap to separate low from high values.
- `vmin = -1` links the minimum value with deep, dark blue, and
- `vmax = 1` links the highest possible correlation value with a vibrant, red color.
- `ax = ax1` defines that the first heatmap should be plotted on the subplot stored in `ax1`.

We change the title of the left subplot using the `ax1.set_title.`

Let us now reproduce the figure on the right hand side of the plot. To do so, we must first mask the diagonal. We achieve this by creating a masking matrix similar to Output 14.5 and use the following code to achieve this (Code box 14.17):

```
diag = np.zeros_like(correlations)
for cell in range(len(diag)):
 diag[cell,cell] = 1
```

Code box 14.17: Preparing a two-dimensional numpy array with the same shape as the matrix where the pairwise correlations are stored

```
array([[1., 0., 0.],
       [0., 1., 0.],
       [0., 0., 1.]])
```

Output 14.5: Numpy array needed for masking the diagonal of the heatmap

The first line of code creates a matrix filled with zeros in the shape of the correlation matrix. A zero within a masking matrix signals Python to show a cell, whereas a one masks the cell (=leaves it empty). The `for`-loop iterates through the diagonal with cells indexed `[0,0]`, `[1,1]`, and `[2,2]`, and assigns these cells a value of 1. As demonstrated in Code box 14.18, this matrix is

then passed to the `mask = diag` option. This visualizes the heatmap on the right-hand side of the plot.

```
sbs.heatmap(correlations,
            annot = True,
            fmt = ".2f",
            cmap = "coolwarm",
            vmin = -1,
            vmax = 1,
            mask = diag,
            ax = ax2)
ax2.set_title("Heatmap without diagonal values")
fig.tight_layout()
fig.savefig("heatmaps.png", dpi = 600)
plt.close()
```

Code box 14.18: Syntax needed to create the second heatmap on the right of Figure 14.9

Notes

1 https://matplotlib.org/3.5.3/api/_as_gen/matplotlib.pyplot.html provides an overview on the methods integrated into the pyplot-interface, and https://matplotlib.org/cheatsheets/#handouts provides you with cheat sheets and handouts that briefly introduce into the most relevant commands and methods of pyplot.

2 The complete seaborn documentation is available under: https://seaborn.pydata.org/, and a cheatsheet is available under: https://github.com/jramshur/Coding-Cheat-Sheets/blob/master/Python%20for%20Data%20Science%20-%20Cheat%20Sheet%20-%20Seaborn.pdf

3 Earlier versions were established in March 2024 and September 2024. Some errors in regards to the data downloaded were identified during applied data cleaning procedures, which is why we had to draw a sample of videos repeatedly and ensure to minimized errors by hand-checking the data.

4 The full list of disciplines queried is "sociology", psychology", "economics", "ethnology", "anthropology", "social work", "criminology", "pedagogy", "journalism", communication science", "communication studies", "public administration", "medicine", "public health", "pharmacology", "nursing", "physics", "chemistry", "mathematics", "astronomy", "biology", "geology", "computer science", "engineering", "materials science", "environmental science", "archaeology", "cultural studies", "gender studies", "regional studies", "postcolonial studies", "history", and "philosophy".

5 We did so to prevent us from using too much RAM and crashing the computer. The Python program code was executed on a Dell Precision 5560 Working station with 32 Gigabytes of RAM, with eight Intel I9-cores which handle 16 processes in parallel. Windows 11 was used as operating system.

6 We could also use histograms. However, after inspection, the data was dispersed so much that the histogram did not provide useful information.

7 In pandas, there are other methods available, such as .div() = divide, .sub() = subtract, .mul() = multiplication etc. https://pandas.pydata.org/docs/reference/frame.html provides a comprehensive overview on the methods that can be cast upon a pandas data frame.

8 An overview of the seaborn color palettes can be accessed using the following link: retrieved February 10, 2025, from https://seaborn.pydata.org/tutorial/color_palettes.html.

15

AUTOMATED CONTENT ANALYSIS USING RELATIONAL METHODS

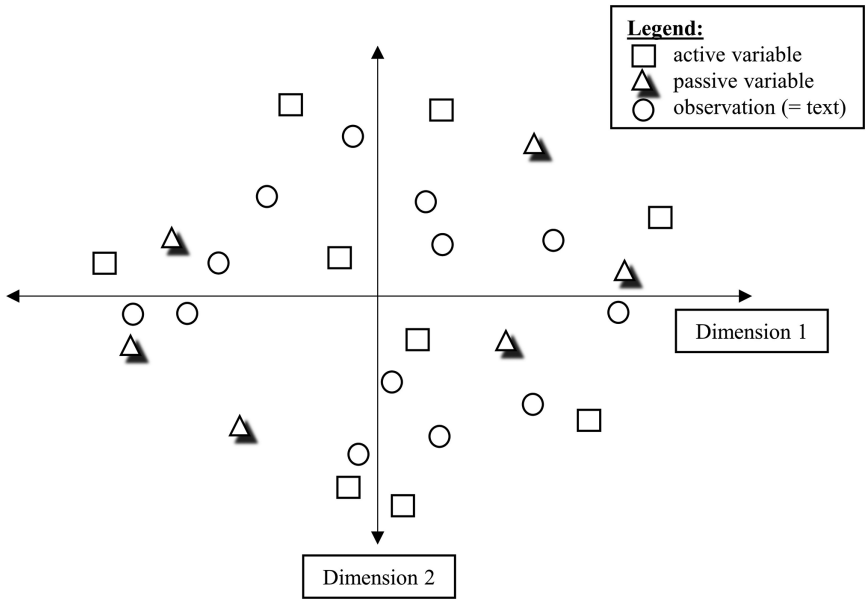
15.1 Geometric data analysis as a family topic modeling methods

At its most abstract level, geometric data analysis (GDA) is twofold: a family of methods[1] suited to extract structures from complex, multifaceted data, as well as a methodological perspective (Lebaron, 2021, S. 127). Regarding the former, GDA consists of correspondence analysis (CA), multiple correspondence analysis (MCA), principal component analysis (PCA), and multiple factor analysis (MFA) (Blasius et al., 2019). The methodology of GDA is based on Bourdieu's habitus field theory (used, for example, in Bourdieu, 1984, 1988, 1996) and was developed in parallel (Lebaron, 2018). In this vein, society at large is interpreted as a social space with fields therein. A space is constructed from the volumes and distributions of different forms of capital (economic, social, cultural, and symbolic), whereas fields consist specifically of actors (a) struggling for the accumulation and worth of capitals, while (b) following tacit rules and values how to correctly take part in these struggles. Despite the focus on capitals and struggles among actors, GDA has been employed successfully to integrate different types of data, including text, to uncover underlying symbolic dimensions, which are then subjected to qualitative interpretation (Blasius & Greenacre, 2014; Hladík & Renisio, 2025; Schmidt-Wellenburg & Schmitz, 2023; Schmitz et al., 2018; Wieczorek et al., 2023; Wieczorek & Malzahn, 2024).

In case of automated content analysis, GDA tries to find structures, so-called *dimensions*, in the data. Each dimension consists of one opposition between variables (tokens related to science versus religion) and observations (text documents as carriers of the tokens). Content-wise, these oppositions between the variables and observations situated on the poles of an axis can be interpreted as topics. This results in a structure in which tokens are variables/columns, and texts are observations. In other words, GDA relies on transforming a text corpus

DOI: 10.4324/9781003496397-18
This chapter has been made available under a CC-BY-NC-ND 4.0 license.

Figure 15.1 Exemplary depiction of a space extracted by GDA, called biplot, including active variables, passive variables, and observations.



into a document-term matrix. As shown in Figure 15.1, the extracted dimensions are mathematically translated into a Euclidean space. As such, we can interpret the distances and positions of both variables and observations if plotted onto a system of coordinates. Variables in close vicinity co-occur systematically across the observations. In our case, this indicates that tokens co-occur systematically throughout the corpus. In turn, texts, that are in proximity are similar in regards to the tokens they use. But how do CA, MCA, PCA, and MFA differ?

We will start with PCA, as it is the oldest of the four named methods. **PCA** was introduced by Pearson (1901) and seeks to reduce the dimensionality and thus complexity of the data. *Dimensionality reduction* simply means that we want to preserve the complex structure present in the dataset (= its variance and the possible combinations of values across the variables), but in the simplest way possible. PCA works for metric data (such as the number of likes, shares, or views of a video on a social media platform) and assumes that the extracted dimensions are orthogonal, meaning that these dimensions are not correlated. In case of the tokens and texts, PCA tries to find topic dimensions (two per axis, one per pole). It is a direct precursor to CA, MCA, and MFA (Abdi & Williams, 2010). It is not robust to outliers, assumes a Gaussian distribution of the data, and linearity (=additivity) among the variables (Greenacre et al., 2022).

CA was introduced in 1963 by Benzecri (Le Roux & Rouanet, 2010, p. 3). It is similar to PCA but is applied where relative values and relative differences between observations are of interest (Greenacre, 2010). Different from PCA,

CA deals well with sparse, non-normally distributed data, meaning that it can deal with data where many entries are either 0 or missing values. This is usually the case with text data, especially when many tokens are used only in a small number of text documents. CA can be used to describe and visualize relations of count data (e.g., number of appearances of tokens across text documents), ratio-scaled data (e.g., income), compositional data (percentages), or binary data (where all the variables are coded either 1 or 0). For the model to work, all data needs to be of the same type.

MCA is used to extract interpretable structures from the categorical data. This is the case, when a text contains different categories of tokens (such as "appears in text", "appears in text more than x times") and text metadata, for example year of publication, language, type of speaker. To unfold a space, the data matrix is converted so each column represents a category, and the values of the columns will equate to one if the category is present in a text and zero otherwise. This is the so-called Burt indicator matrix (Husson & Josse, 2014). Despite ignoring the full information regarding the distribution of tokens and the frequency of their appearances across the texts, MCA allows us to extract topic dimensions purely on the appearance and non-appearance of tokens.

Finally, **MFA** is used to find latent structures, such as topic dimensions, in mixed data types, such as categorical data (e.g., text metadata) and metric data (e.g., times tokens appear in texts) (Escofier & Pages, 1994). It can also be used to relate the distributions of extracted topics (e.g., via LDA) with metadata (see Wieczorek et al., 2023 for an example) to substantiate the interpretation of topics and to link them with authors, publishers, or the publication date of the text. To achieve this, MFA extracts dimensions in a two-step process. In a first step, it uses either MCA on groups of categorical variables present in the data, or PCA on groups of metric variables present in the data. Then, it uses the Eigenvalues (= scaling factors of the data per dimension) of the first dimension per variable group to normalize the data. This normalized data is then subjected to another PCA and, by doing so, creating a representation of the initial data, as well as the data groups underlying the normalization.

In the following, we will use specific MCA (Le Roux & Rouanet, 2010) to reduce high-dimensional statistical spaces, consisting of (a) many tokens and (b) of selected metadata. It is a type of MCA, in which only a share of the tokens is used to find topic dimensions in the text corpus, whereas others are purely illustrative. The resulting dimensions are then subjected to qualitative interpretation and represent either one or two topics.

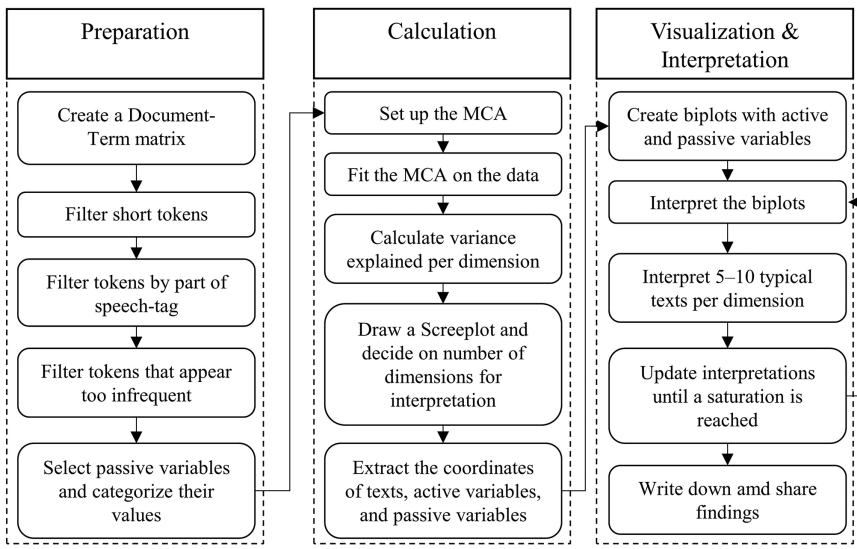
The variables used in specific MCA are active and passive variables (Greenacre & Blasius, 2006). **Active variables**, or rather the distribution of variables across the data, construct the dimensional space. The dimensional space, in turn, explains the variance present in the dataset (e.g., the degree to which tokens co-appear systematically across texts). The greater the variance explained, and the better the extracted dimensions account for the complexity of the text data, as well as metadata, and the more clearly topics emerge from the text corpus. **Passive variables**, in turn, are illustrative and can be seen as

casted onto the dimensions like a kind of shadow (Blasius & Mühlichen, 2010). They illustrate properties that co-occur with the active variables, for example, sociodemographic variables in the study of lifestyles (Bourdieu, 1984; Le Roux et al., 2008) or metadata (e.g., Wieczorek & Malzahn, 2024). They are placed in the extracted dimensions based on the combination of active variables seen in the observations (= text documents). In our case, this enables us to investigate, if some topic dimensions are more liked, commented, or tied to specific points in time.

Despite all this computational effort, we must still interpret our findings manually, as this technique produces results in the sense of the distant-reading methodology (Moretti, 2000). Fortunately, MCA allows us to visualize the dimensional structure onto a simple graph (called a biplot, as seen in Figure 15.1), which then facilitates our interpretation (Greenacre, 2012). This plot permits us to simultaneously map the texts on one side, and text contents and metadata on the other. Additionally, the fact that we can locate the position of different texts within this space enables us to select those texts that are most illustrative for each extracted topic.

In the remainder of this chapter, we follow the workflow depicted in Figure 15.2. Overall, we start with some data preparations consisting of (1a) the conversion of a Pandas data frame into a document-text matrix, (1b) filtering out tokens with a single character, (1c) filtering tokens by part-of-speech tag, (1d) filtering tokens that appear too infrequent for further analysis, and (1e) by selecting a small number of passive variables from the document metadata and categorize their values. Next, we perform the MCA and create a low-dimensional

Figure 15.2 Workflow applied for using the MCA on text data.



representation of our data by (2a) setting up the MCA algorithm; (2b) fitting the data; (2c) calculating the variance explained by each extracted dimension; (2d) drawing a scree plot and using the elbow-method to decide how many dimensions should be kept; (2e) extracting the positions of the active variables, passive variables, and texts (= observations) on the dimensions kept for further investigation. Finally, (3a) we create biplots to visualize the distribution of active and passive variables to (3b) interpret the dimensions manually. We then step back from the biplots and (3c) interpret three to five documents per dimension to substantiate the visual interpretation of the topics. We (3d) reiterate these steps until we reach a theoretical saturation and (3e) finally write down and share the findings.

15.1.1 Create a document-term matrix

In this subsection, we will follow the left part of the workflow depicted in Figure 15.2. As mentioned earlier, the methods that comprise GDA require a document-term matrix, which is a structured data format that enables us to extract topic dimensions for qualitative interpretation. Table 15.1 provides a minimum example of the information needed to conduct a specific MCA.

In this example, the document-term matrix consists of a numbered index (from 0 to M, where 0 follows Python's indexing convention, and M represents the last observation, e.g., 999 if we analyze 1,000 texts) and a currently undefined number of columns N that indicates the number of tokens per text. Please keep in mind that an index cannot only contain consecutively ordered numbers but may also contain different information, for example the comment ID. In fact, we can pass any information we deem suitable as an index to a Pandas data frame, if it helps to distinguish between observations – in our case: comments. To the right of the index, you will find the actual data columns. In other words, each column represents a token, while the cells list the times a token appears per document, which is why this format is called *document-term matrix*.

As you may notice, this data format strips the tokens from their context. This format is known as the "bag of words" format. It assumes that knowing about the distribution of our tokens distributions across documents is sufficient to assign meaningful topics. This assumption rests on the idea that, for example, texts about different topics, such as astrophysics, use different tokens than a

Table 15.1 Example of a Document-Term Matrix

Index	*Token1*	*Token2*	...	*Token N*
1	1	0	...	4
2	3	2	...	0
...	...	...	...	...
M	1	0	...	0

Table 15.2 Document-Term Matrix Extended by Document Metadata

Index	*Topic*	*Comment likes*	*Token1*	*Token2*	...	*Token N*
1	Sociology	21	1	0	...	4
2	Physics	100	3	2	...	0
...			...	...	...	...
M	Chemistry	51	1	0	...	0

religious topic. For example, the former could contain tokens like "Hawking radiation", "black hole", "neutron star", "neutrino", etc., whereas the latter could be characterized by tokens such as "bible", "god", "heaven", and so on. Consequently, we will use only the number of appearances of the tokens per document to construct our topic space (CA, MFA, PCA), or we categorize the appearances of the tokens (e.g., token appears: yes/no).

Furthermore, additional metadata, such as comment timestamps, the number of likes, or the sentiment of the comment (see Section 11) can enhance topic modeling interpretation. Combining text data and metadata results in a mixed data structure. In this sense, our data structure is not strictly a *document-term-matrix*, but a *metadata extended document-term matrix* (see Table 15.2).

We will now turn our attention to the technical part of this section, namely how to prepare our data for MCA. We begin with loading the packages, spaCy's language model, and the data (Code box 15.1).

Here, we import `re`, `os`, and `pandas` (with alias `pd`) for data handling. We proceed the `text` module (as `tx`) from `sklearn.feature_extraction`, `spacy`, the `detect` module from `langdetect`, and the `en_core_web_sm` language model for text preprocessing. We furthermore import `prince` to conduct the MCA, and finally, `itertools` (with alias `it`), `matplotlib.pyplot` (as `plt`) as well as the `adjust_text` function from the `adjustText` libraries to create biplots.

In a next step, we load the data table which contains the text corpus and its metadata. To do so, we call the `os.chdir()` function to navigate through our hard drive, before we import the data with `pd.read_csv(MCA_df.csv)` and assign it to `df` (short for data frame). To keep only unique comments, we apply `drop_duplicates()` on `df`, but specify that duplicates should be dropped based on `subset = "textOriginal"`. We remove duplicates, as there are commenters who posted the exact same comment multiple times and also across multiple videos. Keeping those comments introduces bias into our data and makes the extraction of topics more likely which mirror the contributions of a single commenter. The resulting data frame contains 60,619 unique comments, which is a subset of a larger corpus containing contains millions of comments posted in different languages under videos in multiple disciplines (such as physics, chemistry, history, economics, or sociology). Besides the comment text, the data frame includes variables such as the video ID, comment ID, reply

```
import re
import os
import pandas as pd

from sklearn.feature_extraction import text as tx
import spacy
from langdetect import detect
nlp = spacy.load("en_core_web_sm")

import prince

import itertools as it
from matplotlib import pyplot as plt
from adjustText import adjust_text

root = "C:\\Users\\[YOUR_PATH]\\"
data_path = root + "Data\\"
figures = root + "Figures\\"

os.chdir(data_path)

df = pd.read_csv("MCA_df.csv", sep = ";")
df = df.drop_duplicate(subset = "textOriginal")
```

Code box 15.1: Importing the packages necessary to conduct an MCA plus loading the data into the work space

count, like count, the original text of the comment, and the date of publication on YouTube.

We will now generate a document-term matrix (Code box 15.2):

```
count_vectorizer = tx.CountVectorizer(min_df = 100,
                                      max_features = 10000)
dtm = count_vectorizer.fit(df.textOriginal)
dtm_matrix = dtm.transform(df.textOriginal).A
dtm_df=pd.DataFrame(dtm_matrix,columns=dtm.get_feature_names_out())
```

Code box 15.2: Creating the document-term matrix from the YouTube comments posted below sociology videos

We use sklearn's `CountVectorizer()` and keep:

- Tokens that appear in at least 100 documents by using the `min_df = 100`.
- The 10,000 tokens that appear most often in the text corpus with `max_features = 10000`.

Although these (especially the value set with `min_df`) might seem arbitrary, we set these thresholds to increase the stability of the specific MCA and to account to the diversity and sparsity of tokens used in our corpus. We usually include items in GDA, which appear in at least 5% and at most 95% of the observations (Hjellbrekke, 2019; Greenacre, 2010). However, as tokens characteristic for some topics are usually spread relatively sparsely across the texts, this threshold is set much lower – sometimes even to 5, 10, or 25. Nevertheless, using such a threshold enables us to draw clearer boundaries between topics and between topic dimensions.

We then use `count_vectorizer.fit()` method and build a vocabulary of tokens used in the comments stored in `df.textOriginal` and assign the fitted dictionary to `dtm`. This variable is then used to transform the comments into a sparse document-term matrix with `transform(df.textOriginal)`. The chained `A` method converts the matrix into a NumPy array and assigning it to `dtm_matrix`. Finally, we create a Pandas data frame with `pd.DataFrame()` and pass (a) `dtm_matrix` as input data and (b) token vocabulary extracted from dtm with `columns=dtm.get_feature_names_out()`. This results in a data frame assigned to `dtm_df` (short for document-term matrix data frame), which serves as basis for further feature selection operations and as basis for the MCA.

15.1.2 Apply cleaning procedures on the document-term matrix

We continue the data cleaning procedure with filtering stop words and non-English tokens, and with retaining only nouns, adjectives, and verbs (Code box 15.3).

We apply the feature selection techniques introduced in Section 13.5 to achieve this. As the tokens are the basis for the feature selection, we use a list comprehension to extract them from the `dtm` variable first. We continue with our first filtering procedure in the second list comprehension, which consists of the following steps. It keeps a token, if the following two conditions are met:

1. Check if the token consists of characters belonging to the ascii family using the `isascii()` method.[2]
2. Check if the token consists of more than two characters.[3]

```
tokens = [token for token in dtm.get_feature_names_out()]
tokens_filtered = [token for token in tokens if
                   token.isascii() == True and len(token) > 2]

tokens_filtered_spacy = []
pos_list = ["NOUN", "ADJ", "VERB"]

for token in nlp(" ".join(tokens_filtered)):
    if token.pos_ in pos_list and detect(token.text) == "en":
        tokens_filtered_spacy.append(token.text)

dtm_df  = dtm_df[tokens_filtered_spacy]
```

Code box 15.3: Token filtering by token length and by part of speech tags

This filtered token list is assigned to `tokens_filtered`.

The next steps are to retain only nouns, adjectives, and verbs in our token list and to keep only English tokens. To check for the former, we define the parts of speech first that we wish to keep with `pos_list = ["NOUN", "ADJ", "VERB"]`. We begin the subsequent flow-control sequence (starting with the `for` loop and ending with the `pass` argument) with concatenating the list of tokens into a single string and transforming this string into a spaCy NLP object with `nlp(" ".join(tokens_filtered))`. We do so as spaCy's `nlp()` function requires us to pass a string instead of a list of tokens for its (pre-)processing procedures. See Section 13.2 for a detailed account of this process. The nested `if` operator controls:

- If the part of speech tag of a token is a member of `pos_list`.
- If the language of a token is exactly equal to English ("`en`").

The token is added to the `tokens_filtered_spacy` list via the `append()` method, if both conditions are met. Otherwise, the loop continues with the next token. The returned list is used to reduce the variables in `dtm_df` using the Pandas selecting operation `dtm_df[tokens_filtered_spacy]`.

15.1.3 Explore the token distribution

Let us now examine the most frequent tokens that appear in `dtm_df.` To do so, we will first sort the tokens by descending frequency of appearance as follows (Code box 15.4):

```
tokens_sum = dtm_df.sum().sort_values(ascending = False)
```

Code box 15.4: Counting the number of token appearances throughout the corpus and sorting them by number of appearances (in descending order)

Here, we call the Pandas' `sum()` method to count the tokens and sort the tokens by descending frequency with `sort_values(ascending = False)`. We assign this new object to a variable we call `tokens_sum`, which will be used as input for the plotting functions of Matplotlib or Seaborn.

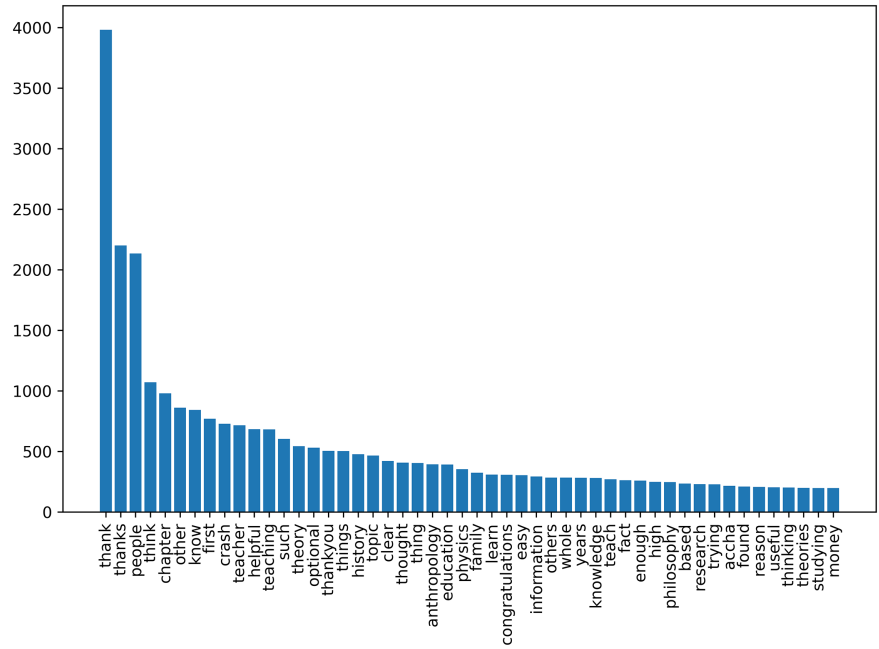
To visualize the frequency of the 50 most used tokens across our data corpus, we use the code in Code box 15.5, which creates Figure 15.3:

```
plt.figure(figsize = [8, 6])
plt.bar(tokens_sum.head(n = 50).index, tokens_sum.head(n = 50))
plt.xticks(rotation = 90)
plt.tight_layout()

os.chdir(figures)
plt.savefig("most_frequent_50_tokens.png", dpi = 600)
plt.close()
```

Code box 15.5: Visualizing the 50 most frequent tokens in descending order as a bar chart using matplotlib

The `plt.figure(figsize = [8, 6])` function creates an eight-inch-wide and six-inch-high figure canvas. The `plt.bar()` function in the next line of code initiates a bar plot with the 50 most frequent tokens on the x-axis (`tokens_sum.head(n = 50).index`), and token frequencies on the y-axis (`tokens_sum.head(n = 50)`). In the next line, we rotate the x-tick labels by 90 degrees using the `plt.xticks(rotation = 90)` function. We then adjust the plot to fit tightly onto the canvas with `plt.tight_layout()`. We then change the working directory with `os.chdir(figures)` and save a high-resolution image of the figure with by `plt.savefig(dpi = 600)`, and close the canvas using the `plt.close()` function.

Figure 15.3 The 50 most frequent tokens in the document-term matrix of the YouTube comments regarding sociology videos.



As you see in Figure 15.3, the most frequent tokens include phrases of gratitude ("thank", "thanks", "people"), references to lessons (e.g., "chapter"), learning ("thinking", "thought"), and comparisons between disciplines ("philosophy", "anthropology", "physics"). This distribution provides insights into potential topics that may appear during automated content analysis with MCA.

15.1.4 Make the document-term matrix ready for MCA

In the next step, we remove all texts from our corpus that do not contain any tokens listed as variable in the document-term matrix. We do so, as empty rows cause the MCA algorithm to crash.[4] To avoid this, we use the code snippet presented in Code box 15.6, which results in an index-list shown in Output 15.1.

This step serves two purposes:

1) We link the comments stored in `df` with the results of the MCA by setting the `dtm_df`'s index to `comment_id`. Initially, the document-term matrix used a consecutive numbering for observations, which we now replace with the comment.
2) We retain only those comments for further investigation where at least a single token from the document-term matrix is present.

```
dtm_df.index = df.comment_id
keep_comments = dtm_df.sum(axis = 1)
keep_comments_comment_id = keep_comments[keep_comments.values >
0].index
print(keep_comments_comment_id)
```

Code box 15.6: Identification of comments without appearance of any token listed as variable in the document-term matrix

```
Index(['UgxrSWFwlmX9bYj-WH54AaABAg', 'Ugzp86BU-paXFcWuMgB4AaABAg',
       'UgzNnTm69HTvEl8lWHh4AaABAg', 'UgzH8rxYvkrWr4p0izF4AaABAg',
       'Ugz3MsaUobwU4davXx94AaABAg', 'Ugwv3MIfwfjb7qshNJJ4AaABAg',
       'UgyXRzaqdQ-w4SSOKbF4AaABAg', 'UgzI5E_CL1Axgq1NFV94AaABAg',
       'Ugxav0a7VtmbKV8U4Ih4AaABAg', 'Ugxwpszqhgy6ViC6jkp4AaABAg',
       . . .
       'UgyQ26IK2B9wwyhi98l4AaABAg', 'UgxN_vYo97Aigr6tzsp4AaABAg',
       'Ugxy7DAagrY2PcFlRYt4AaABAg', 'Ugyc7cTX84xDSnXraBt4AaABAg',
       'UgyilYtQvhknCBDHNwJ4AaABAg', 'Ugyh5YXeruiIN88Bm0d4AaABAg',
       'Ugwgj-5LzrQqM0wqvEZ4AaABAg', 'UgxGTV1d_G42hSi4K3h4AaABAg',
       'UgzwqGkdAhozR6pqiph4AaABAg', 'UgwkoYGwLQSVN5RetnR4AaABAg'],
      dtype='object', name='comment_id', length=19080)
```

Output 15.1: Index list of comment IDs with at least one token present in the document-term matrix

The first line of code in Code box 15.6 assigns comment IDs from `df` to the index of `dtm_df`. The second line of code sums all values per row by calling `dtm_df.sum(axis = 1)`. The returned values indicate the number of tokens per document available for topic extraction. We assign the resulting Pandas series to `keep_comments`, and extract index entries (=comment IDs) with `keep_comments[keep_comments.values > 0].index`, where at least one token eligible for the MCA is present. This results in an index object with 19.080 entries, as shown in Output 15.1.

In the following, we filter the document-term matrix to exclude rows that do not contain any token eligible for the MCA (Code box 15.7):

```
dtm_df=dtm_df[dtm_df.index.isin(list(keep_comments_comment_id))]
```

Code box 15.7: Removal of comments without appearance of any token listed as variable in the document-term matrix

```
[False False False . . . False False False]
```

Output 15.2: Output of the code snippet present in the square brackets of Code box 15.7

This filtering operation resembles the selection procedure in Code box 15.6 but differs in its condition. Here, we use `isin()` to check if `dtm_df` entries are in `keep_comments_comment_id`. As shown in Output 15.2, the result is a list of Boolean operators. When used during a selection procedure, `True` statements equate with an order to keep an entry, whereas `False` statements correspond to an order to drop an entry. This results in a document-term matrix with 19.080 documents.

Next, we dichotomize the values in the cells of the `dtm_df` that will be passed to the MCA. Dichotomization implies converting cell values to 1 if a token appears at least one time in a document and 0 otherwise. We do this for two reasons:

1) First, comments vary in length, meaning that longer texts have higher chances to including multiple instances of the tokens listed in Figure 15.3.[5]
2) Secondly, MCA treats different counts as separate categories. This would further sparsen our data matrix and could cause the MCA to fail in extracting meaningful topic dimensions.

To implement this, we use the conditional assignment procedure in Code box 15.8:

```
dtm_df[dtm_df >= 1] = 1
```

Code box 15.8: Binarization of each cell of the document-term matrix

15.1.5 Add passive variables to the modified document-term matrix

To incorporate passive variables into the analysis, we select and recode relevant metadata variables. We extract `["likeCount", "publishedAt"]` from `df`, as the former indicates the popularity of a topic, which could vary by publication date. We prepare the publication date for grouping using the following code (Code box 15.9):

```
passive_df = df[["likeCount", "publishedAt"]]
passive_df.publishedAt = [re.search("^[0-9]{4}", x).group() for x in \
                      passive_df.publishedAt]

## generate Groups of likes
print(passive_df.likeCount.describe())
```

Code box 15.9: Selecting and descriptive summary of selected passive variables

The `publishedAt` variable contains timestamps in the YEAR-Month-Day-Time format (for example, `2024-10-02T19:17:03Z`). As you might guess, it is extremely seldom that two comments are posted at the exact same second. For this reason, extract only the year using regular expressions (introduced in Section 12.4.10). The publication year of the comment is stored at the beginning of the time stamp. Consequently, the Regex pattern that will extract the first four digits that capture a year is "`^[0-9]{4}`". We implement this pattern in a list comprehension using `re.search()` and call the `group()` method to return the first detected string pattern.

Next, we categorize likes per comment (`likeCount`). To get an idea how to group the number of likes, we apply the Pandas `describe()` method (see Section 14.4), and calculate univariate point and dispersion parameters:

```
count    60619.000000
mean         5.196292
std         64.823061
min          0.000000
25%          0.000000
50%          0.000000
75%          1.000000
max       5041.000000
Name: likeCount, dtype: float64
```

Output 15.3: Point and dispersion measures of the number of likes per comment selected as passive variable

As indicated by Output 15.3, the number of likes per comment is extremely skewed. To deal with this issue, we could create categories including:

1) Comments with no likes.
2) Comments with 1–9 likes (which already comprise like 99% of the comments).

3) Comments with 10–49 likes.
4) Comments with 50–99 likes.
5) Comments with at least 100 likes.

These categories are to a certain extent arbitrary. You could also create fewer categories, or categories with different thresholds. Usually, we advise you to use, for example, the quartiles, or different thresholds if, for example, the distribution of the values in the respective passive variable is bimodal or multimodal.[6]

To implement this categorization on the `likeCount` variable, we use the following flow control sequence (Code box 15.10):

```
like_cat = []
for count in passive_df.likeCount:
    if count >= 100:
        like_cat.append("100+")
    elif count < 100 and count >= 50:
        like_cat.append("50-99")
    elif count < 50 and count <= 10:
        like_cat.append("10-49")
    elif count < 10 and count >= 1:
        like_cat.append("1-9")
    else:
        like_cat.append("0")
passive_df.likeCount = like_ cat
```

Code box 15.10: Categorization procedure for the number of likes using a for loop if, elif, and else statements

We start with a `for` loop with nested `if`, `elif`, and `else` statements. The outcomes of this nested flow control structure are assigned `like_cat` (short for number of likes categorized). The first `if` statement checks, if the number of likes of a comment is greater than or equal to 100. If this condition is met, then the string "`100+`" is appended to `like_cat`. The next condition is checked by the `elif` statement if the first is not met. In our case, the first `elif` checks if `passive_df.likeCount` is less than 100 and greater equal 50. If this applies, the character string "`50-99`" is appended to the `like_cat`. This continues until the `else` statement is reached, which, in our case, comprises only values less equal 0. The last line of code reassigns the `passive_df.likeCount` variable with the categories defined in the flow control sequence.

15.1.6 Applying and fitting the MCA on the data

We now turn to the middle box of the workflow depicted in Figure 15.2 and fit the MCA on the document-term matrix and passive variables (Code box 15.11):

```
MCA = prince.MCA(n_components = 10,
                 check_input = True,
                 engine = "sklearn")
MCA_Fit = MCA.fit(dtm_df)
```

Code box 15.11: Initializing and fitting a multiple correspondence analysis on the document-term matrix of the comments of sociology YouTube videos

Similar to `sklearn`'s `CountVectorizer()`, we initialize `prince.MCA()` to conduct the MCA and adjust it with the following options:

- `n_components` = The number of dimensions to be extracted from the data. In our case, we extract the first ten dimensions that structure the data most clearly and cover the highest amount of variance.
- `check_input` = Tests, if our input data is an array, a sparse matrix, or similar. These data types are eligible to extract topics using an MCA.
- `engine` = Defines, if either sklearn or the fbpca package should be used to run the singular value decomposition of the MCA.

After initiating the MCA algorithm, we call `MCA.fit(dtm_df)` to fit the MCA on our document-term matrix, and assign the result to `MCA_Fit`. The resulting MCA object stores:

- The eigenvalues per dimension.
- The share of variance covered by each dimension.
- The name of the variables used to construct the topic-space (`active_cols`).
- The name of the observations used to construct the topic-space (`active_rows`).
- The contribution of each variable to the variance explained per dimension (`column_contributions_`).
- The contribution of each text document to the variance explained per dimension (`row_contributions_`).
- The coordinates of the active variables (`column_coordinates`).
- The coordinates of the text documents (`row_coordinates`).

15.1.7 Selecting an appropriate number of dimensions for interpretation

After fitting the data, we must decide on the number of dimensions to interpret. We use the **elbow-criterion** introduced by Thorndike (1953) as a heuristic to select the number of meaningful dimensions for further investigation. It states that – given the percentage of variance covered (or: explained) by each dimension – only a few dimensions which significantly add more explained variance and are thus considered meaningful. A dimension ceases to be meaningful when the additional explained variance drops substantially compared to the former dimensions. This marks the "point of diminishing returns". We use a **scree plot** to draw the number of dimensions on the x-axis and the variance explained per dimension on the y-axis. This helps us to find the point of diminishing returns.

To create a scree plot, we extract the explained variance per dimension from `MCA_Fit`. The variance explained per dimension, as well as the eigenvalues and cumulative explained variance is stored in `MCA_Fit.eigenvalues_summary`. This summary is a Pandas data frame containing the columns: "`eigenvalue`", "`% of variance`", and "`% of variance (cumulative)`". Such variable names are inconvenient for referencing, data modification, and calculation. We change them by passing new values (`["eigenvalue", "variance", "variance_cumulative"]`) to `variance_summary.columns` (Code box 15.12) to make further data preparation and analysis easier:

```
variance_summary = MCA_Fit.eigenvalues_summary
variance_summary.columns = ["eigenvalue", "variance",
"variance_cumulative"]
print(variance_summary)
```

Code box 15.12: Extracting a Pandas data frame containing the eigenvalues, variance explained, and cumulative variance explained per dimension extracted from our document-term matrix

Results in Output 15.4.

When we look at the output, we observe that:

1) The variance explained per dimension is relatively low. This means that our space is relatively unstructured and contains potentially dozens of topics.
2) Eigenvalues correspond to the variance explained per dimension.

```
           eigenvalue variance variance_cumulative
component
0               0.029    2.90%                2.90%
1               0.018    1.78%                4.68%
2               0.014    1.39%                6.07%
3               0.013    1.34%                7.41%
4               0.013    1.33%                8.74%
5               0.013    1.32%               10.06%
6               0.013    1.30%               11.36%
7               0.013    1.28%               12.64%
8               0.013    1.26%               13.89%
9               0.012    1.25%               15.14%
```

Output 15.4: Data frame containing the eigenvalues, variance explained, and cumulative explained variance of the conducted MCA

3) Both the `variance` and `variance_cumulative` variables are neither integers nor floating point numbers, but strings.

In order to plot the variance against the respective extracted dimensions, we convert the character strings to floating point numbers as follows (in Code box 15.13):

```
variance_summary.variance = [float(re.sub("%", "", x)) for x \
                             in variance_summary.variance]
print(variance_summary)
```

Code box 15.13: Changing the entries of the variance column from a character string into floating point numbers

To do so, we nest an `re.sub()` function, which allows us to replace a string, inside a `float()` function, which changes the variable type of an object to a floating-point number. We utilize a list comprehension to iterate through all values stored in `variance_summary.variance`. More precisely, we call `float(re.sub("%", "", x))` within the list comprehension to delete the %-sign, and then convert the resulting output into a floating point number. We replace the values within `variance_summary.variance` and check if the operation was successful with `print(variance_summary)` (Output 15.5):

```
           eigenvalue variance variance_cumulative
component
0               0.028     2.84                2.84%
1               0.017     1.68                4.52%
2               0.014     1.44                5.96%
3               0.014     1.40                7.36%
4               0.013     1.34                8.70%
5               0.013     1.34               10.04%
6               0.013     1.32               11.36%
7               0.013     1.31               12.67%
8               0.013     1.28               13.95%
9               0.013     1.26               15.21%
```

Output 15.5: Testing, if the changes performed in the variance variable were successful

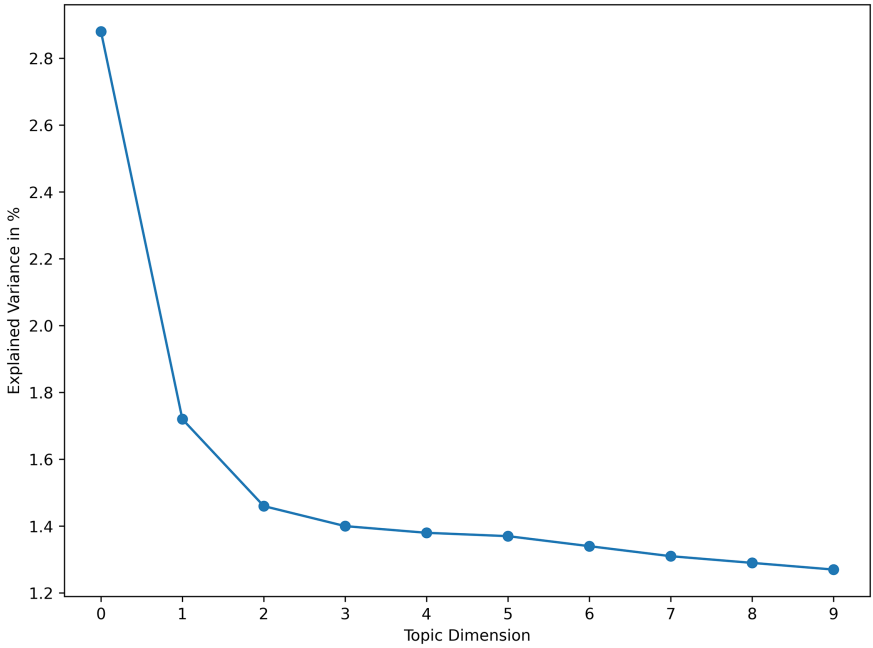
We can now pass the index and the `variance` column of `variance_summary` as demonstrated in Code box 15.14 to generate the scree plot in Figure 15.4:

```
plt.figure(figsize = [8, 6])
plt.plot(variance_summary.index,
         variance_summary.variance
         )
plt.scatter(variance_summary.index,
         variance_summary.variance
         )
plt.xticks(range(0,10))
plt.xlabel("Topic Dimension")
plt.ylabel("Explained Variance in %")
plt.tight_layout()
plt.savefig("MCA_screeplot.png",
           dpi = 600)

plt.close()
```

Code box 15.14: Creating and saving a scree plot using matplotlib.pyplot functions

Here, we initialize a figure canvas with a width of eight inches and a height of six inches by calling `plt.figure(figsize = [8, 6])`. We then initialize a line plot (`plot()`), pass `variance_summary.index` (extracted dimensions) to the x-axis, `variance_summary.variance` to the y-axis, and adjust the color

Figure 15.4 Scree plot of the variables explained by topic dimension extracted by the MCA.



with the `color` option. The `plt.scatter()` function generates a scatter plot, which helps us to find the elbow more easily. We add tick marks for the first ten dimensions with `xticks(range(0,10))`, and change the labels of the x- and y-axis with `xlabel()` and `ylabel()`. Next, we adjust the plot to the canvas with `tight_layout()` and save a high-resolution image of the scree plot by calling `plt.savefig(dpi = 600)`. The scree plot depicted in Figure 15.4 suggests to interpret the first three dimensions extracted by the specific MCA.

There are now three more steps to take until we can interpret our three dimensions. We need to extract:

1) The coordinates of the active variables (i.e., the coordinates of the tokens).
2) The coordinates of the passive variables (categorized publication year and likes).
3) The coordinates of the text documents.

For the first two steps, we need exactly one method, namely `column_coordinates()`, pass `dtm_df`, and assign the column coordinates to a variable we call `token_coordinates` (Code box 15.15):

```
token_coordinates = MCA_Fit.column_coordinates(dtm_df)
print(token_coordinates)
```

Code box 15.15: Extracting the coordinates of the tokens on the first ten dimension of the MCA

This results in the Pandas data frame seen in Output 15.6. It contains the coordinates of the MCA categories, which align with the appearance (indicated by `_1`) or absence (indicated by `_0`) of each token on the first ten dimensions.

```
                       0         1         2  ...         7         8         9
accha_0         0.007705 -0.002435 -0.002547  ...  0.000466 -0.004767  0.003147
accha_1        -0.703654  0.222403  0.232583  ... -0.042520  0.435402 -0.287461
age_0          -0.008813  0.005520  0.001102  ... -0.015267  0.002057  0.009813
age_1           1.483051 -0.928853 -0.185445  ...  2.568936 -0.346159 -1.651173
anthropology_0 -0.029510 -0.054726  0.003649  ... -0.000603  0.006010 -0.009795
                  ...       ...       ...     ...   ...       ...       ...
write_1         0.312788 -0.600412 -0.582232  ...  0.229244  0.819166  1.525725
writing_0      -0.003974  0.000020  0.006930  ... -0.006537 -0.005693 -0.001913
writing_1       0.576951 -0.002934 -1.006181  ...  0.949073  0.826527  0.277720
years_0        -0.015094  0.007385  0.002152  ... -0.027051  0.017217  0.010485
years_1         1.118214 -0.547144 -0.159418  ...  2.004059 -1.275511 -0.776759

[184 rows x 10 columns]
```

Output 15.6: Coordinates of the tokens on the first ten dimensions of the MCA

Except for one slight modification, we do the same with the passive variables. As seen in Code box 15.16, we use `passive_df` as input for `MCA_Fit.column_coordinates()` before assigning the calculated positions to `passive_coordinates`. This results in a data frame shown in Output 15.7.

```
passive_coordinates=MCA_Fit.column_coordinates(passive_df)
print(passive_coordinates)
```

Code box 15.16: Extracting the coordinates of the passive variables (categorized number of likes per comment, year of publication) on the ten dimensions of the MCA

```
                          0         1         2 ...         7         8         9
likeCount_0        0.233253 -0.146174 -0.123552 ...  0.081142  0.033227 -0.011813
likeCount_10-49   -0.027383  0.014449  0.003228 ... -0.009395  0.002937 -0.004244
likeCount_100+     0.538449 -0.316610  0.154454 ...  0.217434 -0.180188  0.276903
likeCount_50-99    0.562719 -0.124358  0.058419 ...  0.132456 -0.152328  0.007400
publishedAt_2011 -0.165759 -0.281476 -0.187612 ... -0.301632 -0.537681 -0.398487
publishedAt_2012  0.576025 -0.526088 -0.040401 ...  0.110107 -0.140184 -0.095275
publishedAt_2013  0.695297 -0.466335 -0.114302 ...  0.209648 -0.085055  0.083199
publishedAt_2014  1.104522 -0.680735 -0.153383 ... -0.038331  0.232574  0.177135
publishedAt_2015  0.712772 -0.350140 -0.139384 ...  0.076060  0.119774  0.201565
publishedAt_2016  0.550256 -0.301280 -0.155922 ... -0.010291  0.190147  0.065671
publishedAt_2017  0.511415 -0.226829  0.142598 ...  0.368543 -0.243095  0.005346
publishedAt_2018  0.434018 -0.189207 -0.028326 ...  0.125346 -0.011543  0.033151
publishedAt_2019  0.295516 -0.176691 -0.149730 ...  0.031512 -0.013585  0.052321
publishedAt_2020 -0.055151 -0.027283 -0.161544 ... -0.090869  0.024009 -0.032960
publishedAt_2021 -0.189058  0.091177 -0.035676 ... -0.102433  0.075591  0.025382
publishedAt_2022 -0.182345  0.295596 -0.076586 ... -0.134754  0.135706 -0.030075
publishedAt_2023 -0.305471  0.065432  0.059040 ... -0.095901  0.055246 -0.015739
publishedAt_2024 -0.328403  0.103485  0.030290 ... -0.132348  0.008472 -0.021006

[18 rows x 10 columns]
```

Output 15.7: Coordinates of the passive variables (categorized number of likes per comment, year of publication) on the 10 dimensions of the MCA

Finally, we extract the coordinates of the comments posted under the YouTube videos `MCA_Fit.transform(dtm_df)` (Code box 15.17), assign this data frame to `comments_coords`, and use the `print()` function to generate Output 15.8. This allows us to analyze the relationship between tokens, active variables, passive variables, and text documents.

```
comments_coords = MCA_Fit.transform(dtm_df)
print(comments_coords)
```

Code box 15.17: Extracting the coordinates of the YouTube comments on the first ten dimensions of the MCA

```
                                  0         1 ...         8         9
comment_id                                    ...
UgxrSWFwlmX9bYj-WH54AaABAg   0.042119 -0.093022 ...  0.111541  0.090347
Ugzp86BU-paXFcWuMgB4AaABAg   0.040837 -0.066418 ... -0.076456  0.320817
UgzNnTm69HTvEl8lWHh4AaABAg  -0.045089  0.015309 ... -0.131605  0.019704
UgzH8rxYvkrWr4p0izF4AaABAg  -0.106793  0.028636 ... -0.088218 -0.016115
Ugwv3MIfwfjb7qshNJJ4AaABAg  -0.106793  0.028636 ... -0.088218 -0.016115
                              ...       ...     ...   ...       ...
Ugyh5YXeruiIN88Bm0d4AaABAg  -0.093715  0.011742 ...  0.025103 -0.084535
Ugwgj-5LzrQqM0wqvEZ4AaABAg   0.039023 -0.013715 ... -0.082566  0.010892
UgxGTV1d_G42hSi4K3h4AaABAg   0.151996  0.284472 ... -0.102054  0.246663
UgzwqGkdAhozR6pqiph4AaABAg   0.125478  0.351006 ... -0.000920  0.216529
UgwkoYGwLQSVN5RetnR4AaABAg  -0.046664 -0.041863 ...  0.075978  0.146664

[19297 rows x 10 columns]
```

Output 15.8: Coordinates of the YouTube comments on the first ten dimensions of the MCA

15.1.8 Creating the biplots

After we gathered all the necessary information to analyze the topic dimensions, we now proceed with analyzing the dimensions (right box in Figure 15.2). In the following, we start with a visual analysis of the first three topic dimensions and the positions of the active and passive variables on each dimension. But before doing so, we must create bi-plots and select a few categories for inspection. Without this selection, our figures would be overcrowded and difficult to interpret.

We begin by selecting tokens that appear throughout the comments. These are indicated with the `_1` suffix, for example `sociology_1`. We achieve this using the following list comprehension combined with an if-clause and a regex-search (Code box 15.18).

```
keep_index = [token for token in token_coordinates.index if
re.search("1$", token)]
variable_coordinates = token_coordinates[token_coordinates.
index.isin(keep_index)]
```

Code box 15.18: Selecting only tokens that appear throughout the text documents

Here, filter tokens ending with a "1" by calling `re.search("1$", token)` and store them in `keep_index`. In the second line, we call the `isin()` method to keep only rows in `token_coordinates` whose index is included in `keep_index`.

Next, we generate a list of axis pairs (dimensions) eligible for the X- and Y-axes of the biplot. To do so, we call `it.combinations()` as follows (Code box 15.19).

```
axes = [combination for combination in it.combinations
([0, 1, 2], 2)]
print(axes)
```

Code box 15.19: Preparing a list of dimension-pairs to visualize using the biplot

The `it.combinations()` generator generates combinations from a sequence (array, list, tuple). Here, we produce combinations of two numbers present in `[0, 1, 2]` and pass these combinations to a list. This results in a list of tuples, similar to Output 15.9.

```
[(0, 1), (0, 2), (1, 2)]
```

Output 15.9: List of tuples containing all combinations of dimensions produced by the itertools combinations() function

Let us now generate the biplots. Each biplot consists of the following:

1) A scatterplot of the active variables.
2) A scatterplot of the passive variables.
3) Labels of the active variables.
4) Labels of the passive variables.
5) Axis labels including the variance explained per axis.

We will explain these steps and then automatize the process using a for loop. In this regard, we begin with defining the coordinates of the active and passive variables (Code box 15.20):

```
axis = (0,1)
xs = variable_coordinates[axis[0]]
ys = variable_coordinates[axis[1]]
xs_passive = passive_coordinates[axis[0]]
ys_passive = passive_coordinates[axis[1]]
passive_names = list(passive_coordinates.index)
```

Code box 15.20: Defining x- and y-coordinate for the active and passive variables

Here, we assign a tuple with column-values that resembles the first element of the axes list to `axis`. Accordingly, `xs = variable_coordinates[axis[0]]` stores all variable coordinates we wish to plot on the x-axis, whereas `ys = variable_coordinates[axis[1]]` stores the variable coordinates used for the y-axis. We repeat this procedure for the x- and y- coordinates of the passive variables, and store the names of the passive variables in `passive_variables`. We then initialize the figure canvas and generate scatter plots for the active and passive variables (see Code box 15.21).

```
plt.figure(figsize = [8, 6])
## plot the active variables
plt.scatter(xs,
            ys,
            color = "steelblue",
            alpha = 0.7)
## plot the passive variables
plt.scatter(xs_passive,
            ys_passive,
            color = "orange",
            edgecolor = "black",
            marker = "^",
            alpha = 0.7)
```

Code box 15.21: Scatterplot visualization of active and passive variables

Here, we call `plt.figure(figsize = [8, 6])` to create a figure canvas with a width of eight and a height of six inches. We then draw the scatter plots of (a) the selected active variables and (b) the passive variables using `plt.scatter()`. In the first case, we pass `xs` and `ys`, which contain the respective position of the active variables on the x- and y-axes. We then adjust the color of the points to grey with `color = "steelblue"`, and make these points slightly transparent with `alpha = 0.7`. The subsequent scatterplot draws the coordinates of the passive variables (`xs_passive`, `ys_passive`) onto the same figure canvas, but with a different color (`color = "orange"`) and a different edge color (`edgecolor = "black"`). We also change the marker to a triangle with `marker = "^"` to differentiate between active and passive variables.

Now, we adjust the labels of the X- and Y-axes (see Code box 15.22).

```
plt.xlabel(f"Dim: {axis[0]}, Variance explained: {variance_
summary.loc[axis[0], 'variance']} %")
plt.ylabel(f"Dim: {axis[1]}, Variance explained: {variance_
summary.loc[axis[1], 'variance']} %")
```

Code box 15.22: Include the explained variance per dimension to the x- and y-axis labels

To adjust the labels, we call the `plt.xlabel()` and `plt.ylabel()` functions and pass a formatted string (indicated by the `f` in front of the string). In a formatted string, we embed values by enclosing them in curly brackets `{}`. Here, we pass, firstly, the dimension number, as denoted by `{axis[0]}` in case of `plt.xlabel()`, and extract the explained variance for the first dimension stored in the cell `variance_summary.loc[axis[0],` '`variance`']. The same applies to the second axis used to generate the y axis.

We variables extract the five most typical variables at the poles of each dimension, which help us in our attempt to interpret the topics (Code box 15.23).

```
xs_sorted = xs.sort_values()
ys_sorted = ys.sort_values()
variable_names = [xs_sorted.head().index, xs_sorted.tail().index,
                  ys_sorted.head().index, ys_sorted.tail().index]
variable_names = [item for sublist in variable_names for item
in sublist]
variable_names = list(set(variable_names))
xs_typical = xs[xs.index.isin(variable_names)]
ys_typical = ys[ys.index.isin(variable_names)]
```

Code box 15.23: Extracting the most typical variables of the poles of each dimension

We use the `sort_values()` method to order the variable positions stored in `xs` and `ys`. We then create two lists that contain the names of the variables which yield the five lowest and highest values on the x- and y-axes and extract their names as a list using, for example, `xs_sorted.head().index`. Without setting any options, the `head()` method retrieves the first five rows of a Pandas data frame or series, while `tail()` retrieves the last five rows. We assign the list of lists of variable names to `variable_names`, which we subsequently flatten with `variable_names = [item for sublist in variable_names for item in sublist]`.

Sometimes, the most typical variables of one dimension are also characteristic for other extracted topic dimensions. This is why we keep only a union of the token names by calling `list(set(variable_names))` and reassigning `variable_names` with this union. Finally, we create two variables (`xs_typical` and `ys_typical`) to store the coordinates of the most typical variables per dimension. As before, we use the Pandas `isin()` method in a Pandas' selection to select the x- and y-coordinates of the selected tokens.

To prevent overlapping annotations, we use the texts_adjust() function (Code box 15.24):

```
text_adjusts = []
for x, y, text in zip(xs_typical,ys_typical,variable_names):
    text_adjusts.append(plt.text(x, y, text))
for x, y, text in zip(xs_passive, ys_passive, passive_names):
    text_adjusts.append(plt.text(x, y, text))
adjust_text(text_adjusts, arrowstyle = "->")
```

Code box 15.24: Applying the adjust_text() function to prohibit overlapping annotations

We begin by assigning an empty list to `text_adjust`. We then initiate two for loop control sequences to ensure that the annotations of the active and passive variables do not overlap. As you see, we iterate through three variables at once, as indicated `for x, y, text in zip(xs_typical, ys_typical, variable_names)` in the case of the active variables. Within the loop, we call `plt.text()` to annotate a specific point on the figure and append the coordinates of the annotated text, as well as the text itself, to the `text_adjusts` list. We repeat this procedure for the passive variables in the second for-loop. Afterwards, we pass the list of Matplotlib coordinates and annotations to `adjust_text(text_adjusts, arrowstyle = "->")`. The `arrowstyle` option defines the arrows used to indicate the position of the annotations. With this in mind, we automatize the entire process of generating biplots using a for-loop and adding the code snippets within the flow control sequence (Code box 15.25).

```
for axis in axes:
    xs = variable_coordinates[axis[0]]
    ys = variable_coordinates[axis[1]]

    xs_passive = passive_coordinates[axis[0]]
    ys_passive = passive_coordinates[axis[1]]
    passive_names = list(passive_coordinates.index)
```

Code box 15.25: Code including a for loop to create the biplots for interpretation

```
    plt.figure(figsize = [8, 6])

    ## plot the active variables
    plt.scatter(xs,
                ys,
                color = "steelblue",
                alpha = 0.7)
    ## plot the passive variables
    plt.scatter(xs_passive,
                ys_passive,
                color = "orange",
                edgecolor = "black",
                marker = "^",
                alpha = 0.7)
    ## set the axes names
    plt.xlabel(f"Dim: {axis[0]}, Variance explained: {variance_
summary.loc[axis[0], 'variance']} %")
    plt.ylabel(f"Dim: {axis[1]}, Variance explained: {variance_
summary.loc[axis[1], 'variance']} %")

    ## sort the values for the first and second axis to extract the
    ## most typical tokens in the following
    xs_sorted = xs.sort_values()
    ys_sorted = ys.sort_values()

    variable_names = [xs_sorted.head().index, xs_sorted.tail().
index,
                      ys_sorted.head().index, ys_sorted.tail().
index]
    variable_names = [item for sublist in variable_names for
item in sublist]
    variable_names = list(set(variable_names))

    xs_typical = xs[xs.index.isin(variable_names)]
    ys_typical = ys[ys.index.isin(variable_names)]

    text_adjusts = []
       ## get the coordinates of the active variables for further
adjustment
```

Code box 15.25: (Continued)

```
   for x, y, text in zip(xs_typical, ys_typical, variable_names):
            text_adjusts.append(plt.text(x, y, text))

   ## get the coordinates of the passive variables for further
adjustment
   for x, y, text in zip(xs_passive, ys_passive, passive_names):
            text_adjusts.append(plt.text(x, y, text))

    adjust_text(text_adjusts, arrowstyle = "->")
    figname = "mca_variables_"+ str(axis[0]) + "_" + str(axis[1])
+ ".png"
    plt.savefig(figname,
                dpi = 600)
    plt.close()
```

Code box 15.25: (Continued)

15.1.9 Extract the most typical texts per dimension

Now it is time to extract the ten most typical texts per MCA dimension (five high, five low values) (Code box 15.26).

```
id_data_frame = []
for axis in [0, 1, 2]:
    comments_coords.sort_values(by = axis, inplace = True)
    id_list = pd.concat([comments_coords.head(), comments_
coords.tail()])
    id_list["axis"] = axis
    id_data_frame.append(id_list)
id_data_frame = pd.concat(id_data_frame)

id_data_frame = pd.merge(left = id_data_frame,
                                right = df,
                                how = "left",
                                left_index = True,
                                right_index = True)
os.chdir(output)
id_data_frame.to_csv("top_texts_per_dimension.csv", sep = ";")
```

Code box 15.26: Extracting and saving top texts per dimension

We start with assigning an empty list to `id_data_frame`, and initiate a for-loop to iterate over the combinations of the first three dimensions stored in `axis`. In the first line within the `for` loop, we sort `comments_coords` in ascending order using `sort_values()`, and set `inplace = True` to replace the `comments_coords` variable. We then retrieve the most typical texts per pole of `comments_coords` with the `head()` and `tail()` methods and store both data frames in a list. We concatenate the data frames in the list with the `pd.concat()` function. We then add a column called "`axis`", which stores the dimension name. Finally, the new data frame is added to `id_data_frame`. After the for-loop terminates, we use the `pd.concat() function` to stack the most typical texts in a single data frame.

However, this data frame contains neither comments nor the metadata of `df`. This is why we merge both data frames with `pd.merge()`. This function follows the logic of Venn diagrams with a "left" and a "right" dataset. Consequently, we define rules on (1) how to merge and (2) which columns or indices to use for merging the two data frames. We set the `id_data_frame` as left data frame, and `df` as right data frame, and define with `how = left` to use `id_data_frame` as basis for the merging procedure. As both data frames store the names of the comments in their indices, we use the options `left_index = True` and `right_index = True`. We overwrite `id_data_frame`, and save export the resulting data frame using `to_csv()`.

15.1.10 Topic interpretation

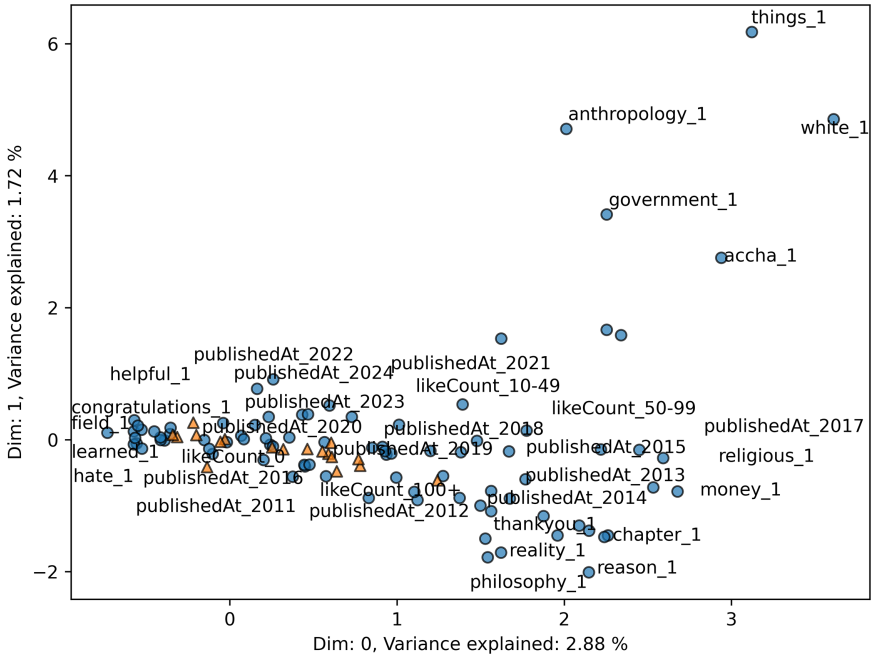
Dimension 0

Let us begin by interpreting the first dimension (Dimension 0), which is depicted on the x-axis of Figure 15.5. We first examine the right-hand side of the x-axis, before turning to the left-hand side to analyze the tokens that characterize this first dimension.

On the right-hand side, we find tokens such as "hate_1", "religious_1", as well as "anthropology_1", "chapter_1," "field_1", and "philosophy_1". On the left-hand side of the plot, we see "politics_1", "accha_1", "learned_1", "white_1", and "batch_1". At first glance, this dimension appears as a contrast between disciplines adjacent to sociology with a focus on philosophical and religious topics (right), and some sort of focus on politics, performed or focused on white people (left).

When we consider the publication dates, we observe that videos associated with the right-hand side of the X-axis were typically published between 2014 and 2018, while those on the left were either published earlier or later. This pattern might suggest changes in the video content. However, this interpretation remains tentative and, keep in mind, that the tokens themselves provide a condensed representation of the text documents.

Figure 15.5 Biplot of the first two dimensions extracted by the MCA form the document-term matrix.



This is why we examine the comments (and excerpts thereof) typical for the poles of the first dimension to substantiate our interpretation. So, what themes emerge at the poles of dimension 0?

Dim0: right side, Comment 1
(excerpt, italics and bold text by author)

> I've lived the life of a monastic-secular, similar to a monk for Humanism, but of other developments & views for 9 years now. It takes great ideas from; *Comte, St. Benedict, Saint-Simon, Spinoza, Emerson, Nietzsche, Gracian*, with many & more of the great Exemplars of Humanity you (The School of Life) so brilliantly explore in your video-essays & curriculum series/studies. Until about 2 years ago, I had thought I was the only one (or 1 of a very small number of people) who thought in these ways, aside from the very rare & few authors, **philosophers** & great minds from the past – & then I found, The **School of Life**. It has exponentially & immensely increased my thirst & absolute need for continuing the endeavors of *Humanity* & the *Art*(s) of *Worldly Wisdom*. [...]

Now, is there anyway we can assemble "All This" with the intent to go further down the rabbit hole?? Is this already trying to become a "Thing", with The School of Life, Humanism, Secularism, Positivism, or even; **Religion**, Government, Military, Business & Societal paradigms already?? [...] Why we're not more "openly" searching for these Persons and collectively assembling their aptitudes for solving "ALL" Humanly & Worldly Plights, not just some, few or less, is beyond me. We'd like to think that we are, but I truly believe that we are not. Our brightest minds are being auctioned off to the highest bidders in business, privatized & government programs that are, for lack of better terms, redundant or non-effective in the long run – or worse yet, digressive or degrading to societal, national & global affairs, or just down-right in it for money or attention in some way. That's just 1 of many, many examples of "old metaphors running amuck", that are being blatantly played out before us everyday, and yet we still do very little about it. If you have honestly read this much of my blabbering, then you may sense a bit of hostility or negative views towards humanity by this very author, but I assure you, that is not the case. Though my texts here may appear to be ungrateful to the comforts afforded me by what Human Progress has already offered, or even unsatisfied with the works of The School of Life or the great ideas, men & women they so lovingly, and therapeutically teach, I am genuinely & mostly content with the current states of things. It is for the future generations & beyond that my urgency may seem more lively in this passage, or perhaps my lack of findings in souls of similar tastes out there/here [...]

This comment appears to be a self-reflection by the commenter, who engages deeply with the video's content and reflects on his relation to diverse philosophical schools of thought. At first glance, this supports our interpretation, even though the comment also touches on other topics not covered by the tokens alone. In this sense, a subsequent qualitative analysis of this comment could provide further insights. Let us now turn to the next commentary.

Dim0: right side, Comment 2 (excerpt):

In this video lesson, I learned about Anthropology, Sociology and Political Science.

Anthropology

- from the word "anthropologia" which means study of humanity.
- it also come from that Greek word "anthropus" means human being.

- study of human being and their ancestors through time and space in relation to physical character, environment, social relations and culture.
- Anthropology has a major American influence it's because of the famous anthropologist Franz Boaz

Franz Boaz

- father of modern anthropology
- he is an expert when it comes to human culture and studied historical particularism.

Field of Anthropology

1. BIOLOGICAL OR PHYSICAL ANTHROPOLOGY- study of human origin such as genetics, race, evolution, fossils, and primates.
2. CULTURAL ANTHROPOLOGY- sutdy of the living people their religion, social system, language, clothing, food, beliefs and tradition.
3. LINGUISTIC OR ANTHROPOLOGICAL LINGUISTICS- study of language, it's evolution, it's connection to other languages and other Society.
4. ACHEOLOGY- study of the material remains of past human life ang activities such as religion, social system, language, atire, food, beliefs and tradition

Goals of Anthropology

- Observe the common things among people (traditions, language, others)
- Discover what make people difference from one another
- Create new knowledge through researchers about humankind and behavior.

Sociology

- The study of sociology, patterns of social interactions, social life, social change and social causes and consequences of human behavior.
- "Sociologie" came from the French word means companion and "ology" meaning the study of. It was coined or first developed by french philosopher named AUGUSTE CONTE.

Fields of Sociology

- Social organization
- Social psychology

- Applied sociology
- Human ecology

Goals of Sociology

1. Study of the nature of humanity
2. Appreciate Sociology [...]

This comment summarizes the video's content, indicating that the video was likely structured as a university lesson comparing anthropology, sociology, and political science. The same applies to the third comment:

Dim0: right side, Comment 3 (excerpt):

I've always known that change is inevitable, it happens all the time and that nothing permanently stays the same. We are always bound to develop ourselves. It may be through new discoveries in science and technology, personal and shared experiences. And also the urge to explore and learn more since there is no such thing as forever. People should adapt the changes as it is also necessary to remember and understand the obstacles brought by these changes.

Through this module, I was able to visit places virtually and discover facts and history of certain places. I have learned and got **deeper understanding** about **Anthropology**, **Sociology**, and **Political Science**. And it broadens my knowledge by addressing the corresponding goals associated in it. [...]

In contrast, the other side of the first dimension appears to be more intuitively interpretable. It represents expressions of gratitude for the lessons taught in the video, as illustrated by the three comments here:

Dim0 left side, Comment 1:

Thank you so much 😘 mam really your way of teaching is very good and nice explation and I understand everything in this chapter ❤️‍🩹 again thanks

Dim0 left side, Comment 2:

> mam please start teaching psychology or haier any teacher whose teaching skill is like you the way you teach i like your teaching skill very much **thank you** for explaining this chapter you have clear my all the doubts.

Dim0 left side, Comment 3:

> Hello ma'am good morning
> I am shivani i am i am preparation for upsc i decided my optional subject sociology ma'am yours teaching style so good ma'am i need your help ma'am.
> Notes and... . . . . .
> **Thank you** ma'am 😊

Overall, we might label the first dimension as *Summarization of learned content versus Praise and gratitude for teaching*.

Now let us turn to the most representative texts associated with to the two poles (topics) of the second dimension. As seen on the biplot in Figure 15.5, the tokens characteristic for the top plot overlap greatly with the tokens characteristic for one pole of the first dimension. So we may treat this pole as a variation of the right-side of dimension 0:

Dimension 1 Top, Comment 1 (excerpt):

> In this video lesson, I learned a lot about the concepts of Anthropology, Sociology and Political Science.
>
> ***Anthropology***
>
> - Is from the Latin word "antropologia", which means "the study of humanity." It is the study of human beings and their ancestors through time and space in relation to physics character, environmental, social relations, and culture. It's goals are to observe the common things among people, discover what makes people different from each other and to create new knowledge through researches about humankind and behavior.

Four Fields of Anthropology
✓ Biological and Physical Anthropology
✓ Cultural Anthropology
✓ Linguistics or Anthropological Linguistics
✓ Archaeology
[…]

Dimension 1 Top, Comment 2:

I learned a lot from this video. It gives me a **deeper understanding about the Anthropology, Sociology and Political Science**. First i learned that anthropology is from the the new latin word anthropologia which means the study of humanity. Anthropology is the scientific study of humanity's behavior, biology, cultures, communities, and linguistics in the present and past, including previous human species. The goal of anthropology is observing the common things among people, second is discover what makes people different from each other, third is create a new knowledge through researches about humankind and behavior. [...]

Dimension 1 Top, Comment 3:

Thank you Ma'am for this wonderful lesson. I enjoyed traveling with you even it's just virtual. Hope to be there when time comes.✨

I learned a lot from this video. Based from what Ma'am Dawn discussed, we can be socially and culturally competent individual by broadening our knowledge. Change is inevitable so we are bound to improve ourselves in many aspect of our life. Things may have changed but all for the better.

The first thing I learned is about Anthropology which is from the Latin word "antropologia", which means "the study of humanity." It is the study of human beings and their ancestors through time and space in relation to physics character, environmental, social relations, and culture.

Next is the sociology which is the study of society, patterns of social interactions, social life, social changes, and social causes and consequences of human behavior. [...]

In contrast, the texts associated with the opposite pole of Dimension 1, at the bottom of the graph, criticize sociology as a discipline, pointing out problems with the study of human society and social action.

Dimension 1 Bottom, Comment 1 (excerpt):

i feel like this video only **points out historically negative things**, like these were problems of the past but this video portrayed them as relevant to right now. It points out absolutly no benefits that society has done! This video makes normal folk sound like victims.

This translates to me, "Lets **put labels on everything and everyone and consider large groups of people with similar problems stereotypes**. It considers that a single demographic, aka a single type of person with a similar problem must stem from the same source, in other words, "Similar people have identical problems for the same reason". It lumps everyone a group when, in fact, they may identical problems, but are not caused by the same thing.

I'm **embarrassed to say that i study sociology in college**. In one of my sociology classes I swear it should be called, "Why women should hate men, and why black people should hate white people."

Sociology looks completely outside a person for reasons of 'Why is this a certain way?',**'Why do people act like this**?' But that does not mean when you find a reason for a cause, that the reason is the only or even valid one, there could be many reasons and those reasons can add up. Some reasons may tie into others and contribute to the cause combined not alone.[...]

Dimension 1 Bottom, Comment 2 (excerpt):

I think the topic of **sociology is generating a bad name** because as a science it is **incapable of producing perfect or repeatable results** when humans and society are far too complex to understand enough to properly model. I don't think this is a failing that disqualifies it as a science though, just look at another science with the same problem, meteorology. [...]

Now as to why I think there is a **mild degree of hatred for sociology** out there, I think most of that fits into a dislike for what was labeled "critical sociology". As was mentioned, empirical data must be collected properly and from non-biased methods to have any merit, but when the scientists conducting a study have a pre-determined bias and desire for their results to confirm that bias and spur social change those scientists are far more likely to create a flawed study that is nothing more than propaganda to support that social change. Just look up videos on why most science studies are misleading or false and they can explain dozens or even hundreds of ways bias or flaws can seep into a study by accident or on purpose. Of course you don't need to learn anything to know that devoted activists are unlikely to publish results that actually hurt their positions, so that alone makes having an initial bias unhealthy for actually understanding the truth. [...]

Dimension 1 Bottom, Comment 3 (excerpt):

> Well, **Marx didn't see coming that banks would give money too poor working class people to "save" the economy from collapsing**, or stalling it at least for a few decennia. I think banks are the thing that keeps us standing, how unstable they might be. [...]
>
> Balls to the walls Communism ahs been proven to not work exactly as advertised, or the version that is claimed to work is impractical to reach. However, Marx's observations were correct

Against this backdrop, we might label the second dimension *Summarization of learned content vs Harsh critique of sociology*.

Finally, we take a closer look at the third extracted dimension. At the top of the third dimension, we find tokens such as "field_1", "physics_1", "government_1", "batch_1", and "politics_1". These comments were published from

Figure 15.6 Biplot of the position of the active and passive variables on dimensions 0 and 2.

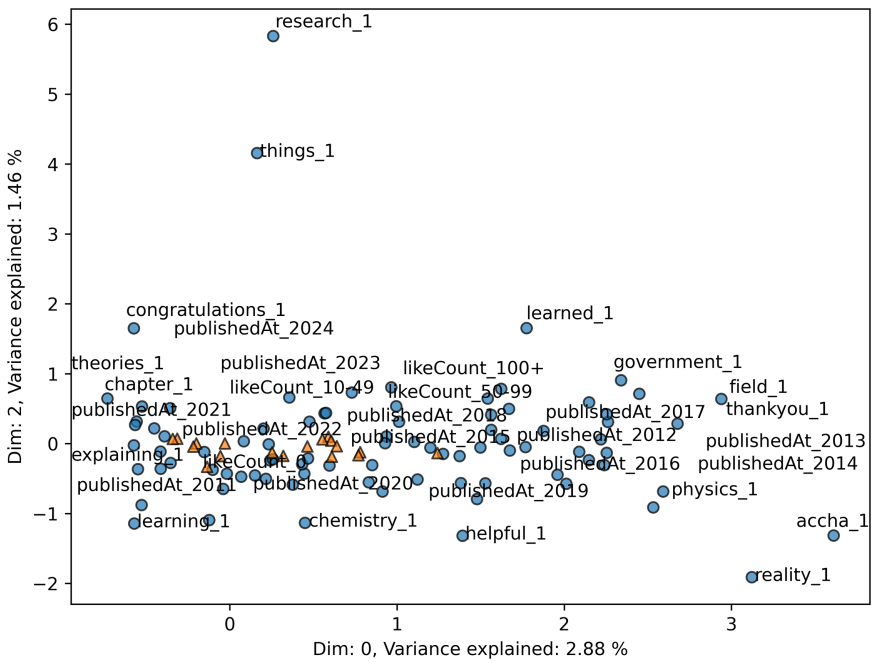
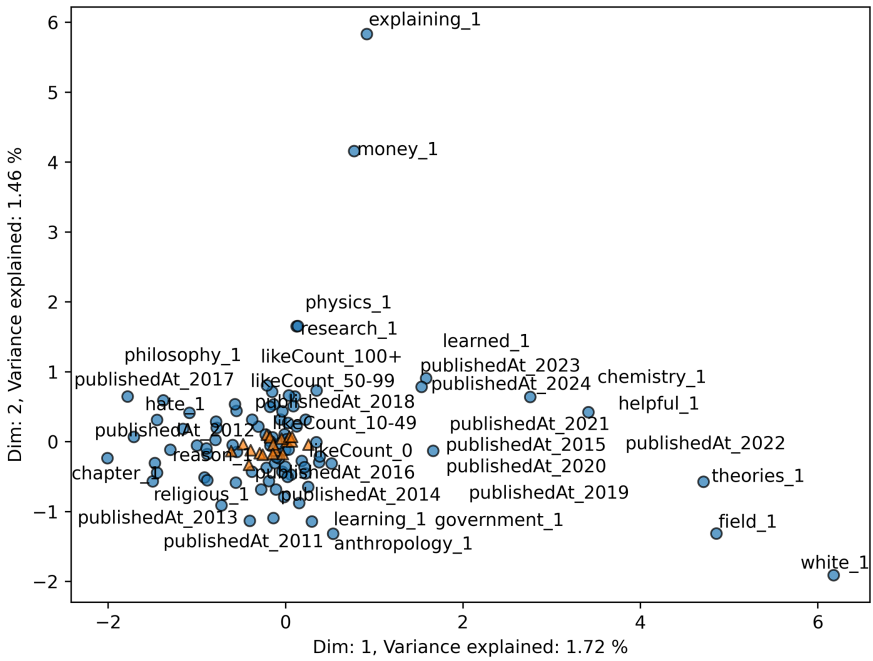


Figure 15.7 Biplot of the position of the active and passive variables on dimensions 1 and 2.



2017 onwards and received more likes compared to other comments. At the bottom of the third dimension, we see terms like "politics_1", "community_1", "research_1", "learned_1", "chapter_1", which may suggest a connection to the first dimension. However, upon closer inspection of the texts, a different story emerges. The three, exemplary comments at the top of dimension 2 indicate discussions between different scientific disciplines and the appreciation of these fields:

Dimension 2 Top, Comment 1 (excerpt):

> I simply can't understand, w**hy don't we study World History in the analytical manner as we study Sociology.** It is important to note that Humanities and Social Sciences do a similar job, however the Ethics are separated from Sciences. Now this must not be done, the Philosophical approach of the humanities must be combined with the sensible approach of the sciences. [...]

Dimension 2 Top, Comment 2:

I love Crash Course, but I am just not interested in this series. John and Hank, even Mike, Andre, Carrie Anne, and ESPECIALLY Phil present the material like they know what they are talking about and enjoy teaching it. Nicole just sounds like she is reading a script without making it relatable. Ranking every Crash Course to date (not including Kids' Science because I haven't seen it, nor Film History because there has only been one episode) **in terms of quality and enjoyment, it would look like this**:

1. World History
2. Astronomy
3. Mythology
4. Anatomy & Physiology
5. U.S. History
6. Computer Science
7. Psychology
8. Games
9. Biology
10. Literature
11. Chemistry
12. Philosophy
13. U.S. Government & Politics
14. Big History
15. Physics
16. Economics
17. Intellectual Property
18. Ecology
19. Sociology
20. Human Geography (LOL)

Dimension 2 Top, Comment 3 (excerpt):

I am still **not very convinced**. Take Jonathan Haidt and Joshua Greene, and the **revolution in moral psychology** of these last 10 or 20 years. The field is thriving. I may be wrong, but **sociology seems to be disconnected from what is going on in other fields of research**. What are the great advances in this area in these last 10 or 20 years? What is cool in science is that the fields talk to each other, and they try to fit everything together in a big theoretical body that goes from **Quantum Field Theory** to human brains, minds and groups [...]

Dimension 2 Bottom, Comment 1 (excerpt):

Thanks for educating in a way that I don't feel like I am being propagandized to agree with a point of view.

I would label myself a critical thinker who is often critical of our world society because I perceive it to be run by royal families then own these nations in governments and businesses openly proclaim that the worst they fight are against us. [...]

I am also **critical of my personal behavior** as well as the other truth-seeking citizen journalists like myself.

so I am writing in this comments section today in order to extend an olive branch to other individuals within our world community in hopes that we could combine this critical conspiracy point of view with this area of studying human culture.

Paradoxically because I feel the **need to defend my own individuality and that aspect of the entire human species** as it is being attacked by fake behavior that is promoted by the bank or run media and artificial intelligence pushing us all around towards this anti human point of view.

Now that I've learned all this information like that NASA released a warfare of the future document that explains that the plan is for this **banker run government** to attack the people by 2025 I want to learn how to organize a viable folk community locally that is connected with other independent folk communities worldwide. [...]

Dimension 2 Bottom, Comment 2:

All those **latent benefits of education** could be more easily achieved in a less structured social setting, like a playground or a park or a community project or event. The only real purpose of school is to give children information that they wouldn't have researched on their own... . and thanks to government **programs like No Child Left Behind**, the schools have been failing in this function.

Personally, when I think back on my years of public education, the only class where I learned anything useful was shop class. (I learned math and reading at home. History didn't stick. I forgot my Spanish. I never really found a use for algebra. As for music, well, I just kind of sucked at that.) Meanwhile, things I actually needed to know, like how taxes work, how to get a job, how to vote, how to do laundry, etc., were not taught in schools. **So tell me, what did I spend those K-12 years learning**???

Dimension 2 Bottom, Comment 3 (excerpt):

I'm just gonna openly ask anyone for criticisms of my 12 marker I did.

*Discuss how far sociologists would agree that the main purpose of education is to promote social mobility.

Social mobility is the movement between social classes, usually referring to people "climbing the social ladder". The **education system** is said to promote social mobility by giving students knowledge and qualifications that they then move on to use to get a job and earn money.

Functionalists believe in the organic analogy which is the idea that all institutions work together to make an individual. They **believe that education prepares students for the real world by making it a miniature society where real-world problems and skills are taught.** Aswell as the national curriculum, the hidden curriculum provides students with skills they need in the world of work, like time management and acceptance of social hierarchy. These skills prepare students for the world of work, and the meritocratic education system teaches students that with effort, they can achieve upwards social mobility.

[...]

Overall, **Sociologists believe that education has the intentions to promote social mobility but aren't achieving this due to numerous unaddressed issues**. These is a narrow criteria of students that the education system works really well for, while others are forgotten. These are usually working-class students, and especially boys when it comes to results.

If you've read that all wow thank you so much and good luck with your exam!!!

Table 15.3 Interpretation of the First Three Dimensions Extracted by the MCA From the YouTube Comments

Dimension	*Interpretation*	*Variance explained (cumulative)*
1	Summarization of learned content Vs Praise and gratitude for teaching	2.82 %
2	Summarization of learned content Vs Harsh critique of sociology	4.50 %
3	Hierarchy of sciences and relation of sociology and other disciplines Vs Critique of institutions and (state) organizations	5.86 %

Overall, we can interpret the third dimension as opposition between the "hierarchy of sciences and relation of sociology and other disciplines versus the Critique of institutions and (state) organizations". Table 15.3 summarizes the three oppositions found in the data, accounting for five topics in total.

15.2 Semantic network analysis

The next, relational method we will focus on in the current chapter is **semantic network analysis**. Semantic network analysis (SNA) comes in a variety of flavors, but we will use it as a means to analyze networks of concepts and employ network measures as well as community detection algorithms to extract topics. Within this framework, we can analyze, for example, Habitus-Field Theory in how it relates "field", "habitus", "practices", (kinds of) "capital", and "position" (Fuhse, 2022, S. 106ff.). Or we can examine the changing meaning of concepts like "sociology" through their links of association and opposition with other concepts. Semantic network analysis has two distinct advantages according to Hecking and Leydesdorff (2019). First, we can use semantic network analysis for small and sparse datasets. Secondly, they are more stable compared to generative topic modeling approaches, including Latent Dirichlet Allocation, when tokens are removed from the dataset. Currently, semantic network analysis is being extended to socio-semantic network analysis, which provides us with a profound theoretical underpinning if we decide on applying this approach (Fuhse et al., 2020; Roth & Cointet, 2010). In this vein, socio-semantic networks are networks of social and symbolic relationships (e.g., between ideas) that are studied in conjunction (Fuhse, 2021). In essence, semantic network analysis opens the door to studying the relations of concepts, topics, and their speakers over time, and allows us to conceive topics as products and initiators of social interaction.

In the following, we focus on the application of semantic network analysis as a method of automated content analysis. But to do so, we must first understand the basic concepts underlying network analysis. These are introduced in Section 15.2.1. Afterwards, we will get more technical in Section 15.2.2 and show you how to prepare your data to conduct SNA. We do this mainly by converting the document-term matrix used for GDA into a relational data structure. We then continue with establishing a network graph and extracting topics from the semantic network in Section 15.2.4 and visualize it in Section 15.2.6. Finally, we will show you how to select texts most closely associated with the extracted topics and how to interpret these topics in Section 15.2.7.

15.2.1 Basic concepts of network analysis

Basically, networks are **graphs**. Graphs are abstract objects that consist of **nodes** (or vertices) and **edges** (ties) that connects them. Nodes and edges can

be virtually anything. In our case, we equate nodes with tokens and edges with the co-occurrence of tokens across texts. A network graph yields edges that are either directed or undirected. **Undirected edges** signal reciprocal relations between the nodes (in our case: co-appearance), whereas **directed edges** symbolize that one node refers to another node (e.g., one actor refers to another actor, one text cites another text). Edges can take either binary values (0 = no connection between nodes, 1 = connection exists between nodes), or continuous values, also called "weights", which indicate the strength of the connection between nodes. In our case, we will use weighted, undirected edges as they indicate the co-occurrence of tokens across texts.

In a network, changes in the local structure (connections among nodes, here: tokens) entail changes in the global structure (also called topology, or the overall connectivity and "holes" in the network) and vice versa (Kossinets & Watts, 2006). This is why we must consider both global and local structure when we extract and topics from a semantic network and when interpreting both network and topics. But what are local and global network structures, and how can we measure them?

Local structures comprise the (more or less) direct neighborhood of a given node, or a set of nodes. Local structures indicate the embeddedness of nodes (e.g., actors, tokens) within a given network. This gives a hint on: (a) what concepts and topics are relevant in the data, (b) how densely or loosely the net of topics and concepts is woven, and as a consequence, (c) how sharply we can distinguish between topics. The term *global structure* refers to overall properties of a network, which are related to their topology (= porosity and connectedness). The properties are number of nodes (= tokens), the share of realized edges on the total number of possible edges, or the degree of fragmentation between isolated subgroups.

To capture different aspects of the local structures, we can employ the canonical centrality measures described by Freeman (1978), namely degree centrality, eigenvector centrality, closeness centrality, and betweenness centrality, and include network constraint described by Burt (2017).

Degree centrality measures the number of ties assigned to each node. In our case, we can use the weighted degree centrality to measure the strength of co-occurrence of tokens across texts (e.g., the tokens "sociology" and "lesson" are co-cited ten times together, so this link will have an initial weight of 10). According to Freeman (1978, 219f.), degree centrality is associated with the visibility of a node within a network, and thus its potential to be a focus of activity. In our case, this may give a hint on how strongly certain tokens are embedded within its local structure, highlighting their importance for the potential, underlying topic.

Eigenvector centrality measures the relative influence of a node in a network, as it takes the connectivity (e.g., weighted number of edges) of the neighbors of a given node into account. For example, a token might yield connections to a few other tokens, which, in turn, might be very well connected to multiple other tokens. In our case, eigenvector centrality signals the influence of tokens on the meaning structure of larger parts of the network.

Betweenness centrality measures the weighted share of shortest paths between two nodes, which pass through a node under observation (Freeman, 1978, pp. 222–224). In other words, betweenness centrality measures the probability of a node to control the flow of information in a given network. High betweenness values indicate the possibility that a token links different concepts both within and between topics. However, we will focus on the shortest paths within a topic to identify tokens that may be most characteristic for the respective topic.

Next, *closeness centrality* follows the idea of independence of nodes from intermediaries in a network (Freeman, pp. 224–225). It also indicates the possibility of a node to easily spread information to even distant outskirts (= periphery/peripheries) of a network. Tokens with high closeness centrality-values may connect many distinct regions of the semantic networks and may thus be considered as concepts that link topics together.

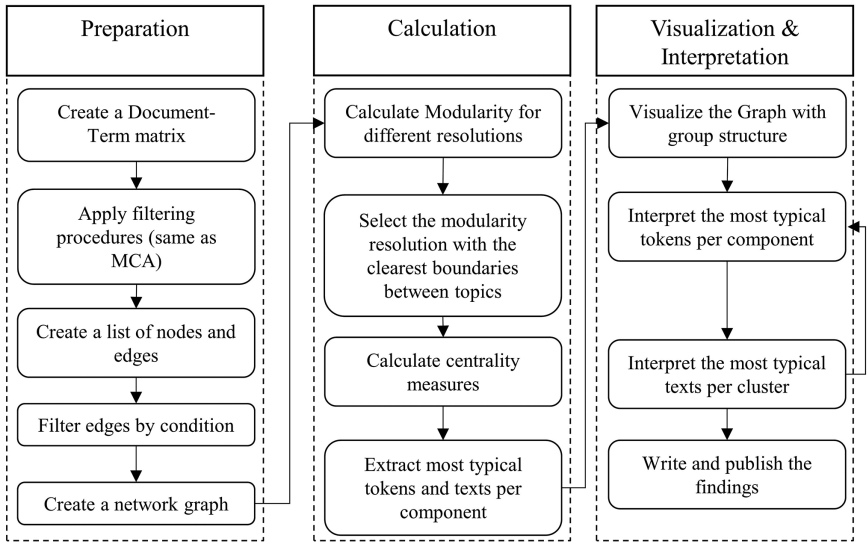
Finally, *constraint* is a measure of brokerage, or the lack thereof. It is calculated as the share of nonredundant ties of a node and its neighbors (Burt, 2017). In this regard, it is assumed that a token with low constraint values may be able to span multiple topics. In our case, high constraint values indicate the local closure of topics, and the embeddedness of tokens associated with a topic.

Furthermore, there are several measures available to characterize the global network structure. One of those is *transitivity* (Newman, 2006), which means the share of triplets of all possible triplets. A triplet consists of three nodes (triad), which are connected among each other (the token sociology co-occurs with lecture, and both co-occur with critique). Transitivity measures how densely connected the semantic network is overall and how structured or unstructured to the co-occurrences of tokens are. If transitivity values are high, this would mean that (nearly) all tokens are co-occurring in all texts.

Modularity measures the degree to which a graph (in our case, the semantic network) is fragmented into different densely connected subgroups. Modularity is the sum of all realized connections among nodes (= tokens), divided by the expected strength of the connections if these edges were placed randomly. The calculation of this difference is restricted to nodes (= tokens) within the detected subgroups. These subgroups are called *components* and are detected, if a number of nodes (tokens) are more strongly connected than average, while their connection to other nodes is weaker than average. Consequently, the clearer cut the boundaries between these groups, the higher the *modularity* score.[7] In a semantic network, the detected components relate to the topics extracted from the graph. Modularity scores can take values between –0.5 and 1. Values below zero indicate a lower modularity – or clustering – than expected by pure chance, the value zero is equal to network clustering by pure chance, and positive values indicate a stronger clustering of nodes than we would expect if the edges were set by pure chance.

Equipped with this knowledge, we may now proceed to the technical part of our semantic network analysis. Analogous to the MCA, the workflow in Figure 15.8 depicts the steps to be taken to use semantic network analysis as a

Figure 15.8 Workflow depicting the steps taken to conduct a semantic network analysis.



method for automated content analysis. However, a small addition should be made at this point. To get an overview on the topic dimensions, we will make use of the most central tokens in each community in terms of betweenness centrality. Although we could use any other centrality measure to define the most central tokens, we opt for betweenness centrality as it is a measure of the shortest distances between all nodes in a given cluster, as well as in the overall network. In our case, we can interpret this measure as importance for the local, as well as for the global structure of the semantic network.

15.2.2 Converting our data frame into a relational structure

Similar to the MCA, we start the data preparation process with importing the necessary packages (see Code box 15.27) and use the same data to conduct a semantic network analysis. We import the `os`, `re`, and `pandas` (as `pd`) packages to handle the data table. For text preparation, including feature selection, feature extraction, and the construction of a document-term matrix, we import the `text` module from `sklearn.feature_extraction`, `spacy`, `langdetect`, and spacy's `en_core_web_sm` language model. Additionally, we import `itertools` to create an `edgelist`, `numpy` (as `np`) for calculation purposes, `networkx` for network graph construction and to calculate network measures, and the `community` package (as `cm`) to extract potential topics communities. Lastly, we import `matplotlib.pyplot` (as `plt`) and the `adjust_text` from `adjustText` for visualization purposes.

```
import os
import re
import pandas as pd

from sklearn.feature_extraction import text as tx
import spacy
from langdetect import detect
nlp = spacy.load("en_core_web_sm")

import itertools as it
import numpy as np
import networkx as nx
import community as cm

from matplotlib import pyplot as plt
from adjustText import adjust_text
```

Code box 15.27 Import of packages needed to prepare the data for and to conduct a semantic network analysis

As in MCA, we start with data preparations. Since the first two steps of the SNA workflow correspond to the first four steps in the MCA workflow (depicted in Figure 15.2), we skip them and proceed directly with creating a list of nodes (tokens) and edges (co-occurrences of tokens across texts). We use the same feature extraction (Code box 15.2) and filtering procedures (Code box 15.3) as before on `dtm_df`. This leaves us with a document-term matrix filled with zeros (=token does not appear in a text) and ones (=token appears in a text). The columns of this matrix represent tokens, while the index names represent comment IDs.

To establish a semantic network, we derive an edge list from `dtm_df`. In our case, an edge can only be spanned between two tokens that co-occur in a text. This is why we filter out comments (rows) of our document-term matrix with less than two tokens (see Code box 15.28).

```
dtm_df = dtm_df[dtm_df.sum(axis = 1) > 1]
```

Code box 15.28 Keeping the rows in the document-term matrix with at least two tokens

To do so, we use the conditional indexing operation `dtm_df[dtm_df.sum(axis = 1) > 1]`. We then replace the `dtm_df` variable.

As the columns of `dtm_df` represent the tokens, which will be the nodes of our network, we extract them with the following list comprehension and assign them to a new variable we will call `nodelist` (see Code box 15.29):

```
nodelist = [x for x in dtm_df.columns]
```

Code box 15.29 Creating the node list from dtf_dm column names

The next lines of code depicted in Code box 15.30 show you how to establish a Pandas data frame that contains all the information needed to create the edge list, which we will then pass to our network graph. The following steps are required to do so:

1) We create an empty list that stores the edges of the co-occurring tokens (edgelist).
2) We iterate through each row (=comment) of our document-term matrix `dtm_df.`
3) We check, where the values of the cells (=tokens per comment) is greater than 0 (=token appears in the comment) and keep the name of the tokens with `dtm_df.loc[i][dtm_df.loc[i] > 0].index`.
4) If this condition holds, create a list of all token pairs (=edges) per text document and append it to the edgelist with `edgelist.append([c for c in it.combinations(nodes, 2)])`. This results in a list of lists of token pairs.
5) We flatten the list to generate an edgelist with token-pairs across texts with `[item for sublist in edgelist for item in sublist]`.
6) We create a Pandas data frame containing the node-pairs and edge weights, which will be passed to the network graph-constructor with `pd.DataFrame(edgelist, columns =["in", "out"]).size().reset_index()`.

```
edgelist = []
for i in dtm_df.index:
    nodes = dtm_df.loc[i][dtm_df.loc[i] > 0].index
    edgelist.append([c for c in it.combinations(nodes, 2)])

edgelist = [item for sublist in edgelist for item in sublist]
edge_df = pd.DataFrame(edgelist, columns = ["in", "out"])
edge_df = edge_df.groupby(by = ["in", "out"]).size().reset_index()
```

Code box 15.30 Constructing a pandas data frame containing node pairs with co-occurrence counts

This procedure generates a data frame with two edges (`in` and `out`), as well as their number of co-appearances across the text documents.

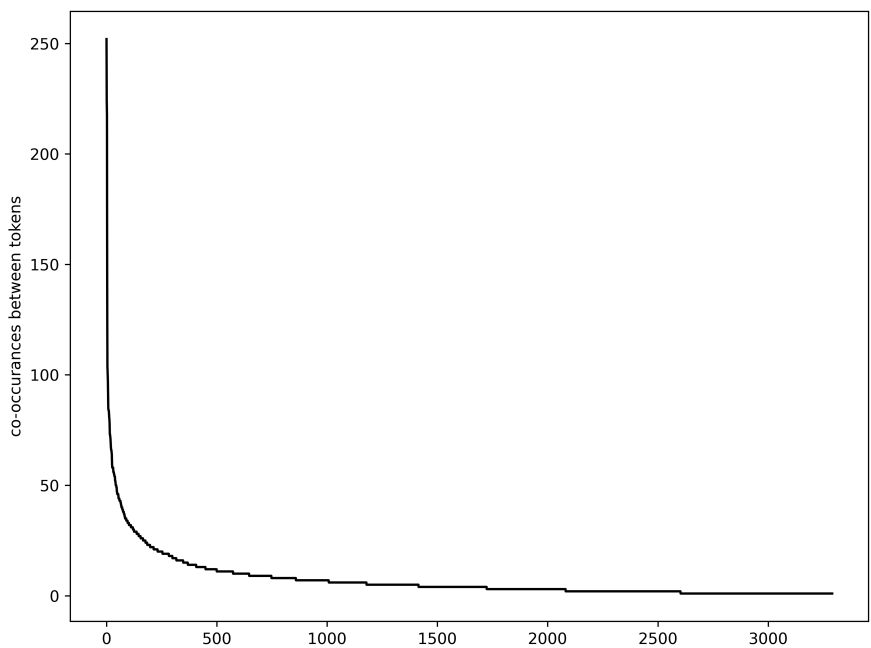
15.2.3 Filtering meaningful edges

In the next step, which corresponds to the fourth box on the left of the workflow in Figure 15.8, we determine which edges to retain when constructing our semantic network. This decision is crucial, as it significantly impacts the topics extracted by the community detection-algorithm. Here, our goal is to keep informative edges that reveal both local and global structures within the network. We can apply different approaches, including:

1) Setting a fixed lower threshold (e.g., a minimum of five token co-occurrences).
2) Retaining edges with a weight above the mean or median edge weight.
3) Using measures such as negative pointwise mutual information (NPMI) to retain only edges constructed from node pairs that co-occur more often than expected by chance.

All three approaches address the issue illustrated in Figure 15.9, namely that only a small number of token-pairs co-occur frequently, while most

Figure 15.9 Distribution of the co-occurrence of token pairs (= edges) across the comments.



appear only a few times. Statistically this results in a highly right-skewed distribution, where many co-occurrences of tokens-pairs may be due to pure chance. Hence, it is important to consider not only raw co-occurrences but also the overall frequency of each token. For example, if a token appears only 15 times in the corpus, its maximum possible co-occurrence count with any other token is limited to 15. If we follow strategy (1) and set the threshold to 20 co-occurrences, or if (2), if the mean edge weight is greater than 15, then all edges related to this token would be dropped. This is problematic when rare but meaningful terms, which are necessary to understand a topic (e.g., "discourse analysis", "liquid modernity", or "world society") would be excluded. To address this issue, we opt for strategy (3) and use the Negative Pointwise Mutual Information (NPMI) to preserve informative edges.

This procedure was successfully applied to a topic modeling approach based on semantic networks by Austin et al. (2024). They define the formula of the NPMI as follows (Figure 15.10):

Figure 15.10 Formula of the negative pointwise mutual information between two tokens/nodes present in the semantic network.

$$NPMI(token_1, token_2) = \frac{\log\left(\frac{p(token_1, token_2)}{p(token_1) * p(token_2)}\right)}{-\log p(token_1, token_2)}$$

This function assesses if a token pair $token_1, token_2$ co-occurs more frequently than expected by pure chance. The numerator on the right-hand side of the equation computes the normalized probability ratio between observed ($p(token_1, token_2)$) and expected co-occurrences if the token pair were independent ($p(token_1) * p(token_2)$). As the probabilities for tokens to co-occur range from 0 to 1, the logarithm of the numerator is zero (=logarithm of one) at most, but is typically negative. This is why we calculate $-\log p(token_1, token_2)$ to allow (a) for positive values and (b) to normalize the numerator on the co-occurrence of tokens, so that values greater than 0 yield the information that the token-pair indeed co-occurs more often than by pure chance and, consequently, that the edge may contribute to the topics extracted.

To compute NPMI values, we iterate through the edge list, calculate the probability that two tokens co-occur, and apply the formula (Code box 15.31).

```
NPMI_edgelist = []
for i in edge_df.index:
    print(f"Currently calculating value for edge {i} of
    {len(edge_df)}")
    token1 = edge_df.loc[i, "in"]
    token2 = edge_df.loc[i, "out"]

    p_token1_token2 = edge_df.loc[i, 0] / edge_df[0].sum()
    p_token1 = edge_df[(edge_df["in"] == token1) |
    (edge_df["out"] == token1)]
    p_token1 = p_token1[0].sum() / edge_df[0].sum()
    p_token2 = edge_df[(edge_df["in"] == token2) |
    (edge_df["out"] == token2)]
    p_token2 = p_token2[0].sum() / edge_df[0].sum()

    numerator = np.log(p_token1_token2 / (p_token1 * p_token2))
    denominator = -np.log(p_token1_token2)

    NPMI = numerator / denominator
    NPMI_edgelist.append((token1,token2,NPMI))

NPMI_edge_df = pd.DataFrame(NPMI_edgelist,
    columns = ["in","out","NPMI"])
NPMI_edge_df = NPMI_edge_df[NPMI_edge_df.NPMI > 0]
```

Code box 15.31: Calculating the negative pointwise mutual information per edge, and keeping the edges of nodes that co-appear more often than expected

We begin with initializing an empty list to fill (`NPMI_edgelist`) before we use a `for` loop with iterator `i` to iterate through the rows in `edge_df`. We proceed as follows within the for loop to calculate the NPMI values:

1) Pull the first node from a row of the edge list by calling `edge_df.loc[i, "in"]`.
2) Pull the second node from the edge list by calling `edge_df.loc[i, "out"]`.
3) Select rows where both tokens co-occur and calculate the number of co-occurrences across texts with `p_token1_token2 = edge_df.loc[i, 0] / edge_df[0].sum()`.

4) Select all rows which contain the first token with `p_token1 = edge_df[(edge_df["in"] == token1) | (edge_df["out"] == token1)]`.
5) Calculate the probability to encounter only token 1 across the data frame with `p_token1 = p_token1[0].sum() / edge_df[0].sum()`.
6) Repeat steps 4 and 5 with the second token (`p_token2`).
7) Calculate the numerator of the NPMI value using `np.log(p_token1_token2 / (p_token1 * p_token2))`.
8) Calculate the denominator of the NPMI value by calling denominator = `-np.log(p_token1_token2).`
9) Calculate the NPMI value of the token-pair with `NPMI = numerator / denominator`.
10) Add the value to the list which stores the token pairs and NPMI values with `NPMI_edgelist.append((token1,token2,NPMI))`.
11) Initialize a Pandas data frame from the `NPMI_edgelist` with `NPMI_edge_df = pd.DataFrame(NPMI_edgelist, columns = ["in", "out", "NPMI"])`.
12) Filter only informative edges with an NPMI greater than 0 with `NPMI_edge_df = NPMI_edge_df[NPMI_edge_df.NPMI > 0]`.

We exemplify this procedure using the token-pairs present in the first row of our `edge_df`, namely "people" and "think" (Output 15.10), which yield an NPMI value of −0.0196 according to the formula (Output 15.11). The negative value indicates that the two tokens appear less often than by pure chance, meaning that they are not informative and are dropped from the edgelist.

```
people think
```

Output 15.10: Example of tokens stored in the token1, and token2 variables within the for loop in Code box 15.31

```
NPMI
-0.01956672381264082
```

Output 15.11: NPMI value of the token-pair "people" and "think"

The final filtering procedure of Code box 15.31 results in the following data frame, which serves as input for the creation of the semantic network.

	in	out	NPMI
0	accha	age	0.024191
1	accha	background	0.103379
2	accha	chapter	0.136749
3	accha	chemistry	0.029199
4	accha	coaching	0.164863
..	. . .	. . .	. . .
447	trying	using	0.025983
448	trying	watched	0.004100
449	watched	whole	0.028843
450	white	years	0.003086
451	write	writing	0.066794

Output 15.12: Data frame that stores the edge list with NPMI values greater than 0

As demonstrated in Code box 15.32, we use the `nx.Graph()` function to initialize an empty, undirected graph and assign it to `N` (short for network). We then iterate simultaneously through the edges (`e1` and `e2`), as well as the `NPMI` stored in `NPMI_edge_df`, and use the `N.add_edge()` method to add the weighted edge (e.g., `N.add_edge("sociology", "study", weight = 3)`) to the network graph object.

```
N = nx.Graph()
for e1,e2,NPMI in zip(NPMI_edge_df["in"], NPMI_edge_df["out"],
NPMI_edge_df["NPMI"]):
 N.add_edge(e1, e2, weight = NPMI)
```

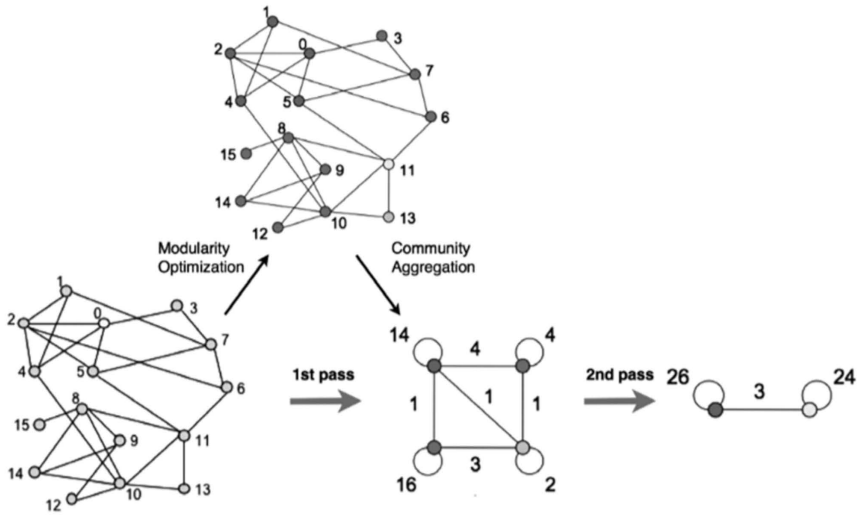
Code box 15.32: Construction of an undirected networkx graph object

After this step, we are ready to extract the topic communities from the semantic network and use these communities for topic interpretation.

15.2.4 Topic extraction and topic assignment to the texts

Now, we turn to the box on top of the middle section of the workflow in Figure 15.8. In this step, we extract communities of co-occurring tokens, which

Figure 15.11 Visualization of the community detection algorithm.



Source: (Blondel et al., 2008, p. 5)

can be interpreted as topics. However, the number of topic-communities to extract from our semantic network is unknown. In other words, we apply an unsupervised learning-algorithm (community detection) which identifies patterns from the data, but requires us researchers to assign meaning to these patterns. This raises the following questions: Should we rather extract topics that consist of few or many tokens? How can we draw boundaries between the communities? And how does the community detection algorithm, implemented in the `python-louvain` package, actually work?

The *community detection algorithm* is an agglomerative clustering algorithm (Blondel et al., 2008). *Clustering* describes the process of grouping entities (e.g., tokens, persons, organizations) based on some shared traits. *Agglomeration* means that entities (= tokens) are grouped iteratively according to the similarity of their traits, merged into groups, which are then treated as nodes and compared in regards to their similarity.

The community detection algorithm consists of two iterative steps. First, during initialization, each node (token) is assigned a unique cluster number. In a second step, the algorithm evaluates whether moving each node into a different community increases the modularity (= measure of fragmentation of the network). The nodes are then placed in their respective communities, which gain members by agglomeration. These newly formed communities are treated as single nodes in the next iteration, and this procedure is repeated until no further modularity gain is possible.

In network terms, agglomeration is possible when nodes within a network are hihgly interconnected, while their connections to other nodes or potential groups within the network are comparatively weaker. Each iteration starts with previously detected communities, which are treated as nodes. Then, the algorithm tests whether merging communities increases modularity. If this is the case, the communities are merged, and the merged communities are then treated as new nodes with self-loops. This process repeats until no further gain in modularity is possible. Figure 15.11 depicts this process.

This procedure explains how the community detection algorithm performs the grouping procedure iteratively. However, this does not solve the question of how many tokens should ideally form a community and how many communities may be present in the network graph. The parameter that governs the sizes and number of communities to extract is called *resolution*. Smaller resolution values result in smaller communities, larger values in larger communities.

It is our task to select a model with the highest modularity value. To achieve this, we proceed with calculating the modularity values of the network for different resolutions (Code box 15.33).

```
modularities = []
for resolution in np.arange(0.05, 1.55, 0.05):
    communities = cm.best_partition(N,
                        resolution = np.round(resolution, 2),
                        weight = "weight",
                        random_state = 1)

modularity = cm.modularity(communities, N, weight = "weight")
modularities.append((resolution,modularity))
modularities = pd.DataFrame(modularities,
               columns = ["resolution", "modularity"])

best_resolution = modularities[
                  modularities.modularity ==
                  modularities.modularity.max()
                  ]
best_resolution = best_resolution.iloc[0]
```

Code box 15.33: Extracting the optimal number of clusters using community detection and modularity comparison

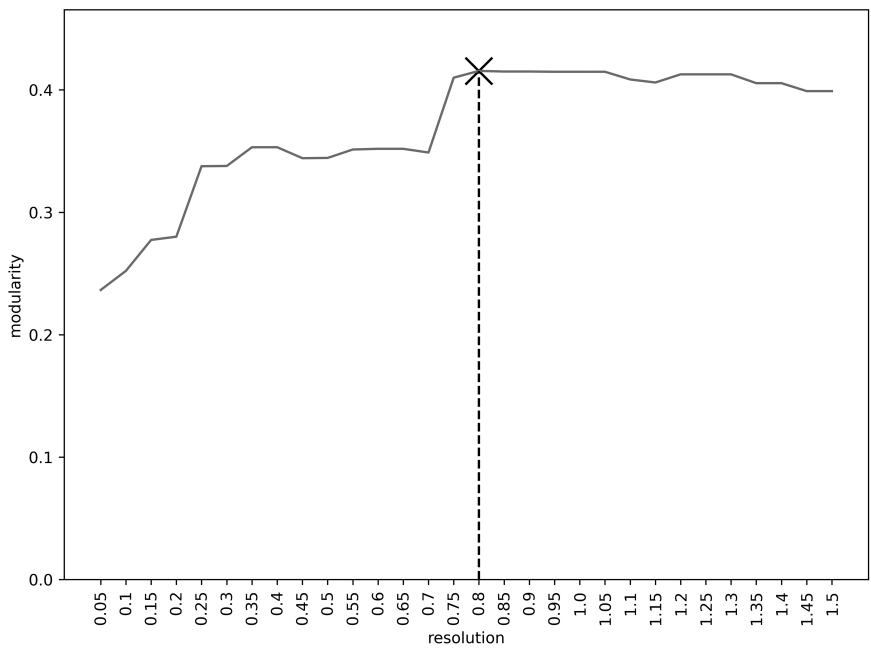
As usual, we start with assigning an empty list to a variable (here: `modularities`). We then initialize a for loop to calculate the modularity values for resolutions between 0.005 (very small clusters) and 1.5 (large clusters) in steps of 0.05 using `np.arange(0.05, 1.55, 0.05)`.[8]

As a first step within the `for` loop, we call the `best_partition()` function and initialize the community detection algorithm. To extract the optimal solution, we pass:

- `N` = The initialized, undirected network graph.
- `resolution =` The resolution of the communities to extract.
- `weight =` The key which stores the NPMI value of each token pair.
- `random_state =` To make our calculation as replicable as possible.

Next, we call `cm.modularity()` to calculate the modularity of the network graph and pass the detected `communities`, the semantic network `N`, and the weights of the token pairs (`weight`). In a next step, we append a tuple which contains the resolution and the modularity to the `modularities` list, which serves as input data for the Pandas data frame (with column names `columns = ["resolution", "modularity"]`). Next, we extract the model with the highest modularity by using the pandas selecting operation and test, where the highest modularity value is stored: `modularities[modularities.modularity == modularities.modularity.max()]`. This leaves us with a data frame with a single row, which is why we finish our selection procedure with assigning `best_resolution.iloc[0]` to `best_resolution`.

Figure 15.12 Line plot representing the extracted modularity value for each resolution parameter.



Before we continue with extracting the centrality measures and assigning the texts to the communities we check, if the modularity algorithm really chose an optimal model. To do so, we use the code shown in Code box 15.34 to visualize the modularity values for each resolution as depicted in Figure 15.12. The figure suggests extracting relatively small clusters with a resolution of 0.8 and a value of approximately 0.4, which roughly equates to a medium level of clustering.

```
plt.figure(figsize = [8, 6])
plt.plot(modularities.resolution,
        modularities.modularity,
        color = "grey")

plt.scatter(best_resolution.resolution,
        best_resolution.modularity,
        marker = "x",
        s = 300,
        color = "black")
plt.vlines(x = best_resolution.resolution,
        ymin = 0,
        ymax = best_resolution.modularity,
        color = "black",
        linestyles = "--")
plt.ylim(0, best_resolution.modularity.values + 0.05)
plt.xlabel("resolution")
plt.ylabel("modularity")
plt.xticks(ticks = [x for x in modularities.resolution],
        labels = [np.round(x, 2) for x in modularities.
resolution],
        rotation = 90)
plt.tight_layout()
os.chdir(figures)
plt.savefig("network_modularity_resolutions.png", dpi = 600)
plt.close()
```

Code box 15.34: Visualization modularity by the resolution

In regards to the code, we start with creating a figure canvas of 8 by 6 inches by calling `plt.figure(figsize = [8, 6])`. Then, we draw the line plot with `plt.plot()` and assign modularity values to the x-axis (`modularities.resolution`), modularity scores to the y-axis (`modularities.modularity`), and color the line gray. Next, we use `plt.scatter()` to highlight the resolution

of the model with the highest modularity value. To do so, we pass the best_resolution variable to the scatterplot, color this point black, set the size to 300 (= font size of 300), and use marker = "x" to change the marker. We do so to emphasize the optimal modularity-resolution. Furthermore, we create the black, dashed vertical line seen in Figure 15.12, we use plt.vlines(x = best_resolution.resolution, ymin = 0, = best_resolution.modularity, color = "black", linestyles = "--").

The next block of matplotlib functions adjusts the appearance of the graph. We change the extension of the y-axis with `plt.ylim(0, best_resolution.modularity.values+0.05)`, and set its lower bound to 0, and the upper bound dynamically to the maximum modularity value plus 0.05. We then change the x- and y-labels with `plt.xlabel()` and `plt.ylabel()`, and change the tick marks on the x-axis the `xticks()` function. We pass lists of equal length to `tickmarks` (`[x for x in modularities.resolution]`) and `labels` (`[np.round(x, 2) for x in modularities.resolution]`), and change their rotation by 90 degree (`rotation = 90`). We then fit the plot to the figure canvas with `tight_layout()`, before we change the folder, save the figure in high resolution `plt.savefig("network_modularity_resolutions.png", dpi = 600)`, and close the figure canvas with `plt.close()`.

In a next step, we create and export a Pandas data frame that contains the token centrality values and the assigned communities (Code box 15.35):

```
best_partition = cm.best_partition(N,
                        resolution = best_resolution.resolution,
                        random_state = 1)
best_partition_values = best_partition.values()
node_values = pd.DataFrame(dict(
    token       = node_values.index,
    degree      = nx.degree_centrality(N),
    eigenvector = nx.eigenvector_centrality(N),
    closeness   = nx.closeness_centrality(N),
    betweenness = nx.betweenness_centrality(N),
    constraint  = nx.constraint(N),
    modularity  = best_partition))
```

Code box 15.35: Extracting centrality values and assigned communities

```
   token     degree eigenvector . . . closeness constraint modularity
0  accha       0.152941 0.178715 . . . 0.454545 0.134999   5
1  age         0.082353 0.051074 . . . 0.410628 0.191316   1
2  background 0.211765 0.193536. . .  0.500000 0.094214   2
3  chapter     0.200000 0.194258  . . .  0.461957  0.112375    3
4  chapters    0.200000 0.189352. . .   0.452128 0.103853   3
. . .
```

Output 15.13: First five rows of the data frame containing the token centrality values and assigned communities

NetworkX provides us with a wide range of centrality measures. Here, we include the four canonical centrality measures (Freeman, 1978), the constraint measure (Burt, 2017), and the community assignment determined by the community algorithm (Newman, 2006). We pass the network graph, N, to the following functions to calculate the following measures:

- `nx.degree_centrality()` to calculate the node degree centrality.
- `nx.eigenvector_centrality()` to calculate eigenvector centrality.
- `nx.betweenness_centrality()` to calculate relative betweenness centrality.
- `nx.closeness_centrality()` to calculate closeness centrality.
- `nx.constraint()` to calculate constraint values.
- The community assignment (`modularity`) by passing `best_partition`.

These values are passed to a dictionary to create the data frame depicted in Output 15.13.

We then sort our data frame by the community number in ascending order and by betweenness centrality in descending order before saving it as a CSV file (Code box 15.36):

```
node_values.sort_values(by = ["modularity", "betweenness"],
                        ascending = [True, False],
                        inplace = True)
node_values.to_csv("node_descriptives.csv", sep =";")
```

Code box 15.36: Export of the node centralities and assigned communities as CSV file

This data preparation ensures that we can assign the most typical tokens per topic, which is an essential step in our process.

15.2.5 Assigning the most typical tokens to extracted topics

We use the code snippet in Code box 15.37 to create and export a portion of the data frame that contains just the five tokens with the highest betweenness centrality per extracted community:

```
cluster_most_typical_tokens = []
for m in set(node_values.modularity):
tokens = ", ".join(list(node_values[node_values. modularity == m].\
    head().token))
    cluster_most_typical_tokens.append((m, tokens))

cluster_most_typical_tokens = pd.DataFrame(cluster_most_typical_
tokens,
    columns = ["modularity", "tokens"])
cluster_most_typical_tokens.to_csv("top_tokens_per_cluster.
csv", sep = ";")
```

Code box 15.37: Generating and exporting the five most central tokens per community

The procedure to do so is as follows:

1) Assign an empty list to `cluster_most_typical_tokens`.
2) Iterate through the unique community numbers stored in `set(node_values.modularity)`.
3) Use a Pandas conditional selecting procedure to select rows that matches the iterator value `m` (`node_values[node_values.modularity == m]`).
4) Extract the first five tokens with the highest betweenness centrality per community using the `.head()` function.
5) Convert the extracted tokens to a list using the `list()` function.
6) Using the `", "` `.join()` – to merge the five most typical tokens per community into a single string and assigning it to `tokens`.
7) Append this string (`tokens`) to `cluster_most_typical_tokens`.
8) Following the loop, we convert the list into a Pandas data frame and save it using pandas' `to_csv()` method.

Next, we assign potential topic clusters – indexed by community – to individual texts. This requires us to do the following:

1) Reduce our document-token matrix to contain only tokens used as nodes in the semantic network graph.
2) Calculate the association of a document with the topic using betweenness centrality as weight.
3) Merge topic loadings with the original data frame (`df`) to allow us to analyze all of the full texts as well as their metadata.

These steps are realized in Code box 15.38:

```
dtm_df_reduced = dtm_df[list(node_values.token)]
for c in set(node_values.modularity):
    modularity_class = node_values[node_values.modularity == c]
    modularity_values = dtm_df_reduced[modularity_class.token]

    for token, betweenness in zip(modularity_class.token,
    modularity_class.betweenness):
    modularity_values[token]   =   modularity_values[token]   *
    betweenness
    modularity_values = modularity_values.sum(axis = 1) / dtm_
    df_reduced.sum(axis = 1)

    variable_name = "Modularity_Class_" + str(c)
    dtm_df_reduced.insert(c, variable_name, modularity_values)
select_variables = [x for x in dtm_df_reduced.columns if \
                  re.search("^Modularity",x)]
community_weighting = dtm_df_reduced[select_variables]
```

Code box 15.38: Code scheme used to assign the respective texts to the potential topic communities

The first step selects only the token names stored as columns in the document-term matrix and creates `dtm_df_reduced`. The second step loops through the extracted communities and

1) Selects only the tokens associated with a community (modularity_class = node_values[node_values.modularity == c]).
2) Assigns betweenness values to the tokens appearing in the texts (`modularity_values = dtm_df_reduced[modularity_class.token]`).

3) Creates a weighting scheme which represents topic loadings (= association between texts and topics). We normalize the weighting scheme to values between 0 and 1, with 0 indicating that a given text is not associated with a potential topic at all.[9] We use the second loop as follows to do so:

   a. We initialize the loop using the `zip()` and iterate simultaneously through the tokens and betweenness centrality values per community `modularity_class.token` and `modularity_class.betweenness`.
   b. We multiply token values with the betweenness centrality, to generate a weighted topic loading.
   c. We add all weighted betweenness values using the `modularity_values.sum(axis=1)`, and divide it by the total number of tokens per document that is also part of the semantic network with `dtm_df_reduced.sum(axis=1)`.

For example, if a text contains eight tokens, but only two of the tokens are associated with a community yielding betweenness values of 0.1 and 0.02, then the topic loading would be calculated as (0.1 + 0.02)/8 = 0.015.

We then append the topic loadings as new columns in `dtm_df_reduced`, and format the column names with `variable_name = "Modularity_Class_" + str(c)`, before inserting the topic loadings per text with `dtm_df_reduced.insert(c, variable_name, modularity_values)`. In case of the first community, this would result in the variable name "`Modularity_Class_0`", or "`Modularity_Class_1`" for the second community. The last two functions, seen in Code box 15.38, select the columns where the topic loadings are stored. We use a Regex pattern which starts with "`Modularity`" for column selection and assign it to `select_variables`.

We now merge `community_weighting` with `df`, which contains YouTube comments and comment metadata. The goal is to filter the ten comments with the highest loadings per topic and to save the resulting data frame. We use the following code to achieve this (Code box 15.39):

```
df_with_modularities = pd.merge(community_weighting, df,
       how = "left", left_index = True,  right_index = True)
top_texts_per_modularity_class = []
for variable in select_variables:
       top_texts   =   df_with_modularities.sort_values(by   =
       variable,
       ascending = False).head(n = 10)
       top_texts_per_modularity_class.append(top_texts)
```

Code box 15.39: Establishing a data frame with the ten most typical texts per extracted community from the semantic network

```
top_texts_per_modularity_class =
pd.concat(top_texts_per_modularity_class)

os.chdir(output)
top_texts_per_modularity_class.to_csv
("top_texts_per_modularity_class.csv", sep = ";")
```

Code box 15.39: (Continued)

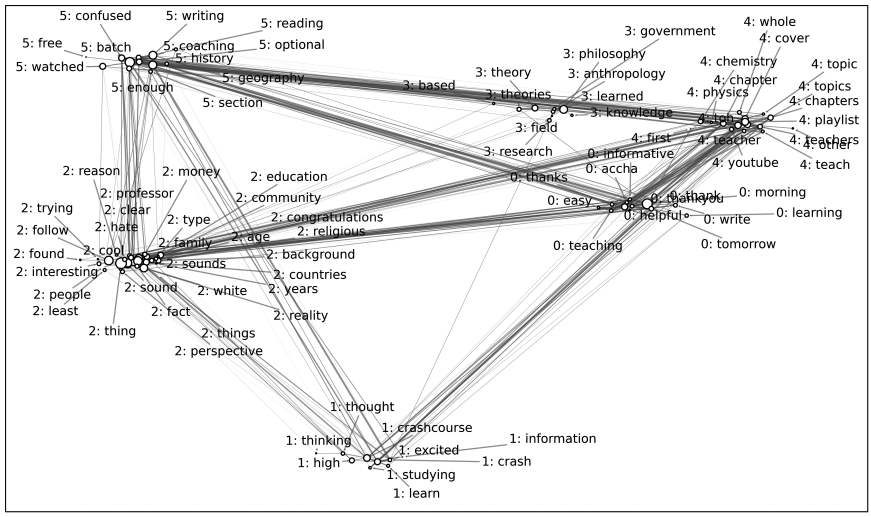
We merge both datasets with `pd.merge()`. Here, we merge `df` on `community_weighting` using a left join (`how = "left"`). We merge both data frames on the index using the options `left_index = True` and `right_index = True`. We then assign an empty list to top_texts_per_modularity_class, and iterate through the community classes stored in `select_variables`. We then sort the data frame in descending order using `df_with_modularities.sort_values(by = variable, ascending = False)` and extract the first ten rows by chaining the `head(n = 10)` method. We append these rows to `top_texts_per_modularity_class.append(top_texts)`, concatenate the resulting data frames with the Pandas' `concat()` function, and save the resulting data frame as CSV for further evaluation and interpretation.

15.2.6 Visualize the semantic network (with topic clusters)

The next step is visually interpreting the constructed semantic network. This can be a challenging task, as network graph layouts (e.g., the spring layout) often resemble a furball when too many nodes are included. Additionally, if we include the corresponding token names, it would render our graph completely unreadable. To address this issue, we generate a clustered layout combined with a *spring layout* (Fruchterman & Reingold, 1991), ensuring that the nodes within the extracted communities remain close while the communities are positioned further apart.

Figure 15.13 depicts the visualization. Here, we see the six communities (numbered 0 to 5) with node labels. That includes both the community number and the token. Different from MCA, the horizontal and vertical positions in this layout do not carry any specific meaning. Yet the algorithm places tokens that co-occur often in proximity. The edges are shaded gradually from light to dark gray. Darker colors indicating higher NPMI (edge weights), and larger nodes represent more central tokens. For example, in community 0, the most central nodes are *morning*, *thankyou*, *tomorrow*, *write*, and *thank*, whereas *easy*, *learning*, *teaching*, or *informative* are less central. From this alone, we might suspect that this community encapsulates *acknowledgements and courtesies* in regards

Figure 15.13 Depiction of the semantic network using a combination of cluster layout and spring layout.



to the video's content. However, the depiction of our semantic network provides further insights. As you may notice, co-occurring edges extend beyond individual communities, and by doing so, linking different topic clusters. This suggests that similar to other topic modeling approaches (e.g., Latent Dirichlet Allocation), topics are not separated but overlap. For example, topics 0, 4, and 5 are strongly linked and possibly form another, latent topic.

Let us now proceed with the technical part and construct this graph step by step. The rationale to do so is as follows:

1) Define the positions of the communities (in a circle) and nodes (spreading around the positions in the circle) in a two-dimensional space.
2) Create immutable (frozen) sets of nodes (tokens) per community (topic).
3) Adjust the node positions relative to the community positions.
4) Create the figure canvas and draw the nodes and edges.
5) Generate node labels.
6) Push node labels apart to avoid overlapping annotations.
7) Save and export the semantic network graph.

The following code snippet shown in Code box 15.40 captures the first two steps.

This initializes the positions of the communities (`nx.cycle_graph(best_partition_values)`), and positions them equidistant in a cycle (`nx.cycle_graph(supergraph)`) with an increase scale to fit all tokens onto the figure (`scale = 50`). The resulting positions are assigned to `supergraph.` A **supergraph** contains another graph (a subgraph of nodes) but focuses on larger structures (communities/topics). Consequently, the next step is to define the node

```
supergraph = nx.cycle_graph(best_partition_values)
supergraph_pos = nx.circular_layout(supergraph, scale = 50)
node_pos = nx.spring_layout(N,
                            scale = 50,
                            seed = 1,
                            k = 0.25,
                            weight = "weight",
                            iterations = 500)
```

Code box 15.40: Initialization of the basic layout for the communities, and the nodes in our network graph

positions with `nx.spring_layout()` function. This initializes the Fruchterman and Reingold (1991) algorithm, which treats the edges as springs between nodes, drawing strongly connected nodes (with high NPMI values) closed together, while pushing less-connected nodes (with low NPMI) further away. We pass the following options to the spring layout:

- `N` = The semantic network graph.
- `scale` = The size of the graph. Larger values ensure larger spaces between communities.
- `seed` = The starting positions of the nodes. Set to ensure reproducibility of the graph.
- `k` = Defines the node spacing. Smaller values keep nodes assigned to the same community in proximity.
- `weight` = Sequence or data frame column that yields the edge weights (NPMI). Higher values draw the nodes together.
- `iterations` = Number of times the algorithm reposition the nodes.

To create a link between the nodes and their communities, we create a frozen set and append this set to a list, which is used as input to update the node positions on the figure (Code box 15.41):

```
communities = []
for c in list(set(best_partition_values)):
    token_community = frozenset([key for key, value in best_
    partition.items() if value == c])
    communities.append(token_community)
```

Code box 15.41: Force the positions of the nodes on the position of their respective community

Here, we iterate through each community (stored in iterator c) and assign the token names stored in the `best_partition` dictionary to a frozen set. This is then passed to `token_community`, and appended to the communities list. This results in the following list depicted in Output 15.14:

```
[frozenset({'accha',
            'easy',
            'helpful',
            'informative',
            'morning',
            'perspective',
            'teacher',
            'teaching',
            'thank',
            'thanks',
            'thankyou',
            'tomorrow',
            'write'}),
. . .
frozenset({'crash',
            'crashcourse',
            'excited',
            'high',
            'information',
            'learn',
            'learning',
            'mentioned',
            'studying',
            'thinking',
            'thought'})]
```

Output 15.14: Excerpt from the list of frozen sets comprising the nodes per community in a sequence that can no longer be changed

Now, we relocate our nodes to the vicinity of the communities using the code snippet in Code box 15.42.

To do so, we assign an empty dictionary to a variable we call `pos` (short for "position") first. We need this dictionary, as NetworkX uses dictionaries to map the node names as keys, and their position as values. We then initiate a for loop

```
pos = {}
for position, community in zip(supergraph_pos.values(),
communities):
      com = nx.subgraph(N, community)
      pos.update(nx.spring_layout(com,
                                  center = position,
                                  scale = 15,
                                  k = 0.25,
                                  seed = 42))
```

Code box 15.42: Update the node positions in regards to the position of the respective community

and use `nx.subgraph(N, community)` to extract the communities, edges, and nodes from the semantic network `N`. The subsequent `nx.spring_layout()` arranges these nodes around their respective community centers, indicated by the `center=position` option. The `pos.update()` updates the node coordinates of the respective topic community.

We now initiate the figure canvas and draw the semantic network (see Code box 15.43).

As usual, we initialize a figure canvas, this time ten inches wide and six inches high, with plt.figure(figsize = [10, 6]). We then draw the node positions

```
plt.figure(figsize = [10, 6])
nx.draw_networkx_nodes(N,
            pos = pos,
            edgecolors = "black",
            node_color = "white",
            node_size = [x*1000 for x in \
            nx.betweenness_centrality(N).values()]
            )
nx.draw_networkx_edges(N,
            pos = pos,
            width=[edge[2]*10 for edge in N.edges(data="weight")],
            edge_color = "grey",
            alpha = 0.5)
```

Code box 15.43: Creating the figure canvas and adding nodes and edges

of the tokens calling `nx.draw_network_nodes()` and passing the following options to adjust their appearance to our needs:

- `N` = the semantic network containing the tokens and edges.
- `pos =` the updated node positions in relation to the position of the communities in the supergraph.
- `edgecolors` = defines the colors of the outer lines of the nodes (black).
- `node_color` = sets the color that fills the nodes (white).
- `node_size` = adjusts the size of the nodes. Nodes are either sized uniformly, if a floating point number or integer is passed, or differently if a list of values is passed. In our case, we calculate the node size by multiplying betweenness centrality with 1,000.

We then continue to draw the edges by calling `nx.draw_networkx_edges()`. The function yields the same options with exception of `width`, which sets the thickness of the edges, and `edge_color`, which adjusts the edge color.

We now generate the node labels which will contain the number of the community, and the token name (Code box 15.44):

```
node_coordinates = [(x, y) for x, y in pos.values()]
node_names = [f"{best_partition[token]}: {token}" for token in
pos.keys()]
```

Code box 15.44: Generating the node labels using the community number and token name

We start with assigning two lists, one storing the x- and y- position of each node as a tuple (`node_coordinates`), and one empty list for the modified token names (`node_names`). The second uses a formatted string operation within a list comprehension, to generate node names like "4: chemistry" with the community number and the token name. We now pass the node names and node coordinates to the `adjust_text()` function (Code box 15.45), which we already introduced in Code box 15.25. This ensures a clear labeling of tokens by topic community and aids us in interpreting the topic clusters within the semantic network.

```
text_to_adjust = []
for name, coords in zip(node_names,node_coordinates):
    text_to_adjust.append(plt.text(coords[0], coords[1], name))
```

Code box 15.45: Adjusting the position of the node labels

```
adjust_text(text_to_adjust, arrowprops=dict(arrowstyle = '->',
                                            color = "grey",
                                            alpha = 0.5))
```

Code box 15.45: (Continued)

Lastly, we fit the network graph to the figure canvas using `tight_layout()`, save it as a high-resolution figure with `plt.savefig(dpi = 600)`, and close the figure using the `plt.closefig()`.

```
plt.tight_layout()
plt.savefig("token_network.png", dpi = 600)
plt.close()
```

Code box 15.46: Saving the semantic network figure and closing the canvas

15.2.7 Interpretation of the extracted clusters as topics

We can now proceed to interpret the topics extracted from the semantic network. Beyond the information stored in the network graph depicted in Figure 15.13, we can now utilize the tokens with the highest betweenness centrality values associated with each community (see Table 15.4), and the texts with the highest association with the communities, as calculated in Code box 15.38 and selected in Code box 15.39.

The top tokens in community 0 are "morning", "thankyou" (sic!), "tomorrow", "write", and "thank". On a first glance, this hints at *acknowledgements and courtesies*. The tokens in community 1, "crashcourse", "excited", "high", "crash", and "thought", and others seen in the figure like "information" and "thinking", and "studying" appears to be linked to a *crash course in sociology*, maybe also the popular science channel of the same name. Community 2 includes the tokens "sounds", "follow", "cool", "white", and "congratulations", but also other tokens seen in the figure such as "countries", "years", "reality", "education", "money", or "religion". It is somewhat connected to Topics 0 and 4, and may be a topic that may encapsulate *central problems linked to sociology*. Topic 3 hints at *social theory*, as tokens such as philosophy, theory, theories, research, and anthropology indicate. It is not solely sociological theory, as the tokens are also linked to different fields of study. Topic 4 includes tokens such as "playlist", "toh", "cover," "chapters", and "Chapter", but also "play", "teach", "other", "topic", "topics", "whole", or "physics". This might hint at a topic where the *video content is critically compared to content from STEM disciplines*. The fifth and final topic appears to be some sort of *critique on the*

Table 15.4 Tokens With the Highest Betweenness Centrality Values Per Extracted Topic Community

Community	*Tokens with highest betweenness centrality*
0	morning, thankyou, tomorrow, write, thank
1	crashcourse, excited, high, crash, thought
2	sounds, follow, cool, white, congratulations
3	philosophy, theory, theories, research, anthropology
4	playlist, toh, cover, chapters, chapter
5	batch, geography, writing, confused, watched

video, but tokens such as "geography", or "batch", or less prominently, "free" and "history" might indicate. For this reason, we must take a closer look at some texts most clearly aligned to the respective communities for qualitative interpretation (see Chapter 7) to substantiate our findings.

Community 0:

The following three short comments (including typos and spelling errors) express gratitude towards the presenter and greet the instructor of the video. The three excerpts support our interpretation of Topic 0 as *acknowledgements and courtesies*.

Comment 1:

Thankyou sir good morning

Comment 2:

Good morning sir thankyou so Mach(sic!)

Comment 3:

A very nice over view. I have loved it. I have an exam on rural sociology tomorrow morning, that's exactly why am here listening.

Community 1:

Now let us turn to the three texts most clearly associated with Topic 1:

Comment 1:

I think that the Sapir-Whorf Hypothesis needs more elaboration, and perhaps a look at competing or related hypotheses. Perhaps CrashCourse Linguistics?

Comment 2:

CrashCourse: Fabulous! I would like the text of this vid! Think about offering it!

Comment 3:

Whoop! Notification squad! Definitely excited with all these new CrashCourse subjects

These comments suggest that community 1 is linked to videos from the CrashCourse channel. Commenters express a positive sentiment and enthusiasm towards the content, which is why we call Topic 1 *Positive attitude towards the content of the crash course channel.*

Community 2:

We initially labeled Topic 2 *central problems linked to sociology*. However, the following text documents suggest a more specific interpretation:

Comment 1:

it kinda sounds like she's describing casual white supremacy, but not acknowledging it or using language usually associated with the subject nowadays

Comment 2:

Of course marxism was mentioned. My only griped is the "creating reality" point. You're not creating or augmenting reality by assigning a meaning to a gesture. That's just how language works. You make a noise or a gesture and we assign meaning to it. You have a penis you're a boy. You have a vagina you're a girl. Sounds with meanings.

Comment 3:

"Crime can't be a disease, because every society has it." Sounds like Durkheim tried to force reality to fit the analogy.

The first comment deals with white supremacy, which is a right-wing political ideology in the United States. The commenter addresses this problem, but also criticizes the narrator. Comment 2 also criticizes the narrator, but also mentions Marxism and some content that appears to be a mixture of symbolic interactionism and gender studies. The third and final comment also criticizes the narrator of the video, and relates some content to the macro-sociological, functionalist view of Durkheim on social problems. Taken together, we may interpret Topic 2 as c*ritical comments of knowledgeable commenters on the narrator linked to social problems*.

Community 3:

The next extracted topic community was tentatively named *social theory*. However, the following comments suggest that this initial interpretation of community 3 was too coarse:

Comment 1:

Never understood why he is automatically marxist, since marx stole the majority of his philosophy without ctediting others such as proudhon.

Comment 2:

Good old Marxism refurbished. Old European ideas. I really do not like this destructive Critical Theory/Philosophy.

Comment 3:

i really do enjoy these videos, i wold love to see some videos about African philosophy and African political theory.

The commentators seem to refer much more to Marxism, critical theory, and postcolonial theorizing than to general social theory. The comments also point towards the linkage between critical theory and Marxism on the one hand, and philosophy, on the other hand. Despite this is also characteristic for social theory (and the tokens seen in the network graph), we must modify our interpretation and call Topic 3 from now on *Marxism, critical theory, and postcolonial theory*.

Community 4:

A *closer* examination of the content of community 4 reveals that these *texts are partially English, partially Indian*:

Comment 1:

Sir i have watch your previous playlist of sociology completely... . Is there any need to watch this new one ... Please reply ... I think I need to revise the previous one

Comment 2:

Didi baki k chapters kaha gye playlist m se

Comment 3:

Mae toh abhi start kr rhi hun pta nhi cover hoga yah nhi syllabus

In this sense, we must abandon community 4 as an own topic. However, the topic may hint at the fact that we have to clean the texts better in the future.

Community 5:

The comments with the highest topic loadings on the fifth community include partial critique on the videos (comment 3), but mostly contain advertisements for further instruction videos on sociology. Here, the "batch" token makes perfect sense, as these videos can be grouped into playlists and seen continuously. Against this backdrop, we must rename our fifth topic to *advertisements of instructional courses*.

Comment 1:

"UPSC IAS (Mains) Sociology Optional Live 2024 Batch (Comprehensive)
Batch Starting on 7th April 2023 | Class Timing -1:00 PM
To know more visit: http://bit.ly/3ZWbxmq"

Comment 2:

Sociology optional batch for 2021?

Comment 3:

Wow, this is such a dated concept that I am surprised they chose to cover it. I guess they want to be completionists lol

15.3 Concluding comments

Now, let us pause for a second and reflect what we have learned in this section and what the similarities and differences between the MCA and SNA are that were applied on the same data. We have demonstrated how to transform a text corpus into a document-term matrix and a network graph, how to preprocess a text corpus to retain relevant text features, and how to conduct MCA and SNA. We also introduced their theoretical underpinnings (Habitus-Field theory and socio-semantic network theory).

Let us now highlight and reflect on the similarities and differences between the MCA and SNA applied to the same dataset. While both extract meaningful topics from small text corpora, they extracted topics quite differently, namely:

1) MCA identifies a latent structure in the text corpus, whereas in SNA focuses on local structures to extract topics.
2) While the MCA tends to view long comments as typical, SNA returns short texts as typical for each extracted topic cluster. In other words, MCA returns

Table 15.5 Comparison Between the Topics Extracted Using Multiple Correspondence Analysis (MCA) and Semantic Network Analysis (SNA)

Topic #	*SNA: Interpretation*	*MCA: Dimension/topic*	*MCA: Interpretation*
0	*Summarization of learned content*	1–0	*Acknowledgements and courtesies*
1	*Praise and gratitude for teaching*	1–1	*Positive attitude towards the content of the crash course channel*
2	*Summarization of learned content*	2–2	*Critical comments of knowledgeable commenters on the narrator linked to social problems*
3	*Harsh critique of sociology*	2–3	*Marxism, critical theory, and postcolonial theory*
4	*Hierarchy of sciences and relation of sociology and other disciplines*	3–4	*Partially Indian communities*
5	*Critique of institutions and (state) organizations*	3–5	*Advertisements of instructional courses*

topics that are more suitable for further qualitative inspection, whereas SNA returns topics that are ready to be interpreted immediately.

3) MCA allows us to link topics with the document metadata, which facilitates interpretation beyond the tokenized text content.

The extracted topics are compared in Table 15.5. Please note that there is no topic overlap, except for the somewhat similar topics: 1 in case of the SNA and 0 in regards to MCA.

Overall, we recommend using either MCA or SNA for analyzing small text corpora, as we remove too much information from large text corpora. On the one hand, we opt for SNA if texts are of small size and should be readily analyzed. On the other hand, we recommend MCA if you have longer text documents at hand and seek to select suitable texts for further qualitative inquiry. On a final note: both methods should be closely integrated with more qualitative methods in a mixed-methods research design – either to substantiate topics or to evaluate whether the interpretations performed on a small portion of the text corpus hold for different texts in the text corpus.

Notes

1 More precisely, singular value decomposition (SVD) is underlying the GDA family of methods. SVD first maps the more complex data structures onto a space with a small number of dimensions, then "stretching" and "rotating" the respective axes in such a way that the axes contain a large amount of the original information. As each dimension of the resulting space incorporates a part of the initial structure, these axes are suitable for interpretation.
2 Ascii is an acronym for American Standard Code for Information Exchange and does not contain Hindi or Chinese characters. Consequently, checking for these characters allows us to remove any tokens of languages that may have slipped to our previous filtering procedures.
3 This is used to filter extremely short tokens, special characters, and emojis that may also count as tokens. We might preserve these for sentiment analysis, however, they might not be of interest in regards to our relatively simple, relational methods employed in this chapter.
4 In one of the formulas used to calculate the position of the texts, as well as the tokens in the space spanned by the MCA, the algorithm tries to divide by the sum of all tokens present in the observation (in our case: comments). Dividing by zero would result in infinities – and we cannot reckon with infinities.
5 We could also divide the number of token appearances by the total number of tokens per comment. This would allow us to conduct either a CA, or a MFA.
6 Bimodal distributions have two local peaks in the distribution of values, whereas multimodal distributions have three or more peaks in the distribution of values.
7 According to Newman (2006, p. 8578), "modularity is, up to a multiplicative constant, the number of edges falling within groups minus the expected number in an equivalent network with edges placed at random".
8 Remember to set the upper threshold to 1.55 instead of 1.5, as Python starts indexing and counting with 0 instead of 1. If we want to include 1.5 in the range, we need to set a threshold higher than 1.5. Each value of this is stored separately in the iterator named resolution and passed to the loop.
9 However, we do not expect to see high values. In fact, a value of 1 could only be achieved in this weighting scheme, if all tokens present in a community (= potential topic) yield a betweenness centrality of 1. This means that these tokens would cover all the shortest paths between all other nodes in the network, making it extremely unlikely to identify meaningful topics. Usually, the distribution of betweenness centrality values among is highly skewed and betweenness centrality values themselves take small values. This is why we do not expect to see high values of the topic loadings. In fact, the highest betweenness centrality values present in across the extracted communities ranges between 0.0312 and 0.0949.

16
SENTIMENT ANALYSIS

16.1 Introduction

Sentiment analysis – also called *opinion mining* – aims to extract sentiments from text.[1] Sentiments are views and opinions that are held and expressed. They are usually targeted towards an object or actor (individual or organizational), and tend to be more stable over time compared to emotions (Munezero et al., 2014). Furthermore, sentiments are a socially constructed and learned combination of autonomic responses, expressive behaviors, and cultural meanings (Gordon, 2017).

Sentiment analysis relies on either lexical approaches or machine learning approaches to extract and classify sentiments in corpora of texts.[2] It usually comprises of three main tasks, namely:

1. **Polarity detection**, which classifies sentiments into discrete groups (e.g., positive, neutral, or negative).
2. **Subjectivity detection**, which distinguishes subjective sentences from objective ones.
3. **Sentiment strength detection**, which either categorizes text documents into fine-grained sentiment classes (e.g., strong positive, slightly positive), or along a continuous scale (e.g., 0 to 100, where 0 represents a strongly negative sentiment, and 100 a strongly positive sentiment).

Sentiment analysis first appeared in public opinion research in the 1940s and remained relatively dormant until the 2000s (Mäntylä et al., 2018). It only gained significant scholarly attention after the widespread use of the Internet. Initially, sentiment analysis was applied in consumer research to analyze product reviews and to map sentiments in social media discourse

DOI: 10.4324/9781003496397-19
This chapter has been made available under a CC-BY-NC-ND 4.0 license.

on social platforms such as Twitter (now X). Today, sentiment analysis is applied in diverse fields, including healthcare, stock markets, entertainment, education, e-commerce, politics, social media, product reviews, and consumer feedback (Bashiri & Naderi, 2024; Jim et al., 2024). Take politics for example. In this case, sentiment analysis is used to predict election outcomes, to track voter sentiments on political issues, to capture politicians' perception of political opponents, or to monitor public reactions to political controversies or crises.

In this chapter, we will introduce one **lexical approach** for polarity detection and one **deep learning approach** for sentiment classification and prediction. The lexical approach utilizes predefined dictionaries and rules to assign sentiment scores to texts. The deep learning approach leverages a **pretrained deep learning model**, RoBERTa, to classify sentiments in text corpora. Pretraining refers to the training of a deep artificial neural network on large text corpora (e.g., Wikipedia, IMDB, and the such) to extract patterns, before they are fine-tuned to detect sentiment classes or topics in the text documents. We will also demonstrate how we can fine-tune a BERT model on our YouTube comments dataset. Fine-tuning enables us to (a) detect custom sentiment classes and (b) to adjust the pretrained model to capture the specific vocabulary of our text corpus.

In the following sections, we use a subset of the text corpus introduced in Section 14.2, which comprises of approximately 1,100,000 comments from YouTube related to sociology, economics, and psychology. We draw a stratified sample of 30,000 comments for fine-tuning (Section 16.4), equally distributed across positive, neutral, and negative sentiment classes. Due to memory constraints in regards to the computer used by the author of this chapter, we use a smaller BERT model (DistilBERT), as larger models exceed the accessible RAM available and cause the author's programming environment to crash. We then demonstrate how you evaluate the model (Section 16.5) and conclude with some suggestions on how you can utilize the extracted sentiment classes for further qualitative, quantitative, or automated content analysis tasks (Section 16.6).

16.2 Lexical approach to sentiment analysis

A lexical approach to sentiment analysis relies on a sentiment lexicon and predefined set of rules to assign a sentiment class or polarity value to text documents (Wilson et al., 2005). A sentiment lexicon is a dictionary that contains token – value pairs, where the values encode either the sentiment or the polarity of a token. These values are annotated either by a single or multiple annotators (e.g., a researcher). In the latter case, a token's sentiment value is determined by either the mean or median of assigned values, or by selecting the category chosen by the majority of the annotators.[3]

16.2.1 Introducing VADER

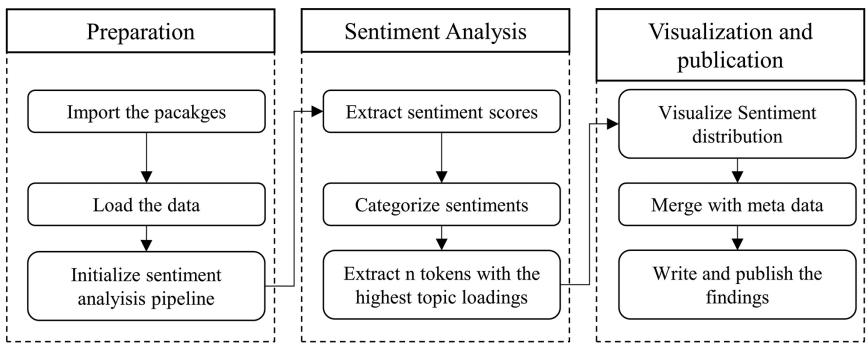
Let us illustrate how polarity values are assigned first to tokens, and then to text documents. In a lexical approach, token polarity scores typically range from –1 (extremely negative) to +1 (extremely positive). For example, 'awesome' might have a polarity value of +1, while 'shocking' might yield a value of –0.8. Additional rules then adjust the token polarity based on context, such as negations (e.g., 'not good' flips polarity sign) or amplifications (e.g., 'very good' increases the polarity value of 'good'). The maximum distance allowed between a negator or amplifier to allow the modification of the polarity value is called **context window**.

In the following, we use the `vaderSentiment` package to assign polarity values to text documents (Hutto & Gilbert, 2014). VADER (**V**alence **A**ware **D**ictionary and s**E**ntiment **R**easoning) was designed for sentiment analysis and trained on microblogs (such as Twitter/now X), movie reviews, product reviews, and opinion news articles. It consists of a lexicon of 7,500 tokens (including slang and emojis), rated on a Likert scale from –4 (extremely negative) to +4 (extremely positive). Furthermore, five additional rules are applied to modify the polarity score (Hutto & Gilbert, 2014, p. 221):

1) An **exclamation mark** increases the sentiment intensity (e.g., 'This book is great.' versus 'This book is great!').
2) **Capitalization** amplifies sentiment magnitude (e.g., 'This book is GREAT!').
3) **Adverbs** (e.g., 'extremely' or 'marginally') also modify the polarity score (e.g., 'This book is extremely great!').
4) **Contrastive conjunctions** (e.g., 'but') shift sentiment polarity (e.g., 'This book is great, but I don't like the content.').
5) **Trigrams** (a sequence of three tokens) **preceding a sentiment-laden token** can negate sentiment polarity (e.g., 'This book *is sometimes a* great read.').

In the next steps, we employ the workflow depicted in Figure 16.1 to conduct sentiment analysis using VADER:

Figure 16.1 Workflow – lexical approach to sentiment analysis – VADER.



```
import pandas as pd
import os
import re
from vaderSentiment import vaderSentiment
from matplotlib import pyplot as plt
import seaborn as sbs

root = "C:\\[YOUR_USERNAME]\\[YOUR_FOLDER]\\"
path = root + "Data\\"

os.chdir(path)
df = pd.read_csv("soc_sci_prepared_for_sentiment_analysis.csv",
sep = ";")
```

Code box 16.1: Import of packages and data necessary to conduct a sentiment analysis with vaderSentiment

To demonstrate how VADER works, we start with importing `pandas` (with alias `pd`), `os`, and `re` for data preparation, the `vaderSentiment` module from `vaderSentiment`[4] to assign sentiment values across text documents, and the `matplotlib.pyplot` and `seaborn` packages – with aliases `plt` and `sbs` – to plot the resulting figures. We then import a comma separated file for by calling `read_csv(soc_sci_prepared_for_sentiment_analysis.csv)` and assign it to `df` (Code box 16.1). The data frame includes the columns listed in Table 16.1:

Table 16.1. Overview on the Columns/Variables of the soc_sci_prepared_for_sentiment_analysis Data Frame Used for Sentiment Analysis With VADER

Column name	*Description*
video_id	YouTube video IDentifier
comment_id	YouTube comment IDentifier
totalReplyCount	Number of relies to the comment
authorDisplayName	Author nickname (removed afterwards)
textOriginal	Content of the Comment
likeCount	Number of likes to the comment
publishedAt	Publication time (format: Year-Month-Day:Hour:Minute:Second)
language	Language assigned by langdect

16.2.2 Calculating sentiment polarity scores

Next, we initialize the `SentimentIntensityAnalyzer()` algorithm and apply it to a sentence ('This is a great book!') by using the `polarity_scores()` method and see what it measures (Code box 16.2):

```
analyzer = vaderSentiment.SentimentIntensityAnalyzer()
analyzer.polarity_scores("This is a great book!")
```

Code box 16.2: Initialization and application of the SentimentAnalyzer algorithm

```
{'neg': 0.0, 'neu': 0.477, 'pos': 0.523, 'compound': 0.6588}
```

Output 16.1: Exemplary VADER output

The output dictionary yields four key-value pairs (Output 16.1). But what do the keys mean?

- `'neg'`, `'neu'`, `'pos'` indicate the proportion of tokens with negative, neutral, and positive sentiments in the provided sentence.
- The `'compound'` score calculates the overall polarity score by summing the negative, neutral, and positive scores, and adjusting this score by the five context-specific rules. The polarity score is then normalized by text length and takes values between –1 (extremely negative sentiment) and +1 (extremely positive sentiment).

Consequently, the scores tell us that our sentence yields no negative tokens (`'neg': 0.0`), a good share of neutral tokens (`'neu': 0.477`), and many positive tokens (`'pos': 0.523`) and, overall, carries a strongly positive sentiment (`'compound': 0.6588`).

Next, we apply VADER to our YouTube comments dataset and calculate the sentiment polarity of the text documents stored in `df.textOriginal` (Code box 16.3):

```
analyzer = vaderSentiment.SentimentIntensityAnalyzer()
sentiments = [analyzer.polarity_scores(text) for text in
df.textOriginal]
compound = [sentiment["compound"] for sentiment in sentiments]
df.insert(1,"sentiment_score",compound)
```

Code box 16.3: Calculating sentiment polarity scores for the YouTube comments in the fields of sociology, psychology, and economics

Running these lines of code returns a list of dictionaries with `'neg'`, `'neu'`, `'pos'`, and `'compound'` values. In a next step, a second list comprehension extracts the `'compound'` value from each dictionary. Lastly, we call Pandas' `insert()` method and add the compound values to our Pandas data frame with column name `sentiment_score`.

16.2.3 Classifying sentiment polarity scores into sentiment categories

Since the BERT models we will use in Sections 16.3 and 16.4 categorize text into three sentiment categories – positive, neutral, and negative – we need to ensure that the compound scores generated by VADER are mapped **onto** the same categories. This allows us to directly compare the VADER sentiment assignments to those assigned by the RoBERTa model.

To achieve this, we follow the categorization scheme proposed by Hutto and Gilbert (2014)[5] and set the following thresholds to map the compound values to the three sentiment classes:

1. **Negative** if the compound value is less than or equal –0.05,
2. **Positive** if the compound value is greater than or equal 0.05, and
3. **Neutral** for values between –0.05 and 0.05.

As demonstrated in Code box 16.4, we use a `for` loop with nested `if`, `elif`, and `else` conditions to perform this categorization. The mapping procedure is as follows:

1) We create an empty list (`sentiment_scores_categorized`) to store the sentiment labels.
2) The subsequent `for` loop iterates over `df.sentiment_score`. Within the score, the condition-operators check
   a. `if` the compound value of a text document is less than or equal to –0.05. If so, it appends `"negative"` to `sentiment_scores_categorized`.
   b. If not, the `elif` condition checks if the compound value is greater than or equal 0.05 and appends `"positive"` if it is.
   c. Otherwise, "`neutral`" is appended (for values between –0.05 and 0.05).

Finally, we call the Pandas `insert()` method to add at a new column named `"sentiment_scores_categorized"` – that contains the sentiment classes – to `df` as the second column.

```
sentiment_scores_categorized = []
for sentiment_score in df.sentiment_score:
    if sentiment_score <= -0.05:
        sentiment_scores_categorized.append("negative")
    elif sentiment_score >= 0.05:
       sentiment_scores_categorized.append("positive")
    else:
       sentiment_scores_categorized.append("neutral")
df.insert(2, "sentiment_scores_categorized", sentiment_scores_
categorized)
```

Code box 16.4: Mapping compound polarity scores to discrete sentiment categories

16.2.4 Exploring and visualizing the sentiment categories

Let us now examine the distribution of sentiment classes in our text corpus. In Code box 16.5, we call pandas' `value_counts()` method with `normalize=True` on `df.sentiment_scores_categorized` to compute the proportion of comments with positive, negative, and neutral sentiments. As depicted in Output 16.2, this method returns the proportion of sentiment classes as a Pandas series object. To facilitate their visualization in Seaborn, we use the `reset_index()` method to pull the index values (positive, negative, and neutral) from the index and assign them to `sentiment_scores_categorized` (same as in `df`), and transform the series object into a Pandas data frame.

```
sentiments_categorized = df.sentiment_scores_categorized.\
    value_counts(normalize = True)
sentiments_categorized_reset_index = sentiments_categorized.reset_
index()
```

Code box 16.5: Calculating the proportion of categorized sentiment polarity.

```
sentiment_scores_categorized
positive    0.506907
negative    0.272372
neutral     0.220722
Name: proportion, dtype: float64
```

Output 16.2: Proportion of the sentiment categories

```
plt.figure(figsize = [8, 6])
sbs.barplot(data = sentiments_categorized_reset_index,
            x = "sentiment_scores_categorized",
            y = "proportion",
            hue = "sentiment_scores_categorized")
plt.xlabel("VADER sentiments (categorized)",
           fontdict={"size" : 12, "weight" : "bold"})
plt.ylabel("% assigned", fontdict ={"size" : 12, "weight" : "bold"})
plt.tight_layout()
plt.savefig("barplot_sentiments_categorized_VADER.png",
dpi = 600)
plt.close()
```

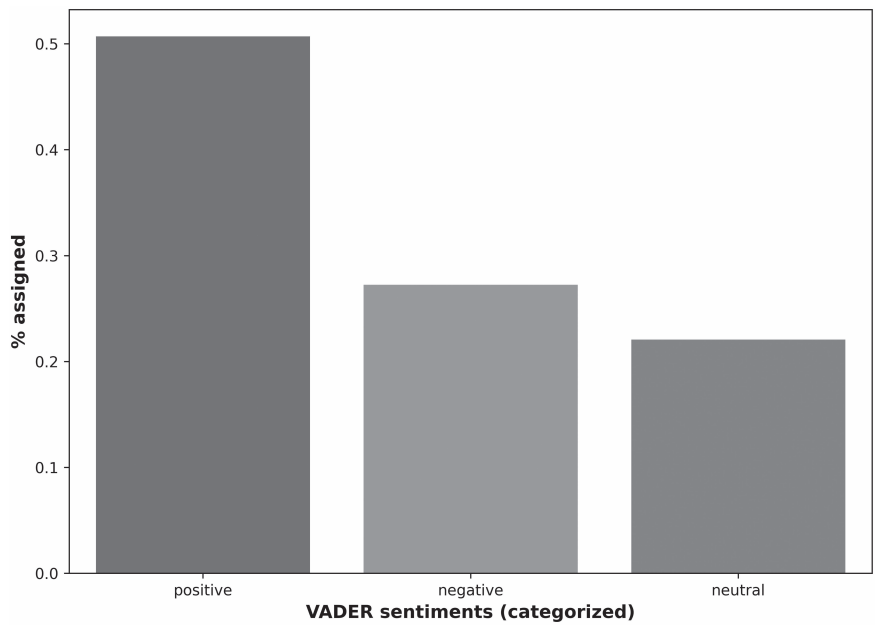
Code box 16.6: Visualizing the proportion of sentiment categories assigned by VADER

Next, we use the lines of code seen in Code box 16.6 to create a bar plot to illustrate the distribution of sentiment classes shown in Output 16.2.

Here, we begin with initializing the figure canvas and set the width to eight inches and the height to six inches with `plt.figure(figsize = [8,6])`. Next, we visualize the sentiment distribution as bar plot using `sbs.barplot()`. We pass the data frame containing sentiment categories and their proportions to Seaborn's `barplot()` function with `data = sentiments_categorized_reset_index`. The x-axis is mapped to sentiment categories (`x = "sentiment_scores_categorized"`), and the y-axis to relative frequencies (`y = "proportion"`). The `hue = "sentiment_scores_categorized"` option colorizes each bar according to the sentiment category assigned by VADER.

The next lines of code change the labels and appearance of the x- and y-axis labels and enhances the readability of the graph. Here, we change the axes labels by calling matplotlib's `xlabel()` and `ylabel()`. The `fontdict` option customizes the text. In our case, we change the size of the text to point size 12 and the weight to bold with `fontdict = {"size" : 12, "weight" : "bold"}.`[6] Before we save and close the figure with `savefig()` and `close()`, we use the `tight_layout()` function to fit the bar plot to the figure canvas. The final plot in Figure 16.2 reveals that – according to VADER – positive sentiments dominate our text, whereas negative and neutral sentiments appear comparatively seldom.

Figure 16.2 Categorized sentiment score assigned by VADER.



We conclude with saving the data frame for future use, for example with BERT-based sentiment classifications (Sections 16.3 and 16.4) with `to_csv()` (see Code box 16.7):

```
os.chdir(path)
df.to_csv("VADER_soc_sci_Sentiment_values.csv", sep = ";")
```

Code box 16.7: Saving the DataFrame containing metadata, compound values, and categorized sentiment scores

16.3 Applying sentiment analysis using a pretrained Transformer model

We now move beyond dictionary-based sentiment analysis to state-of-the-art methods. Nowadays, the application of deep neural networks and large language models (LLMs) has become the gold-standard for sentiment analysis (Alslaity & Orji, 2024; Mäntylä et al., 2018). Many of these models, pretrained and fine-tuned on large text corpora, are freely available for direct use or fine-tuning. One of these large language models we will employ from now on for sentiment analysis is BERT (**B**idirectional **E**ncoder **R**epresentations from **T**ransformers).

16.3.1 The building blocks of BERT and its learning procedure

But what exactly is BERT, and what does its name signify?[7] **Bidirectional** means that our artificial neural network contains layers that read the text sequences forward (from left to right) as well as backwards (from right to left).[8] The network retains a fixed, predefined number of tokens in its memory (e.g., 256, 512, or 1024), which are then read forward and backward to map the meaning of each word in specific contexts. These predefined numbers of tokens are also called '**context windows**' and, as the term suggests, serve to contextualize the words present in the respective text sequence.

An **Encoder** processes a high-dimensional input (e.g., a text corpus containing many thousands of unique words) and projects the input onto a low-dimensional space. This low-dimensional space preserves information of a text in a condensed manner, such as tokens, their dependencies, semantic as well as syntactic information, and contexts.

A **Representation** is a vector of encoded values that contains (a) information on the tokens themselves and (b) on the sequence (e.g., sentence, paragraph) in which the tokens appear.[9] As such, it is an abstraction that allows the BERT model to distinguish between different meanings of a token given context. For

example, the meaning of "theory" differs if it co-occurs with "sociology" or "conspiracy".

A **Transformer** is a deep learning architecture that uses a **self-attention mechanism** which allows the model to (a) capture the relative importance of words within a text and (b) to contextualize the relationship among words over long distances within the texts. According to Vaswani (2017), who introduced Transformer models, self-attention is a "mechanism relating different positions of a single sequence in order to compute a representation of the sequence" (Vaswani, 2017, p. 2), and thus, enables the model to learn dependencies between tokens even if these tokens are distant within the text sequence.

After clarifying the meaning of the abbreviation BERT, we need to explain another important specificity of this architecture, namely the multi-head self-attention layers. These are used as hidden layers where the pattern recognition is performed. These **multi-head self-attention layers** enable the Transformer model to simultaneously encode multiple properties of a document at once, for example the tokens themselves, their context, syntactic relationships, or their semantic similarity. Each of these features is projected onto a mathematical space. As a consequence, all of them combined represent the text. These are then concatenated into a single context space and multiplied with a weight matrix to produce a feature vector (= continuous numerical representation of the text). Such abundant information can then be used to identify sentiments in a text if linked to sentiment values or polarity classes.

Finally, the **output layer** returns (a) the most probable sentiment class and (b) a confidence value which quantifies that this class is assigned "correctly" in accordance with the pretrained model. The output layer uses a softmax activation function, which enables the model to output values of each sentiment class is between 0 and 1.[10]

But how does a BERT model make sense of the text content and how is it trained? In regards to the original BERT model proposed by Devlin et al. (2018), two tasks were provided:

1) **Masked Language Modeling** (MLM), which "randomly masks some of the tokens from the input, and the objective is to predict the original vocabulary id of the masked word based only on its context" (Devlin et al., 2019, p. 4171). Take, for example, the following sentence: "A cat is a small, furry animal with paws that likes to scratch surfaces". In the MLM task, the word cat is masked, leaving us with "A [masked] is a small, furry animal with paws that likes to scratch surfaces". The task for the model is then to assign "cat" to the [masked] part of the sentence.
2) **Sentence prediction tasks**, for which the BERT model tries to predict the most likely sentence that follows a given sentence. For example, if we provide our model with the sentence "A cat is a small, furry animal with paws

that likes to scratch surfaces", and we talked a lot about how they use their claws to catch mice and other small animals, our BERT model might predict the following sentence: "It often sharpens its claws on the furniture to keep them in good condition".

The Transformer architecture introduced by Devlin et al. (2018) came in two different versions, a base version and a large version. The base version contains 12 stacked encoders with 12 self-attention heads. The final, fully connected (dense), hidden layer contains 768 artificial neurons. The large version contains 24 stacked encoders with 16 attention heads and a fully connected, hidden layer yielding 1024 artificial neurons. Although this might sound like a small difference at first, the large version yields more than three times the number of parameters of the BERT base model, making the latter more precise, but computationally more expensive.[11]

To work properly, BERT requires all texts to be of same length, as it tries to transform the documents to an attention matrix of a fixed size, for example 512 columns by 512 rows (Soydaner, 2022). As a consequence, texts with less than the given size are padded (= padding token "[PAD]" is added until the text reaches a specified length), whereas long texts are truncated (= elements that exceed the specified text length are deleted).

16.3.2 Initialize a RoBERTa sentiment classifier

For the remainder of this subsection, we will use a pretrained, freely available RoBERTa model for sentiment classification (Liu et al., 2019).[12] RoBERTa is an abbreviation of **R**obustly **o**ptimized **BERT a**pproach and was trained using the same text corpora that Devlin et al. (2018) used to train the first BERT version. However, there are also notable differences between RoBERTa and standard BERT, which root in their training procedures. These differences are that the RoBERTa model (1) was trained longer (= more epochs); (2) used large batches of up to 8,000 sentences at once; (3) the next sentence prediction objective was removed; and (4) it was trained over longer sequences of text. Furthermore, the authors introduced a (5) dynamically changing masking pattern to improve its ability to contextualize each token.[13] Dynamic masking simply means that different words are masked with a [mask] token during the training procedure. In the case of the sentence *A cat is a small, furry animal with paws that likes to scratch surfaces*, the following tokens could be masked during model training:

1) *A [mask] is a small, furry animal with paws that likes to scratch surfaces.*
2) *A cat is a small, [mask] animal with paws that likes to scratch surfaces.*
3) *A cat is a small, furry [mask] with paws that likes to scratch surfaces.*
4) *A cat is a small, furry animal with [mask] that likes to scratch surfaces*
5) *A cat is a small, furry animal with paws that likes to [mask] surfaces*

```
import os
import numpy as np
import pandas as pd
from tqdm import tqdm
from transformers import pipeline
import sklearn.metrics as metrics
classifier = pipeline(task = "text-classification",
            model = "terrencewee12/xlm-roberta-base-sentiment-
            multilingual-finetuned-v3")
```

Code box 16.8: Importing packages and initializing the RoBERTa model for sentiment classification

Let us use the pretrained RoBERTa model to assign sentiment classes to text documents.[14] As usual, we begin with importing the packages necessary to conduct our analysis. As demonstrated in Code box 16.8, we import the `os`, `numpy` (alias: `np`), `pandas` (with alias `pd`), the `tqdm` function from the `tqdm` package, as well as the pipeline function from the `transformers` package. Lastly, we import `sklearn.metrics` (alias `metrics`) to calculate the agreement between RoBERTa and VADER in regards to the assigned sentiment categories. We then initialize the Transformer with the `pipeline()` function and assign it to a variable we call `classifier`. For now, we pass the bare minimum information to the `pipeline()` function to make it work, namely:

1) The definition of the task to accomplish. In our case, we want to use our model for text classification, hence the `task = "text-classification"`. Other tasks include text feature extraction, question answering, calculating sentence similarity or translation.
2) The model to conduct sentiment analysis, in our case: `model = "terrencewee12/xlm-roberta-base-sentiment-multilingual-finetuned-v3"`. The path directs Python to the website on the Hugging Face directory, where the model is stored. If you initialize the model for the first time, this option ensures that your computer will automatically download the model and initialize it. If you already downloaded the model, then this option simply initializes the BERT model, in our case the pretrained RoBERTa model.

Let us test the RoBERTa sentiment classifier on texts of different lengths before we apply it on the whole text corpus[15]:

```
test_texts = ["This is an easy and understandable sentence",
        """This whole book is mostly a deluded version of
other textbooks.
        It is overtly complicated and the knowledge provided
is unnecessary.
        The concepts are messy and no one will ever read this
book, trust me.
        """,
        """
        Precision is not Accuracy, and it's idiosyncratic to
each Individual.
        But every concept is a degree of conformity to some
proportion of "Truth in labelling" Nomenclature.
[APPROXIMATELY 700 WORDS OMMITTED]
        WYSIWYG, and if you don't "get" the concepts of
Absolute Zero Kelvin sync-duration here-now-forever Singu-
larity, it's only a matter of Time thinking for your actual,
nearest-to-completeness self, and realising embedded integra-
tion of AM-FM Actuality. (Over to you, fellow students)
        """
        ]

for text in test_texts:
 print(classifier(text))
```

Code box 16.9: Applying the RoBERTa sentiment classifier with minimal passed options applied on three documents of different length

The following output is displayed in Output 16.3 when the code displayed in Code box 16.9 is executed:

```
[{'label': 'positive', 'score': 0.9678682684898376}]
[{'label': 'negative', 'score': 0.9421517252922058}]
Traceback (most recent call last):
...
RuntimeError: The expanded size of the tensor (851) must match
the existing size (514) at non-singleton dimension 1. Target
sizes: [1, 851]. Tensor sizes: [1, 514]
```

Output 16.3: Output of the RoBERTa sentiment classifier with minimal passed options applied on three documents of different length

The first two texts were labeled as *positive* and *negative*, with confidence scores of 0.968 and 0.942, respectively. As the confidence score ranges from 0 to 1, these values indicate that our model has little doubt that "This is an easy and understandable sentence" yields a positive sentiment, whereas the sentiment of "This whole book is mostly a deluded version of other textbooks. It is overtly complicated and the knowledge provided is unnecessary. Concepts are messy and no one will ever read this book, trust me" is (almost) certainly negative. The last input, however, produced a runtime error. What happened here? What does this error mean and why does it occur?

This error occurs because the third text is too long to fit into the attention head. More precisely, the RoBERTa algorithm has a context window of 512 tokens plus two tokens that indicate the beginning (`[bos]` = beginning of sequence) and the end (`[eos]` = end of sequence) of a text document. The error message indicates that the size of the text document, which is converted to a tensor,[16] exceeds the maximum number of tokens. Consequently, we must either

a) Slice the text document into multiple, smaller chunks with maximum 512 tokens, or
b) Truncate the text to the first 512 tokens and assign a sentiment class for the beginning of our text.

Although the first strategy yields more information, it is easier to follow the second strategy. To apply this strategy, we have to pass three additional options to our `pipeline()` function and reassign it to the classifier variable. The three options are as follows:

1) `max_length = 512`: The `max_length` option defines the maximum number of tokens that are passed to our BERT algorithm. In our case, we set the maximum number of tokens to 512, which equates to the context window used by the RoBERTa algorithm.
2) `Padding = True`: This option forces the BERT model to fill text documents with less tokens than defined by `max_length` with special `[pad]` tokens.
3) `Truncation = True`: This option is used to mask all tokens with index values greater than defined by `max_length`.

16.3.3 Using a pipeline to predict sentiment classes

We will now initialize our classifier variable again, pass the additional information to the `pipeline()` function, and apply the classifier on our list of texts (`test_texts`) afterwards to get the list of dictionaries depicted in Output 16.4:

```
classifier = pipeline(task = "text-classification",
             model = "terrencewee12/xlm-roberta-base-sentiment-
multilingual-finetuned-v3",
             truncation = True,
             max_length = 512,
             padding = True)
classifier(test_texts)
```

Code box 16.10: Initializing a new RoBERTa pipeline for sentiment classification including options for text padding and truncation.

```
[{'label': 'positive', 'score': 0.9678682684898376},
 {'label': 'negative', 'score': 0.9421517252922058},
 {'label': 'negative', 'score': 0.4828781187534332}]
```

Output 16.4: Output of the RoBERTa classifier for sentiment analysis for the three example texts

As you may have already noticed, adding these three options enables us to pass the long text (with more than 512 tokens) to the classifier without causing errors. But keep in mind that only the first 512 tokens will be used for sentiment classification tasks. We will now use this behavior and classify the sentiment of all documents in the text corpus. To do so, we opt for a list comprehension to classify each text separately (Code box 16.11). Although it is not strictly necessary to do so, this enables us to enclose the documents stored in `df.textOriginal` with the `tqdm()` function. This function activates a progress bar and calculates the time until a calculation is finished.[17] We included this progress bar to ensure that neither Python nor our computer crashed during this procedure.

As shown in Output 16.5, the list comprehension applied produces a list of lists of dictionaries. In order to compare the sentiment classes assigned to each

```
sentiment_scores = [classifier(text) for text in tqdm(df.text
Original)]
```

Code box 16.11: Assigning sentiment classes and sentiment scores to all documents in our text corpus

```
print(sentiment_scores)
print(sentiment_scores[0])
print(sentiment_scores[0][0])
[[{'label': 'neutral', 'score': 0.484528511762619}],
 [{'label': 'positive', 'score': 0.9678808450698853}],
 [{'label': 'neutral', 'score': 0.6840695142745972}],
 [{'label': 'neutral', 'score': 0.7190048694610596}],
 [{'label': 'neutral', 'score': 0.7931422591209412}],
...]
[{'label': 'neutral', 'score': 0.484528511762619}]
{'label': 'neutral', 'score': 0.484528511762619}
```

Output 16.5: Output of the sentiment classification process

text document, we must iterate through this list of lists of dictionaries and pull the assigned labels and scores from this data structure. As also demonstrated in Output 16.5, if we navigate to the first element of the list of lists of dictionaries, Python outputs a list containing a single dictionary. We use `sentiment_scores[0][0]` to further navigate to the dictionary stored as first element of the first list in `sentiment_scores`. We can make use of this data structure and extract the sentiment classes assigned to each text document using `[x[0]["label"] for x in sentiment_scores]`, and we can extract the assigned sentiment scores using `[x[0]["score"] for x in sentiment_scores]`. As demonstrated in Code box 16.12, we add the sentiment classes and confidence scores using the pandas `insert()` method:

```
df.insert(1, "RoBERTa_label", [x[0]["label"] for x in sentiment_
scores])
df.insert(1, "RoBERTa_score", [x[0]["score"] for x in sentiment_
scores])
df.to_csv("soc_sci_RoBERTa_sentiment.csv")
```

Code box 16.12: Extracting and inserting the sentiment categories and sentiment scores assigned by the RoBERTa algorithm

16.3.4 Comparing sentiment classes predicted by VADER and RoBERTa

Next, we calculate the proportion of sentiment classes assigned to the text documents and compare the sentiment classes assigned by RoBERTa

with VADER. As shown in Code box 16.13, we call the Pandas `value_counts(normalize = True)` method to calculate the proportions of sentiment classes assigned to the text documents, and the Pandas `crosstab()` function to calculate the absolute number of documents that were assigned either the positive, neutral, or negative sentiment classes by VADER or RoBERTa. We divide these numbers by the total number of documents to calculate the percentage of agreement between the two algorithms. Output 16.6 displays the proportion of assigned sentiment categories by the RoBERTa algorithm and a cross-table which contains the relative values of the sentiment classes assigned by VADER and RoBERTa:

```
df.RoBERTa_label.value_counts(normalize=True)
pd.crosstab(df.RoBERTa_label,  df.sentiment_scores_categorized)
/ len(df)
```

Code box 16.13: Calculating the proportion of sentiment classes assigned to the documents by RoBERTa, and cross-tabulation of the agreement of the sentiment classes assigned by VADER and RoBERTa.

```
RoBERTa_label
negative    0.463686
positive    0.360027
neutral     0.176287
Name: proportion, dtype: float64

sentiment_scores_categorized  negative   neutral   positive
RoBERTa_label
negative                      0.224353   0.082415  0.156917
neutral                       0.019540   0.090322  0.066425
positive                      0.028479   0.047984  0.283564
```

Output 16.6: Relative frequencies of the assigned sentiment for all text documents.

We see notable differences between VADER and RoBERTa if we compare the sentiment classes assigned by the two algorithms (see Table 16.2). First of all, RoBERTa assigns much less positive and neutral sentiment classes to the text documents compared to VADER. Secondly, RoBERTa assigns negative sentiment more than twice as often as VADER. But how well do both algorithms agree on the sentiment class assigned to the text documents? We can, for example, calculate Cohen's Kappa as demonstrated in Code box 16.14:

```
agreement = metrics.cohen_kappa_score(df.sentiment_scores_
categorized, df.RoBERTa_label)
np.round(agreement, 4)
```

Code box 16.14: Calculating Cohen's Kappa to quantify the agreement of VADER and RoBERTA in regards to the assigned sentiment classes

Table 16.2 Comparison and Agreement Between the Sentiment Categories Assigned by VADER and RoBERTa

Sentiment Categories	*Proportion assigned by VADER*	*Proportion Assigned by RoBERTa*	*Cohen's Kappa*
Positive	0.5069	0.3600	0.3841
Neutral	0.2723	0.1763	
Negative	0.2207	0.4637	

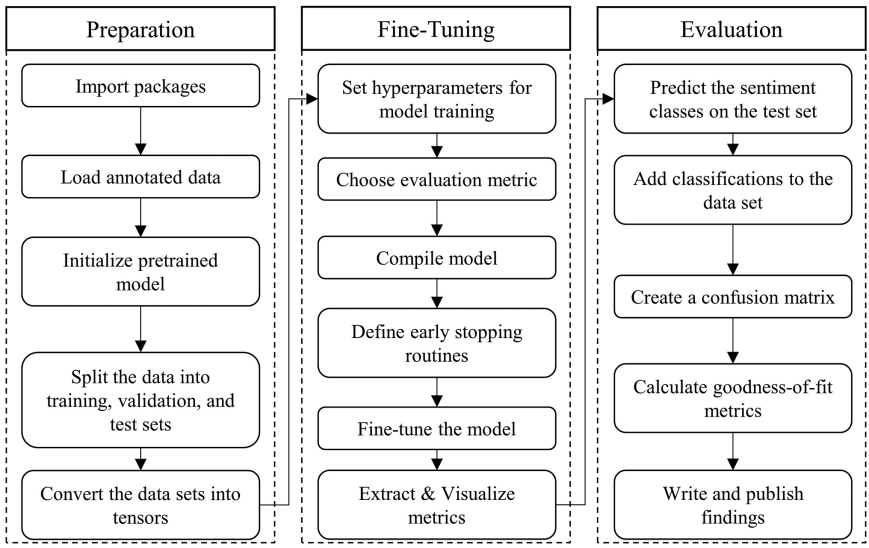
Cohen's Kappa calculates the proportion of the shared categories assigned by two raters (see Warrens, 2015). The Kappa score takes values ranging between 0 and 1. In our case, we treat VADER and RoBERTa as raters, and according to the Kappa value, they do not agree too often on the same sentiment class (Kappa = 0.3841). In other words, both algorithms produce vastly different sentiment distributions based on different assignments of sentiment classes on the texts. To decide which model to use, it would be up to us to interpret a subset of text documents and assign a sentiment class based on our interpretation. Only by doing so could we decide on which of the two models to use for sentiment classification.

16.4 Fine-tuning a Transformer model for sentiment analysis

Besides using a pretrained model to extract sentiments or to calculate polarity scores, we could also use pretrained deep neural networks and adjust them to our data or task. This adjustment is called **fine-tuning** and is considered a transfer learning technique (see Section 11.2). Fine-tuning uses parameters of a pretrained model and updates them to accomplish our novel task or novel corpus. Furthermore, we remove the output layer from the pretrained model and exchange it for a task-specific output layer (Wankmüller, 2022, p. 1692). We then retrain our model to adapt to our text corpus or the novel task.

One example where fine-tuning might be necessary is to analyze the sentiment of newspaper articles. Newspapers are usually written in a relatively

Figure 16.3 Workflow applied for fine-tuning the sentiment analysis and model evaluation.



neutral language or journalists can mask their sentiments or opinions on a topic or actors (such as politicians) using different stylistic devices. As large language models are usually trained on different corpora (e.g., on social media or Wikipedia), they might not be able to catch the sentiments conveyed in journalistic texts properly. Another example are citations in scientific texts, for example journal publications. Scholars and scientists may agree (e.g., positive sentiment) or disagree (negative sentiment) with the content of cited references. Compared to authors on Wikipedia or users on a social media platform, scholars use a different vocabulary to express their (dis-) agreement and thus their sentiment towards a cited reference. Therefore, a model trained on either Wikipedia or social media is neither able to correctly identify sentiments conveyed in newspaper articles, or agreement or disagreement in the highly objective style of a scientific publication, which is why it needs to be fine-tuned on the previously unseen corpus with adequate sentiment categories.

In the following, we will fine-tune a DistilBERT model and use the sentiment classes assigned by RoBERTa on the YouTube comments posted below social science videos as ground truth. We use DistilBERT for fine-tuning, as it is 60% smaller and needs approximately 60% less time for training compared to the initial BERT version, but retains over 95% of its task performance (Sanh et al., 2019). We must do so due to limited computing capacities of the author writing this chapter.

We will now begin with importing the relevant packages to fine-tune the DistilBERT model as shown in Code box 16.15:

```
import os
import re
import pandas as pd
from sklearn.model_selection import train_test_split
from datasets import Dataset, DatasetDict

from transformers import AutoTokenizer
from transformers import TFAutoModelForSequenceClassification
from transformers import pipeline

import tensorflow as tf

import sklearn.metrics as metrics
import numpy as np

import seaborn as sbs
from matplotlib import pyplot as plt
```

Code box 16.15: Importing the necessary packages for fine-tuning the BERT models

As always, we begin our workflow with importing the packages, modules, and functions necessary to prepare and analyze our data, and to evaluate the results. We import the `os`, `re` and `pandas` (alias: `pd`) packages for data munging and data preparation. Furthermore, we import the `Dataset` and `DatasetDict` functions from Hugging Face's `datasets` package. Both enable us to create a training set, validation set from the data, which are then used for fine-tuning and model evaluation. We also import the `train_test_split` function from `sklearn.model_selection` to split our text documents into training set, test set, and validation set. We furthermore add the `AutoTokenizer`, `pipeline`, and `TFAutoModelForSequenceClassification` from the transformers package, as well as `tensorflow` (alias: `tf`) to prepare and fine-tune the `DistilBERT model`. We also import `sklearn.metrics` as well as `numpy` (alias: `np`) to evaluate how well our model predicts the sentiment classes assigned by RoBERTa, and import `seaborn` (as `sbs`) and `matplotlib.pyplot` (alias `plt`) to visualize our findings.

We then define the paths, change the working directory, and use the Pandas `read_csv()` function as demonstrated in Code box 16.16 and import the `soc_sci_RoBERTa_sentiment.csv` data table that contains not only the document metadata, but also the VADER and RoBERTa sentiment classifications:

```
root = "C:\\Users\\[PATH]\\"
path = root + "Data\\"
model_path = root + "Trained_Model\\"
output = root + "Output\\"

os.chdir(path)
df = pd.read_csv("soc_sci_RoBERTa_sentiment.csv", sep = ";")
```

Code box 16.16: Setting the paths, changing the working directories, and loading the data table with VADER and RoBERTa sentiment classifications

16.4.1 Initializing the DistilBERT architecture

We then use the code displayed in Code box 16.17 to initialize a tokenizer that enables us to transform our text documents into a format fit for model training, as well as the DistilBERT model without a task-specific output layer itself:

```
tokenizer = AutoTokenizer.from_pretrained("distilbert-base-
uncased", return_tensors = "np",
                                truncation = True,
                                padding = 512)

model = TFAutoModelForSequenceClassification,
    from_pretrained("distilbert-base-uncased", num_labels = 3)
model.summary()
```

Code box 16.17: Initializing the DistilBERT tokenizer and the DistilBERT model without task-specific output layer

Let us take a closer look at the two functions `AutoTokenizer.from_pretrained()` and `TFAutoModelForSequenceClassification.from_pretrained()`. The first function and method initializes the tokenizer of the DistilBERT architecture, splitting the text into token strings, converting the tokens into integers, potentially adding new tokens into the vocabulary used for the pretrained model, as well as adding special tokens (e.g., masking tokens [MASK], padding tokens [PAD], or a token that symbolizes the beginning of a text sequence [CLS]). The second function and method imports the DistilBERT architecture without task-specific output layer into our working space. The option `num_labels` defines the number of classes we want to assign to the text. We have to pass `num_labels = 3`, as we want to assign either a positive,

neutral, or negative sentiment (three classes in total). The last method `model.summary()` returns a text string that contains information on the model architecture displayed in Output 16.7:

```
model.summary()

_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 distilbert (TFDistilBertMa  multiple                  66362880
 inLayer)

 pre_classifier (Dense)      multiple                  590592

 classifier (Dense)          multiple                  2307

 dropout_19 (Dropout)        multiple                  0 (unused)

=================================================================
Total params: 66955779 (255.42 MB)
Trainable params: 66955779 (255.42 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Output 16.7: Model summary of the DistilBERT architecture

The output shows the layer types which are stacked upon each other, the output shape (e.g., the batch size, the number of tokens to be passed to each layer, and the number of neurons per layer), and the number of weights. Let us dig a little into the layers:

1. `distilbert (TFDistilBertMainLayer)`: This layer stores the DistilBERT architecture and multiple, stacked Transformer-layers that perform the embedding and normalization procedure. This layer has 66,362,880 parameters.
2. `pre_classifier (Dense)`: This layer pools all the different types of information into a single output which is then used to predict the sentiment categories. This layer consists of 590,592 parameters, namely (768 * 768) weights of the connections between the artificial neurons placed in this layer and the previous and subsequent layers plus 768 parameters to capture the biases (=error terms).
3. `classifier (Dense)`: This layer stores the information of the three sentiment classes. Each neuron represents a sentiment and is fully connected (=dense) with the previous layer. Thus, this layer yields 3 * 768 weights, which explains the 2,304 parameters.

4. `dropout_19 (Dropout)`: This layer is not used, as it has 0 parameters. The _19 just tells us that it is the 19th layer that was initialized during our session.

We furthermore see the total number of parameters of the DistilBERT model, which equals the size of the model (66.955.779), the number of parameters that can be fine-tuned (66.955.779), and the number of parameters that are not eligible for fine-tuning (0). These numbers tell us that all parameters of this model are eligible for fine-tuning.

16.4.2 Preparing and splitting the data into training set, validation set, and test set

Let us now draw stratified samples of 10,000 text documents (= YouTube comments) each for model training with the sentiment classes assigned by the RoBERTa architecture as strata as shown in Code box 16.18:

```
df_sample = df.groupby(by = "RoBERTa_label").\
    sample(n = 10000, random_state = 42)
```

Code box 16.18: Drawing a stratified sample of text documents by sentiment assigned by RoBERTa

To perform a stratified sampling, we chain the Pandas `groupby()` and `sample()` methods. In the former case, we group our data frame by the sentiment classes assigned by the RoBERTa algorithm with `groupby(by = "RoBERTa_label")`, draw a reproducible sample of 10,000 documents per sentiment class with `sample(n = 10000, random_state = 42)`, and assign this sample to `df_sample`. We will use these annotations as ground truth for model training. We draw samples of equal sizes, as unequal class sizes reduce the accuracy of the model prediction and introduce bias. For example, if a single class dominates the dataset, a classifier might assign this dominant class to all text documents.

As TensorFlow cannot deal properly with strings as classes, we must convert the sentiment classes stored in `"RoBERTa_label"` into integers. To do so, we use the following code in Code box 16.19:

```
sentiment_values = {"neutral" : 1,
                    "negative" : 0,
                    "positive": 2}

sentiment_values_mapped = df_sample["RoBERTa_label"].map(sentiment_
values)
df_sample.insert(0, "label", sentiment_values_mapped)
```

Code box 16.19: Converting the sentiment classes to integer values

As you see, we first define a dictionary that maps sentiment class to integer values (0 = negative, 1= neutral, and 2 = positive). We then pass this dictionary to pandas' `map()` method to convert the sentiment classes assigned by RoBERTa to the integer values. We then call the Pandas' `insert()` method to add the integer values as a new column (`label`) in `df_sample`.

Next, we split the data into training set, validation set, and test sets (see Code box 16.20):

```
train_validation, test = train_test_split(df_sample,
                                          test_size = 0.2,
                                          stratify = df_sample
                                          ["RoBERTa_label"],
                                          random_state = 42)

train, validation = train_test_split(train_validation,
                                     test_size = 0.25,
                                     stratify = train_validation
                                     ["RoBERTa_label"],
                                     random_state = 42)
```

Code box 16.20: Splitting the dataset into training set, test set, and validation set

As you see, we use the `train_test_split()` function from `sklearn` to split `df_sample` into a test set (20% of 30,000 documents, or 6,000 documents) and a data frame named: `train_validation` that holds the training and validation set (80% of the text documents, or 24,000 documents). The `stratify = df_sample["RoBERTa_label"]` argument ensures that the sample follows the distribution of the sentiment classes present in `df_sample`. The `random_state = 42` ensures reproducibility of the splitting procedure. We then split `train_validation` into training set and validation set. In this case, we reserve another 25% of the documents stored in `train_validation` as validation set, leaving us with overall 18,000 documents (= 60%) for training, 6,000 documents (= 20%) for validation, and 6,000 documents (= 20%) for model testing.

16.4.3 Converting the training set, validation set, and test set into TensorFlow tensors

Next, we prepare our data to be used in TensorFlow. To do so, we take two preparation steps:

1. Converting the training set, validation set, and test set stored as Pandas' data frame into the Apache Arrow data format used by the Hugging Face's `transformers` package, and
2. Converting the latter into a data format ready for use in TensorFlow.

The first of these two conversions is demonstrated in Code box 16.21:

```
datasets = DatasetDict()
datasets["train"] = Dataset.from_pandas(train)
datasets["validation"] = Dataset.from_pandas(validation)
datasets["test"] = Dataset.from_pandas(test)
```

Code box 16.21: Conversion of the training set, test set, and validation set into a data format suitable for training in TensorFlow

The `transformers` package uses the Apache Arrow data format, which supports batching (= passing multiple text documents at once to our deep learning architecture), and stores the data passed to the DistilBERT model in the random access memory of the computer to speed up the calculations (Groet et al., 2024, S. 2). It also allows for seamless GPU integration – which accelerates the calculations further – and allows for mapping preprocessing steps directly on the data.

In the first line of code, we use `DatasetDict()` to create a dictionary for the storage of Arrow datasets and assign this to `datasets`. In a next step, we call the `Dataset.from_pandas()` method and pass the training set, validation set, and test set – for example `Dataset.from_pandas(train)` – and convert these sets to the Apache Arrow data format. These converted datasets are then passed as values to keys in the `datasets` dictionary.

We can use functions and methods on the Apache Arrow data by using the `map()` method on `datasets`. In our case, we map the DistilBERT tokenizer on

```
def tokenize_function(data):
    return tokenizer(data["textOriginal"],
                     padding = "max_length",
                     truncation = True)

tokenized_datasets = datasets.map(tokenize_function,
                                  batched = True,
                                  batch_size = 32)
```

Code box 16.22: Defining a tokenization function for the Apache data format and mapping this function onto the datasets dictionary

our training set, validation set, and test set. To do so, we define a function for tokenization of the texts in our dictionary (see section for further information on how to define functions 12.4.5) as shown in Code box 16.22:

We define `tokenize_function()` and use `data` as a placeholder for the input data. The aim of this function is to return the tokenized, truncated, and padded text documents which are then used as input for the DistilBERT architecture. To do so, we apply the `tokenizer` initialized in Code box 16.17 on `textOriginal`, where the YouTube comments are stored as text documents. We add `padding = "max_length"` and `truncation = True` to ensure our outputs have a length of 512 tokens.

We then apply `datasets.map()` and pass the `tokenize_function` to ready the documents for sentiment analysis. Besides passing the tokenizer, we also set two options, namely `batched=True`, and `batch_size = 32`. The former tells Python to handle the data as batches, and the latter defines the batch size, meaning that 32 documents are passed simultaneously to the tokenizer function. We will run the code and pass the tokenized data to `tokenized_datasets`.

The second conversion step is now ready to be performed. The code to be used is displayed in Code box 16.23.

```
tf_dataset_train = model.prepare_tf_dataset(tokenized_
datasets["train"], batch_size = 32, tokenizer = tokenizer)

tf_dataset_validation = model.prepare_tf_dataset(tokenized_
datasets["validation"], batch_size = 32, tokenizer = tokenizer)

tf_dataset_test = model.prepare_tf_dataset(tokenized_datasets
["test"], batch_size = 32, tokenizer = tokenizer)
```

Code box 16.23: Converting the training set, test set, and validation set from arrow format into a TensorFlow data format

Here, the `prepare_tf_dataset()` method transforms the Arrow-based dictionary stored in `tokenized_datasets` into TensorFlow tensors. We furthermore pass a batch size and the tokenizer to ensure that the documents are properly converted into tensors which are then used as batches of 32 for model training.[18]

16.4.4 Preparing and compiling the DistilBERT model

In a next step, we prepare and compile our DistilBERT model for fine-tuning. This involves the following:

1) Initializing the optimizer to update the model parameters during training.
2) Defining the loss function to quantify prediction errors.

We begin with initializing the Adam optimizer (see Code box 16.24), which is a variant of stochastic gradient descent (Kingma, 2014) that converges faster and reduces the training time considerably (Zhang, 2018):

```
optimizer = tf.keras.optimizers.Adam(learning_rate = 3e-5,
                                     epsilon = 1e-08,
                                     clipnorm = 1.0)
```

Code box 16.24: Initializing the Adam optimizer

The `tf.keras.optimizers.Adam()` optimizer assigned to the `optimizer` variable has three key parameters:

1. `learning_rate =` This option controls how quickly the weights of the artificial neurons are updated. Larger models require smaller learning rates to reach convergence. Convergence is defined as the point in training where further updates yield minimal improvement in predictive power.
2. `epsilon =` This is a small constant added to the weights of the artificial neurons to avoid divisions of the numerical values by zero. Smaller values improve the precision of the learning procedure, but may cause instabilities if changes in the weights become too small over time.
3. `clipnorm =` This option limits the values of gradients to 1.0 to prevent the exploding gradients problem. Exploding gradients, namely weights that grow exponentially during training, can be interpreted as overreacting neurons (like a brain on drugs like cocaine), which destabilize the training procedure.

We then define the loss function to quantify the prediction error on the validation set (see Code box 16.25):

```
loss = tf.keras.losses.SparseCategoricalCrossentropy
(from_logits = True)
```

Code box 16.25: Initializing the loss function

We use the `SparseCategoricalCrossentropy()` method from the `tf.keras.losses` to quantify prediction errors. Sparse categorical cross-entropy is the standard loss function for multi-class classification, which represents sentiment labels as integers (e.g., 0 = negative, 1 = neutral, 2 = positive). Setting `from_logits = True` instructs TensorFlow to interpret raw logits (logarithmic odds) rather than probabilities for sentiment class prediction.[19]

The model is then initialized using the `compile()` method (Code box 16.26).

```
model.compile(metrics = "accuracy",
              loss = loss,
              optimizer = optimizer)
```

Code box 16.26: Compiling the DistilBERT model

This method specifies the metric, loss function, and optimizer to use during model training. We use the accuracy metric to quantify the share of correctly predicted sentiment categories on our training set.

Furthermore, we must prevent our model from overfitting. Overfitting occurs when the trained model becomes overly specialized in recognizing patterns in the training set, which reduces its ability to generalize its predictions to unseen data. To prevent overfitting, we use the `EarlyStopping()` method from the `tf.keras.callbacks` module (see Code box 16.27):

```
early_stopping = tf.keras.callbacks.EarlyStopping (monitor =
                                   "val_loss",
                                   patience = 2,
                                   verbose = 1)
```

Code box 16.27: Defining early stopping behavior

This method monitors validation loss (`monitor = "val_loss"`) and stops the training procedure if the validation loss (= sparse categorical cross-entropy increases in two consecutive training epochs. The `verbose = 1` option is set to inform you if – and when – early stopping happens.

16.4.5 Fine-tuning the DistilBERT architecture for sentiment analysis

After setting all hyperparameters, compiling the model, and setting our early stopping routine, it is now time to fine-tune our DistilBERT model. To do so, we use `model.fit()` as displayed in Code box 16.28:

```
model.fit(tf_dataset_train,
          epochs = 10,
          validation_data = tf_dataset_validation,
          callbacks = [early_stopping])
```

Code box 16.28: Fine-tuning the DistilBERT model

Besides passing the training data (`tf_dataset_train`), we set the following key parameters:

1) `epochs` = the maximum number of training iterations (set to `10`).
2) `validation_data` = data used to evaluate the predictive power of our model (here on `tf_dataset_validation`).
3) `callbacks` = list of early stopping options (`[early_stopping]`).

Fine-tuning takes time, so we advise you to read some scientific papers, write some notes, or enjoy one or multiple cups of coffee. Fine-tuning our comparatively small training set of 18,000 documents requires multiple hours per iteration.[20] Each iteration consists of training with batches of 32, meaning that we update the weights of the artificial neurons via backpropagation 562 times (18,000/32 = 562,5). After each epoch, our model predicts sentiments for the 6,000 documents in `tf_dataset_validation`, before computing loss and accuracy scores to assess the ability of our DistilBERT model to generalize beyond the training set.

After model training is completed, we save the model using the `save_pretrained()` method as displayed in Code box 16.29:

```
model.save_pretrained(save_directory = model_path, version = 1)
```

Code box 16.29: Saving the trained DistilBERT model

This saves our model in the defined `save_directory = model_path`, while specifying the version number allows future refinements of the fine-tuned DistilBERT model.

Let us now extract and save the model history for model assessment. In the following, we will use the code displayed in Code box 16.30 to extract the loss and accuracy per epoch, and assign it to a Pandas data frame (seen as table in Table 16.3):

```
model_history = pd.DataFrame(model.history.history)
model_history.insert(0, "epoch", range(1, len(model_history)+1))
```

Code box 16.30: Extracting training history

Table 16.3 Model History With Values of the Sparse Categorical Cross-Entropy and Accuracy for the Training and Set Validation Set Per Epoch

epoch	*loss*	*accuracy*	*val_loss*	*val_accuracy*
1	0.53837162	0.78077775	0.46708369	0.81116664
2	0.29757461	0.88983333	0.50656378	0.79983336
3	0.14437251	0.95038891	0.63116771	0.81583333

In the first line of code, we convert the training history stored in `model.history.history` from the fine-tuning process into a Pandas data frame. The `model.history.history` attribute stores the training metrics recorded by TensorFlow during model training. This structure is the result of TensorFlow initializing a `History` object to store loss and accuracy values, appending them as a dictionary upon completion of training.

The resulting table tells us that model training terminates after three epochs to prevent overfitting. This is done in accordance with the increase in the validation loss metric (`val_loss`). Overall, our DistilBERT model was able to predict the correct sentiment class correctly in 80% of the text documents stored in the validation set.

Next, we pivot the model history for visualization purposes (Code box 16.31):

```
model_history_long = pd.melt(model_history, id_vars = "epoch")
train_val = ["validation" if re.search("^val_", x) else "training"
         for x in model_history_long.variable]
model_history_long.insert(1, "train_val_set", train_val)
```

Code box 16.31: Pivoting the training history into a long format for data visualization

In the first line of code, the `melt()` function transforms the Pandas data frame from wide to long format. In the wide format, each column is equated with a variable, while in long format, these columns are stacked upon each other. The following list comprehension iterates over the column names, namely `loss`, `accuracy`, `val_loss`, and `val_accuracy`, and assigns `validation` to those strings starting with `val_`, and `training` otherwise. This information is inserted as a new column (`train_val_set`) into `model_history_long`, which helps us to distinguish between training and validation metrics, resulting in Table 16.4.

Finally, we save the model history and the pivoted model history (see Code box 16.32):

```
model_history.to_csv("finetuned_albERT_history.csv")
model_history_long.to_csv("finetuned_albERT_history_long.csv")
```

Code box 16.32: Saving model history

Table 16.4 Pivoted Model History

epoch	*train_val_set*	*variable*	*value*
1	training	loss	0.53837162
2	training	loss	0.29757461
3	training	loss	0.14437251
1	training	accuracy	0.78077775
2	training	accuracy	0.88983333
3	training	accuracy	0.95038891
1	validation	val_loss	0.46708369
2	validation	val_loss	0.50656378
3	validation	val_loss	0.63116771
1	validation	val_accuracy	0.81116664
2	validation	val_accuracy	0.79983336
3	validation	val_accuracy	0.81583333

If you want to load a custom fine-tuned model in the future, you can call the following functions and methods (see Code box 16.33):

```
model = TFAutoModelForSequenceClassification.from_pretrained
(model_path)
model.load_weights(model_path + "tf_model.h5")
```

Code box 16.33: Loading the fine-tuned DistilBERT model

The first line of code is similar to Code box 16.17. Here, we load the model architecture using `from_pretrained()`, and specify the folder where our fine-tuned DistilBERT model is saved. Then, `load_weights()` initializes the model with the pretrained weights stored in `tf_model.h5`. This ensures that our model is ready to predict the three sentiment classes on new, unseen text documents.

16.4.6 Visualization of the metrics used for model optimization

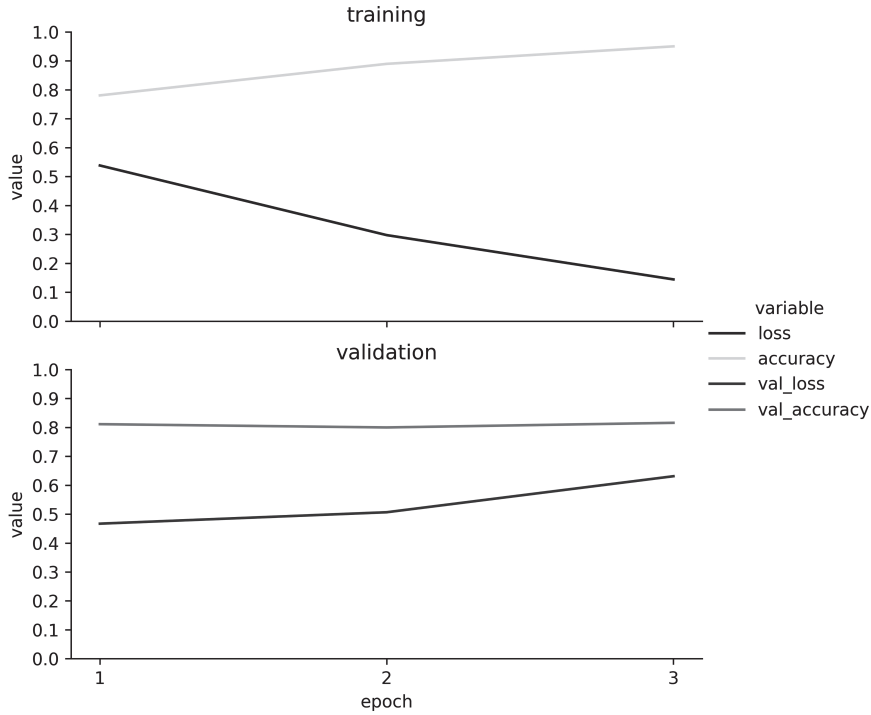
To visualize accuracy and loss values for both training and validation sets (Figure 16.4), we use Seaborn's `FacetGrid()` function and `map()` line plots onto the grid (Code box 16.34):

```
g = sbs.FacetGrid(model_history_long,
                  row = "train_val_set",
                  hue = "variable",
                  aspect = 2
                  )
g.map(sbs.lineplot, "epoch", "value")
g.add_legend()

axes = g.axes.flatten()

for ax in axes:
    ## set titles
    title = ax.get_title()
    title = re.sub("^.+= ","",title)
    ax.set_title(title)

plt.xticks(range(1, len(model_history)+1)
plt.yticks(np.arange(0, 1.1, 0.1))

os.chdir(output)
plt.savefig("training_validation_values.png", dpi = 600)

plt.close()
```

Code box 16.34: Visualizing the accuracy and loss functions of the training set and validation set

We start with initializing `FacetGrid()`, pass `model_history_long` as input data, and organize our plot by the `train_val_set` column. We then color our lines by `variable` (`loss`, `accuracy`, `val_loss`, and `val_accuracy`) and set the aspect ratio to 2 to ensure that the subplots are twice as wide as they are high. After we assign the facet grid to `g` (short for graph), we map Seaborn's `lineplot()` to plot accuracy and loss (y-axis) over epochs (x-axis) using `g.map()`. We further call `g.add_legends()` to place a legend to distinguish between different metrics in both the training and validation sets.

Figure 16.4 Visualization of the accuracy values and values of the loss function (categorical cross-entropy) in the training set and validation set.



We then clean the subplot titles. To do so, we extract the information of both subplot axes and titles with `g.axes.flatten()`. We then proceed with extracting the titles in the subsequent for loop and substitute the start of the title (usually: variable =) with an empty string. We then adjust the ticks on both x- and y-axes using `matplotlib.pyplot`'s `xticks()` and `yticks()` functions. Finally, we change the working directory, save the figure in high resolution, and close the figure canvas to free memory. We can now use this figure to provide an intuitive way to access the predictive power of our trained DistilBERT model, or to compare different training runs (e.g., with different hyperparameter settings or different models, such as RoBERTa).

16.5 Model evaluation

The final steps of our fine-tuning procedure involve model evaluation (right side of the workflow in Figure 16.3). In our case, model evaluation consists of three steps:

1) Assessing performance on the test set by applying our fine-tuned DistilBERT classifier to evaluate how well it predicts sentiment classes on unseen data.

2) Visualizing predictions with a confusion matrix to compare the predicted sentiment classes from our fine-tuned DistilBERT model against the ground truth (sentiment classes assigned by the RoBERTa model).
3) Quantifying the predictive performance of our fine-tuned model by calculating precision, recall, and accuracy.

Beyond these three evaluation steps, we could analyze misclassified documents by employing either inductive-qualitative content analysis (Chapter 6) or deductive-qualitative content analysis (Chapter 7). This could help us to interpret why (and for what reason) our model misclassified the sentiments in the test set. Alternatively, we could leverage machine learning techniques to identify features such as tokens, n-grams, phrases or other features on token or document level that are linked to misclassifications. However, these approaches are outside the scope of this chapter, which is why we will solely focus on the three evaluation steps.

16.5.1 Predicting sentiment classes on the test set

We begin with applying our fine-tuned DistilBERT model to predict the sentiment of documents in the test set. For this, we initialize a `transformers pipeline()` for text classification (Code box 16.35):

```
classifier = pipeline(task = "text-classification",
                      model = model,
                      truncation = True,
                      max_length = 512,
                      padding = True,
                      tokenizer = tokenizer)
```

Code box 16.35: Initializing a pipeline with the fine-tuned DistilBERT model for sentiment classification

As you see, the pipeline consists of the following elements:

1) `task` = Specifies the task of our model, namely text classification (`"text-classification"`).
2) `model` = Here, we assign the fine-tuned DistilBERT from Section 16.4.5.
3) `truncation` = Ensures that the input texts are truncated to the defined `max_length`.
4) `max_length` = Limits the number of tokens per text passed as input to the fine-tuned model to 512.
5) `padding` = Adds `[PAD]` tokens to fill texts with less tokens than `max_length`.
6) `tokenizer` = Uses the DistilBERT tokenizer from Section 16.4.1 to preprocess the documents in the test set.

We assign the pipeline to `classifier` and apply it to a list of text documents stored in `test.textOriginal`. As demonstrated in Code box 16.36, this results in a list of dictionaries containing the predicted sentiment label (`Label_0`, `Label_1`, and `Label_2`) and the confidence score displayed in part in Output 16.8:

```
classified_documents = classifier(list(test.textOriginal))
```

Code box 16.36: Applying the classifier pipeline to the comments in the test set

```
[{'label': 'LABEL_0', 'score': 0.9704149961471558},
 {'label': 'LABEL_1', 'score': 0.6171618103981018},
 {'label': 'LABEL_2', 'score': 0.9750478863716125},
 {'label': 'LABEL_0', 'score': 0.998788058757782},
 {'label': 'LABEL_1', 'score': 0.990152895450592},
...]
```

Output 16.8: List of predicted sentiment labels and confidence scores

Before visualizing the confusion matrix or computing precision, recall, and accuracy, we must convert the assigned label (`Label_0`, `Label_1`, and `Label_2` seen in Output 16.8) to the respective sentiment classes (negative, neutral, and positive). To achieve this, we define a dictionary which will be used for this mapping procedure (Code box 16.37):

```
revert_values = {"LABEL_1" : "neutral",
                 "LABEL_0" : "negative",
                 "LABEL_2" : "positive"}
```

Code box 16.37: Create dictionaries to revert the labels returned by the TensorFlow classifier to sentiment classes and sentiment class numbers

Now, it is time to insert the sentiment labels as well as probability scores assigned by our DistilBERT model before we map the assigned labels to sentiment classes (see Code box 16.38):

```
test.insert(2,
            "BERT_classified", [x["label"] for
            x in classified_documents])
test.insert(3,
            "BERT_classified_score",  [x["score"] for
            x in classified_documents])
test.insert(4,
            "BERT_classfied_label",
             test.BERT_classified.map(revert_values))

test.to_csv("annotated_test_set.csv")
```

Code box 16.38: Adding sentiment classification results as columns to the test set and saving the latter as a CSV file

We use the pandas `insert()` method to add three new columns to the test set: the predicted labels (`"BERT_classified"`), their confidence scores (`"BERT_classified_score"`), and the mapping of the predicted labels to the sentiment class (`"BERT_classfied_label"`). The labels and scores are extracted from the list of dictionaries using list comprehensions, which allows us to pull the values from the respective dictionary keys. We then use the Pandas `map()` method to convert `Label_0`, `Label_1`, and `Label_2` into the original, human-readable sentiment classes `negative`, `neutral`, and `positive`. Finally, we save the updated data frame containing the test set as a CSV file. This makes the classification results accessible for future analysis, including model comparisons and qualitative content analysis of misclassified texts.

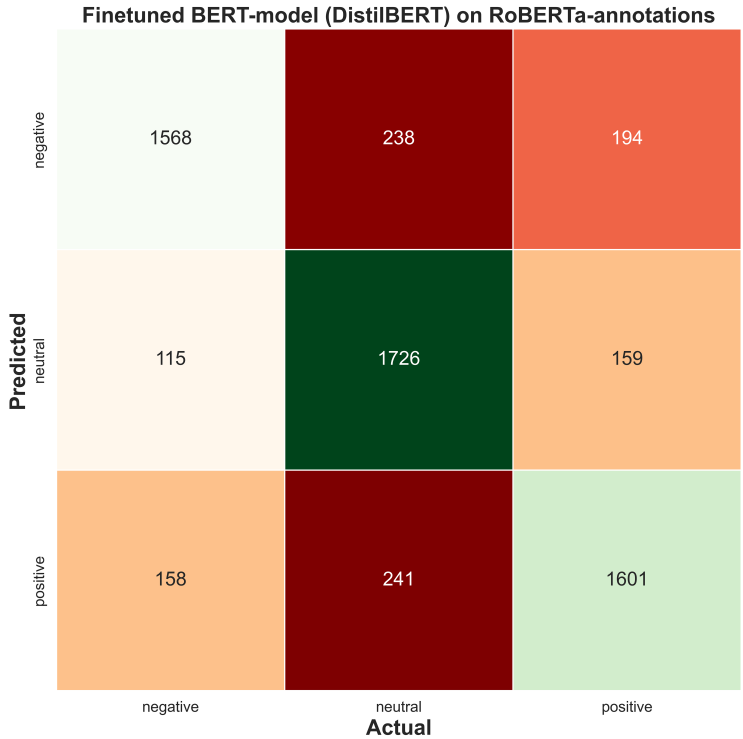
16.5.2 Creating and visualizing the confusion matrix

We will now generate a confusion matrix from a cross table that includes the predicted labels as rows and the ground truth (= sentiment classes assigned by the RoBERTa algorithm) as columns. To establish the cross table, we call Pandas' `crosstab()` and pass `test["label"]` as index values, and `test["BERT_classified"]` as column values (Code box 16.39):

```
confusion_matrix = pd.crosstab(test["label"], test["BERT_
classified"])
```

Code box 16.39: Creating a confusion matrix using pandas crosstab() function

Figure 16.5 Confusion matrix of the predicted versus the actual sentiment classes of the test set.



From this starting point, we create the confusion matrix depicted in Figure 16.5. But what are the key insights we can gather from this confusion matrix? First, the diagonal values indicate that our DistilBERT model predicted the correct sentiment class in most cases. Secondly, the off-diagonal values reveal misclassifications. Thirdly, the confusion matrix shows us that our model has a slight bias towards positive sentiments, as indicated by more false positives in the top-right corner compared to false negatives in the bottom-left.

We will use the code displayed in Code box 16.40 to visualize the confusion matrix, which bears similarities to the Python code used to create the heatmap in Section 14.6:

```
sbs.set_theme(font_scale = 1.5)
off_diag_mask = np.eye(len(confusion_matrix.index),
                       len(confusion_matrix.columns),
                       dtype = bool)
```

Code box 16.40: Creating a confusion matrix with seaborn

```
fig,ax = plt.subplots(figsize = [10, 10])
# plot the diagonal values (correctly predicted data)
sbs.heatmap(data = confusion_matrix,
            annot = True,
            mask = ~off_diag_mask,
            cmap = 'Greens',
            fmt = '.0f',
            cbar = False,
            linewidths = 1,
            ax = ax)

# plot other values (falsely predicted data)
sbs.heatmap(data = confusion_matrix,
            annot = True,
            mask = off_diag_mask,
            cmap = 'OrRd',
            fmt = '.0f',
            cbar = False,
            linewidths = 1,
            ax = ax)

## change axis details
plt.title("Finetuned BERT-model (DistilBERT) on RoBERTa-annotations",
           fontdict = {"size" : 20, "weight": "bold"})

labels = ["negative", "neutral", "positive"]
ax.set_xticklabels(labels = labels, fontdict = {"size" : 14})
ax.set_yticklabels(labels = labels, fontdict = {"size" : 14})
plt.xlabel("Actual", fontdict= {"size" : 20,
                                "weight" : "bold"})
plt.ylabel("Predicted", fontdict= {"size" : 20,
                                "weight" : "bold"})

plt.tight_layout()
## save output
os.chdir(output)
plt.savefig("Confusion_Matrix.png",
           dpi = 600)
plt.close()
```

Code box 16.40: (Continued)

Let us walk you briefly through the code to visualize the confusion matrix which highlights correct and incorrect predictions made by our DistilBERT sentiment classifier.

In the first line, we use Seaborn's `set_theme()` to define the graph properties displayed in Figure 16.5. In our case, we scale the labels and annotations by a factor of 1.5 by using `set_theme(font_scale = 1.5)` to ensure better readability.[21] We then call `np.eye()` and create a square NumPy matrix with `True` values on the diagonal and `False` off-diagonal (`type = bool`). We furthermore pass the number of rows and columns of this square matrix. By doing so, we create a mask for diagonal elements that helps us to differentiate between correct predictions and false positives and false negatives. Next, we initialize a figure ten inches wide and ten inches high (`fig`) as well as its axis elements (`ax`) by using the `matplotlib.pyplot's plt.subplots(figsize = [10, 10])` function and plot the confusion matrix using two Seaborn `heatmap()` plots. The heatmap plots the correct predictions on the diagonal (`mask = ~off_diag_mask`) in green (`cmap = 'Greens'`), whereas the second plots the misclassifications (`mask = off_diag_mask`) in oranges and red colors (`cmap = 'orRd'`). Besides these differences, both share common options:

1) `data =` Here, we pass the confusion_matrix to Seaborn's `heatmap()`.
2) `annot =` Enables annotations if set to True. In our case, this option annotates the number of (in-)correctly predicted sentiment classes by our DistilBERT model.
3) `fmt =` This option formats the numbers displayed in the annotations. In our case, '.0f' formats annotations as whole numbers.
4) `cbar =` This option enables or disables the color bar, which encodes the information on the annotated frequencies in the confusion matrix.
5) `linewidths =` This option defines the space between the cells.
6) `ax =` Axis of the figure onto which the heatmaps are plotted.

Afterwards, we format the titles and labels of our figure. We set the title by calling `plt.title()` and specify size and formatting with the `fontdict` dictionary. We also update the tick marks and tick labels on the X- and Y-axes using `ax.set_xticklabels()` and `ax.set_yticklabels`(). We also update the labels on the X- and Y-axes with the `plt.xlabel()` and `plt.ylabel()` functions respectively. We then fit the confusion matrix to the figure canvas with `plt.tight_layout()`, save the plot in high resolution using `plt.savefig(dpi = 600),` and close the figure by calling `plt.close()`.

This visualization allows us to interpret the model's performance and to distinguish between the share and the direction of incorrectly classified sentiments.

16.5.3 Calculating evaluation metrics

Our final task is to quantify the performance of our fine-tuned DistilBERT model. To achieve this, we follow two steps:

1) We extract the predicted sentiment classes generated by our DistilBERT model and the labels from the data. The latter correspond to the sentiment classes assigned by the pretrained RoBERTa algorithm from Section 16.3.3.
2) We compute the performance metrics using `sklearn.metrics`. We will calculate `precision_score()`, `recall_score()`, and `accuracy_score()`.

The following code displayed in Code box 16.41 extracts the predicted sentiment labels and the ground truth and assigns them to two new variables:

```
test_predict = test["BERT_classfied_label"]
test_true = test["RoBERTa_label"]
```

Code box 16.41: Assigning the predicted sentiment classes and the ground truth to new variables

In a second step, we calculate precision, recall, and accuracy (see Code box 16.42):

```
gof = [metrics.precision_score(test_true,
                               test_predict,
                               average = "weighted"),
       metrics.recall_score(test_true,
                            test_predict,
                            average = "weighted"),
       metrics.accuracy_score(test_true,
                              test_predict)]
gof = pd.DataFrame(gof,
             index = ["precision",
                      "recall",
                      "accuracy"],
             columns = ["values"])
```

Code box 16.42: Calculating and saving the precision, recall, and accuracy for our DistilBERT model on the test set

```
## save output
os.chdir(output)
gof.to_csv("GOF_finetuned_on_RoBERTa.csv", sep = ";")
```

Code box 16.42: (Continued)

Table 16.5 Precision, Recall, and Accuracy of the Fine-Tuned DistilBERT Model on the Test Set

Goodness-of-Fit measure on test set	*Fine-tuned DistilBERT model*
Precision	0.8178
Recall	0.8158
Accuracy	0.8158

Let us walk you through the lines of code one final time for this chapter. In the first lines of code, we calculate `metrics.precision_score()`, `metrics.recall_score()`, and `metrics.accuracy_score()` and assign all of them to a list named gof – or short for goodness of fit. We set the `average = "weighted"` option for precision and recall to compute the weighted average across the three sentiment classes (negative, neutral, and positive). Albeit this is unnecessary in the current case (as all classes yield 2,000 text documents in the test set), this option allows us to put more weight to classes which appear less often in the test set compared to other classes. If we want to calculate precision and recall for all sentiment classes separately, we must set `average = None`. Next, we store the goodness-of-fit metrics in a Pandas `DataFrame()`, where the index contains the metrics names, while the column stores the corresponding values. Finally, we save this data frame to a CSV data table. Table 16.5 presents the results. Overall, about 82% of all sentiments predicted by our fine-tuned DistilBERT model on the test set were correct. This evaluation helps us to assess how well our model predicts sentiments on previously unseen data, based on the "ground truth" assigned by the RoBERTa model.

16.6 Concluding words and suggestions for further analyses

In this chapter, we demonstrated how to conduct sentiment analysis using both a dictionary-based approach and a deep learning-based approach. Specifically, we guided you through applying a pretrained BERT model (RoBERTa) and fine-tuning a DistilBERT model. We also demonstrated how to evaluate the performance of your fine-tuned model on a holdout test set.

Although context-aware deep learning models are powerful, they are far from perfect. First, our model correctly predicted the known sentiment in the

test set only 80% of the time, highlighting its limited ability to generalize to unseen data. Second, assessing the model's performance across different corpora – such as YouTube comments on STEM videos or texts with entirely different stylistic characteristics – requires extensive manual evaluation. In other words, prior knowledge of the dataset and the ability to critically assess the model's performance, as well as the ability to qualitatively interpret the content of the original text corpus, as well as the novel text corpus are essential. This also includes detecting shortcomings and qualitatively interpreting both (a) the textual content and (b) its assigned sentiment labels.

Nevertheless, despite these limitations, sentiment analysis – especially with fine-tuned deep learning models – is a valuable tool for gaining insights into the emotions and sentiments expressed in large text corpora. It can also serve as a guide for further qualitative inquiry, particularly when sentiments are analyzed alongside metadata, such as a speaker's characteristics (e.g., a politician) and topics addressed by the speakers (e.g., their stance on migration or higher education). Such analyses can reveal anomalies, inconsistencies in argumentation, or shifts in opinion. On the one hand, these can then be explored further through in-depth qualitative research or, on the other hand, substantiate the potential shift in tone and content analyzed in preceding qualitative inquiries.

Notes

1 Sentiment analysis must be distinguished from emotion detection. While the former aims at extracting sentiment polarity and strength, for example, if a statement or text generally conveys a positive, neutral, or negative attitude, emotion detection aims to extract either classes of emotion – such as fear, anger, sadness, or surprise – or emotion dimensions from a text, for example the Circumplex model of emotions (Russell, 1980).

2 Lexical approaches use a tagged dictionary, where words or phrases are assigned to sentiment values (Alslaity & Orji, 2024, S. 142). These values can be either discrete or continuous (e.g., good could be labelled as "positive" versus positive = 0.9 on a scale ranging from –1 to +1).

3 It is preferable that more than one person annotates the sentiment value of tokens. Yet there could be differences in the interpretation of the meaning of tokens and, as a consequence, the sentiment carried by a token. In order to quantify the agreement between annotators when assigning sentiment scores, we must employ interrater reliability measures. In case of discrete categories, for example positive, neutral, or negative sentiment, we can use Cohen's Kappa (Warrens, 2015) for two raters and Fleiss' Kappa (Shrout & Fleiss, 1979) for more than two raters to calculate the interrater reliability. If sentiments take continuous values, we must calculate the intra-class correlation coefficient, which quantifies how similar annotators rated the sentiment values per token (Bartko, 1966; Liljequist et al., 2019).

4 This is counterintuitive, but the programmer of the vaderSentiment package included a module of the same name within vaderSentiment. If we only type import vaderSentiment, then we won't be able to use its sentiment analyzer.

5 See: retrieved February 2, 2025, from https://github.com/cjhutto/vaderSentiment/blob/master/README.rst.

6 Other possible inputs are "ultralight", "light", "normal", "semibold", "heavy", or "extra bold".

7 There are different transformer models out there in the wild, for example BERT, RoBERTa, XLNet, ELECTRA, DistilBERT, ALBERT, T5, and GPT.
8 To be more precise, such a layer is called 'bidirectional recurrent layer'.
9 More precisely, we can distinguish between different types of representations, namely dense representations: feature vectors capturing semantic meaning (e.g., word embeddings); latent representations: encoded feature spaces in models like autoencoders; and hierarchical representations: extracted features at different levels (e.g., edges in early CNN layers, objects in later layers).
10 A softmax activation function follows the shape of an s-curve.
11 Computationally expensive means (a) that our model will take more time to finish training, (b) uses more electricity, and therefore (c) produces a larger carbon footprint.
12 For further information, see https://huggingface.co/terrencewee12/xlm-roberta-base-sentiment-multilingual-finetuned-v3.
13 According to Liu et al. (2019, S. 7), RoBERTa outperformed the original BERT version in different natural language tasks, like the General Language Understanding Evaluation (GLUE), and a fine-tuned version for sentiment classification is readily available in the hugging face library.
14 In case you have not installed the necessary deep learning architectures yet, please open the terminal in the environments panel of your anaconda distribution and run pip install "tensorflow<=2.15.0",pip install keras_nlp, and pip install transformers.
15 Please be aware that this takes a long time to finish. Make sure that you have (a) many cups of coffee and (b) other books and papers at hand to read in the meantime. The computer of the author of this section took approximately 20 hours to assign sentiment classes to all of the ~1.100.000 comments stored in df.textOriginal.
16 A **tensor** is a multidimensional vector. For example, a data matrix is a two-dimensional tensor, whereas a stack of data matrices is a three-dimensional tensor.
17 This step takes a very long time. In fact, it took the author's notebook 20 hours to complete.
18 The author of this chapter initially tried to convert the data without specifying the tokenizer which led to a dramatic decrease in accuracy. This means that the model was making severe errors and misclassified sentiment categories frequently.
19 Other loss functions, such as categorical cross entropy, are also applied for multi-class classification. In this case, the model outputs a list of logits or probabilities per input document (e.g., logits: [2.5,0.8,-1.2], which correspond to probabilities [0.8806, 0.09758, 0.0218]). If there are only two output classes, such as positive versus negative sentiment, binary cross-entropy is recommended. For continuous output variables, metrics like R^2, root mean squared error, or mean absolute errors can be used instead.
20 In the author's case, it took one iteration approximately 2 ½ hours to finish.
21 Please consult the Seaborn set_theme() documentation for a comprehensive explanation of the possible options: https://seaborn.pydata.org/generated/seaborn.set_theme.html.

17

TOPIC MODELING WITH LATENT DIRICHLET ALLOCATION

17.1 Introduction

In this chapter, we will introduce you to topic modeling with **Latent Dirichlet Allocation** (LDA). Topic modeling algorithms are a class of unsupervised learning algorithms which allow you to extract topics from a text corpus without the need to read all the documents. Albeit there are many topic modeling algorithms out there in the wild, for example structured topic models (Roberts et al., 2014), latent semantic indexing (Deerwester et al., 1990), or correlated topic models (Blei & Lafferty, 2007), we opted for LDA as it is a widely used and well understood (Jelodar et al., 2019).

LDA is a probabilistic, generative model topic model developed by Blei et al. (2003) with the aim of extracting topics from large text corpora (e.g., Wikipedia). A **topic** is a collection of tokens or n-grams that co-occur throughout the documents of a text corpus. **Probabilistic** means, that each token and each document is assigned to every topic with a certain probability. Across the documents, the sum of all topic probabilities is one, albeit the topic distribution differs between documents. In other words, each text contains a mixture of topics. At the same time, the topic probabilities seen in the documents stem from an underlying, latent probability distribution that shapes the topic distribution across documents.

To calculate the topic probability distributions, LDA uses a **prior**, or belief, on how the topics are distributed before we pass the documents to the algorithm. A prior is similar to our knowledge about a research topic that guides our interpretation of topics in qualitative content analysis. The **latent probability distribution** follows a **Dirichlet distribution** that governs the allocation of topics to tokens and documents. Consequently, the prior (or belief) on how the topics are structured also follows the Dirichlet distribution. This is why this method is called Latent Dirichlet Allocation. Without going into too much mathematical

DOI: 10.4324/9781003496397-20
This chapter has been made available under a CC-BY-NC-ND 4.0 license.

detail, a Dirichlet distribution allows us to calculate the probability distributions within documents while maximizing the discriminatory power between the topics.

This may sound abstract and intimidating at first. But, assuming that the topics seen in documents result from such a latent structure that follows the Dirichlet distribution, enables the LDA to capture different topics within a single text based on some prior knowledge about how the text is structured. The text is structured by tokens (such as words or word sequences, called n-grams) and their number of appearances. Take this book for example. It was written by three authors and contains 18 chapters. Each chapter focuses on different topics and consequently uses different tokens or n-grams to describe the content. Imagine further that the paragraphs were documents passed to the LDA. Based on the vocabulary used in the paragraphs, the LDA would then return a topic probability distribution per paragraph after calculation.

Due to the probabilistic nature of the LDA and its assumption that the document and topics are linked via co-occurring tokens, it also assumes that we can ignore word order, syntactic, and grammatical features of the text. This assumption is called the **bag-of-words assumption** (Blei et al., 2003, S. 994). Although this assumption is clearly unrealistic, it helps to reduce the complexity of the text corpus – and thus the computations to perform – significantly. Furthermore, if the text corpus is sufficiently large, this method allows to extract coherent, human-interpretable topics.

Let us step back for a second and imagine a topic model consisting of five topics and three documents. Let us start with the token-topic distributions of the five synthetic topics seen in Table 17.1. The tokens assigned to each topic enable us to guess what the underlying theme or topic of a document could be. In case of Topic 0, which is characterized by words such as sociology, theory, Bourdieu, habitus, and field, we could guess that a text which includes these words could be concerned with the "Habitus-Field-theory" by the French sociologist Bourdieu. Maybe it is a textbook on sociological theory or, more specific, a chapter focusing on his work. In contrast, topic 1 could be closely

Table 17.1 Token-Topic Distribution of Five Synthetic Topics

Topic	*Top 5 tokens with the highest loadings.*
0	0.8 * sociology, 0.7* theory, 0.65 * Bourdieu, 0.6 * habitus, 0.59 * field
1	0.9 * knowledge, 0.8 * discourse, 0.78 * Foucault, 0.71 * critical, 0.60 * post_structuralist
2	0.7 * topic_modeling, 0.66 * method, 0.65 * methodology, 0.62 * text_corpus, 0.61 * LDA
3	0.85 * market, 0.84 * economy, 0.77 * finance, 0.75 * stock_market, 0.55 * firm
4	0.89 * qualitative, 0.8 * content_analysis, 0.75 * category, 0.71 * coding_ scheme, 0.66 * interviews

related to "(critical) discourse analysis" in the spirit of Foucault, while the tokens assigned to Topic 2 hint at "topic modeling using LDA". Topic 3 could be coined "economic sociology" or "economics and finance", whereas Topic 4 could simply be related to "qualitative content analysis". These synthetic topics are well delineated and the token-topic pairs enable us to label the topics very easily. However, this is seldom the case. It is often the case that we witness at least some overlap between the topics in regards to the tokens and n-grams assigned to each topic. For this reason, it is necessary to take a closer look at texts with the highest topic probabilities – or topic loadings – per topic. These can be considered as typical cases for each of the topics extracted.

A matrix, or a Pandas data frame, that contains the document-topic distributions is similar to Table 17.2. In our case, we see three artificial documents with the respective topic distributions assigned by the LDA algorithm. Each document consists of a mixture of topics, but the distributions vary greatly. While document D1 focuses mainly on topic 0, it also yields a fair share of tokens associated with topics 1 and 4. Based on our initial interpretation of the tokens, we could assume that this document is mainly concerned with Habitus-Field theory, but uses a qualitative-discourse analysis to investigate a phenomenon. The second document D2 does not have a clear propensity towards a topic. Finally, D3 clearly deals with topic modeling, most probably using LDA. Keep in mind that the assignment of topics to documents is not purely random but guided by the belief of the LDA (prior) on how the topics might be distributed across the corpus via the tokens.

But beware, **topics** extracted from the corpus **are not necessarily human-interpretable**. Others might be human-interpretable but are only loosely related to our research questions. If, for example, our aim is to analyze the content of articles published in sociology theory journals, but the authors add acknowledgments and funding information, then the LDA algorithm will find and extract at least one topic that subsumes all tokens related to acknowledgments. Although this is a well-interpretable topic, we may discard it, as it does not add any information on the content of the articles. In both cases, we may (or may not) discard these topics as "junk" topics and refrain from interpreting them.

But how can we possibly know the number of topics to extract and to interpret qualitatively? This is one major drawback of the LDA and similar generative topic modeling algorithms. **We cannot know the number of topics to extract in advance**. Typically, we must calculate multiple topic models with a different

Table 17.2 Document-Topic Distribution of Three Documents and Five Topics

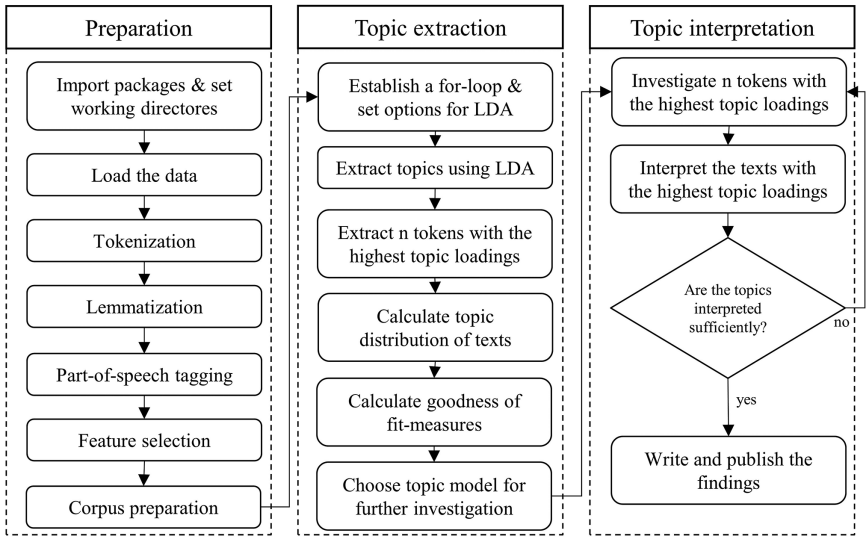
Document	*Topic 0*	*Topic 1*	*Topic 2*	*Topic 3*	*Topic 4*
D1	0.7	0.15	0	0	0.15
D2	0.2	0.3	0.1	0.2	0.2
D3	0	0	0.95	0	0.05

number of topics to find a model suitable for further investigation. In the following, we will use **K** as placeholder for the **number of topics** extracted from the text corpus, as the letter k is used in the literature to denote the number of clusters or groups extracted from data.

Our selection procedure is guided, firstly, by measures that quantify how (a) coherent the topics are and (b) how well the topics explain tokens in unseen documents that were not used during model training. While the former relates to the family of **coherence measures**, the measure for the latter is called **perplexity**. On the one hand, coherence measures calculate the extent of the semantic relations among the most important tokens of the topics extracted. A model with highly coherent topics hints that the topics extracted are human-interpretable (Röder et al., 2015). On the other hand, perplexity splits the text corpus into a training- and test set and measures how "confused" the model is in fitting topics to the test set (Hoffman et al., 2010). Confusion means that the model simply guesses (probably incorrectly) which topics to assign to an unseen document. Consequently, we seek to select a model with high coherence values and low perplexity values. Furthermore, the extracted topics as well as their coherences are also highly dependent on the size of the dictionary (= the mapping between tokens and token-ids), the lower- and upper threshold in regards to the token frequency, in addition to the tokens which are kept by part-of-speech filtering (see Section 13.5). This is why we have to be extremely careful in setting the thresholds and preprocessing steps, especially as there is no gold standard for feature selection except for lemmatization and stop word removal (Schofield, Magnusson, & Mimno, 2017; Schofield, Magnusson, Thompson, et al., 2017).

With this in mind, we will now proceed with the more technical part of this chapter and follow the workflow depicted in Figure 17.1. As in the chapters before, we begin with data preparation (Section 17.1), namely with importing the packages, setting the working directory, and load the data. We proceed with tokenization, lemmatization, and part-of-speech tagging. We then perform a feature selection by part-of-speech tags, create a dictionary with token-id mappings, and remove tokens that were used less than ten times throughout the corpus. We then continue with topic extraction and model selection in Section 17.2. In this subsection, we will show you how to apply Gensim's LDA algorithm, extract the top n-tokens per topic as well as the document-topic distributions, and how to calculate the goodness of fit-measures (coherence and perplexity). We then continue to encapsulate these steps into a for-loop, visualize the goodness of fit-measures, and select a topic model for further inspection. Finally, we will guide you through the model interpretation steps (Section 17.3) based on the token-topic assignments and a more qualitative inquiry into the texts with the highest topic loadings. We will focus on the texts that are associated with the five most prevalent tokens according to the topic distribution in our text corpus. We close this section with a conclusion and reflections on the usage of LDA.

Figure 17.1 LDA workflow.



17.2 Preprocessing

We will start our preprocessing process with importing the necessary packages (see Code box 17.1):

```
import os
import pandas as pd
import pickle
import spacy
nlp = spacy.load("en_core_web_sm")
from gensim import corpora
from gensim.models import LdaModel
from gensim.models import CoherenceModel
from matplotlib import pyplot as plt
import seaborn as sbs
```

Code box 17.1: Importing the packages and functions necessary to perform a LDA

The first three packages seen in Code box 17.1, `os`, `pandas` (imported with alias `pd`), and `pickle` are used to read and write data. While the first two are probably already known, we still need to say a few words about the last one. The `pickle` package offers us functions to serialize complex objects, such as

trained machine learning models. Such serialized objects can be stored to and imported from our hard drive into a Python session very easily. We also import `spacy` and the `en_core_web_sm` language model to perform feature extraction and feature selection procedures on token level similar to Sections 13.4.3. and 13.5. We then proceed with importing the `gensim.corpora` module, which allows us establish a dictionary and convert our corpus into a bag-of-words format. We then proceed with importing the `LdaModel` function from the `gensim.models`, which is used to extract topics from the corpus, as well as the `CoherenceModel` module, which contains the different coherence measures as well as the algorithm to calculate the `perplexity` score. We furthermore import the `pyplot` module (with alias `plt`) from `matplotlib` and `seaborn` as `sbs` for visualization purposes.

We will now define the paths where the data is stored with the help of the code seen in Code box 17.2. As you may notice, we introduce a `try` and `except` flow-control sequence which we did not cover before. More precisely, this sequence is used for **exception handling** purposes. Exception handling is used to capture errors and to tell Python how to proceed when an error occurs. In our case, an error could occur, when we try to create a folder on our hard drive using the `os.makedirs()` function. If, for example, the folder assigned to `model_out` exists on our hard drive, then `os.makedirs()` will cause an `FileExistsError`. This is why we tell Python to try to create the folder, and if the folder exist, to continue the execution of the program code with the `pass` function. Adding this flow control sequence makes Python to change the working

```
root = "C:\\Users\\[YOUR_PATH]\\"
path = root + "Data\\"
model_out = root + "LDAmodels\\"
gof_path = root + "gof\\"
output = root + "output\\"
graphics = root + "graphics\\"
corpora = root + "Corpus\\"
try:
     os.makedirs(model_out)
     os.makedirs(gof_path)
     os.makedirs(output)
     os.makedirs(graphics)
     os.makedirs(corpora)
except:
     pass
```

Code box 17.2: Creating the paths for storing the outputs of the LDA using a try and except flow control sequence

directory using the `chdir()` function in case the folders are created or already exist on our hard drive.

We will now import the dataset using the Pandas `read_csv()` function as seen in Code box Code box 17.3. This specific data frame is a subset of the social sciences data frame introduced in Section 14.1 and includes more than 400,000 comments posted under YouTube videos from the domains of sociology, psychology, and economics, as well as the video metadata:

```
os.chdir(path)
df = pd.read_csv("comments_sociology_psychology_economics_lda.
csv", sep = ";")
```

Code box 17.3: Importing the data frame containing the documents and metadata using pandas read_csv()

Next, we will proceed with tokenization, lemmatization, part-of-speech tagging, and feature extraction on token level using the code demonstrated in Code box 17.4. This relates to the preprocessing steps depicted in the second to fifth boxes on the left-hand side of the workflow in Figure 17.1. To do so, we will make use of a for loop and list flattening procedure similar to Section 13.5.

We will now briefly explain the rationale of the code line by line. We start with assigning a list to `pos_to_keep` that contains the part-of-speech tags of the

```
pos_to_keep = ["NOUN", "ADJ", "ADV", "VERB", "PROPN"]
corpus_preparation = []
for (text, comment_id) in zip(df.textOriginal, df.comment_id):
    nlp_text = nlp(text)
    row = [(token.text,  token.lemma_,token.pos_,  comment_id)
          for token in nlp_text if \
    token.is_stop == False and token.pos_ in pos_to_keep]
    corpus_preparation.append(row)
corpus_preparation = [item for sublist in corpus_preparation for
item in sublist]
corpus_preparation = [item for item in corpus_preparation if
item[0].isascii()]
corpus_df = pd.DataFrame(data = corpus_preparation,
       columns = ["token", "lemma", "pos", "comment_id"])
```

Code box 17.4: Tokenization, lemmatization, part-of-speech tagging (feature extraction), and token filtering (feature selection) using spacy

tokens we wish to keep. We want to keep nouns, adjectives, adverbs, verbs, and proper nouns. In the next line, we assign an empty list to `corpus_preparation`, which serves as container for the extracted token features as well as the `comment_id` of the respective document in the `for` loop. We then initiate a for loop with the two iterators text and `comment_id` and use the `zip()` function to simultaneously iterate through the `df.textOriginal` and `df.comment_id` columns.

In the first line of the `for` loop, we tokenize the document stored in the iterator `text` using `nlp(text)` and assign it to `nlp_text`. We then use a list comprehension on `nlp_text` and extract the token string using `token.text`, the lemma with `token.lemma_`, the part-of-speech tag with `token.pos_`, as a tuple and add the `comment_id` to the tuple. We will make use of it to create a list of lemmas per document next, which serves as foundation for creating the text corpus. We also add two conditions that must be fulfilled, otherwise the extracted token-features will not be included in the list assigned to `row`. The first condition, `token.is_stop == False,` conducts stop word filtering, whereas the second one, `token.pos_ in pos_to_keep,` keeps only tokens whose part-of-speech is part of `pos_to_keep`. Lastly, we append the row that contains the tuple of the extracted token-level features with the `append()` function to `corpus_preparation`. This step allows us to convert `corpus_preparation` to a `Pandas data frame`, as seen in the last two lines of code.

The first of the two following list comprehensions flatten the list (as demonstrated in Section 13.5). The second checks, if the first entry of each tuple (the `token.text`) in the now flattened list of tuples consists of characters belonging to the ascii character system. This second list is the first attempt to remove non-character tokens, such as emojis, or non-Latin characters stemming from languages such as Chinese, Indian, or Indonesian.[1] Finally, we pass the filtered list of tuples in `corpus_preparation` together with a list of column names to the Pandas `DataFrame()` function and assign the novel data frame to `corpus_df`.

However, we did not finish our data preprocessing yet. In the next steps, seen in Code box 17.5, we remove all possible None-type lemmas and also lemmas with less than three characters. To check for the type of lemmas (usually a character string), we use the first list comprehension in Code box 17.5 and assign the result to `lemma_type`. The second list comprehension calculates the number of characters using the `len()` function, if the lemma is a character string, and assigns zero otherwise. The resulting list is then assigned to `token_len`. Both are then inserted as columns to `df` using the pandas `insert()` method. Finally, we use both conditions in `df[(df.lemma_type == str) & (df.no_characters >= 3)] to` remove all lemmas that are not character strings and have less than three characters.

```
lemma_type = [type(x) for x in df.lemma]
token_len = [len(x) if type(x) == str else 0 for x in df.lemma]

df.insert(0,"lemma_type",lemma_type)
df.insert(1,"no_characters",token_len)

df = df[(df.lemma_type == str) & (df.no_characters >= 3)]
```

Code box 17.5: Keeping only character string-tokens with at least three characters

After all these feature extraction and feature selection steps, we continue with establishing the text corpus itself. The `gensim` package requires us to create a corpus from a one-dimensional data structure – such as a `numpy` array, a `Pandas series object`, or a list – containing a list of tokens or a list of lemmas. The same data structure is then used to establish a dictionary to map the tokens to token-ids. However, our data is currently arranged as a two-dimensional Pandas data frame. We opt for converting it into a Pandas series object with comment IDs as index values, and a list of lemmas as values. We do so since this data structure enables us to link the text corpus to both the metadata and the document-topic distributions calculated by the LDA.

```
corpus_df = df[["comment_id", "lemma"]]
corpus_df  =  corpus_df.groupby(by  =  "comment_id")["lemma"].
apply(list)
```

Code box 17.6: Transforming the corpus data frame object into a Pandas series object with comment IDs as index names, and a list of lemmas as values

As you see in Code box 17.6, we first select the `comment_id` and `lemma` columns using the Pandas selecting with `df[[“comment_id”,”lemma”]]`. We assign this selection to `corpus_df`. We then use the `groupby()` method to group the data of `corpus_df` by the `comment_id` column. Different from previous calculations performed on grouped columns, such as counting the tokens per text, we now select the lemma column and chain the pandas `apply()` method. This method allows us to use a function on a whole `Pandas data frame`. In our case, we use the `list` function to convert the lemmas per comment into a list of lemmas. As we only pass a single column, this results in a 1-dimensional Pandas series object with `comment_id` as index names, and the lists of lemmas as values (see Output 17.1).

```
print(corpus_df)
comment_id
Ugg--Lt4XetMB3gCoAEC  [thank, Videos, help, psychology]
Ugg-07Qd2hZueXgCoAEC  [MUPPETS, MUPPETS, MUPPETS, favorite,
                      thing, j . . .
Ugg-08hjNdN8-3gCoAEC  [panic, attack, amusement, park, little,
                      talk, . . .
Ugg-0Bze6Zrk3HgCoAEC  [economic, economic]
Ugg-0GfDygUUdXgCoAEC  [wish, legalize, hallucinogen, soon,
                      study, ex . . .
UgzzzxAzBB9XtGMb0I14AaABAg    [tell, Trump, impose, tariff, com-
                              pany, suppose . . .
Ugzzzxan4qCtyD4-ich4AaABAg    [learn, phycopath, problem, solve,
                              stop, head, . . .
UgzzzxxqRgu-6M4vCUF4AaABAg    [Chad, great, nice, hear, great,
                              communication . . .
UgzzzzScrddiNLk8kSl4AaABAg    [true, pretty, woman, have, unat-
                              tractive, frie . . .
UgzzzztGsnu-i12-woJ4AaABAg    [hate, trump, liar, corrupt,
                              people]
```

Output 17.1: Excerpt from the Pandas series object with comment IDs as index names and list of lemmas as values

As demonstrated in Code box 17.7, we create the dictionary, filter rare tokens or tokens that are too common to be used to discriminate between topics, and convert our corpus into a bag-of-words format in a next step:

```
dictionary = corpora.Dictionary(corpus_df)
dictionary.filter_extremes(no_below = 10, no_above = 0.5)
bag_of_words_corpus = [dictionary.doc2bow(text) for text in
corpus_df.values]

print(bag_of_words_corpus)
print(f"Number of tokens in dictionary after cleaning:
{len(dictionary)}.")
```

Code box 17.7: Creating the token-token id dictionary, filtering rare and common tokens, and transforming the corpus into bag-of-words format

The first line of code maps the tokens to the token IDs using the `Dictionary()` function from `gensim.corpora`, and assign this mapping to `dictionary`. The function assigns a unique number (ID) to each token. For example, if we type `dictionary[203]`, Python would return "`friend`", while `dictionary[15444]` returns "`unattractive`". Finally, we use a list comprehension to translate the list of lemmas into a bag-of-words format. Within the list comprehension, you see the dictionary method `doc2bow()`, which simply is an abbreviation for (translate) **doc**uments to (**2**) a **b**ag **o**f **w**ords format. After this transformation, each document is stored as a list of tuples. Each tuple retains the unique token id at the first place, and the number of appearances are stored in the second place of each tuple. The result is displayed in Output 17.2. These tuples are ordered by token ID in ascending order, making it easier for the LDA algorithm to extract topics.

```
[...[(203,2),  (422,1),  (537,1),  (695,1),  (742,1),  (886,1),
(945,1), (1345,1), (3419,2), (3443,2), (3915,1), (15444,1)]]

Number of tokens in dictionary after cleaning: 21108.
```

Output 17.2: Text corpus converted to a bag-of-word format plus the number of tokens kept in the dictionary

```
os.chdir(corpora)
## save corpus and dictionary for future evaluation and plotting
corpus_df.to_csv("preprocessed_corpus.csv")
dictionary.save("dictionary_lda")
pickle.dump(bag_of_words_corpus,
open("bow_corpus.pickle ","wb"))
```

Code box 17.8: Saving the text document corpus, dictionary, and corpus converted to a bag-of-words format to the hard drive

The final step is to save the corpus and the dictionary. We save the 1-dimensional Pandas Series that contains the corpus using Pandas' `to_csv()` function. We then proceed to save the dictionary using the `gensim`'s built-in `save()` method, while we serialize the corpus in bag-of-words format using the pickle's `dump()` function. We pass the bag_of_words_corpus, and open("bow_corpus.pickle","wb") to make sure the corpus is saved as `bow_corpus.pickle` and tell Python to write it in binary mode, hence the `wb` in the `open()` function. This will enable us in the future to quickly read

large files into the Python workspace and save us from the necessity to do all the preprocessing once again.[2]

17.3 Topic extraction and model selection

Let us now proceed with topic extraction and model selection as depicted in the middle section of the workflow in Figure 17.1. The aim of this procedure is to identify one or a few topic models that are sufficiently complex to recognize an underlying fine-grained topic structure but, at the same time, provide us researchers with human-interpretable topics. Similar to other topic modeling methods (see Sections 15.1 and 15.2), we continue with (a) topic extraction, (b) calculating topic coherence and perplexity (goodness-of-fit), and (c) model selection for further qualitative interpretation. In the following, we begin with presenting the individual functions and methods, before we combine those in a for loop to automatize the topic modeling procedure to calculate topic models with different numbers of topics successively. After automating topic extraction via the flow control sequence, we show you how to visualize topic coherence and perplexity values, and how to select models for in-depth interpretation.

```
model = LdaModel(corpus = bag_of_words_corpus,
                 id2word = dictionary,
                 num_topics = 10)
```

Code box 17.9: Minimum example of topic extraction using a Latent Dirichlet Allocation

In the following, we use the `LdaModel()` function from `gensim.models` to extract topics from our text corpus. To execute the function without any additional options, we need to pass the *corpus*, the *dictionary*, and the *number of topics (k)* to extract from the text corpus to `LdaModel`. Consequently, this minimum example requires us to assign the corpus in bag of words format (here: `corpus = bag_of_words_corpus`); the `dictionary` variable that contains the token to token-id mappings to `id2word = dictionary`; and the number of topics (K) as an `integer` to `num_topics`. Using the code from Code box 17.9, we are now able to run the LDA and extract ten topics from the text corpus. Remember to assign the trained model to a new variable (here: `model`).

```
model = LdaModel(corpus = bag_of_words_corpus,
                     id2word = dictionary,
                     num_topics = 10,
                     chunksize = 20000,
                     alpha = 'auto',
                     eta = 'auto',
                     iterations = 100,
                     passes = 5,
                     eval_every = 5,
                     random_state = 42)
```

Code box 17.10: Using gensim's LdaModel function with additional options to extract five topics from the text corpus

We can, furthermore, adjust the topic modeling process to increase (a) its ability to extract coherent topics and (b) its replicability. For this reason, we will briefly introduce and explain some of the important options of the `LdaModel` function (see Code box 17.10).[3] We begin with the `chunksize` option. This option governs the number of documents 'shown' to the LDA-algorithm at once to calculate its perplexity. The `alpha` option represents a belief (prior) on how the topics are distributed across the documents. It requires us to either pass an array that represents our predefined belief on how the topics are distributed across the text corpus, or a topic selection strategy. In case of the code snippet depicted in Code box 17.10, we pass `alpha='auto'`, and, by doing so, telling the LDA to learn the document-topic priors from the corpus during the training process. The `eta` option represents a prior on the topic-token distribution. Again, we set `eta = 'auto'` and tell `LdaModel` to learn the topic-token priors automatically from the data.

The next option, `iterations`, sets the maximum number of times (= iterations) the algorithm is allowed to see the whole corpus to optimize the perplexity score. In contrast to `iterations`, the option `passes` defines how many times the complete corpus is presented during each iteration. In our case, we set passes to 5, meaning that for each iteration, the whole corpus will be shown five times to our model. The `eval_every` option sets the number of chunks passed to the LdaModel algorithm before the perplexity-metric is calculated. Smaller values might increase the topic coherence but, at the same time, increase calculation time considerably. In our case, we pass five batches

of 20,000 documents before calculating the perplexity metric. Finally, the `random_state = 42` option sets the initial state for the random process (such as drawing a sample of documents with the size assigned to `chunksize`) underlying the topic extraction process. This option is mainly used for reproducibility.

```
model.print_topics(num_topics = 10)
```

Code box 17.11: Gensim's print_topics() method used to extract the topic-token pairs for the ten most characteristic tokens per text

In a next step, we extract the ten, most typical tokens – also called the most *tokens with highest topic loadings* – of each topic. Similar to GDA and semantic network analysis (Section 15), we will use these tokens for topic interpretation. To extract the typical tokens per topic, we use the `print_topics()` method on the `model` variable. Per default, `print_topics()` extracts the 10 tokens with the highest topic loadings – in other words: that are most characteristic for each topic. However, without further options, this method returns the ten tokens with the highest loadings only for the ten *most prevalent topics* in our corpus. Most prevalent simply means that these topics are most probable to appear on average across the documents. We use `num_topics = 10` to extract the tokens with the highest topic loadings of the 10 most prevalent topics and, as you see in Code box 17.11, change it to `model.print_topics(num_topics = 100)` if we want to extract the topics for the 100 most prevalent topics. Setting this option is necessary if we use the LdaModel function to extract more than 10 topics and want to get a glimpse on keywords of all topics extracted. If we now run the code depicted in Code box 17.11, Python returns a list of tuples as displayed in Output 17.3. Notice that the first element of each tuple is the topic number, whereas the second element contains, firstly, floating point numbers, which represent the contribution of each token to the topic. Higher values indicate that it is more likely that a document will be assigned to a topic if a token appears (e.g., if "video" appears, it is more likely that the document is assigned to topic 0). Secondly, it lists the ten tokens with the highest topic loadings. This list of tuples can be used to generate a Pandas data frame with a column containing the topic number, and a second column that contains the most probable tokens.

```
[(0,
  '0.024*"video" + 0.016*"great" + 0.013*"read" + 0.011*"book"
+ 0.010*"study" + 0.010*"fact" + 0.009*"thank" + 0.008*"under-
stand" + 0.008*"psychology" + 0.008*"good"'),
```

Code box 17.12: Using the save-method to export our trained LDA model with k = 10 topics to our hard drive

```
 (1,
 '0.064*"man" + 0.056*"woman" + 0.017*"child" + 0.014*"want" +
0.014*"God" + 0.011*"love" + 0.010*"family" + 0.009*"relation-
ship" + 0.009*"world" + 0.009*"Africa"'),
 (2,
  '0.081*"look" + 0.043*"right" + 0.022*"eye" + 0.021*"bro" +
0.015*"stare" + 0.013*"face" + 0.013*"brain" + 0.013*"funny" +
0.012*"hand" + 0.012*"get"'),
 (3,
 '0.037*"money" + 0.019*"pay" + 0.015*"debt" + 0.012*"government"
+ 0.010*"price" + 0.010*"bank" + 0.010*"inflation" + 0.009*"mar-
ket" + 0.009*"tax" + 0.008*"rate"'),
 (4,
 '0.037*"people" + 0.018*"think" + 0.013*"feel" + 0.012*"thing"
+ 0.012*"know" + 0.011*"person" + 0.009*"life" + 0.009*"way" +
0.008*"experiment" + 0.007*"want"'),
 (5,
 '0.051*"India" + 0.026*"answer" + 0.025*"question" + 0.024*"Trump"
+ 0.015*"trump" + 0.013*"indian" + 0.012*"vote" + 0.011*"win" +
0.011*"game" + 0.009*"white"'),
 (6,
  '0.054*"thank" + 0.047*"sir" + 0.033*"Sir" + 0.032*"class" +
0.022*"video" + 0.017*"good" + 0.016*"hai" + 0.013*"student" +
0.013*"teacher" + 0.012*"sociology"'),
 (7,
  '0.018*"year" + 0.018*"work" + 0.016*"life" + 0.014*"day" +
0.014*"live" + 0.013*"time" + 0.013*"food" + 0.008*"good" +
0.008*"Japan" + 0.006*"home"'),
 (8,
  '0.035*"video" + 0.024*"love" + 0.022*"watch" + 0.019*"talk"
+ 0.019*"know" + 0.018*"guy" + 0.018*"time" + 0.017*"good" +
0.013*"say" + 0.013*"get"'),
 (9,
 '0.021*"country" + 0.015*"world" + 0.014*"economy" + 0.013*"peo-
ple" + 0.012*"China" + 0.008*"economic" + 0.008*"system" +
0.007*"government" + 0.006*"problem" + 0.006*"poor"')]
```

Code box 17.12: (Continued)

```
model.save("10_LDA_model")
```

Output 17.3: The ten most characteristic tokens assigned to the topics by the LDA.

To save the trained model on our hard drive, we use Gensim's `save()` method on `model` (see Code box 17.12) and enter a name (here: "`10_LDA_model`").

```
perplexity = model.log_perplexity(bag_of_words_corpus)
print(perplexity)
```

Code box 17.13: Calculating log perplexity of the topic model

```
-8.1696483
```

Output 17.4: Logarithm of the perplexity value calculated for a model with k = 10 topics

After saving our trained model with K = 10 topics, we calculate the `perplexity` and `c_v` coherence values. Remember that both enable us to gauge how well our model extracted the topics and how well it potentially assigns these topics to previously unseen documents. We begin with calculating the perplexity metric (see Code box 17.13), for which `gensim` provides us with the method `log_perplexity()`. The only variable we need to pass to calculate the perplexity value is the text corpus, in our case the `bag_of_words_corpus`. Output 17.4 shows the corresponding perplexity score.

```
coherence = CoherenceModel(model = model,
                           texts = corpus,
                           dictionary = dictionary,
                           coherence = 'c_v',
                           processes = 1)
print(coherence)
```

Code box 17.14: Establishing the c_v coehence values using the gensim's CoherenceModel() function

In a next step, we calculate the topic coherence. However, there are multiple coherence metrics available, such as `pmi` (pointwise mutual information), `c_v`, `c_uci`, and `u_mass`. Nonetheless, we decided to calculate the `c_v` coherence, since it yields the highest correlation with coherence ratings of extracted topics given by human annotators (Röder et al., 2015). This measure uses a sliding window approach (e.g., 10 tokens prior and after a token under consideration) and calculates the probability that two tokens co-occur across documents

Table 17.3 c_v coherence values per Topic for a topic model with k = 10 topics

Topic	*c_v Coherence Value*
0	0.574456843
1	0.729902656
2	0.594734958
3	0.870263859
4	0.772394196
5	0.429378259
6	0.581974944
7	0.729811573
8	0.644018462
9	0.754094837

within this window.[4] We use the `CoherenceModel()` function from `gensim.models` to calculate the c_v coherence value (see Code box 17.14). To do so, we firstly pass the trained LDA-model using the `model = model` option. We then pass the text corpus with `texts = corpus`, and our dictionary with token – token-id mappings using the `dictionary` option. To define which coherence metric to calculate, we pass the `coherence = 'c_v'` option. Finally, the `processes = 1` option defines the number of processors to use for calculation. Usually, Python tries to use all available processors of your CPU to calculate the coherence value. But beware if you use the Microsoft Windows operating system. In Windows, using multiple cores at once for calculation causes Python to freeze sometimes without returning an error message. We can avoid freezing by telling the `CoherenceModel()` function to use just a single core for calculation. This may take longer, but reliably results in a list of coherence values. We transformed this list to Table 17.3 for convenience.

However, calculating the topic coherences and perplexity score for a single topic model is not sufficient to choose the best model for further investigation. Especially when we consider that perplexity is a relative value without meaning on its own. However, like other measures such as the log likelihood in regression frameworks, lower values equate with better generalizability. Furthermore, perplexity is highly contingent upon the number of documents and the number of tokens included in the analysis. This is why we cannot use this measure to compare the goodness of fit of topic models trained on different corpora. We will address this issue later after we run topic models with different numbers of topics (K) and calculate perplexity scores and topic coherences for each model.

The next step is to calculate the document-topic distributions. To do so, we call `model.get_document_topics(bow=bag_of_word_corpus)`. The option enclosed in this method requires us to pass a corpus in a bag of words-format (hence the `bow`). This results in a `gensim.interfaces.TransformedCorpus`, which is basically an iterable object that provides the topic distribution per document on demand. Consequently, to extract the topic distribution per text, we

employ a list comprehension and assign the resulting list of topic distributions to a new variable we call `topics_per_document (see Code box 17.15)`.

```
topics_per_document = [x for x in model.get_document_topics(bow =
bag_of_words_corpus)]
print(topics_per_document[0])
```

Code box 17.15: Extracting the topic propensities per text using gensim's get_document_topics() method and a list comprehension

Output 17.5 demonstrates how the document-topic distribution is structured per document using the first document stored in `topics_per_document` as an example. As you see, this is a list of tuples, which can be used as input for a Pandas `DataFrame()` function call. The first element of each tuple is the topic number, potentially from 0 to k, and the second element is the topic propensity. In the exemplary document, we see a relatively balanced topic distribution, meaning that the document contains tokens that are indicative of all 10 topics extracted from the text corpus. Keep in mind that this is just the first element of `topics_per_document`, meaning that extracting the topics per document returns a list of lists of tuples.

```
[(0, 0.111118615),
 (1, 0.07020954),
 (2, 0.07235005),
 (3, 0.08082595),
 (4, 0.13121334),
 (5, 0.06601424),
 (6, 0.0789223),
 (7, 0.08633132),
 (8, 0.17926726),
 (9, 0.123747416)]
```

Output 17.5: Topics distribution extracted from the first document included in our text corpus

In order to calculate the propensity of the topics across all documents and to merge the document-topic distributions with the metadata currently stored in `df`, we must convert the list of lists of tuples assigned to `topics_per_document` into a single Pandas data frame. Calling the `DataFrame()` function on a list of lists of tuples would result in the tuples being the content of the cells of the returned data frame. This is why we must iterate through each document-topic

distribution stored within `topics_per_document`, turn each into a data frame, append it to a list, concatenate the list of data frames, and arrange the topics (in columns) in ascending order from 1 to k. We use the procedure depicted in Code box 17.16 to achieve this.

```
topics_per_document_df = []
for doc in topics_per_document:
        topics_doc = pd.DataFrame(data = [x[1] for x in doc],
        index = [x[0] for x in doc]).T
        topics_per_document_df.append(topics_doc)

topics_per_document_df = pd.concat(topics_per_document_df)
topics_per_document_df = topics_per_document_df[range(k)]
```

Code box 17.16: Using a for loop to convert the list of the text and topic-distribution pairs into a Pandas data frame

As you have seen many times before in this book by now, we begin with assigning an empty list to a variable, here: `topics_per_document_df`. We then initiate a for loop – with an iterator named doc – to iterate through each entry of `topics_per_document`. In the subsequent indented lines of code, we assign the second element of each tuple as list as data to the Pandas data frame. A list is interpreted as single data column. Furthermore, we assign the topic number, stored in the first element of each tuple, to the index. The Pandas `T` method transposes the resulting data frame, meaning that it flips the columns and rows. This procedure results in a data frame with a single row that contains the topic distribution of the document, which is then assigned to `topics_doc`. We then append `topics_doc` to `topics_per_document_df` by calling Pandas' `append()` method. After the loop has finished iterating, we concatenate all data frames appended to `topics_per_document_df` using the pandas `concat()` function and rearrange the columns using the Pandas indexer and pass `range(k)` to sort topics from 0 to k (in our case 10).

```
topics_propensities = topics_per_document_df.mean().\
                      reset_index(name = "propensities").\
                      rename(columns = {"index" : "k"})
```

Code box 17.17: Calculating the average topic propensities for a model with k =10 topics

We will now calculate the average topic propensities across the text corpus. As demonstrated in Code box 17.17, we use the Pandas `mean()` method to

Table 17.4 Topic Propensities for a Model With k = 10 Topics

k	*Propensities*
0	0.1182489
1	0.07376764
2	0.0754674
3	0.094093315
4	0.14693831
5	0.069315106
6	0.12431303
7	0.088459484
8	0.1933405
9	0.14182308

calculate the average topic propensities and return them as a Pandas series object. We then chain the `reset_index()` and method and pass the `name =` "`propensities`" option to rename the column that stores the topic propensities from value to propensities. Finally, we chain the `rename()` method and rename the index to `k`, indicating the number of the topics extracted from the corpus. This results in the average topic distribution listed in Table 17.4.

We are now prepared to combine all the functions and methods explained so far to extract topics from the text corpus. But before we initiate a for loop control sequence, we define the hyperparameters that should govern model training (Code box 17.18). Furthermore, we change the working directory to where we

```
ks = range(10, 101, 5) # number of topics (k) that should be
extracted from the corpus
number_of_documents_for_training = 20000 #chunk of documents to
be shown at once
passes = 5 # times the complete corpus is shown to the LDA for
each iteration
iterations = 100 # number of times the corpus is shown to the
LDA algorithm
eval_every = 5 # number of chunks to be shown before evaluation.
os.chdir(model_out)
topics_per_document_and_model = []
overall_topic_propensities = []
goodness_of_fit = []
```

Code box 17.18: Defining the hyperparameters for model training, including the number of topics to extract, the chunk size, the number of passes, iterations, and the number of chunks passed before calculating the perplexity score

want to save the model output and assign three empty lists to variables that will store the document-topic distribution (`topics_per_document_and_model`), the average topic propensities per model (`overall_topic_propensities`), and the perplexity scores as well as the topic coherences (`goodness_of_fit`).

We then initiate the for loop and extract the given K number of topics from the text corpus. As you see in Code box 17.19, we basically use the same functions and methods introduced before, but we also add `print()` functions and comment code snippets that were not described in depth in this section. We use comments to explain the additional lines of code to you. Beware, the following Code box 17.19 contains the longest code snippet present in this book, but remember that you can break it down into different subsections that address a part of the topic modeling procedure explained previously.

```
for k in ks:
    print(f"Currently fitting model with k = {k} topics.")
    model = LdaModel(corpus = bag_of_words_corpus,
                 id2word = dictionary,
                 num_topics = k,
                 chunksize = number_of_documents_for_training,
                 alpha = 'auto',
                 eta = 'auto',
                 iterations = iterations,
                 passes = passes,
                 eval_every = eval_every,
                 random_state = 42)
    model.save(f"{k}_LDA_model")

    print("currently extracting top tokens per topic")
    top_tokens = model.print_topics(num_topics = k)
    top_tokens = pd.DataFrame(top_tokens, columns = ["k", "tokens"])
    # insert a column to top_topkens that contains the number of
    topics extracted from
    # the topic model
    top_tokens.insert(0, "num_topics", k)

    print("Assigning topic propensities per text")
    topics_per_document = [x for x in model.\
       get_document_topics(bow = bag_of_words_corpus)]
```

Code box 17.19: For loop for extracting 10 to 100 topics (in five topic steps each), extracting the document-topic distribution, the tokens with highest topic loadings, and calculating c_v coherence score, perplexity score, and average topic propensities

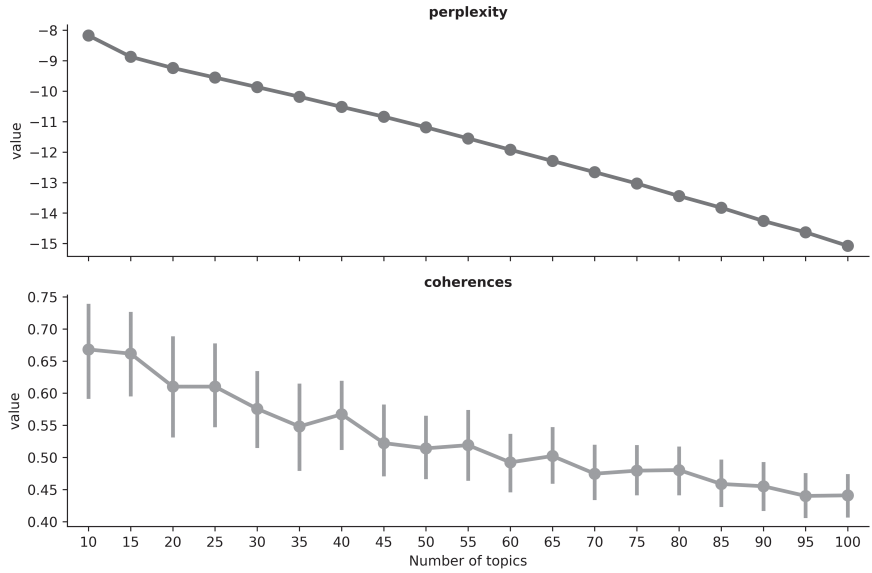
```
 topics_per_document_df = []
 for doc in topics_per_document:
     topics_doc = pd.DataFrame(data = [x[1] for x in doc],
                               index = [x[0] for x in doc]).T

      topics_per_document_df.append(topics_doc)
 topics_per_document_df = pd.concat(topics_per_document_df)
 topics_per_document_df = topics_per_document_df[range(k)]
 topics_per_document_df.index = corpus_df.index

 print("Calculating Topic Propensity")
 topics_propensities = topics_per_document_df.mean().\
                           reset_index(name = "propensities").\
                           rename(columns = {"index" : "k"})
 # we insert the number of extracted topics
 topics_propensities.insert(0, "num_topics", k)
 # append the average topic propensities to the list containing
the average topic
 # propensities for each model.
 overall_topic_propensities.append(topics_propensities)

 print("selecting the top texts per topic")
 highest_loadings = []
 for topic_num in range(k):
    highest_loading_per_topic = topics_per_document_df.\
    sort_values(by = topic_num, ascending = False).head(n = 10)
 highest_loading_per_topic.insert(0, "k", topic_num)
 highest_loadings.append(highest_loading_per_topic)

 highest_loadings = pd.concat(highest_loadings)

 # merge the data frame with the texts with the highest topic
loadings with df
 # that contains the metadata.
 highest_loadings = pd.merge(highest_loadings, df,
         how = "left",
         left_index = True,
         right_on = "comment_id")
```

Code box 17.19: (Continued)

```
 print(f"calculating perplexity and coherence for model with {k}
topics.")
 perplexity = model.log_perplexity(bag_of_words_corpus)
 coherence = CoherenceModel(model = model,
                texts = corpus_df.values,
                dictionary = dictionary,
                coherence = 'c_v',
                processes = 1)

 coherences = coherence.get_coherence_per_topic()

 # establish a data frame where coherence scores and perplexity
scores are stored
 gof = pd.DataFrame(coherences, columns = ["coherences"])
 gof.insert(0, "topic_num", k)
 gof.insert(1, "k", range(k))
 gof.insert(2, "perplexity", perplexity)

 # append the data frame with the goodness of fit measures for
the current model
 # to the list where the coherence values and perplexity scores
per model are stored.
 goodness_of_fit.append(gof)

 print("saving topics per document and top topics")
 top_tokens.to_csv(f"top_tokens_{k}.csv", sep =";")
 topics_per_document_df.to_csv(f"topics_per_document_{k}.csv",
 sep =";")
        highest_loadings.to_csv(f"top_texts_per_topic_{k}.csv",
sep =";")
# concatenate and save the data frames regarding the document-topic
distributions and
# the coherence values as well as the perplexity scores.
overall_topic_propensities   =   pd.concat(overall_topic_propen-
sities)
overall_topic_propensities.to_csv("overall_topic_propensities.
csv", sep =";")
gof_df = pd.concat(goodness_of_fit)
gof_df.to_csv("gof.csv", sep = ";")
```

Code box 17.19: (Continued)

Figure 17.2 Perplexity und c_v coherence values per topic.




```
os.chdir(graphics)
gof_long = gof_df.melt(value_vars = ["perplexity", "coherences"],
               id_vars = "topic_num")
g = sbs.FacetGrid(data = gof_long,
       col = "variable", hue = "variable",
       sharey = False, sharex = True, col_wrap=1, aspect=3)
g.map(sbs.pointplot, "topic_num", "value")
g.set_titles("{col_name}", fontweight = "bold")
g.set_axis_labels(x_var = "Number of topics")
plt.savefig("goodness_of_fit.png", dpi = 600)
plt.close()
```

Code box 17.20: Visualization of the perplexity and c_v coherence scores for $10 \leq \boldsymbol{k} \leq 100$ in steps of 5 topics

Finally, we will visualize the perplexity scores and coherence values for each model with different k topics as depicted in Figure 17.2. To do so, we will use the code presented in Code box 17.20. As you see, we again change the working directory to a folder where we will store the generated figures. You will also notice that we use the Pandas `melt()` method on the data frame that stores the coherence scores and perplexity values. Remember that `melt()` pivots the

Table 17.5 Perplexity Scores and Coherence Values Per Topic and Topic Model with Different ks in Long Format

Topic_num	*K*	*Variable*	*Value*
10	0	perplexity	−8.16964825
10	0	Coherence	0.5744568433517
10	1	Coherence	0.729902655509908
...	...	...	...
100	100	Coherence	0.766808344649101

data frame into the following structure (also called long-format) depicted in Table 17.5.

We then use Seaborn's `FacetGrid()` function to assign the goodness of fit measures to a variable named g (short for graph) and map the Seaborn `pointplot()` onto the facets similar to Code box 17.21 in Section 13.5. We then change the titles and X-axis labels using the `set_titles()` and `set_axis_labels()` respectively.

Based on these measures, which model or models should we choose for further qualitative evaluation? First of all, we observe that both the perplexity scores and coherence values depicted in Figure 17.2 decrease with increasing number of topics k. With regards to perplexity, this is no surprise, as an increase in k means also an increase in the number of parameters our model could use for model optimization and thus for topic prediction on the unseen part of the text corpus. The decrease in c_v coherence is more surprising. Usually, the c_v coherence follows a bell-shaped curve, meaning that it increases, yields an optimal number of topics, before decreasing again. In our case, we see a decrease from 10 topics onwards, which, in combination with the perplexity scores, indicates potential overfitting on our data. Nonetheless, for $10 \leq k \leq 25$, the coherence scores are decent (at least 0.6), of which perplexity is lowest for $k = 25$. Against this backdrop, we opt to investigate a topic model with 25 topics further. However, please keep in mind that we usually investigate more than one topic model, as some topics extracted might not be human-interpretable. If you think that topic coherence values are too low, you should consider adding additional preprocessing steps, for example n-gram concatenation and removing the tokens that are assigned as top tokens to at least two topics. This could further increase topic coherence and might allow your model to extract more meaningful topics.

17.4 Topic interpretation

Let us now begin with the interpretation of the extracted topics. For educational purposes, we did not analyze all 25 topics. Instead, we focus on the five, most prevalent topics extracted from the text corpus: topics 0, 7, 13, 18, and 24. The individual prevalences are displayed in Table 17.6. Remember that albeit we

termed the following analysis "qualitative", it is still different distant reading from qualitative content analysis in Sections 8.

Let us now take a closer look at the most characteristic tokens per topic (Table 17.7) and at the three texts of the five most prevalent topics (highlighted in gray):

Table 17.6 Average Document-Topic Propensities for a Model With k = 25 Topics

Topic	*Propensity*
0	0.093845904
1	0.040357944
2	0.050843474
3	0.060788713
4	0.064061075
5	0.047816575
6	0.059235483
7	0.07125024
8	0.05883425
9	0.059542567
10	0.03928494
11	0.0426206
12	0.041216638
13	0.074675925
14	0.04140613
15	0.049627073
16	0.045732066
17	0.05043862
18	0.07671855
19	0.043050434
20	0.057427913
21	0.05124658
22	0.054305654
22	0.05315659
23	0.0658086
24	0.093845904

Table 17.7 The Ten Most Characteristic Tokens Per Topic for a Model With k = 25 Topics

Topic	*Most characteric tokens per topic*
0	0.154*"video" + 0.053*"watch" + 0.039*"thank" + 0.039*"Sir" + 0.023*"book" + 0.021*"good" + 0.018*"channel" + 0.017*"understand" + 0.017*"read" + 0.014*"amazing"
1	0.060*"child" + 0.045*"God" + 0.026*"care" + 0.022*"send" + 0.022*"kid" + 0.019*"evil" + 0.019*"parent" + 0.017*"bear" + 0.016*"god" + 0.014*"family"

(*Continued*)

Table 17.7 (Continued)

Topic	*Most characteric tokens per topic*
2	0.130*"good" + 0.098*"right" + 0.080*"get" + 0.050*"wrong" + 0.031*"explain" + 0.018*"correct" + 0.017*"say" + 0.014*"way" + 0.014*"thing" + 0.012*"exactly"
3	0.084*"money" + 0.031*"debt" + 0.023*"government" + 0.023*"bank" + 0.021*"interest" + 0.012*"credit" + 0.012*"pay" + 0.011*"create" + 0.011*"currency" + 0.011*"dollar"
4	0.041*"people" + 0.032*"think" + 0.025*"experiment" + 0.020*"person" + 0.015*"group" + 0.013*"power" + 0.012*"act" + 0.011*"know" + 0.011*"different" + 0.010*"team"
5	0.063*"lol" + 0.053*"Trump" + 0.039*"lie" + 0.033*"vote" + 0.032*"trump" + 0.024*"china" + 0.023*"laugh" + 0.021*"joke" + 0.021*"white" + 0.020*"black"
6	0.056*"class" + 0.047*"study" + 0.046*"psychology" + 0.030*"school" + 0.026*"science" + 0.026*"student" + 0.023*"true" + 0.022*"teach" + 0.021*"teacher" + 0.019*"sociology"
7	0.046*"life" + 0.040*"love" + 0.031*"feel" + 0.015*"people" + 0.014*"self" + 0.014*"way" + 0.013*"live" + 0.013*"relationship" + 0.012*"thing" + 0.011*"know"
8	0.143*"thank" + 0.126*"sir" + 0.092*"talk" + 0.054*"nice" + 0.028*"interview" + 0.026*"help" + 0.025*"fast" + 0.021*"good" + 0.019*"class" + 0.019*"lot"
9	0.032*"economic" + 0.025*"system" + 0.015*"market" + 0.015*"government" + 0.014*"state" + 0.014*"economy" + 0.013*"free" + 0.012*"business" + 0.012*"capitalism" + 0.008*"model"
10	0.072*"call" + 0.054*"wait" + 0.034*"future" + 0.029*"definitely" + 0.029*"learn" + 0.021*"music" + 0.019*"sign" + 0.019*"cool" + 0.018*"phone" + 0.018*"background"
11	0.049*"speak" + 0.019*"totally" + 0.019*"fake" + 0.018*"character" + 0.017*"sorry" + 0.016*"dumb" + 0.015*"line" + 0.015*"check" + 0.014*"matter" + 0.014*"trust"
12	0.116*"India" + 0.029*"quality" + 0.029*"indian" + 0.025*"Japan" + 0.024*"trick" + 0.023*"search" + 0.018*"paper" + 0.018*"tha" + 0.017*"development" + 0.014*"Stanford"
13	0.049*"country" + 0.040*"world" + 0.029*"China" + 0.017*"economy" + 0.013*"war" + 0.011*"collapse" + 0.010*"nation" + 0.009*"America" + 0.008*"USA" + 0.008*"people"
14	0.073*"question" + 0.052*"answer" + 0.046*"ask" + 0.028*"lose" + 0.021*"trade" + 0.015*"energy" + 0.015*"source" + 0.013*"trading" + 0.013*"plan" + 0.010*"Harris"
15	0.037*"day" + 0.029*"food" + 0.016*"buy" + 0.016*"car" + 0.013*"store" + 0.013*"hour" + 0.012*"eat" + 0.012*"sell" + 0.011*"build" + 0.010*"month"

(*Continued*)

Table 17.7 Continued)

Topic	*Most characteric tokens per topic*
16	0.030*"BPD" + 0.029*"fool" + 0.028*"anxiety" + 0.020*"disorder" + 0.020*"india" + 0.019*"describe" + 0.017*"depression" + 0.017*"mental" + 0.015*"trait" + 0.015*"personality"
17	0.043*"high" + 0.039*"pay" + 0.034*"price" + 0.030*"low" + 0.028*"tax" + 0.025*"increase" + 0.025*"inflation" + 0.023*"economy" + 0.022*"cost" + 0.022*"income"
18	0.052*"year" + 0.035*"time" + 0.029*"know" + 0.025*"tell" + 0.019*"old" + 0.018*"friend" + 0.018*"happen" + 0.016*"start" + 0.016*"get" + 0.016*"try"
19	0.158*"great" + 0.048*"number" + 0.035*"guess" + 0.031*"miss" + 0.027*"smart" + 0.023*"funny" + 0.015*"seriously" + 0.014*"podcast" + 0.014*"tip" + 0.013*"episode"
20	0.060*"like" + 0.037*"say" + 0.036*"hear" + 0.033*"comment" + 0.032*"bad" + 0.029*"thing" + 0.025*"listen" + 0.022*"sound" + 0.018*"know" + 0.016*"stop"
21	0.048*"human" + 0.032*"change" + 0.023*"play" + 0.023*"game" + 0.012*"climate" + 0.011*"nature" + 0.011*"religion" + 0.008*"base" + 0.008*"exist" + 0.008*"society"
22	0.108*"look" + 0.035*"word" + 0.028*"brain" + 0.026*"eye" + 0.024*"mind" + 0.018*"stare" + 0.017*"body" + 0.015*"mean" + 0.014*"sense" + 0.014*"think"
23	0.136*"man" + 0.118*"woman" + 0.082*"guy" + 0.042*"want" + 0.029*"bro" + 0.020*"girl" + 0.017*"sex" + 0.016*"course" + 0.014*"attractive" + 0.014*"know"
24	0.064*"people" + 0.056*"work" + 0.032*"need" + 0.019*"problem" + 0.018*"job" + 0.015*"rich" + 0.015*"want" + 0.012*"good" + 0.011*"hard" + 0.009*"society"

Topic 0:

Comment 1:

Helpful Video Thanks for providing such a helpful Video and Sir you have cleared all concepts in easy manner.

Comment 2:

A reading list, or suggested further reading from all the authors mentioned would be extremely useful. Thank you for a wonderful channel!

Comment 3:

I love how accessible this channel is. All of the topics are presented in a way that is easy to understand and interesting to watch. I love it.

Topic 7:

Comment 1:

Being unattractive makes you lonely. Being lonely causes mental illness over time. Having mental illness causes more loneliness.
Being physically unattractive makes you mentally unattractive.

Comment 2:

All of these were true, once but I'm managing things better than I used to, so for anyone out there with the same feeling, please try to love yourself, I know it's hard and it feels like you can't, but there is always light somewhere and it will be there for you in whatever form that light take

Comment 3:

I thought we'd be together forever. I can't blame her for leaving. But it hurts to know she doesn't want me anymore. I hope she finds her peace. And i hope I find the strength to move on.

Topic 13:

Comment 1:

@TullyRiven "Before the 19th century there were . . . "
i know it is difficult for u to follow along, but we were comparing the difference between the 21st century and the 19th century. we r not comparing the 19th century to before the 19th century.

Comment 2:

You left out the part about the imperial armies fighting for dominance of weaker nations for thousands of years, stealing the wealth and enslaving the conquered populations, and then enforcing the "law."

Comment 3:

Americans and Europeans keep repeating that China is collapsing, and at the same time politicians in Europe and the United States are increasingly trying to contain China's advance. If China were to collapse, it didn't need the American effort to contain the Chinese advance.

Topic 18:

Comment 1:

Personally, I like Kamala's face more than Trump's. I'm 70 and I hope to outlive him because..... well, everything. He divided the country more than in the Vietnam era. I just hope we can come together again, like Northern Ireland finally did. On 11/6/2024 all hell is going to break loose when he and his losers cry

Comment 2:

The first time I experienced a friend having I panic attack I was with another friend who has panic attacks. I had no idea what to do so I'm glad my other friend was there to help.

Comment 3:

is it normal to not be able to remember your dreams? I used to be able to when I was younger, but for the past like 6 years or more it's rare that I can remember a full scene from any one dream I have in a night for longer than thirty minutes after I wake u

Topic 24:

Comment 1:

Any company that encourages employees to purchase shares, and financially is willing to help out (like Costco) is always going to do extremely well as the majority of shares are actually owned by the workers and not by billionaire inbred families.

Comment 2:

making poor areas rich is easy. doing it in such a way that the poor people benefit from it and aren't forced to leave ... thats the hard part

Comment 3:

Basically most of the jobs now exist becoz people are lazy to do it themselves. Info is available in Internet, but people do not want to find it themselves, they just want someone to curate it for them.

The comments as well as the tokens assigned to **Topic 0** hint at appreciation towards the host of the YouTube videos, as well as the learning resources that might have been named throughout the video (comment 2). Also, there is praise for the teaching style, which is reflected in the tokens video, watch, thank, Sir, book, good, channel, understand, read, amazing. For this reason, we might name Topic 0 *appreciation of learning resources*.

Topic 7 appears to be related to romantic love and relationships. This includes comments focusing indirectly on breakups and self-love. Overall, this is an emotionally laden topic. This is also reflected in the top tokens, which include life, love, feel, people, or relationship among others. For this reason, we name this topic *(Self-)love and relationship*.

Topic 13 includes tokens such as country, world, China, economy, war, collapse, nation, America, USA, and people. Taking also a look at the comments with the highest loadings on Topic 13, it becomes clear that this topic deals with global policy and history, but also from a global perspective. Hence, we coin the topic *Global policy and global history*.

Topic 18 is more difficult to interpret, as it appears to include different topics related to remembering. The most typical tokens, such as year, time, know, tell, old clearly point towards this direction, however, friend, happen, start, get, and try seem to tell a different story, namely that the commentator might have tried something or initiated a process. A closer look at the comments reveals that the commentators indeed remember something from the past, but the events they remember are quite different. While the person who wrote comment 1 remembers the Vietnam era (he or she is 70 years old), he or she compares the current situation in the United States with the 1960s. This was a comment posted in relation to an economics video, which might also explain why political comments are present in our data. The other two describe memories of situations of psychological distress and dreams, meaning that the focus is on past events that are emotionally charged. Hence, we interpret the topic as *emotionally charged memories*.

At last, **Topic 24** is more straightforward in the sense that both the tokens and the texts with the highest topic loading relate to working conditions and the economy. The most typical tokens in regards to this topic are people, work, need, problem, job, rich, want, good, hard, and society. And indeed, the comments focus on shares, finances, companies, and wealth (or poor areas as the opposite), and work ethics (laziness in jobs). For this reason, we might interpret the topic as *working conditions, work ethics, and economic situation*.

Table 17.8 summarizes the topic interpretation per topic, as well as the most prevalent tokens associated with the five topics.

17.5 Conclusion

In the course of this chapter, we have learned how to clean our data and translate it into a bag-of-words corpus so that we can calculate a topic model (LDA). We also learned to extract token-topic loadings and document-topic distributions,

Table 17.8 Topic Interpretations Based on the Texts With Highest Topic Loadings and the Top 10 Most Characteristic Tokens Per Topic

Topic Number	*Interpretation*	*Top tokens*
0	Appreciation of learning resources	0.154*"video" + 0.053*"watch" + 0.039*"thank" + 0.039*"Sir" + 0.023*"book" + 0.021*"good" + 0.018*"channel" + 0.017*"understand" + 0.017*"read" + 0.014*"amazing"
7	(Self-)love and relationship	0.046*"life" + 0.040*"love" + 0.031*"feel" + 0.015*"people" + 0.014*"self" + 0.014*"way" + 0.013*"live" + 0.013*"relationship" + 0.012*"thing" + 0.011*"know"
13	Global policy and global history	0.049*"country" + 0.040*"world" + 0.029*"China" + 0.017*"economy" + 0.013*"war" + 0.011*"collapse" + 0.010*"nation" + 0.009*"America" + 0.008*"USA" + 0.008*"people"
18	Emotionally charged memories	0.052*"year" + 0.035*"time" + 0.029*"know" + 0.025*"tell" + 0.019*"old" + 0.018*"friend" + 0.018*"happen" + 0.016*"start" + 0.016*"get" + 0.016*"try"
24	Working conditions, work ethics, and economic situation	0.064*"people" + 0.056*"work" + 0.032*"need" + 0.019*"problem" + 0.018*"job" + 0.015*"rich" + 0.015*"want" + 0.012*"good" + 0.011*"hard" + 0.009*"society"

to calculate perplexity scores and coherence measures per topic model and how to select a model for interpretation based on these goodness-of-fit measures. Based on our prior knowledge of the extracted topics of the model (with K = 25 topics), we were able to assign a label to some topics that matched the extracted topics and the text content. This procedure and this result indicate the usefulness of LDA to comb through large amounts of text in the sense of distant reading – and to extract meaningful, albeit rather coarse topics. These can be seen as an entry point to a more in-depth interpretation, especially when one considers that each document consists of a mixture of different themes. The texts themselves, as well as the common themes, can be considered typical, prototypical, or "unexpected" in the sense of theme mixtures and thus serve as a sample for further theoretical sampling, for example. Conversely, the extracted themes and associated texts could complement qualitative studies by being used as an irritant for the interpreters and, if necessary, as a corrective for the qualitative analyses.

However, topic modeling with generative models comes also with drawbacks:

1) First, the a priori nature of the parameters and hyperparameters must be set. Even if we do have domain specific knowledge (e.g., on the academic system), we cannot fully use our knowledge to: (a) preprocess the data in all possible combinations and (b) to select an optimal model, as we do not know the exact features which are best for extracting and predicting topics.
2) Second, there are more combinations of feature values than can be covered by training data, therefore making it difficult to generalize to regions of the space for which little or no training data are present. This is called the "curse of dimensionality".
3) "Text is contextual and sequential by nature. Word order carries meaning" (Wankmüller, 2022, S. 1684), which is why topic models with a bag of words approach have problems to understand nuances in the text and may falsely assign texts to topics.
4) Following from 3), this drawback might hamper the quality of our interpretation considerably, as we cannot read all of the documents present in our text corpus and interpret them using methods of qualitative content analysis.

Nonetheless, LDA is still a valid, albeit outdated topic modeling approach that helps you to make sense of large text corpora. In the following section, we will expand on this knowledge and introduce you to BERTopic, which addresses these problems and applies a deep learning-framework to extract coherent and semantically meaningful topics.

Notes

1 Despite the fact that including different languages in our text corpus might bring additional insights, the author of this chapter refrains from doing so due to his insufficient language skills regarding Chinese etc.
2 To do so, we have to run pickle.load(open("bow_corpus.pickle", "rb")), with rb = read binary.
3 These options are not exhaustive. To see an exhaustive list of options in regards to the LdaModel algorithm see: https://radimrehurek.com/gensim/models/ldamodel.html
4 More precisely, it calculates the pointwise mutual information, which is related to the negative pointwise mutual information applied for semantic network analysis in Section 15.2.

18
TOPIC MODELING WITH BERTOPIC

18.1 Introduction

In the final chapter of the third part of our book, we move beyond relational and generative topic modeling approaches and introduce BERTopic. BERTopic employs a deep neural network to generate document embeddings and identifies topics from high-density regions within this embedding space. Similar to sentiment analysis using pretrained RoBERTa and fine-tuned DistilBERT architectures (Section 16.3 and 16.4), we will initialize a pipeline that guides you through the data preparation and data analysis process. Since we already introduced the BERT architecture in the context of sentiment analysis, we refrain from introducing it at this place.

Unlike relational and generative topic modeling approaches (such as LDA), BERTopic leverages all layers of encoded token information, including their dependencies, semantics, morphology, and grammar for topics extraction. Using all these pieces of information at once also relieves us from applying extensive preprocessing steps. However, some post-processing (e.g., stop word removal) and fine-tuning (e.g., using OpenAI's ChatGPT) are applied to increase topic interpretability after topic extraction. Moreover, BERTopic does not necessarily require us to specify the number of topics in advance. Instead, it applies advanced dimensionality reduction techniques such as Uniform Manifold Approximation and Projection (UMAP) (McInnes et al., 2018; Sainburg et al., 2021) and soft clustering techniques, such as **H**ierarchical **D**ensity **B**ased **S**patial **C**lustering of **A**pplications with **N**oise (HDBSCAN) (Campello et al., 2013; McInnes et al., 2017), by default. The latter leverages local document density in the embedding space to assign clusters, which we will illustrate with examples. While this process may initially seem abstract, it simply involves the following five steps:

1) Transfer text documents into an embedding space using a pretrained BERT model (e.g., DistilBERT).

DOI: 10.4324/9781003496397-21
This chapter has been made available under a CC-BY-NC-ND 4.0 license.

2) Reduce dimensionality (from 6 * 768 per text document to a few dimensions) to enable clustering.
3) Use distances between documents in this low-dimensional space to cluster documents based on document similarities.
4) Extract and visualize topic properties and the documents distributions within the embedding space.
5) Interpret the extracted clusters as topics.

Another advantage of BERTopic is its ability to mitigate meaning conflation (Camacho-Collados & Pilehvar, 2020, S. 2151). Meaning conflation, as often encountered in LDA and relational methods, entails that tokens which bear different meanings in different contexts might be interpreted as always having the same meaning. For example, *field* is a concept defined by Bourdieu, is a technical term found in quantum field theory, a scientific discipline, or a plot of land eligible for agricultural cultivation of crops and plants, to name a few.

There are three further advantages of BERTopic compared to LDA, including:

1) Its ability to identify a topic hierarchy which accounts for nested, fine-grained topics within more general topics. In contrast, LDA and geometric data analysis treat topics as equally general or fine-grained.
2) BERTopic allows us to model the topic development over time and provides us with keywords and n-grams that characterize each topic per time-frame in relation to the topic itself.
3) The BERTopic pipeline facilitates integration with large language models (LLMs) such as OpenAI's ChatGPT, to assist in topic interpretation and to offer an alternative perspective on the topics.

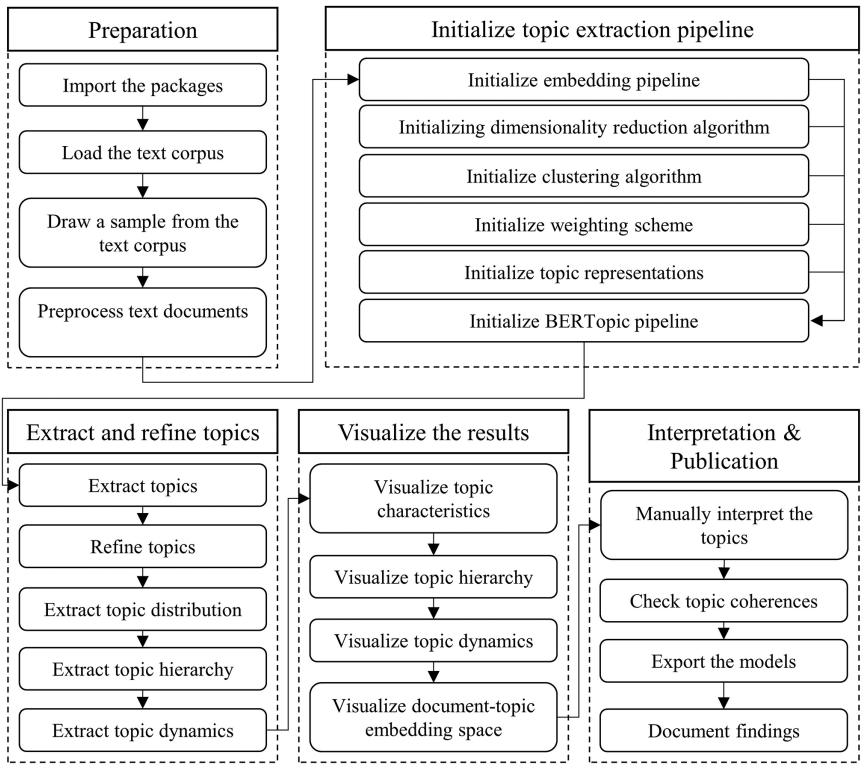
In the following, we will apply a workflow (Figure 18.1) that covers data preparation, BERTopic pipeline initialization, topic extraction, refinement, visualization, interpretation, and evaluation using coherence metrics.

To this end, the remainder of this chapter is structured as follows. We begin with setting up the BERTopic pipeline (Section 18.2), proceed with topic extraction (Section 18.3), and topic visualization at document and topic levels (Section 18.4). We then assess topic coherence (Section 18.5) and conclude with an overview on the advantages and disadvantages of using BERTopic in our research endeavors (Section 18.6).

18.2 Setting up the BERTopic pipeline

The first step to take is to load the necessary packages, prepare the data, and initialize the algorithms to be injected into the BERTopic pipeline. In the following, we will walk you through the procedure.

Figure 18.1 Workflow depicting the steps taken for topic modeling with BERTopic.



18.2.1 Loading the packages and text preprocessing

Similar to the previous chapters, we start with importing Python packages and functions that are essential for the BERTopic pipeline to work (see Code box 18.1).

First, we import `pandas` (with alias `pd`), `os`, `re`, the `tqdm` function, and `pickle` for data handling. Furthermore, we import `spacy` with its `en_core_web_sm` language model for tokenization and text preprocessing. We then import the `BERTopic` pipeline and its components. To create stable embeddings of our text corpus, we import `tensorflow` (alias `ts`) and `numpy` (alias `np`). We do so to ensure the replicability of the procedure, especially in regards to the `transformers pipeline`. Next, we import the `transformers AutoTokenizer` for text tokenization and import the `TFAutoModel` to import the pretrained `DistilBERT` model into our Python coding session.

```
import pandas as pd
import os
import re
from tqdm import tqdm
import pickle

import spacy
nlp = spacy.load("en_core_web_sm")
from bertopic import BERTopic

import tensorflow as tf

import numpy as np
from transformers.pipelines import pipeline
from transformers import AutoTokenizer
from transformers import TFAutoModel

from umap import UMAP
from hdbscan import HDBSCAN
from bertopic.vectorizers import ClassTfidfTransformer
from sklearn.feature_extraction.text import CountVectorizer
from bertopic.representation import KeyBERTInspired
import openai
from bertopic.representation import OpenAI

import seaborn as sbs
import matplotlib.pyplot as plt

from gensim.models.coherencemodel import CoherenceModel
from gensim.corpora.dictionary import Dictionary
```

Code box 18.1: Importing packages for BERTopic

Additionally, we import the `UMAP` function from `umap` for dimensionality reduction, `HDBSCAN` from `hdbscan` to extract topics from the embeddings, `sklearn`'s `CountVectorizer` to remove stop words post-hoc, the `ClassTfidfTransformer` from the `bertopic.vectorizers`, and KeyBERTInspired from `bertopic.representation` to enhance the interpretability of the extracted topics. Finally, we import `openai` and the `OpenAI` interface from `bertopic.representation` to integrate LLMs into our topic interpretation procedure.

Next, as demonstrated in Code box 18.3, we define the folder paths where our text data and according metadata is stored, as well as where the models and the resulting data tables and figures are saved:

```
root = "C:\\Users\\[YOUR_PATH]\\"
path = root + "[YOUR_DATAPATH]\\"
model_out = root + "[YOUR_MODEL_OUTPUT_PATH] \\"
output = root + "[YOUR_OUTPUT_PATH]\\"
```

Code box 18.2: Setting up folder paths for our topic modeling procedure with BERTopic

We now change the working directory to `path`, and load the `soc_sci_RoBERTa_sentiment.csv` using `pd.read_csv()` and assign the data frame to `df`. This allows us to link the sentiments assigned to the texts to the topics assigned using `BERTopic`. We then proceed to drop documents from our data frame without any text, draw a sample of 10,000 text documents with `df.sample()`, and set the `random_state =` option to replicate our sampling procedure (Code box 18.3):

```
os.chdir(path)
df = pd.read_csv("soc_sci_RoBERTa_sentiment.csv")
df.dropna(subset = "textOriginal",inplace = True)
df_sample = df.sample(n = 10000, random_state = 42)
```

Code box 18.3: Importing the text corpus and drawing a random sample of 10,000 texts

The Pandas data frame stores our text corpus – which consists of YouTube comments on sociology, psychology, and economics videos – and the following metadata (see Table 18.1).

The next step is a little trickier. Despite we don't have to preprocess the data manually to use the `BERTopic` pipeline for topic extraction, we still have to remove some tokens that do contain special characters (e.g., emojis) not encoded in the ASCII character set. We must do so to apply `sklearn`'s `CountVectorizer()` to refine the topic names and keywords after topic extraction. We achieve this by using the following code (see Code box 18.4):

```
cleaned_texts = [
        [token.text for token in nlp(text)
         if token.text.isascii() == True] \
              for text in tqdm(df_sample.textOriginal)]
cleaned_texts = [" ".join(text) for text
                 in tqdm(cleaned_texts)]
```

Code box 18.4: Removing non-ASCII tokens

Table 18.1 Overview of the Data Columns Stored in the soc_sci_RoBERTa_sentiment.csv Data Table

Data Column	*Description*
video_id	YouTube video IDentifier
RoBERTa_score	Confidence score assigned by RoBERTa
RoBERTa_label	Sentiment class assigned by the pretrained RoBERTa algorithm
sentiment_score	Sentiment polarity assigned by VADER
Sentiment_scores_categorized	Sentiment class derived from the VADER polarity score
comment_id	YouTube comment IDentifier
totalReplyCount	Number of relies to the comment
textOriginal	Author nickname (removed afterwards)
likeCound	Content of the comment
language	Publication time (format: Year-Month-Day:Hour:Minute:Second)
publishedAt	Number of likes to the comment

Let us explain the list comprehensions in Code box 18.4 in more detail. The first three lines show a nested list comprehension. The outer list comprehension, which contains `for text in tqdm(df_sample.textOriginal)`, iterates over the YouTube comments. The inner list comprehension – `[token.text for token in nlp(text) if token.text.isascii() == True]` – tokenizes the text documents, iterates through each token, and retains only tokens consisting of ASCII characters. The final list comprehension reconstructs the filtered tokens into text strings. These preprocessed text documents are assigned to `cleaned_texts` and are the basis for creating the embedding space via the DistilBERT algorithm.

But before we initialize the BERTopic pipeline itself, we may save the preprocessed text documents as pickle. We advise you to do so to save computation time and energy by using the `pickle.dump()` for saving and `pickle.load()` for reusing the preprocessed texts (Code box 18.5)[1]:

```
pickle.dump(cleaned_texts, open("cleaned_corpus.pickle ","wb"))
cleaned_texts = pickle.load(open("cleaned_corpus.pickle ","rb"))
```

Code box 18.5: Saving and loading the preprocessed text corpus using pickle.dump and pickle.load

With these steps completed, we are ready to initialize the BERTopic pipeline. This will be discussed in the next section.

18.2.2 Setting up the embedding pipeline

Now, we initialize the essential components of the BERTopic pipeline. Similar to the sentiment analysis in Section 16.2.2, we begin with configuring the embedding pipeline. The embedding pipeline consists of a pretrained DistilBERT tokenizer and a feature extractor that generates the embeddings. By injecting the Transformers pipeline as the first element into the `BERTopic` pipeline, we establish the embedding space which serves as basis for the subsequent dimensionality reduction and topic extraction steps. Let us begin with setting up the tokenizer (Code box 18.6) before initializing the pretrained DistilBERT model to translate text documents into an embedding space (Code box 18.7):

```
tf.random.set_seed(42)
np.random.seed(42)
tokenizer = AutoTokenizer.from_pretrained(
                                model ="distilbert-base-uncased",
                                return_tensors = "np",
                                truncation = True,
                                padding = 512)
```

Code box 18.6: Initialization of the transformers tokenizer

The first two lines of code ensure reproducibility by setting `tf.random.set_seed()` and `np.random.seed()`. This step addresses randomness in the initialization of the artificial neurons, Dropout layers, and the shuffling procedures prior to embedding extraction. We then initialize the pretrained BERT tokenizer, which preprocesses text documents. As before, we the `AutoTokenizer.from_pretrained()` function loads the tokenizer that truncates or pads the documents to a length of 512 tokens. Once the tokenizer is initialized, we proceed to set up the embedding pipeline (Code box 18.7):

```
model = TFAutoModel.from_pretrained("distilbert-base-uncased")
model.trainable = False

embeddings = pipeline(task = "feature-extraction",
                      return_tensors = "np",
                      framework = "tf",
                      truncation = True,
                      padding = 512,
                      tokenizer = tokenizer,
                      model = model)

print(embeddings("This is a Test"))
```

Code box 18.7: Initialization of a transformers pipeline to create an embedding space

Here, we initialize the BERT model (“distilbert-base-uncased”) and disable model training with `model.trainable = False` to prevent randomness introduced by the dropout layers.[2] Leaving dropout active would result in different embeddings each run with a huge impact on downstream dimensionality reduction and topic extraction.

We then initialize the embeddings `pipeline()` similar to Code box 16.35 in Section 16.4.1, with one key difference. Instead of setting `task = “text-classification”` as in Sentiment analysis, we set `task = “feature-extraction”`. This extracts embeddings instead of assigning (sentiment) classes to text documents. Other settings remain the same, namely that documents are truncated and padded to 512 tokens length (excluding the special beginning of sequence and end of sequencetokens). The last line in Code box 18.7 applies the pipeline test-wise on the sentence “This is a Test”, producing the tensor depicted in Output 18.1:

```
tf.Tensor(
[[[-0.156513 -0.18619666 0.05277661 . . . -0.11881153 0.0662061
    0.54701537]
  [-0.35751367 -0.648356 -0.06178995 . . . -0.3040195 0.35076845
    0.5220682 ]
  [-0.27718437 -0.44594452 0.18184265 . . . -0.09477907 -0.00757394
    0.99582815]
  [-0.28408557 -0.39167672 0.37525564 . . . -0.21505764 -0.11725163
    1.0526478 ]
  [ 0.26608288 -0.509364 -0.3180135 . . . -0.42029774 0.01444202
   -0.21489538]
  [ 0.9440616 0.01117214 -0.47139424 . . . 0.14394712 -0.72878313
   -0.16194952]]], shape=(1, 6, 768), dtype=float32)
```

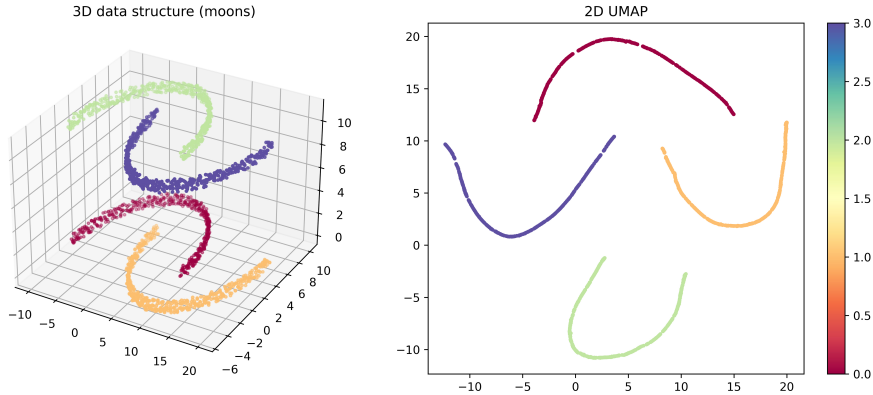
Output 18.1: Tensor representation of text embeddings

The tensor structure consists of nested lists: a corpus level-list, which contains a document level list, which, in turn, contains six token-level lists. The innermost lists contain four embeddings for the actual tokens and two (the first and last) for the special tokens indicating the beginning of a sequence and the ending of a sequence. Each list yields the 768 weights of the artificial neurons present in the dense layer of the DistilBERT model. These are averaged in the BERTopic pipeline to generate document-level embeddings. In other words, these weights represent the high-dimensional space from which we will extract the topics.

18.2.3 Dimensionality reduction

The second component of the `BERTopic` model pipeline reduces the 768-dimensional embedding space. Following Grootendorst (2022), we

Figure 18.2 Projection of three-dimensional shapes (moons) onto a two-dimensional plane using UMAP.



apply the **U**niform **M**anifold **A**pproximation and **P**rojection (UMAP) algorithm (McInnes et al., 2018) to create low-dimensional representations of the high-dimensional embedding space. UMAP assumes that abstract forms, such as curves or folded sheets, are able to approximate the relevant information of the high-dimensional space. The core assumption of UMAP is that the high-dimensional space (the 768-dimensional space derived from the DistilBERT's dense layer weights) is the product of an underlying process that generates the data. This aligns with our case, where text documents are transformed into the weights of our artificial neural network, which encode the token dependencies, semantics, part-of-speech tags, and syntax. This is much like a shadow that captures the shape of a three-dimensional body. The following Figure 18.2 captures this intuition. It projects Euclidean distances from three-dimensional shapes (four synthetically created moons) onto a two-dimensional plane.

To recognize these shapes and to approximate the distribution of data in the high-dimensional space adequately, UMAP reduced dimensionality in two steps:

1. Build a network graph which that captures the neighboring relations by linking similar text documents. The resulting lines and triangles are used to reconstruct the shapes of the high-dimensional structures.
2. Apply an artificial neural network to learn a low-dimensional representation (curves, forks, spheres, etc.) that captures the information on the graph.

In the first step, the UMAP algorithm requires a radius to calculate the distance between a given data point (= text document) and neighboring (similar) text documents. If two documents are sufficiently close, a new edge between the text documents is added. As new edges are spanned between text documents, the probability of adding new edges and extending the local "neighborhood"

decreases. Based on this, it is then calculated how likely it is that this neighborhood can be extended. If the link between the text documents in the established neighborhood and potential candidate text documents exceeds the modified threshold, then the candidates are linked to the closest data points within the already established neighborhoods.

In the second step, an artificial neural network (with SGD algorithm, see Section 11.3) is applied to learn the best, low-dimensional representation of the graph (Sainburg et al., 2021). In our case, we create a 25-dimensional embedding, which captures the information of the original 768-dimensional space. The number of extracted dimensions is arbitrary, and we recommend experimenting with different values and checking coherence scores afterwards.

We will now demonstrate how to initialize the `UMAP()` algorithm (see Code box 18.8):

```
umap_model = UMAP(n_neighbors=5,
                  n_components=25,
                  min_dist=0.0,
                  n_epochs = 200,
                  metric='cosine',
                  random_state=42)
```

Code box 18.8: Initializing UMAP for dimensionality reduction of the embedding space

We could directly pass the embedding space to UMAP(). However, we specify key parameters to align it better to our data. These parameters include the following:

- `n_neighbors` = The number of neighbors to consider while learning the manifold structure of the data. Small values emphasize local structures, large values capture the broader structures present in the data.
- `n_components` = Reduces the original number of dimensions to the defined number of dimensions (here: 25).
- `min_dist` = Controls allowed proximity of text documents in a low-dimensional space. Small values result in a finer topological structure with more fine-grained topic clusters.
- `n_epochs` = Defines the number of training iterations for the SGD algorithm.
- `metric` = Specifies the distance metric to generate the neighborhood of data points (for example Euclidean, Manhattan, or Cosine Distance).
- `random_state` = sets the state of a random number generator to ensure reproducibility.

Besides UMAP, `BERTopic` also supports different dimensionality reduction techniques we encountered in the context of geometric data analysis (Section 15.1),

namely principal component analysis (PCA). We could also apply correspondence analysis (CA) or multiple factor analysis (MFA). For now, we focus on the UMAP algorithm, as it does not require pre-specifying the number of extracted dimensions by hand like PCA, CA and MFA.

18.2.4 Initializing (soft-)clustering techniques for topic extraction

The next algorithm we introduce into the `BERTopic` pipeline is HDBSCAN (Campello et al., 2013; McInnes et al., 2017). This algorithm is used for clustering and to extract topics present in our data. Albeit we can use different clustering techniques, such as K-Means or single-linkage, we opt for HDBSCAN for several reasons. First, HDBSCAN is a soft-clustering algorithm that assigns data points – or text documents – to clusters based on local density. Each cluster corresponds to one topic, which appears as a peak in local density in the low-dimensional representation of the embedding space. Second, HDBSCAN reconstructs a hierarchical representation – the so-called cluster tree – that captures the documents similarity. It can extract branches at various hierarchical levels and keep them as topics, making it more flexible in handling regions of different densities and shapes. This flexibility renders HDBSCAN suitable for topic extraction.

But before we introduce the technical part, let us explore how the HDBSCAN algorithm works on a toy dataset consisting of 200 data points first. In principle, it:

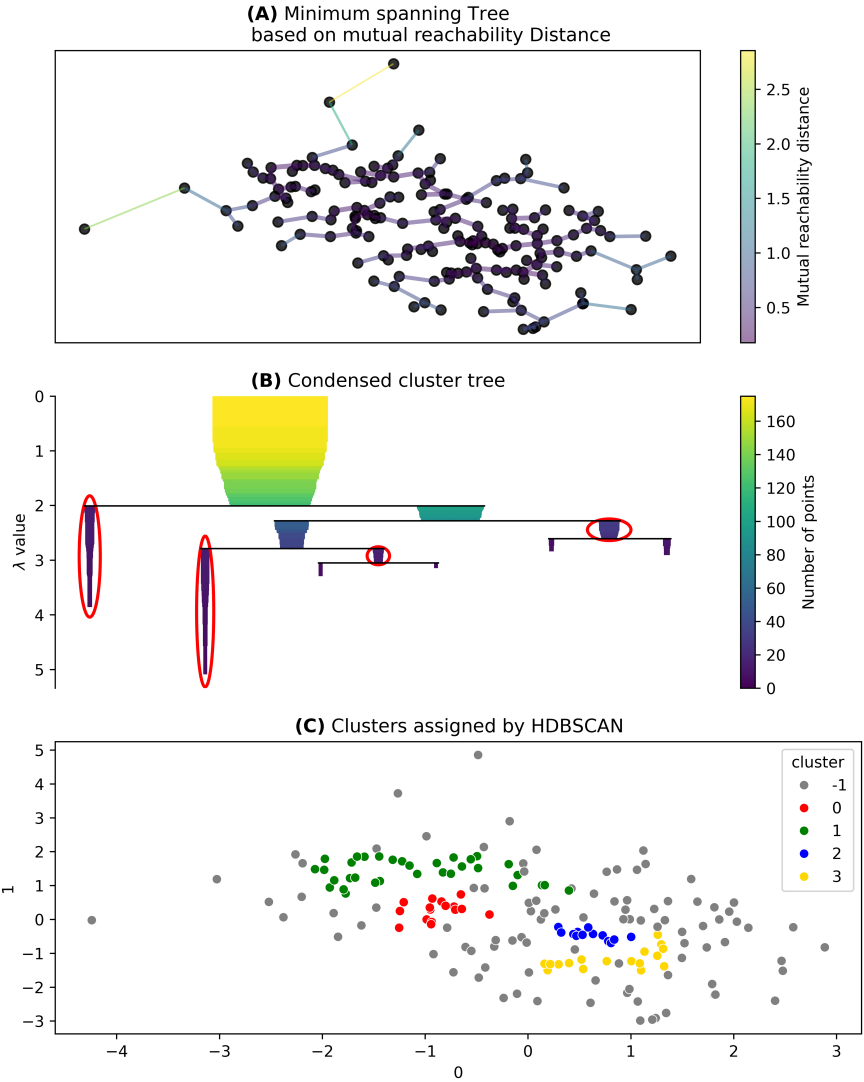
1) **Computes the mutual reachability distance** and links text documents with the shortest mutual distances iteratively. The shortest mutual distance is calculated as follows:

   a. Defining a neighborhood for each text document based on a predefined number of neighbors (e.g., 5).
   b. Extracting the maximum distance from a given text document to its farthest neighbor in regards to the neighborhood.

   Within this step, the distances are adjusted so that the text documents are pushed apart, provided that one or both documents are within the neighborhood radii. In the event of both being situated outside the radii, the unmodified distance is designated as the mutual distance.[3]
2) Repeats this step until all data points **form a connected network graph**. In our case, the text documents are, and the edges are the links between the text documents, weighted by their mutual distance. These two steps correspond to Figure 18.3, Panel A.

3) **Builds a condensed cluster tree** and, while doing so, checks the stability of the branches of this tree. This is a top-down process, meaning that we start with all data points that are assigned to a condensed cluster tree (or dendrogram) depicted on top of panel B in Figure 18.3.

Figure 18.3 Depiction of the steps taken by HDBSCAN to cluster data.



The distance between the data points – or in our case: text documents – is denoted as Lambda (λ).[4] Initially, we start with a single, loose cluster with high distances between points in relation to their mutual reachability distance. Then, the threshold value of λ is slowly increased. During this process, the number of data points assigned to the first cluster decreases, but the cluster remains largely intact. In our case, the removed data points are text documents that are very dissimilar to all other text documents and

therefore denoted as 'noise'. After reaching a critical λ-value, the cluster splits into two more cohesive subclusters. This process is repeated iteratively until all clusters dissolve.

Note that a cluster is stable, if it takes a long time until it dissolves. A 'long time' means that the distance between the λ-value where the cluster came into being and the λ-value where the cluster dissolves is high. This approach is flexible enough to discard clusters that are somewhat stable (middle and right branch of the dendrogram in Panel B), and to keep clusters of different local densities and shapes. This results in the extracted clusters seen in Figure 18.3, Panel C.

Now, let us proceed with the technical implementation of the `HDBSCAN()` algorithm (Code box 18.9):

```
hdbscan_model = HDBSCAN(min_cluster_size = 25,
                        min_samples = 10,
                        metric = 'correlation',
                        approx_min_span_tree = False,
                        n_jobs = 1,
                        prediction_data = True)
```

Code box 18.9: Initialization of HDBSCAN algorithm

Like with the UMAP algorithm before, we define the following key parameters to ensure replicability and the adequacy of the HDBSCAN algorithm for cluster extraction:

- `min_cluster_size` = Minimum number of documents per extracted cluster (here: 25).
- `min_samples` = Number of neighbors used to compute the mutual reachability distance. Smaller values extract fine-grained but potentially unstable clusters, while larger values extract stable clusters, but classify more documents as noise.
- `metric` = Distance measure for cluster extraction (e.g., Euclidean distance, cosine distance, here: '`correlation`').
- `approx_in_span_tree` = Approximates the mutual reachability-distance network graph. Setting it to `False` increases calculation time, but improves replicability.
- `n_jobs` = Number of CPU cores used for cluster detection. If set to one, only one CPU is used to assign clusters in the data. This also increases the calculation time, but avoids rounding errors and thus improves replicability.
- `prediction_data` = Stores information on how clusters were assigned. This enables us to assign previously unseen text documents to new clusters and to calculate cluster stability measures.

By adjusting the minimum cluster size, the number of neighbors, and the distance metric, we can generate and compare different cluster solutions. While we do not systematically vary these parameters here, we recommend experimenting with them to improve topic coherence.

18.2.5 Initializing a topic specific weighting scheme and topic representation algorithms

In a next step, we apply a weighting scheme to highlight topic-specific tokens. This weighting scheme is called class term frequency-inverse document frequency (C-TF-IDF, see Grootendorst, 2022). As the name and abbreviation suggest, it is an adaptation of TF-IDF introduced in Section 13.4.1. It increases the weights of tokens and n-grams that are topic-specific, while decreasing the weights of tokens and n-grams that are more common across topics. We initialize the `ClassTfidfTransformer()`[5] function and assign it to a variable as follows (Code box 18.10):

```
ctfidf_model = ClassTfidfTransformer()
```

Code box 18.10: Initialization of BERTopic's ClassTfidfTransformer

The final step is to initialize a representation model to refine the topic representations (see Code box 18.11):

```
representation_model = KeyBERTInspired()
```

Code box 18.11: Initializing the KeyBERTInspired() algorithm

The `KeyBERT-Inspired()` procedure refines the keywords and n-grams representing the topics in three steps.

1) It extracts
    a. Candidate keywords per topic based on the C-TF-IDF scores of the tokens or phrases.
    b. Representative documents per topics.
2) Then, it embeds the candidate keywords and representative documents.
3) It computes the similarity (e.g., cosine similarity) between the candidate keyword and document embeddings and selects the most relevant keywords or n-grams.

In addition to the `KeyBERTInspired()` algorithm, we can apply large language models refine topic representations. This includes OpenAI's ChatGPT, Mistral,

Zephyr, or Llama. If you have created an account and have an API key of the respective LLM beforehand, you can integrate the respective LLM. In the following, we demonstrate how to initialize ChatGPT for topic refinement (Code box 18.12). Output 18.2 shows the default prompt used for keyword fine-tuning:

```
client = openai.OpenAI(api_key = [YOUR_API_KEY])
representation_model = OpenAI(client,
                              model = "gpt-3.5-turbo",
                              chat = True,
                              delay_in_seconds = 10)

print(representation_model.default_prompt_)
```

Code box 18.12: Initializing OpenAI's ChatGPT 3.5 for topic refinement

```
'\nI have a topic that contains the following documents:
\n[DOCUMENTS]\nThe topic is described by the following keywords:
[KEYWORDS]\n\nBased on the information above, extract a short
topic label in the following format:\ntopic: <topic label>\n'
```

Output 18.2: Default prompt used by BERTopics to interact with ChatGPT

Although we use GPT-3.5-turbo, alternative models, for example alternative models available on https://platform.openai.com/docs/models (e.g., GPT-4o-mini) can be used to fine-tune the topic representations.

18.3 Topic extraction

Now it is time to initialize the `BERTopic` pipeline, to inject the previously initialized algorithm, and to extract the topics from the YouTube comments that constitute our text corpus. In the following, we demonstrate how to extract topics with the `BERTopic` pipeline, the topic hierarchy based on HDBSCAN's condensed cluster tree, and how to map the changes within topics over time.

18.3.1 Fitting the BERTopic model and extracting topics

We initialize the `BERTopic` pipeline as follows (Code box 18.13).

As you see in Code box 18.13, we start with injecting the different initialized algorithms, namely the DistilBERT embeddings to `embedding_model`, the

```
topic_model = BERTopic(embedding_model= embeddings,
                       umap_model = umap_model,
                       hdbscan_model = hdbscan_model,
                       ctfidf_model = ctfidf_model,
                       representation_model = representation_model,
                       n_gram_range = (1,2),
                       verbose = True,
                       language = "multilingual",
                       low_memory = True)
```

Code box 18.13: Initializing the BERTopic pipeline

UMAP-algorithm to `umap_model`, the HDBSCAN soft-clustering algorithm to `hdbscan_model`, the topic-specific weighting scheme to `ctfidf_model`, and the fine-tuning algorithm to `representation_model`. Besides these parts of the pipeline, we set the following options:

- `n_gram_range` = The provided tuple denotes the minimum and maximum number of tokens to be considered as n-grams for topic extraction and topic representation.
- `verbose` = tells Python to print the current state of the calculation if set to `True`.
- `language` = defines the primary language of the text documents. This information is passed to `spaCy` to perform stop word removal, lemmatization, and text normalization. If multilingual is passed, the model tries to infer the language and use an appropriate `spaCy` model for text preprocessing.
- `low_memory` = defines the memory usage behavior of BERTopic. When set to `True`, calculated embeddings will be stored in the random access memory of your computer. If set to `False`, embeddings will be repeatedly calculated. The latter increases the calculation time, but enables us to calculate embeddings for large-text corpora, which exceeds the RAM otherwise.

After we initialized the `BERTopic` pipeline and assign it to `topic_model`, we can call `fit_transform()` and pass our text corpus to extract the topics from the text documents (Code box 18.14):

```
fitted_texts = topic_model.fit_transform(cleaned_texts)
```

Code box 18.14: Extracting topics using BERTopic's fit_transform() method

After the topic extraction and the fitting is finished, we can create a Pandas data frame by calling `topic_model.get_topic_info()` and inspect the data frame afterwards (Code box 18.15):

```
topics_info = topic_model.get_topic_info()
```

Code box 18.15: Extracting a DataFrame with topic details

In our case, this results in a data frame with 58 topics, numbered from 0 to 57, as well as one residual category –1 which captures all documents identified as noise. Besides the topic numbers (`Topic`), the resulting data frame includes the number of documents assigned to this topic (`Count`), the keyword phrase that characterizes this topic (`Name`), the tokens or n-grams with the highest C-TF-IDF score per topic (`Representation`), as well as the three most typical documents for this topic (`Representative_Docs`).

Note that BERTopic assigns only a single topic to a document. We will demonstrate how to approximate topic distributions similar to LDA later in this section. The following Table 18.2 shows an excerpt from the data frame, including the noise topics, as well as topic 2 (which is assigned relatively often) and topic 57 (which appears 25 times, or the least number of times allowed to be identified as topic). Note further that we only included the first few lines of the first representative document, as they tend to be relatively long and would span across multiple pages of this book.

In its current state, our topic names and representations include stop words, making it difficult to interpret the extracted topics despite having representative documents at hand. To address this issue, we initialize sklearn's `CountVectorizer()`, which allows us to remove stop words, and to refine the topic representation:

```
vectorizer_model = CountVectorizer(stop_words = "english",
                                   ngram_range = (1, 2),
                                   min_df = 5)
```

Code box 18.16: Initializing the CountVectorizer() to refine topic representations

Here, the `CountVectorizer()` applies a list of English stop words (`stop_words`), creates a sparse document frequency matrix containing unigrams and bigrams (`ngram_range`), and includes only tokens that appear in at least five YouTube comments (`min_df`). If you wish to customize the `CountVectorizer()` function further, refer to Chapter 13.4.1 where its

Table 18.2 Excerpt From the Topic Representations Extracted by the BERTopic Pipeline

Topic	*Count*	*Name*	*Representation*	*Representative_Docs*
-1	4398	-1_the_to_and_of	['the', 'to', 'and', 'of', 'is', 'in', 'that', 'it', 'you', 'are']	["I try to limit my exposure to 1 hour at a time, particularly when it comes to television and streaming news. Most of the stories get repeated anyway, so there isn't much reason to watch it all day, most of the time ..."]
2	480	2_my_was_me_and	['my', 'was', 'me', 'and', 'to', 'it', 'that', 'have', 'but', 'the']	["I'm 19 years old, father was mostly absent and my mother suffered from crack addiction that inevitably messed her brain up. I was always considered ahead of my age, wise, smart and decisive and i've had a similar instance of the process of it's never too late no matter how old you are, i think most people wouldn't really see themselves up to a certain point where the pain is truly, i mean truly overbearing ..."]
57	25	57_of_the_in_communist	['of', 'the', 'in', 'communist', 'country', 'money', 'and', 'to', 'created', 'problem']	["One good but controversial idea for developing countries – totally unfair low tax benefits for the specific industries they wish to advance. I have a real life example for this – Ukraine. Even during the war with ruzzia but before 2022 and large scale bombings and war crimes, a certain Ukrainian industry was really debating merits of emigration vs. staying at home. The IT industry. Why? Because due to a legal but maybe not quite ethical loophole 99 % of IT employees in the Ukraine were paying 5 % tax, regardless of the gross amounts, age, education, family status etc... ."]

options are introduced. As demonstrated in Code box 18.17, we update the topic representations and extract the new topic names and keyword phrases. We also call the `topic_model.update_topic()` method, with the following parameters to create a Pandas data frame with the updated topic representations:

- `docs` = The text corpus used to train the `BERTopic()` model.
- `top_n_words` = the assigned number of tokens and n-grams which best describe the respective topic.
- vectorizer_model = the previously initialized CountVectorizer().

```
topic_model.update_topics(docs = cleaned_texts,
                          top_n_words = 10,
                          vectorizer_model = vectorizer_model)
updated_topics = topic_model.get_topic_info()
```

Code box 18.17: Applying sklearn's CountVectorizer() to update topic representations

The following Table 18.3 compares the fine-tuned tokens and suggested topic names of topics −1 (noise), 2, and 57. As you can see, these are much more intuitive, and it is easier to get a glimpse of the topic content with the updated names and keyword phrases.

If you are not satisfied with the topic representations, either (1) due to the extraction of highly similar or fragmented topics or (2) if you have an excessive number of documents assigned to the noise topic (−1), then you refine the topics further using the following two methods (Code box 18.18):

```
topic_model.reduce_topics(docs = cleaned_texts, nr_topics="auto")
topic_model_outliers=topic_model.reduce_outliers(cleaned_texts,
                                                 topics=fitted_
representations[0])
```

Code box 18.18: Methods for merging similar topics and to reassigning noise documents to topics

Table 18.3 Comparison Between the Topic Names and Representative Tokens Without and With Updated Topics for Three Exemplary Topics

	Without fine-tuning		*With fine-tuning*	
Topic #	*Suggested name*	*Top tokens*	*Suggested name*	*Top tokens*
-1	-1_the_to_and_of	['the', 'to', 'and', 'of', 'is', 'in', 'that', 'it', 'you', 'are']	-1_people_like_just_money	['people', 'like', 'just', 'money', 'world', 'trump', 'think', 'good', 'does', 'time']
2	2_my_was_me_and	['my', 'was', 'me', 'and', 'to', 'it', 'that', 'have', 'but', 'the']	2_room_like_feel_did	['room', 'like', 'feel', 'did', 'crazy', 'really', 'time', 'anxiety', 'got', 'know']
57	57_of_the_in_communist	['of', 'the', 'in', 'communist', 'country', 'money', 'and', 'to', 'created', 'problem']	57_communist_created_production_country	['communist', 'created', 'production', 'country', 'problem', 'money', 'car', 'war', 'industry', 'benefits']

The first method, `reduce_topics(),` automatically merges similar topics by applying agglomerative clustering. To do so, we pass:

- `docs` = our text corpus.
- `nr_topics` = the number of topics to extract (either determined automatically, or a predefined number by the user).

Alternatively, we can pass a list of lists with topics to merge (e.g., `[[0,1], [56,57]`). However, keep in mind that merging topics (manually or automatically) requires us to inspect and inspect the topics first and decide manually whether we want to keep them or not.

The `topic_model.reduce_outliers()` method requires us to pass the text corpus and a list with the assigned topics per document. This list is stored in the first output of the `fitted_representations` variable, whereas its second list contains the probability that the document is assigned to this respective topic. The default strategy is to calculate the cosine similarity between the C-TF-IDF representations of each document and the C-TF-IDF representations of each topic. The topic with the highest cosine similarity value is then assigned.

Additionally, we might also want to approximate the topic distributions within our text documents. We can use this distribution, for example, to compare the topics extracted with BERTopic with topics extracted with different topic modeling approaches, or directly as an input for methods of geometric data analysis. To do so, BERTopic uses a sliding window approach (e.g., five tokens) and calculates the C-TF-IDF representation per topic per window. These are aggregated to represent the topic distribution per text. This is realized as follows (Code box 18.19):

```
topics_distribution = topic_model.approximate_distribution
                      (cleaned_texts,
                      batch_size = 32,
                      window = 5)
```

Code box 18.19: Calculating the topic distribution across text documents

The `topic_model.approximate_distribution()` method needs us to pass the text corpus, the batch size (= the number of documents for which the topic distribution is calculated in parallel), as well as the window size to calculate the C-TF-IDF representations.

18.3.2 Extracting the topic hierarchy

Unlike LDA (Section 17), which treats topics as independent, BERTopic leverages HDBSCAN to extract (nested) topics based on local density. This allows

hierarchical topic modeling, based on the cluster dendrogram, the mutual distance scores, and the cluster stability. Treating the branches of the dendrogram as topics allows us to simultaneously interpret fine-grained topics and more general topics (e.g., network theory sociological theory sociology). To extract and print the topic hierarchy, we use the following two methods (Code box 18.20):

```
topic_hierarchy = BERTopic.hierarchical_topics(topic_model,
                                               docs = cleaned_texts)
topic_tree = topic_model.get_topic_tree(topic_hierarchy)
```

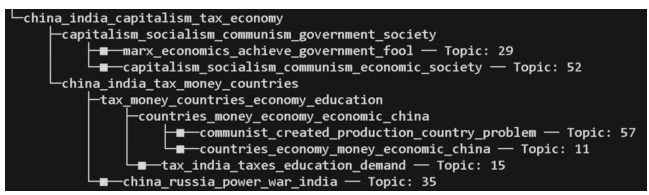
Code box 18.20: Extracting the topic hierarchy using BERTopic model

`BERTopic.hierarchical_topics()` requires us to pass the trained BERTopic model and the text corpus. It returns a Pandas data frame that contains all topic relations. The `topic_model.get_topic_tree(topic_hierarchy)` method generates a textual representation of the topic hierarchy. Figure 18.4 illustrates a branch of this hierarchy.

18.3.3 Dynamic topic modeling

In addition to extracting the hierarchical topic structure, BERTopic supports dynamic topic modeling. This approach enables us to analyze how topics evolve over time. To do so, we begin with extracting timestamps from our data and combining them into discrete time slices. For demonstration purposes, we will use years as time slices. But before doing so, let us take a look at the first five timestamps in our data frame and investigate how they are structured (Code box 18.21 and Output 18.3):

Figure 18.4 Topic branch of the BERTopic hierarchy.



```
df_sample.publishedAt.head()
```

Code box 18.21: Extracting the first timestamps of our data frame

```
635058    2024-11-02T16:38:56Z
521872    2024-08-01T08:45:04Z
714215    2017-12-17T02:49:03Z
732813    2014-09-26T01:27:33Z
872013    2021-11-12T23:13:52Z
Name: publishedAt, dtype: object
```

Output 18.3: Structure of the timestamps in our data frame

To extract the year of commenting stored in `df_sample.publishedAt`, we use a regular expression nested in a list comprehension (Code box 18.22).[6] To extract the first four digits of the time stamp (the year) we use the `"^\d{4}"` regex-pattern and pass it to the `re.search()` function. The chained `group()` method extracts the first matched pattern per string, which are assigned to the years variable:

```
years = [re.search("^\d{4}",x).group() for x in df_sample.
publishedAt]
```

Code box 18.22: Generate years as timestamp for dynamic topic extraction

We now initialize `BERTopic.topics_over_time()`, pass the trained topic model our text corpus (to docs), and the years (to timestamps), and sort the returned Pandas data frame by topic and timestamp (Code box 18.23):

```
topics_over_time = BERTopic.topics_over_time(topic_model,
      docs = cleaned_texts, timestamps = years)
topics_over_time = topics_over_time.sort_values(by = ["Topic",
"Timestamp"])
```

Code box 18.23: Extracting and assigning the topics per time frame (year)

Table 18.4 Keywords and Frequencies of the 57th Extracted Topic Over Time

Topic	*Words*	*Frequency*	*Timestamp*
57_communist_created_ production_country	investing, stock, idiots, ppl, market	1	01.01.2020
57_communist_created_ production_country	currency, education, total, nation, choose	4	01.01.2022
57_communist_created_ production_country	capitalism, country, ukraine, tax, problem	6	01.01.2023
57_communist_created_ production_country	communist, production, car, money, created	14	01.01.2024

A snapshot of the resulting data frame is presented in Table 18.4. Here, we see how the topic, labeled "57_communist_created_production_country" evolves. We skip the first timestamp, as there is only one document assigned to topic 57. In 2022, the documents are mainly characterized by currency, education, total, nation, and choose, which could relate to cultural capital in a comparative manner. In 2023, this shifts to Eastern European countries (post-communist societies), especially in regards to Ukraine and maybe related to the Russo-Ukrainian war. Finally, it shifts to potential economic needs and modes of production of (post-) communist countries.

Although this provides us with a first glimpse on how this topic (and thus, the YouTube comments) evolved over time, we might further merge the topics to interpret the global topic as well as its evolution over time in more detail. Furthermore, we could consult the document meta data to assess if properties of the speakers (e.g., in terms of location, gender, political preference) changed over time, or relate the topics to text sentiments. This rich information could, in turn, be used as guide for theoretical sampling, as qualitative codes, or databases for relational methods, such as multiple correspondence analysis, to analyze the co-evolution of the topic space and the actors therein, which relate to symbolic and objective structures of a field or social space in accordance to the habitus field-theory (Blasius et al., 2019; Bourdieu, 1984; Schmitz et al., 2017; Wieczorek et al., 2023). The connection to other theoretical approaches, such as socio-semantic network analysis (Fuhse, 2021; Roth & Cointet, 2010), is also possible.

18.4 BERTopic visualizations

The next step is to visualize topic distributions, topic hierarchies, topic developments over time, and document-topic relationships within the embedding space. To achieve this, we will use the BERTopic built-in methods to generate interactive HTML plots for in-depth inspection.[7]

18.4.1 Topic representations

The first visualization employs bar charts to capture the C-TF-IDF word scores of top tokens per topic. These charts help to identify:

1) The most influential tokens or n-grams for a predefined number of topics.
2) The distribution of the scores per topic. This distribution allows us to detect dominant tokens or n-grams that might be eligible as 'code' in the sense of qualitative content analysis.

We use the following code (Code box 18.24) to visualize and save the bar chart with the topics per word for two exemplary topics. We chose topics 9 and 57.[8]

```
fig_barchart = topic_model.visualize_barchart(topics = [9, 57],
title  = "")
fig_barchart.write_html("barchart_topic9_57.html")
```

Code box 18.24: Creating bar charts for topics 9 and 57

To generate Figure 18.5, we call `topic_model.visualize_barchart()`, assign it to `fig_bachart`, and save it as interactive HTML by calling `fig_bachart.write_html()`.[9] We can also customize the `visualize_barchart()` method using the following options:

- `top_n_topics =` Sets the number of the most frequent topics in the corpus to be plotted (e.g., 5 plots the five most frequent topics).
- `topics =` requires us to pass a list of topic numbers to be plotted.
- `n_words =` number of tokens or n-grams with the highest C-TF-IDF score per topic.
- `custom_topics =` set to True if custom labels were previously defined and should be passed as titles for the bar charts.
- `title =` defines the overall figure title. The default value is "Topic Word Scores".

Figure 18.5 Bar charts of selected topics and their highest c-tf-idf score tokens.

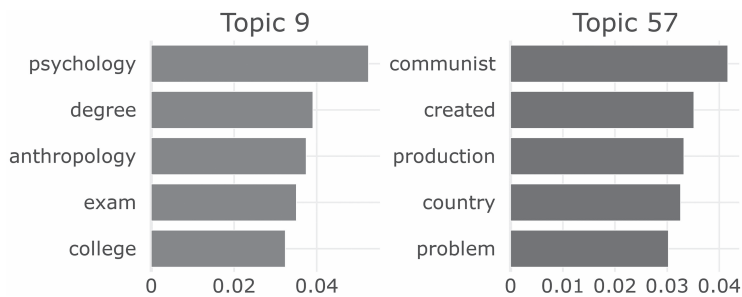
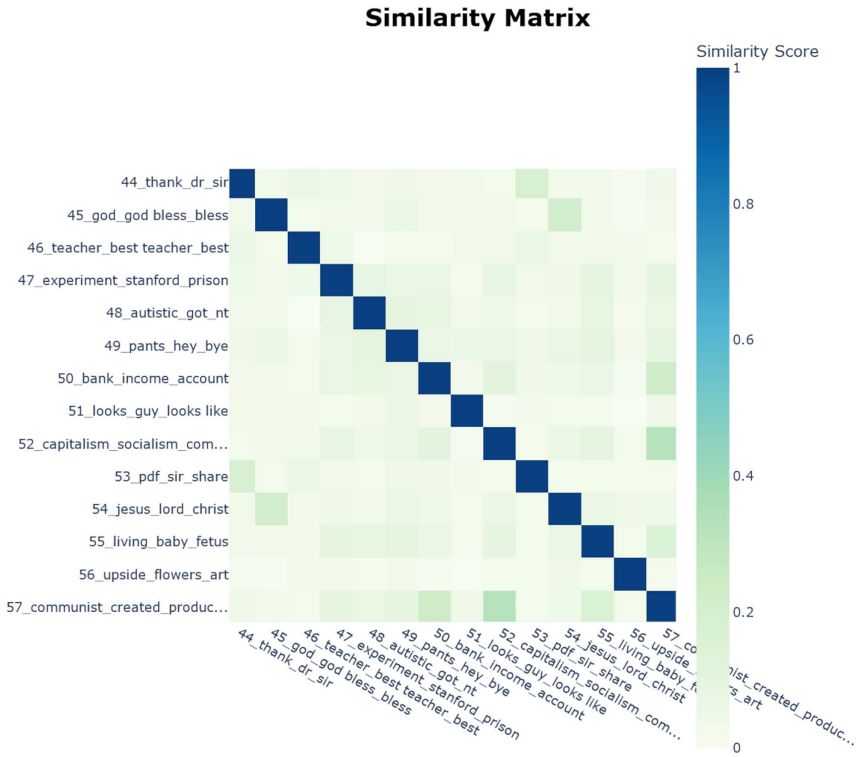


Figure 18.6 Snapshot of a heatmap of cosine similarities between topics.



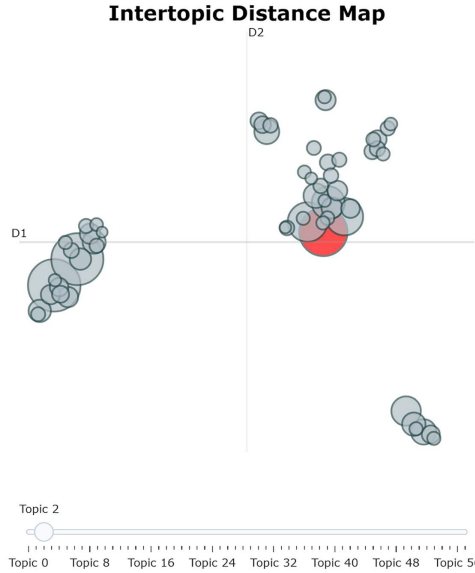
We will use similar method calls to generate and save subsequent visualizations.

In a next step, we create a heatmap that encapsulates cosine similarities between the extracted topics. Ideally, these similarities are low, as high cosine similarity values between topics may indicate redundancy or a strong overlap. If two or more topics appear redundant, we should employ techniques of qualitative content analysis to determine whether they are aspects of a single topic. If this is the case, then we can merge the topics by calling the `merge_topics()` method. BERTopic will then automatically update the topic list. Technically, we can call `topic_model.visualize_heatmap()` (see Code box 18.25) to create Figure 18.6:

```
fig = topic_model.visualize_heatmap(use_ctfidf = True)
fig.write_html("heatmap.html")
```

Code box 18.25: Creating a heatmap of cosine similarities between topics

Figure 18.7 Intertopic distance map.



As you may notice, we zoomed on a part of the heatmap. It indicates low to moderate cosine similarity values between 0 and 0.4, and suggest that the depicted topics are mostly distinct. The diagonal always equals 1, as each topic is identical with itself. You may have also noticed that we passed the `use_ctfidf = True` option. By default, BERTopic uses embeddings to calculate cosine similarities which results – in our case – higher cosine similarities. This may stem from the fact that the DistilBERT model used to generate the embedding space was trained on the English Wikipedia and the Toronto Book Corpus (Sanh et al., 2019, S. 3) and might thus not capture all of the nuances present in the YouTube comments analyzed.

Furthermore, we can visualize topic similarities by projecting them onto a two-dimensional plane. This visualization is called an intertopic distance map and is achieved by applying the UMAP algorithm to the cosine distances (calculated as 1-cosine similarity).[10] The following code (Code box 18.26) captures the code to create the intertopic distance map depicted in Figure 18.7:

```
fig_intertopic_distances = topic_model.visualize_topics(use_
ctfidf = True)
fig_intertopic_distances.write_html("intertopic_distance_map.
html")
```

Code box 18.26: Initializing the intertopic distance map with c-tf-idf weighting scheme

Complementary to the topic space, we may visualize the topic hierarchy using the visualize_hierarchy() method (Code box 18.27).

```
fig_hierarchy = topic_model.\
   visualize_hierarchy(hierarchical_topics = topic_hierarchy)
fig_hierarchy.write_html("topic_hierarchy.html")
```

Code box 18.27: Visualizing the topic hierarchy

We pass the Pandas data frame (`topic_hierarchy`) to the `hierarchichal_topics` argument. Remember that this data frame contains information on the topic hierarchy, representative keywords or n-grams of each topic, and the branches of the dendrogram. Hovering our cursor over the nodes depicted in Figure 18.8 reveals representative keyword phrases. For instance, hovering over the rightmost node of the black branch of the dendrogram displays `china_money_india_countries_economic`, suggesting that this broader topic encompasses more specific subtopics related to emerging economies. However, this interpretation is tentative, and we recommend corroborating it by qualitative analysis.

Finally, we visualize topic evolution over time. BERTopic allows us to do this in a dual sense. First, it allows us to plot the number of documents assigned to a topic per time frame (in our case years). Secondly, it allows us to depict the most characteristic tokens per time frame and topic according to the C-TF-IDF values of the tokens or n-grams. Figure 18.9 exemplifies such a plot for topics 2, 42, and 57 generated by the `topic_model.visualize_topics_over_time()` method (Code box 18.28):

```
fig_topics_over_time = topic_model.visualize_topics_over_time
(topics_over_time,
 topics = [2, 42, 57])
fig_topics_over_time.write_html("topics_over_time_2_42_57.html")
```

Code box 18.28: Visualizing topic evolution over time for topics 2, 42, and 56

To generate this figure, we pass the Pandas data frame where the topics, their appearances, timestamps, and the keyword phrases are stored. We furthermore use the `topics =` option to pass the topics 2, 42, and 57 as a list. Otherwise, we could pass `top_n =` to depict the N (e.g., five) most frequent topics across the corpus. To interpret these trends, we recommend extracting documents with the highest topic probabilities per time frame and to conduct a qualitative content analysis. This could help uncover topic transitions and their relationship to evolving discourses.

Figure 18.8 Cluster dendrogram of the topic hierarchy.

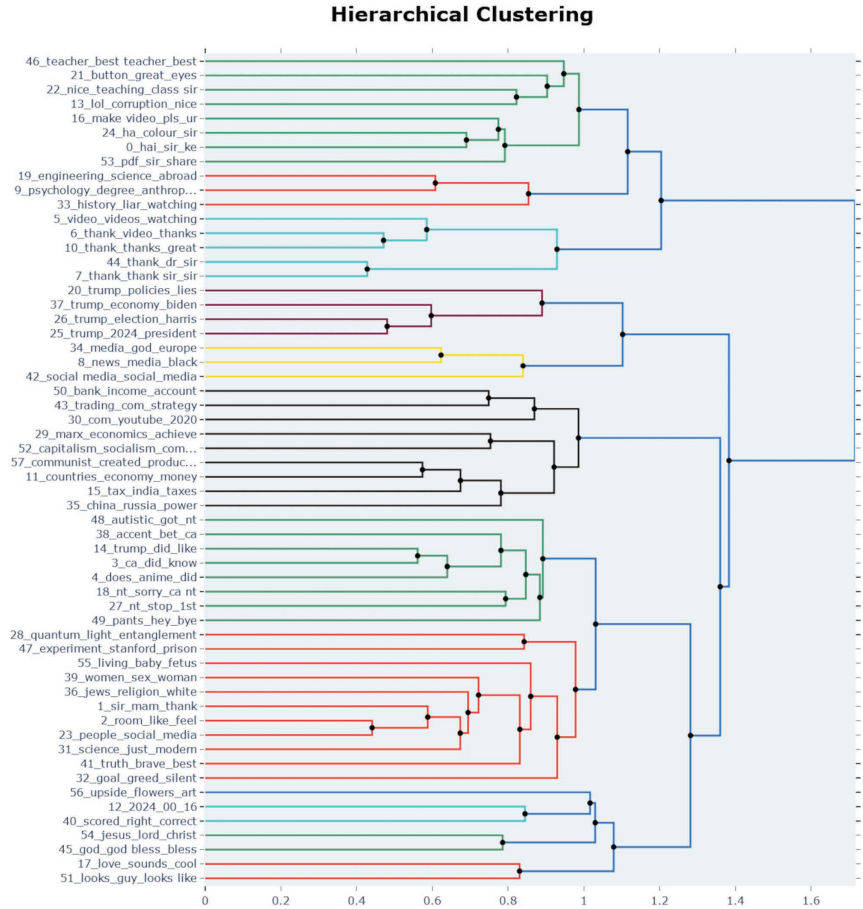
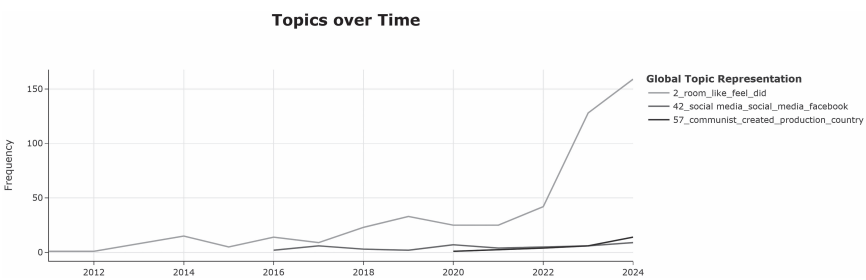


Figure 18.9 Evolution of topics 2, 42, and 57 over time.



18.4.2 Document-topic representations

Let us now project the documents, both with and without a topic hierarchy, onto a two-dimensional plane using the UMAP algorithm. To achieve this – without re-calculating the entire BERTopic model – we recommend creating document embeddings first and then applying the UMAP algorithm to reduce the dimensionality of the 768-dimensional embeddings. Code box 18.29 demonstrates how to do this:

```
doc_embeddings = [embeddings(x) for x in tqdm(cleaned_texts)]
doc_embeddings_rows = [tuple(pd.DataFrame(doc[0]).mean()) for \
                       doc in tqdm(doc_embeddings)]
df_doc_embeddings = pd.DataFrame(doc_embeddings_rows)
doc_reduced = umap_model.fit_transform(doc_embeddings_rows)
```

Code box 18.29: Code for creating and reducing document embeddings using UMAP

In the first line of code, we use the DistilBERT pipeline inside a list comprehension to extract embeddings from the text corpus and assign the resulting Tensor Flow object – composed of threefold nested lists – to `doc_embeddings` (similar to Chapter 18.1.2). The innermost list contains token-level embeddings, which must be averaged to generate document-level embeddings. This averaging is done in the second list comprehension.

But let us break down the list comprehension further, as the expression `tuple(pd.DataFrame(doc[0]).mean())` might initially seem cryptic at first. Here, `doc[0]` retrieves the token-level embeddings per document. Passing these to `pandas.DataFrame()` function converts the embeddings into a data frame with 768 columns and one row per token (e.g., six if the document contains the two BERT-specific tokens indicating the start and end of a sequence, and four tokens for the text content). The `mean()` method averages the embedding values across all 768 columns, returning a Pandas series object. We then convert the series into a tuple to leverage pandas' ability to treat a list of tuples as a data frame structure. Subsequently, we pass this list of tuples to another `pd.DataFrame()` and assign it to `df_doc_embeddings`. Simply stacking Pandas series objects would result in a data frame with a single column, but 7,680,000 rows (768 * 10,000). Instead, we use the data frame with 10,000 rows and 768 columns as input for the `umap_model` initialized in Section 18.1.3. The `fit_transform()` method then creates an array with 10,000 rows and 25 dimensions per row. We visualize the first two as follows (Code box 18.30):

```
fig_documents = topic_model.visualize_documents(docs = cleaned_texts,
topics = topics, reduced_embeddings = doc_reduced)
fig_documents.write_html("document_embeddings.html")
```

Code box 18.30: Visualizing document embeddings in relation to assigned topics

Calling `topic_model.visualize_documents()` and the `fig_documents.write_html()` results in Figure 18.10. We use the following parameters to generate the figure:

- `docs` = the text corpus to display.
- `topics` = a list of assigned topic numbers per document.
- `reduced_embeddings` = the coordinates returned by the UMAP algorithm.

Hovering over points in the scatter plot displays the text of the corresponding document. We can use this display to get insights into the structure of the documents at a specific region of the two-dimensional space.

The `visualize_hierarchical_documents()` method is structured similarly, except that we pass a Pandas data frame that stores the hierarchical

Figure 18.10 Documents and assigned topics positioned in the reduced embedding space.

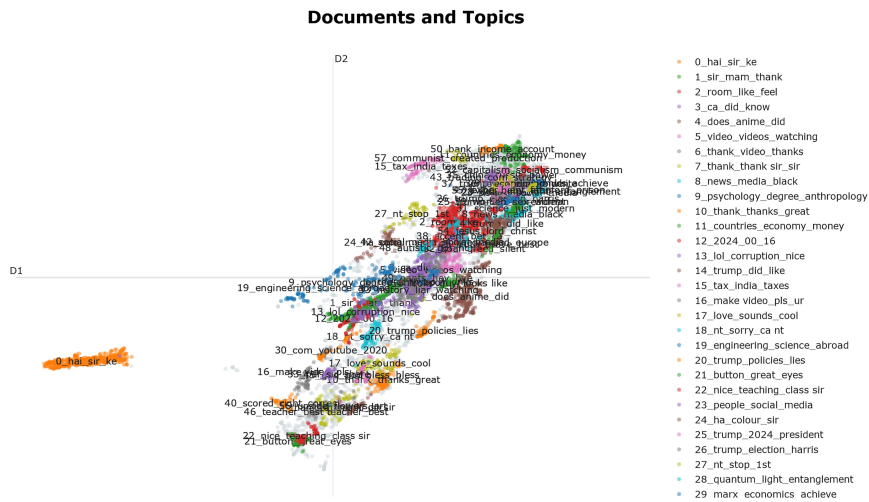
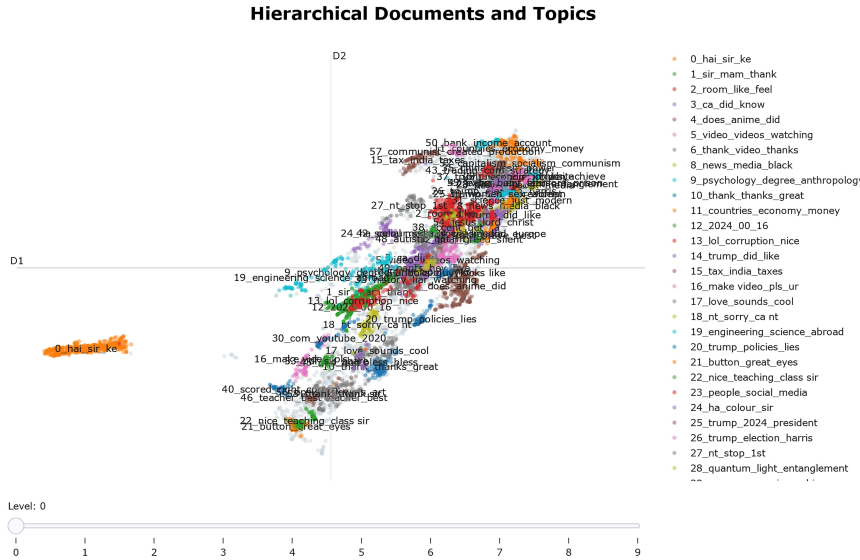


Figure 18.11 Hierarchical visualization of documents and assigned topics in the reduced embedding space.



clustering information to the `hierarchical_topics` parameter (Code box 18.31). This includes a slide at the bottom of the plot (Figure 18.11), which allows us to select different levels of topic granularity. Higher numbers correspond to more general topics – akin to moving along the branches of the cluster dendrogram from more fine-grained topics on the left to more general topics on the right. This method allows for an in-depth analysis of topics-document relationships across different hierarchical levels.[11]

```
fig_hierarchical_documents = topic_model.\
   visualize_hierarchical_documents(docs = cleaned_texts,
hierarchical_topics = topic_hierarchy,
reduced_embeddings = doc_reduced)
fig_hierarchical_documents.write_html("hierarchical_document_
embeddings.html")
```

Code box 18.31: Creating a hierarchical visualization of documents and assigned topics in the reduced embedding space

18.5 Calculating coherence

Although BERTopic does not natively implement topic coherence measures, we can still utilize sklearn's `CountVectorizer()` and `gensim` (Řehůřek & Sojka, 2011) to calculate C_v-Coherence (Röder et al., 2015). To do so, we start by extracting the most typical tokens or n-grams per topic weighted by c-tf-idf while excluding noise (topic = −1) (Code box 18.32):

```
topics_gensim = topic_model.get_topics()
topic_words = [[token for token, score in topics] for topic_no,
               topics in topics_gensim.items() if topic >= 0]
```

Code box 18.32: Extracting the tokens and n-grams with the highest c-tf-idf score per topic

The `get_topics()` method returns a dictionary where the keys correspond to topic numbers, and values contain lists of tuples. Each tuple consists of a token or n-gram, and its corresponding C-TF-IDF score (e.g., `("economy", 0.1454)`). The nested list comprehension extracts the tokens or n-grams from these tuples, while the outer list comprehension iterates over the dictionary, retrieving topic numbers (`topic_no`), and their associated lists of tuples `(topics)`. The inner list comprehension extracts only the first element of the tuples equals to token and returns a list of lists, containing contain topic-specific tokens for coherence evaluation. Next, we fit and tokenize the text corpus as follows (Code box 18.33):

```
fitted_model = vectorizer_model.fit(cleaned_texts)
count_vector_tokenizer = fitted_model.build_tokenizer()
tokenized_docs = [count_vector_tokenizer(text) for text in
cleaned_texts]
dictionary = Dictionary(tokenized_docs)
```

Code box 18.33: Tokenizing the text corpus and creating a Gensim dictionary for c_v coherence calculation

Here, we fit the `CountVectorizer()` initialized in Section 18.2.1 to the text corpus and store the results in `fitted_model`. We then create a custom tokenizer based on the vocabulary learned by the vectorizer. Using a list comprehension, we then tokenize the YouTube comments, before storing them in `tokenized_docs`. We then pass `tokenized_docs` to gensim's `Dictionary()` function and create a document frequency-matrix. All of these are then passed

to the `CoherenceModel()` function to calculate the c_v coherence score (Code box 18.34):

```
coherence_model = CoherenceModel(
                  topics = topic_words,
                  texts = tokenized_docs,
                  dictionary = dictionary,
                  coherence = 'c_v')

mean_coherence_score = coherence_model.get_coherence()
print(f"Coherence Score (c_v): {mean_coherence_score}")
```

Code box 18.34: Calculating the mean c_v coherence score for the BERTopic model

It is important to note that the `coherence = 'c_v'` option and call the `get_coherence()` method to calculate the mean coherence over all topics extracted using BERTopic. The result is approximately 0.4 (Output 18.4):

```
Coherence Score (c_v): 0.40074071537670064
```

Output 18.4: Mean c_v coherence score of the topics extracted by the BERTopic pipeline

This score is lower compared to LDA, but can be improved by fine-tuning the UMAP and HDBSCAN hyperparameters. Subsequent qualitative analysis, analogous to LDA (see Section 17.3), should be conducted to interpret the topics and to identify topics to merge to increase topic coherence and interpretability.

18.6 Conclusion and outlook

In this chapter, we have guided you through automated content analysis with BERTopic and focused on its technical aspects. We have explored the algorithms injected in the BERTopic pipeline, including UMAP for dimensionality reduction, HDBSCAN for topic extraction, and methods for topic weighting and fine-tuning. Additionally, we demonstrated how to extract and visualize topics, analyze their hierarchy, and track their evolutionary change over time. We also demonstrated how to visualize the topic–document relations.

As with any model applied for automated content analysis, BERTopic presents both advantages and challenges. Unlike traditional relational methods or generative topic models based on bag-of-words approaches, embeddings

allow us to capture richer linguistic information, such as token dependencies, morphology, grammar, and semantics. Furthermore, fine-tuning options and the integration of large language models (LLMs) – such as ChatGPT – further enhance interpretability. However, to avoid over-reliance on LLMs, which might foster a "bot delusion" (Wieczorek et al., 2025), we should critically engage with them to refine our understanding of the topics and possibly to add nuances to our interpretations. Furthermore, BERTopic allows us to trace changes within topics and to unveil the hierarchical structure of topics. This additional information can be used as further information on how to interpret the topics, as well as the relations among them.

Despite these advantages, BERTopic has notable drawbacks. These are, firstly, that topic extraction is computationally intensive and requires substantial amounts of memory and time, which, in turn, increases its negative impact on the environment. This leads us to the second drawback. There is a huge number of hyperparameters to tune. Tuning these hyperparameters for UMAP, HDBSCAN, and the topic representation algorithms is complex and requires multiple iterations to achieve optimal performance. Therefore, finding a model with optimal hyperparameters can take disproportionately much longer and potentially more runs compared to the LDA or the deterministic relational methods. Thirdly, underlying neural network architectures, such as DistilBERT or ChatGPT, operate as black boxes, making their training and decision processes difficult to interpret. This is especially true if the computing power (and time!) at hand are insufficient to train and analyze them from scratch.

Given these considerations, you should assess whether you have either enough data or a high level of complexity in the data at hand to justify the use of BERTopic and LLMs. If one of those two conditions is met, BERTopic can help you substantiate your qualitative inquiry through refined keyword phrases, extraction of representative documents, topic hierarchies, or the evolution of topics over time. These might even facilitate the discovery of otherwise unnoticed patterns or structures underlying the data, which might even point at phenomena not anticipated by prior research. Moreover, the quantitative and automated analysis of topic dynamics, hierarchies, and embedding spaces can assist you in selecting documents for further qualitative inquiry or serve as initial coding framework in the sense of open coding.

Overall, automated content analysis provides powerful tools for researchers when integrated within mixed-methods research designs. In our opinion, its effectiveness is maximized when domain-specific expertise is combined with an understanding of the underlying algorithms and qualitative methodologies.

Notes

1 If you wish to refresh your knowledge about the pickle package, how and why we use the pickle data format, you can reread the explanation in Section 17.1.
2 See Output 56 for an overview of the DistilBERT architecture.

3 In more technical terms, this procedure is a modified version of the single-linkage clustering technique.
4 λ is calculated as $\lambda = \frac{1}{MRS}$ (MRS = mutual reachability score) with

$$MRS(i,j) = max\left(core_dist(i), core_dist(j), euclidean_dist(i,j)\right).$$

5 There are different weighting schemes available, as well as options to increase the weight of specific tokens. If you have an intuition which words might capture some topics best, you may pass a list to the seed_words option and a floating point number to seed_multiplier to multiply the weight of the given words by this value.
6 For an introduction into list comprehensions, see Chapter 12.4.9. Furthermore, regular expression patterns and associated functions are introduced in Section 12.4.10.
7 Keep in mind that they have a lower resolution compared to matplotlib figures. This is why we zoomed in the htmls, took snapshots, and further refined them with GIMP.
8 According to the ChatGPT fine-tuning, topic 9 addresses *the pursuit of a career chance in anthropology*, and 57 encompasses *economic systems and reforms in former Soviet Union and developing countries*.
9 Note that the figure was created using the windows snippet-tool, using a zoom of approximately 250%. The screenshot was then imported into GIMP and scaled to 600 dpi. By default, four bar charts are plotted per row, which is why the topic word score is aligned to the right of the zoomed-in figure.
10 Nonetheless, we could also apply alternative approaches, such to create the inter-topic distance map, such as CA or MFA. Applying methods of geometric data analysis allow us not only to investigate topic similarities, but to relate them to metadata attached to the text documents, offering insights into potentially underlying dimensions along which the topics are positioned.
11 Notably, the interactive plot allows us to activate or deactivate topics. Deactivating them greys the topics and documents out and allows for an easier inspection of the remaining topics and associated documents.

REFERENCES

Abbott, A. (2002). The disciplines and the future. In S. Brint (Ed.), *The future of the city of intellect: The changing American University* (pp. 205–230). Stanford University Press.

Abdi, H., & Williams, L. J. (2010). Principal component analysis. *Wiley Interdisciplinary Reviews: Computational Statistics*, *2*(4), 433–459.

Abend, G. (2008). The meaning of 'Theory'. *Sociological Theory*, *26*(2), 173–199.

Abercrombie, G., & Batista-Navarro, R. (2020). Sentiment and position-taking analysis of parliamentary debates: A systematic literature review. *Journal of Computational Social Science*, *3*, 245–270.

Akremi, L. (2019). Stichprobenziehung in der qualitativen Sozialforschung. In N. Baur & J. Blasius (Eds.), *Handbuch Methoden der empirischen Sozialforschung* (2nd ed., pp. 313–331). Springer.

Albu, O. B., & Etter, M. (2016). Hypertextuality and social media: A study of the constitutive and paradoxical implications of organizational Twitter use. *Management Communication Quarterly*, *30*(1), 5–31.

Alexander, J. C. (1982). *Positivism, presuppositions, and current controversies. Vol. 1 of theoretical logic in sociology*. University of California Press.

Alslaity, A., & Orji, R. (2024). Machine learning techniques for emotion detection and sentiment analysis: Current state, challenges, and future directions. *Behaviour & Information Technology*, *43*(1), 139–164.

Andreotta, M., Nugroho, R., Hurlstone, M. J., Boschetti, F., Farrell, S., Walker, I., & Paris, C. (2019). Analyzing social media data: A mixed-methods framework combining computational and qualitative text analysis. *Behavior Research Methods*, *51*(4), 1766–1781.

Angelov, D., & Inkpen, D. (2024). Topic modelling: Contextual token embeddings are all you need. In Al-Onaizan Y., Bansal, M., & Chen Y.N. (Eds.): *Proceedings of the association for computational linguistics: EMNLP 2024* (pp. 13528–13539). Association for Computational Linguistics.

Arazo, E., Ortego, D., Albert, P., O'Connor, N. E., & McGuinness, K. (2020). Pseudo-labeling and confirmation bias in deep semi-supervised learning. In *2020 International joint conference on neural networks (IJCNN)* (1–8). IEEE.

Armat, M. R., Assarroudi, A., Rad, M., Sharifi, H., & Heydari, A. (2018). Inductive and deductive: Ambiguous labels in qualitative content analysis. *Qualitative Report*, *23*, S. 219–221.

Arndt, C., Ladwig, T., & Knutzen, S. (2020). Zwischen Neugier und Verunsicherung – Interne Hochschulbefragungen von Studierenden und Lehrenden im virtuellen Sommersemester 2020. https://doi.org/10.15480/882.3090

Ascher, D., Dubois, P. F., Hinsen, K., Hugunin, J., & Oliphant, T. (2001). *Numerical python*. Lawrence Livermore National Laboratory.

Austin, E., Makwana, S., Trabelsi, A., Largeron, C., & Zaïane, O. R. (2024). Uncovering flat and hierarchical topics by Community Discovery on Word Co-occurrence Network. *Data Science and Engineering*, *9*(1), 41–61.

Autor:innengruppe AEDiL. (2021). Corona-Semester reflektiert Einblicke einer kollaborativen Autoethnographie. wbv.

Bail, C. A. (2014). The cultural environment: Measuring culture with big data. *Theory and Society*, *43*(3–4), 465–482.

Baier, C. (2017). *Reformen in Wissenschaft und Universität aus feldtheoretischer Perspektive: Universitäten als Akteure zwischen Drittmittelwettbewerb, Exzellenzinitiative und akademischem Kapitalismus*. UVK Verlagsgesellschaft.

Baker, M. (2016). 1,500 scientists lift the lid on reproducibility. *Nature*, *533*, 452–454. https://doi.org/10.1038/533452a

Barany, A., Nasiar, N., Porter, C., Zambrano, A. F., Andres, A. L., Bright, D., Shah, M., Liu, X., Gao, S., Zhang, J., Mehta, S., Choi, J., Giordano, C., & Baker, R. S. (2024). ChatGPT for education research: Exploring the potential of large language models for qualitative codebook development. In A. M. Olney, I.-A. Chounta, Z. Liu, O. C. Santos, & I. I. Bittencourt (Eds.), *Artificial intelligence in education* (pp. 134–149). Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-64299-9_10

Baron, S. (2025). Trust, explainability and AI. *Philosophy & Technology*, *38*(1), 4. https://doi.org/10.1007/s13347-024-00837-6

Bartko, J. J. (1966). The intraclass correlation coefficient as a measure of reliability. *Psychological Reports*, *19*(1), 3–11.

Bashiri, H., & Naderi, H. (2024). Comprehensive review and comparative analysis of transformer models in sentiment analysis. *Knowledge and Information Systems*, *66*(12), 7305–7361. https://doi.org/10.1007/s10115-024-02214-3

Basov, N., Breiger, R., & Hellsten, I. (2020). Socio-semantic and other dualities. *Poetics*, *78*, 101433.

Basov, N., & Kholodova, D. (2021). Networks of context. Three-layer socio-cultural mapping for a Verstehende network analysis. *Social Networks*, *69*, 84–101.

Baur, N., & Blasius, J. (2019). *Handbuch Methoden der empirischen Sozialforschung* (2nd ed.). Springer.

Baur, N., Graeff, P., Braunisch, L., & Schweia, M. (2020). The quality of big data. Development, problems, and possibilities of use of process-generated data in the digital age. *Historical Social Research/Historische Sozialforschung*, *45*(3), 209–243.

Becher, T. & Trowler, P. (1984/2001). *Academic tribes and territories: Intellectual enquiry and the culture of disciplines*. Maidenhead:Open University Press

Beck, C., Booth, H., El-Assady, M., & Butt, M. (2020). Representation problems in linguistic annotations: Ambiguity, variation, uncertainty, error and bias. In S. Dipper & A. Zeldes (Eds.), *Proceedings of the 14th Linguistic annotation workshop* (pp. 60–73). Association of Computational Linguistics.

Beck, K. (2010). Soziologie der Online-Kommunikation. In W. Schweiger & K. Beck (Eds.), *Handbuch online-kommunikation* (pp. 15–35). VS Verlag.

Beckert, J. (2022). Durable wealth: Institutions, mechanisms, and practices of wealth perpetuation. *Annual Review of Sociology*, *48*, 233–255. https://doi.org/10.1146/annurev-soc-030320-115024

Benvenuto, N., & Piazza, F. (1992). On the complex backpropagation algorithm. *IEEE Transactions on Signal Processing*, *40*(9), 967–969.

Benz, P., Kropp, K., Nobel, T. C., & Rossier, T. (2024). Homologies in fields of cultural production. Evidence from the European scientific field. *Poetics*, *107*, 101945.

Bick, W., & Müller, P. J. (1984). Sozialwissenschaftliche Datenkunde für prozeßproduzierte Daten: Entstehungsbedingungen und Indikatorenqualität. In W. Bick, R. Mann & J. P. Müller (Eds.), *Sozialforschung und Verwaltungsdaten* (pp. 123–159). Klett-Cotta.

Blasius, J., & Greenacre, M. (2014). *Visualization and verbalization of data*. CRC Press.

Blasius, J., Lebaron, F., Le Roux, B., & Schmitz, A. (2019). Conclusion: State of the art and future of Bourdieusian relational methodology. In J. Blasius, F. Lebaron, B. Le Roux, & A. Schmitz (Eds.), *Empirical investigations of social space* (pp. 429–440). Springer.

Blasius, J., & Mühlichen, A. (2010). Identifying audience segments applying the "social space" approach. *Poetics*, *38*(1), 69–89.

Blei, D. M. (2012). Probabilistic topic models. *Communications of the ACM*, *55*(4), 77. https://doi.org/10.1145/2133806.2133826

Blei, D. M., & Lafferty, J. D. (2007). A correlated topic model of science. *The Annals of Applied Statistics*, *1*(1), 17–35.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet allocation. *Journal of Machine Learning Research*, *3*(1), 993–1022.

Bleich, E., Callison, J. P., Edwards, G. G., Fichman, M., Hoynes, E., Jabari, R., & van der Veen, A. M. (2021). The good, the bad, and the ugly: A corpus linguistics analysis of US newspaper coverage of Latinx, 1996–2016. *Journalism*, *22*(6), 1522–1539.

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., & Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, *2008*(10), P10008.

Blumenberg, H. (1979a). *Die Lesbarkeit der Welt (The legibility of the world)*. Suhrkamp.

Blumenberg, H. (1979b/1985). *Work on myth* (R. M. Wallace, Trans.). MIT Press.

Blumenberg, H. (1984). To bring myth to an end. *New German Critique*, *32*, 109–140. https://doi.org/10.2307/488158.

Bohnsack, R., Krüger, H.-H., & Pfaff, N. (2013). Einleitung: Rekonstruktive Milieuforschung. *Zeitschrift für Qualitative Forschung*, *14*(2), 171–178.

Boltanski, L. & Thévenot, L. (1991/2006). *On justification*. Princeton University Press.

Bonn, A., Richter, A., Vohland, K., Pettibone, L., Miriam, B., Feldmann, R., Goebel, C., Grefe, C., Hecker, S., Hennen, L., Hofer, H., Kiefer, S., Klotz, S., Kluttig, T., Krause, J., Küsel, K., Liedtke, C., Mahla, A., Neumeier, V., . . . Ziegler, D. (2016). Grünbuch Citizen Science Strategie 2020 für Deutschland. Projekt "Bürger schaffen Wissen – Wissen schafft Bürger" (GEWISS). https://www.buergerschaffenwissen.de/sites/default/files/assets/dokumente/gewiss-gruenbuch_citizen_science_strategie.pdf

Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010: 19th international conference on Computational StatisticsParis France, August 22–27, 2010 Keynote, Invited and Contributed Papers* (pp. 177–186). Physica-Verlag HD.

Bourdieu, P. (1977). *Outline of a theory of practice*. Cambridge University Press.

Bourdieu, P. (1984). *Distinction. A social critique of the judgement of taste*. Harvard University Press.

Bourdieu, P. (1988). *Homo Academicus*. Stanford University Press.

Bourdieu, P. (1996a). *The state nobility: Elite schools in the field of power*. Polity Press.

Bourdieu, P. (1996b). *Physical space, social space and habitus*. Vilhelm Aubert Memorial Lecture, Report, 10.

Bourdieu, P. (1998). *Practical reason: On the theory of action*. Stanford University Press.

Bourdieu, P., Wacquant, L., & Farage, S. (1994). Rethinking the state: Genesis and structure of the bureaucratic field. *Sociological Theory*, *12*(1), 1–18. https://doi.org/10.2307/202032

Braun, V., & Clarke, V. (2006). Using thematic analysis in psychology. *Qualitative Research in Psychology*, *3*(2), 77–101. https://doi.org/10.1191/1478088706qp063oa

Braxton, J. M., Milem, J. F., & Sullivan, A. S. (2000). The influence of active learning on the college student departure process. Toward a revision of Tinto's theory. *The Journal of Higher Education*, *71*(5), 569–590.

Breiger, R. L., Wagner-Pacifici, R., & Mohr, J. W. (2018). Capturing distinctions while mining text data: Toward low-tech formalization for text analysis. *Poetics*, *68*, 104–119.

Brodeur, A., Cook, N., & Heyes, A. (2020). Methods matter: p-Hacking and publication bias in causal analysis in economics. *American Economic Review*, *110*(11), 3634–3660. https://doi.org/10.1257/aer.20190687

Browne, M. W. (2000). Cross-validation methods. *Journal of Mathematical Psychology*, *44*(1), 108–132.

Burt, R. S. (2017). *Social capital*. Routledge.

Burt, R. S. (2017). Structural holes versus network closure as social capital. *Social Capital*, 31–56.

Burzan, N. (2016). *Methodenplurale Forschung: Chancen und Probleme von Mixed Methods*. Beltz Juventa.

Butterfield, A., & Ngondi, G. E. (2016). *A dictionary of computer science*. Oxford University Press.

Camacho-Collados, J., & Pilehvar, M. T. (2018). On the role of text preprocessing in neural network architectures: An evaluation study on text categorization and sentiment analysis. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and interpreting neural networks for NLP* (pp. 40–46). Association for Computational Linguistics.

Camacho-Collados, J., & Pilehvar, M. T. (2020). Embeddings in natural language processing. In *Proceedings of the 28th international conference on computational linguistics: Tutorial abstracts* (pp. 10–15). Association for Computational Linguistics.

Campello, R. J. G. B., Moulavi, D., Zimek, A., & Sander, J. (2013). A framework for semi-supervised and unsupervised optimal extraction of clusters from hierarchies. *Data Mining and Knowledge Discovery*, *27*(3), 344–371. https://doi.org/10.1007/s10618-013-0311-4

Cao, K., Liu, Y., Meng, G., & Sun, Q. (2020). An overview on edge computing research. *IEEE Access*, 8, 85714–85728.

Carius, A. C., & Teixeira, A. J. (2024). Artificial Intelligence and content analysis: The large language models (LLMs) and the automatized categorization. *AI & Society*. https://doi.org/10.1007/s00146-024-01988-y

Carlsen, H. B., & Ralund, S. (2022). Computational grounded theory revisited: From computer-led to computer-assisted text analysis. *Big Data & Society*, *9*(1), 20539517221080146.

Carr, C. T., & Hayes, R. A. (2015). Social media: Defining, developing, and divining. *Atlantic Journal of Communication*, *23*(1), 46–65.

Chai, C. P. (2023). Comparison of text preprocessing methods. *Natural Language Engineering*, *29*(3), 509–553.

Chakraborty, J., Majumder, S., & Menzies, T. (2021). Bias in machine learning software: Why? How? What to do? In D. Spinellis, G. Gousios, M. Chechik, & M. Di Penta (Eds.), *Proceedings of the 29th ACM joint meeting on European software engineering conference and symposium on the foundations of software engineering* (pp. 429–440). Association for Computing Machinery.

Chang, J., Gerrish, S., Wang, C., Boyd-Graber, J. L., & Blei, D. M. (2009). Reading tea leaves: How humans interpret topic models. *Advances in Neural Information Processing Systems*, 288–296. NIPS'09: Neural Information Processing Systems.

Charalabidis, Y., Zuiderwijk, A., Alexopoulos, C., Janssen, M., Lampoltshammer, T., & Ferro, E. (2018). *The world of open data: Concepts, methods, tools and experiences* (Bd. 28). Springer International Publishing. https://doi.org/10.1007/978-3-319-90850-2

Charmaz, K. C. (2011). Den Standpunkt verändern: Methoden der konstruktivistischen grounded theory. In G. Mey & K. Mruck (Eds.), *Grounded theory reader* (pp. 181–205). VS Verlag.

Chen, P. P.-S. (1976). The entity-relationship model. Toward a unified view of data. *ACM Transactions on Database Systems (TODS)*, *1*(1), 9–36.

Chetty, R., Friedman, J. N., Saez, E., Turner, N., & Yagan, D. (2017). Mobility report cards: The role of colleges in intergenerational mobility. NBER Working Paper No. 23618.

Chomsky, N. (2009). *Syntactic structures*. De Gruyter Mouton.

Chowdhary, K. (2020). Natural language processing. In *Fundamentals of artificial intelligence* (pp. 603–649). Springer.

Chu, X., Ilyas, I. F., Krishnan, S., & Wang, J. (2016). Data cleaning: Overview and emerging challenges. *Proceedings of the 2016 International Conference on Management of Data*, 2201–2206. https://doi.org/10.1145/2882903.2912574

Chubb, L. A. (2023). Me and the machines: Possibilities and pitfalls of using artificial intelligence for qualitative data analysis. *International Journal of Qualitative Methods*, *22*, 16094069231193593. https://doi.org/10.1177/16094069231193593

Churchill, R., & Singh, L. (2022). The evolution of topic modelling. *ACM Computing Surveys*, *54*(10s), 1–35.

Collier, D., LaPorte, J., & Seawright, J. (2008). Typologies: Forming concepts and creating categorical variables. In *The Oxford handbook of political methodology*. Oxford University Press. https://doi.org/10.1093/oxfordhb/9780199286546.003.0007.

Collier, D., LaPorte, J., & Seawright, J. (2012). Putting typologies to work: Concept formation, measurement, and analytic rigor. *Political Research Quarterly*, *65*(1), 217–232.

Corvo, E., & De Caro, W. (2020). COVID-19 and newspapers: A content & text mining analysis. *European Journal of Public Health*, *30*(Supplement 5), ckaa165–064.

Cosentino, V., Izquierdo, J. L. C., & Cabot, J. (2017). A systematic mapping study of software development with GitHub. *IEEE Access*, *5*, 7173–7192.

Coulmas, F. (Ed.). (2017). *The handbook of sociolinguistics*. Wiley.

Courtois, C., Slechten, L., & Coenen, L. (2018). Challenging Google Search filter bubbles in social and political information. Disconforming evidence from a digital methods case study. *Telematics and Informatics*, *35*(7), 2006–2015.

Creswell, J. W., & Plano Clark, V. L. (2011). *Designing and conducting mixed methods research* (2nd ed.). Sage.

Crichton, M. (1999). Essays on science and society: Ritual abuse, hot air, and missed opportunities. *Science*, 283(5407), 1461–1463. https://doi.org/10.1126/science.283.5407.1461

Czedik-Eysenberg, I., Wieczorek, O., Flexer, A., & Reuter, C. (2024). Charting the universe of metal music lyrics and analyzing their relation to perceived audio hardness. *Transactions of the International Society for Music Information Retrieval*, *7*(1). https://doi.org/10.5334/tismir.182

Daenekindt, S., & Huisman, J. (2020). Mapping the scattered field of research on higher education. A correlated topic model of 17,000 articles, 1991–2018. *Higher Education*, *80*, 1–17.

Dahm, G., & Lauterbach, O. (2016). Measuring students' social and academic integration. Assessment of the operationalization in the national educational panel study. In H.-P. Blossfeld, J. von Maurice, M. Bayer, & J. Skopek (Eds.), *Methodological issues of longitudinal surveys. The example of the national educational panel study* (pp. 313–329). Springer.

Davies, M. (2010). The Corpus of Contemporary American English as the first reliable monitor corpus of English. *Literary and Linguistic Computing*, *25*(4), 447–464.

De Paoli, S. (2023). Performing an inductive thematic analysis of semi-structured interviews with a large language model: An exploration and provocation on the limits of the approach. *Social Science Computer Review*, 08944393231220483. https://doi.org/10.1177/08944393231220483

Deelman, E., Peterka, T., Altintas, I., Carothers, C. D., Van Dam, K. K., Moreland, K., Parashar, M., Ramakrishnan, L., Taufer, M., & Vetter, J. (2018). The future of scientific workflows. *The International Journal of High Performance Computing Applications*, *32*(1), 159–175. https://doi.org/10.1177/1094342017704893

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, *41*(6), 391–407. https://doi.org/10/db4ft5

Deng, L., & Liu, Y. (Hrsg.). (2018). *Deep learning in natural language processing*. Springer Singapore. https://doi.org/10.1007/978-981-10-5209-5

Denzin, N. K. (2004). Reading film. In W. Flick, E. v. Kardorff & I. Steinke (Eds.), *A companion to qualitative research* (pp. 237–242). Sage.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint* arXiv:1810.04805.

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: Human language technologies, volume 1 (long and short papers)* (pp. 4171–4186).

Diaz-Bone, R. (2019). Formen des Schließens und Erklärens. In N. Baur & J. Blasius (Eds.), *Handbuch Methoden der empirischen Sozialforschung* (pp. 49–66). Springer.

Dillon, L., Neo, L. S., & Freilich, J. D. (2020). A comparison of ISIS foreign fighters and supporters social media posts: An exploratory mixed-method content analysis. *Behavioral Sciences of Terrorism and Political Aggression*, *12*(4), 268–291.

DiMaggio, P., Nag, M., & Blei, D. (2013). Exploiting affinities between topic modelling and the sociological perspective on culture: Application to newspaper coverage of US government arts funding. *Poetics*, *41*(6), 570–606.

Dimock, M., & Wike, R. (2020). *America is exceptional in the nature of its political divide*. Pew Research Center. Retrieved November 18, 2024, from https://www.pewresearch.org/fact-tank/2020/11/13/america-isexceptional-in-the-nature-of-its-political-divide/.

Doersch, C., & Zisserman, A. (2017). Multi-task self-supervised visual learning. In *Proceedings of the 2017 IEEE international conference on computer vision* (pp. 2051–2060). IEEE.

Dröge, K. (2023) Horizontal Coding: AI-Based Qualitative Data Analysis in QualCoder, Free & Open Source. Youtube: https://youtu.be/FrQyTOTJhCc?si=qgbYwyh8wxz991Vq

Duarte, J. M., & Berton, L. (2023). A review of semi-supervised learning for text classification. *Artificial Intelligence Review*, *56*(9), 9401–9469.

Duriau, V. J., Reger, R. K., & Pfarrer, M. D. (2007). A content analysis of the content analysis literature in organization studies. Research themes, data sources, and methodological refinements. *Organizational Research Methods*, *10*(1), 5–34.

Ebbinghaus, B. (2005). When less is more. Selection problems in large-N and small-N cross-national comparisons. *International Sociology*, *20*(2), 133–152.

Ebbinghaus, B. (2009). Vergleichende Politische Soziologie: Quantitative Analyze-oder qualitative Fallstudiendesigns? In V. Kaina & A. Römmele (Eds.), *Politische Soziologie* (pp. 481–501). VS Verlag.

Edelmann, A., & Mohr, J. W. (2018). Formal studies of culture: Issues, challenges, and current trends. *Poetics*, *68*, 1–9.

Eisewicht, P. & Grenz, T. (2018). Die (Un)Möglichkeit allgemeiner Gütekriterien in der Qualitativen Forschung – Replik auf den Diskussionsanstoß zu "Gütekriterien qualitativer Forschung" von Jörg Strübing, Stefan Hirschauer, Ruth Ayaß, Uwe Krähnke und Thomas Scheffer. *Zeitschrift für Soziologie*, *47*(5), 364–373.

Ellis, C., Adams, T. E., & Bochner, A. P. (2010). Autoethnography. An overview. *Forum Qualitative Sozialforschung/Forum: Qualitative Social Research*, *12*(1).

Elmasri, R., Weeldreyer, J., & Hevner, A. (1985). The category concept: An extension to the entity-relationship model. *Data & Knowledge Engineering*, *1*(1), 75–116.

Elo, S., & Kyngäs, H. (2008). The qualitative content analysis process. *Journal of Advanced Nursing*, *62*(1), 107–115.

Elster, J. (1989/2006). *Nuts and bolts for the social sciences*. Cambridge University Press.

Epstein, J. (2002). *Snobbery: The American version*. Houghton Mifflin.
Erhard, L., Hanke, S., Remer, U., Falenska, A., & Heiberger, R. H. (2024). PopBERT. Detecting populism and its host ideologies in the German Bundestag. *Political Analysis*, *33*(1), 1–17.
Erhard, L., & Heiberger, R. (2023). Regression and machine learning. In J. Skopek (Ed.), *Research handbook on digital sociology* (pp. 130–145). Edward Elgar Publishing.
Escofier, B., & Pages, J. (1994). Multiple factor analysis (AFMULT package). *Computational Statistics & Data Analysis*, *18*(1), 121–140.
Fawcett, P., Jensen, M. J., Ransan-Cooper, H., & Duus, S. (2019). Explaining the "ebb and flow" of the problem stream: Frame conflicts over the future of coal seam gas ("fracking") in Australia. *Journal of Public Policy*, *39*(3), 521–541.
Federer, L. M. (2022). Long-term availability of data associated with articles in PLOS ONE. *Plos One*, *17*(8), e0272845.
Fellbaum, C. (2005). WordNet and wordnets. In *Encyclopedia of language and linguistics* (S. 665–670). Elsevier.
Findler, N. (1979). A heuristic information retrieval system based on associative networks. In N. Findler (Hrsg.), *Associative networks* (pp. 305–326). Academic Press.
Fink, A. (2019). Conducting research literature reviews. From the internet to paper. Sage.
Fleck, L. (1935/2019). *Entstehung und Entwicklung einer wissenschaftlichen Tatsache. Einführung in die Lehre vom Denkstil und Denkkollektiv*. Suhrkamp.
Flick, U. (2007/2018). Managing quality in qualitative research: A focus on process and transparency. In U. Flick (Ed.), *Managing quality in qualitative research* (pp. 131–140). Sage. https://doi.org/10.4135/9781529716641
Flick, U. (2018). Gütekriterien qualitativer Sozialforschung. In N. Baur & J. Blasius (Hrsg.), *Handbuch Methoden der empirischen Sozialforschung* (pp. 473–488). Springer.
Flick, U. (2014/2019). Gütekriterien qualitativer Sozialforschung. In N. Baur & J. Blasius (Hrsg.), *Handbuch Methoden der empirischen Sozialforschung* (pp. 473–488). Springer VS.
Frankfurt, H. G. (1958). Peirce's notion of abduction. *The Journal of Philosophy*, *55*(14), 593–597.
Frayling, C. (2005). *Mad, bad and dangerous: The scientist and the cinema*. Reaktion.
Freeman, L. C. (1978). Centrality in social networks conceptual clarification. *Social Networks*, *1*(3), 215–239.
Friese, M., & Frankenbach, J. (2020). p-Hacking and publication bias interact to distort meta-analytic effect size estimates. *Psychological Methods*, *25*(4), 456–471. https://doi.org/10.1037/met0000246
Fruchterman, T. M. J., & Reingold, E. M. (1991). Graph drawing by force-directed placement. *Software: Practice and Experience*, *21*(11), 1129–1164. https://doi.org/10.1002/spe.4380211102
Früh, W. (2011). *Inhaltsanalyze: Theorie und Praxis* (7th ed.). UVK.
Fuhse, J. (2021). *Social networks of meaning and communication*. Oxford University Press.
Fuhse, J. (2022). How can theories represent social phenomena? *Sociological Theory*, *40*(2), 99–123.
Fuhse, J., Stuhler, O., Riebling, J., & Martin, J. L. (2020). Relating social and symbolic relations in quantitative text analysis. A study of parliamentary discourse in the Weimar Republic. *Poetics*, *78*, 101363.
Gadamer, H. G. (2010). *Hermeneutik I. Wahrheit und Methode: Grundzüge einer philosophischen Hermeneutik* (Vol. 1). Mohr Siebeck.
Galassi, A., Lippi, M., & Torroni, P. (2020). Attention in natural language processing. *IEEE Transactions on Neural Networks and Learning Systems*, *32*(10), 4291–4308.
Gao, J., Choo, K. T. W., Cao, J., Lee, R. K.-W., & Perrault, S. (2023). CoAIcoder: Examining the effectiveness of AI-assisted human-to-human collaboration in qualitative

analysis. *ACM Transactions on Computer-Human Interaction*, *31*(1), 38. https://doi.org/10.1145/3617362
Gauthier, R. P., & Wallace, J. R. (2022). The computational thematic analysis toolkit. *Proceedings of the Association for Computing Machinery Human-Computer Interaction, 6(GROUP)*, *25*. https://doi.org/10.1145/3492844
Geertz, C. (1973). *The interpretation of cultures*. Basic Books.
Geertz, C. (1983/2002). *Dichte Beschreibung. Beiträge zum Verstehen kultureller Systeme*. Suhrkamp.
Gieryn, T. (2018). *Truth-spots. How places make people believe*. University of Chicago Press.
Gill, S. L. (2020). Qualitative sampling methods. *Journal of Human Lactation*, *36*(4), 579–581. https://doi.org/10.1177/0890334420949218
Glaser, B. G., & Strauss, A. L. (2017). *Discovery of grounded theory: Strategies for qualitative research*. Routledge.
Goble, C., Cohen-Boulakia, S., Soiland-Reyes, S., Garijo, D., Gil, Y., Crusoe, M. R., Peters, K., & Schober, D. (2020). FAIR computational workflows. *Data Intelligence*, *2*(1–2), 108–121.
Gold, Z., & Latonero, M. (2017). Robots welcome: Ethical and legal considerations for web crawling and scraping. *Washington Journal of Law, Technology & Arts*, *13*, 275.
Goldberg, D. (1991). What every computer scientist should know about floating-point arithmetic. *ACM Computing Surveys (CSUR)*, *23*(1), 5–48.
Gordon, S. L. (2017). The sociology of sentiments and emotion. In M. Rosenberg & R. Turner (Eds.), *Social psychology* (pp. 562–592). Routledge.
Goyanes, M., Lopezosa, C., & Jordá, B. (2024). Thematic analysis of interview data with ChatGPT: Designing and testing a reliable research protocol for qualitative research. *OSF*. https://doi.org/10.31235/osf.io/8mr2f
Graeff, P. & Baur, N. (2020). Digital data, administrative data, and survey compared: Updating the classical toolbox for assessing data quality of big data, exemplified by the generation of corruption data. *Historical Social Research*, *45*(3), S. 244–269.
Graneheim, U. H., & Lundman, B. (2004). Qualitative content analysis in nursing research: Concepts, procedures and measures to achieve trustworthiness. *Nurse Education Today*, *24*(2), 105–112. https://doi.org/10.1016/j.nedt.2003.10.001
Graneheim, U. H., Lindgren, B.-M., & Lundman, B. (2017). Methodological challenges in qualitative content analysis: A discussion paper. *Nurse Education Today*, *56*, 29–34.
Greenacre, M. J. (2010). Correspondence analysis. *Wiley Interdisciplinary Reviews: Computational Statistics*, *2*(5), 613–619.
Greenacre, M. J. (2012). Biplots: the joy of singular value decomposition. *Wiley Interdisciplinary Reviews: Computational Statistics*, *4*(4), 399-406.
Greenacre, M., & Blasius, J. (2006). *Multiple correspondence analysis and related methods*. Chapman and Hall/CRC.
Greenacre, M., Groenen, P. J., Hastie, T., d'Enza, A. I., Markos, A., & Tuzhilina, E. (2022). Principal component analysis. *Nature Reviews Methods Primers*, *2*(100).
Grimmer, J., Roberts, M. E., & Stewart, B. M. (2021). Machine learning for social science: An agnostic approach. *Annual Review of Political Science, 24*(1), 395–419.
Grimmer, J., & Stewart, B. M. (2013). Text as data: The promise and pitfalls of automatic content analysis methods for political texts. *Political Analysis*, 21(3), 267–297.
Grisold, T., Kremser, W., Mendling, J., Recker, J., Vom Brocke, J., & Wurm, B. (2024). Generating impactful situated explanations through digital trace data. *Journal of Information Technology*, *39*(1), 2–18.
Groet, P., Hoozemans, J., Grapentin, A., Eberhardt, F., Al-Ars, Z., & Hofstee, H. P. (2024). Leveraging Apache arrow for zero-copy, zero-serialization cluster shared memory (version 1). *arXiv preprint*. https://doi.org/10.48550/arxiv.2404.03030
Grootendorst, M. (2022). BERTopic: Neural topic modelling with a class-based TF-IDF procedure. *arXiv preprint* arXiv:2203.05794.

Grosman, J., & Reigeluth, T. (2019). Perspectives on algorithmic normativities. Engineers, objects, activities. *Big Data & Society*, *6*(2), 1–12.

Grothe-Hammer, M., & Kohl, S. (2020). The decline of organizational sociology? An empirical analysis of research trends in leading journals across half a century. *Current Sociology*, *68*(4), 419–442.

Groves, R. M., & Lyberg, L. (2010). Total survey error: Past, present, and future. *Public Opinion Quarterly*, *74*(5), 849–879. https://doi.org/10.1093/poq/nfq065

Grudniewicz, A., Moher, D., Cobey, K. D., Bryson, G. L., Cukier, S., Allen, K., Ardern, C., Balcom, L., Barros, T., Berger, M., Ciro, J. B., Cugusi, L., Donaldson, M. R., Egger, M., Graham, I. D., Hodgkinson, M., Khan, K. M., Mabizela, M., Manca, A., . . . Lalu, M. M. (2019). Predatory journals: No definition, no defence. *Nature*, *576*(7786), 210–212. https://doi.org/10.1038/d41586-019-03759-y

Guetzkow, J., Lamont, M., & Mallard, G. (2004). What is originality in the humanities and the social sciences? *American Sociological Review*, *69*(2), 190–212.

Gupta, A., & Bosco, F. (2023). Tempest in a teacup: An analysis of p-Hacking in organizational research. *Plos One*. https://doi.org/10.1371/journal.pone.0281938

Häder, M., & Häder, S. (2019). Stichprobenziehung in der quantitativen Sozialforschung. In N. Baur & J. Blasius (Eds.), *Handbuch Methoden der empirischen Sozialforschung* (2nd ed., pp. 333–348). Springer.

Hagberg, A., Swart, P., & Chult, D. S. (2008). *Exploring network structure, dynamics, and function using NetworkX*. Los Alamos National Lab (LANL).

Hamilton, L., Elliott, D., Quick, A., Smith, S., & Choplin, V. (2023). Exploring the use of AI in qualitative analysis: A comparative study of guaranteed income data. *International Journal of Qualitative Methods*, *22*, 16094069231201504. https://doi.org/10.1177/16094069231201504

Händel, M., Bedenlier, S., Gläser-Zikuda, M., Kammerl, R., Kopp, B., & Ziegler, A. (2020a). Do students have the means to learn during the coronavirus pandemic? Student demands for distance learning in a suddenly digital landscape. *PsyArXiv*. https://doi.org/10.31234/osf.io/5ngm9

Händel, M., Stephan, M., Gläser-Zikuda, M., Kopp, B., Bedenlier, S., & Ziegler, A. (2020b). Digital readiness and its effects on higher education student socio-emotional experiences in the context of COVID-19 pandemic. *PsyArXiv*. https://doi.org/10.31234/osf.io/b9pg7

Harrington, A. (2005). *Modern social theory. An introduction*. Oxford University Press.

Hartmann, M. (2010). Die Exzellenzinitiative und ihre Folgen. *Leviathan*, *38*(3), 369–387.

Harzing, A.-W. (2019). Publish or perish (version 7) [Windows]. Harzing.com. https://harzing.com/resources/publish-or-perish

Hecking, T., & Leydesdorff, L. (2019). Can topic models be used in research evaluations? Reproducibility, validity, and reliability when compared with semantic maps. *Research Evaluation*, *28*(3), 263–272.

Heiberger, R. H., & Riebling, J. R. (2016). Installing computational social science: Facing the challenges of new information and communication technologies in social science. *Methodological Innovations*, *9*, 1–11.

Hensel, P. G. (2021). Reproducibility and replicability crisis: How management compares to psychology and economics – A systematic review of literature. *European Management Journal*, 39(5), 577–594. https://doi.org/10.1016/j.emj.2021.01.002.

Hicks, M. T., Humphries, J., & Slater, J. (2024). ChatGPT is bullshit. *Ethics and Information Technology*, *26*(2), 38. https://doi.org/10.1007/s10676-024-09775-5

Hintze, J. L., & Nelson, R. D. (1998). Violin plots: A box plot-density trace synergism. *The American Statistician*, *52*(2), 181–184.

Hirose, M, & Creswell, J. W. (2023). Applying core quality criteria of mixed methods research to an empirical study. *Journal of Mixed Methods Research*, *17*(1), 12–28. https://doi.org/10.1177/15586898221086346.

Hitch, D. (2023). Artificial intelligence augmented qualitative analysis: The way of the future? *Qualitative Health Research*, 10497323231217392. https://doi.org/10.1177/10497323231217392

Hjellbrekke, J. (2019). *Multiple correspondence analysis for the social sciences*. Routledge, Taylor & Francis Group.

Hladík, R., & Renisio, Y. (2025). Mapping knowledge: Topic analysis of science locates researchers in disciplinary landscape. *Poetics*, *108*, 101950.

Hoffman, M., Bach, F. R., & Blei, D. M. (2010). Online learning for latent Dirichlet allocation. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, & A. Culotta (Hrsg.), *Advances in neural information processing systems 23* (pp. 856–864). NeurIPS Foundation.

Honnibal, M., Montani, I., Van Landeghem, S., & Boyd, A. (2020). spaCy: Industrial-strength natural language processing in python.

Howe, K., & Eisenhart, M. (1990). Standards for qualitative (and quantitative) research: A prolegomenon. *Educational Researcher*, *19*(4), 2–9. https://doi.org/10.3102/0013189X019004002.

Hsieh, H.-F., & Shannon, S. E. (2005). Three approaches to qualitative content analysis. *Qualitative Health Research*, *15*(9), 1277–1288.

Husson, F., & Josse, J. (2014). Multiple correspondence analysis. *Visualization and Verbalization of Data*, *22*, 164–181.

Hutto, C., & Gilbert, E. (2014). Vader: A parsimonious rule-based model for sentiment analysis of social media text. *Eighth International Conference on Weblogs and Social Media (ICWSM-14)*, *8*(1), 216–225.

Icarte, R. T., Klassen, T. Q., Valenzano, R., & McIlraith, S. A. (2022). Reward machines: Exploiting reward function structure in reinforcement learning. *Journal of Artificial Intelligence Research*, *73*, 173–208.

IMDB – Internet Movie Database (2021). *Don't look up storyline*. Retrieved October 30, 2024, from https://www.imdb.com/title/tt11286314/.

Jäger, G., & Rogers, J. (2012). Formal language theory: Refining the Chomsky hierarchy. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *367*(1598), 1956–1970.

Jalali, M. S., & Akhavan, A. (2024). Integrating AI language models in qualitative research: Replicating interview data analysis with ChatGPT. *System Dynamics Review*, *40*(3), e1772. https://doi.org/10.1002/sdr.1772

Jaton, F. (2021). Assessing biases, relaxing moralism. On ground-truthing practices in machine learning design and application. *Big Data & Society*, *8*(1), 1–15.

Jelodar, H., Wang, Y., Yuan, C., Feng, X., Jiang, X., Li, Y., & Zhao, L. (2019). Latent Dirichlet allocation (LDA) and topic modelling: Models, applications, a survey. *Multimedia Tools and Applications*, *78*(11), 15169–15211.

Jia, W., Sun, M., Lian, J., & Hou, S. (2022). Feature dimensionality reduction: A review. *Complex & Intelligent Systems*, *8*(3), 2663–2693. https://doi.org/10.1007/s40747-021-00637-x

Jim, J. R., Talukder, M. A. R., Malakar, P., Kabir, M. M., Nur, K., & Mridha, M. (2024). Recent advancements and challenges of NLP-based sentiment analysis: A state-of-the-art review. *Natural Language Processing Journal*, *6*, 100059.

Jones, K. (1994). Natural language processing: A historical review. In A. Zampolli, N. Calzolari, & M. Palmer (Eds.), *Current issues in computational linguistics: In honour of Don Walker* (pp. 3–16). Springer.

Kallio, H., Malo, P., Lainema, T., Bragge, J., Seppälä, T., & Penttinen, E. (2022). Generating research questions from digital trace data: A machine learning method for discovering patterns in a dynamic environment. *Communications of the Association for Information Systems*, *51*(12), 556–581.

Kelle, U. (2008). *Die Integration qualitativer und quantitativer Methoden in der empirischen Sozialforschung*. Theoretische Grundlagen und methodologische Konzepte. Springer.

Kelle, U. (2019). Mixed methods. In N. Baur & J. Blasius (Eds.), *Handbuch Methoden der empirischen Sozialforschung* (2nd ed., pp. 159–172). Springer.

Kery, M. B., Radensky, M., Arya, M., John, B. E., & Myers, B. A. (2018). The story in the notebook: Exploratory data science using a literate programming tool. In R. Mandry, M. Hancock, M. Perry & A. Cox (Eds.), *CHI '18: Proceedings of the 2018 CHI conference on human factors in computing systems* (pp. 1–11). Association for Computing Machinery.

Kim, M., & Thorne, J. (2024). Epistemology of language models: Do language models have holistic knowledge? *arXiv preprint*. https://doi.org/10.48550/arXiv.2403.12862

Kingma, D. P. (2014). Adam: A method for stochastic optimization. *arXiv preprint* arXiv:1412.6980.

Kirby, D. (2013). *Lab coats in Hollywood. Science, scientists, and cinema*. MIT Press.

Kitchin, R., & McArdle, G. (2016). What makes Big Data, Big Data? Exploring the ontological characteristics of 26 datasets. *Big Data & Society*, *3*(1), 2053951716631130.

Knoblauch, H., Baur, N., Traue, B., & Akremi, L. (2018). Was heißt "Interpretativ forschen"? In L. Akremi, N. Baur, H. Knoblauch, & B. Traue (Eds.), *Handbuch interpretativ forschen* (pp. 9–36). Beltz Juventa.

Knorr Cetina, K. (1999). *Epistemic cultures*. Harvard University Press.

Knuth, D. E. (1984). Literate programming. *The Computer Journal*, *27*(2), 97–111. https://doi.org/10.1093/comjnl/27.2.97

Kossinets, G., & Watts, D. J. (2006). Empirical analysis of an evolving social network. *Science*, *311*(5757), 88–90.

Kozlowski, A. C., Taddy, M., & Evans, J. A. (2019). The geometry of culture: Analyzing the meanings of class through word embeddings. *American Sociological Review*, *84*(5), 905–949.

Kracauer, S. (1952). The challenge of qualitative content analysis. *Public Opinion Quarterly*, *16*(4), 631–642.

Krähnke, U., Pehl, T., & Dresing, T. (2025). *Hybride Interpretation textbasierter Daten mit dialogisch integrierten LLMs: Zur Nutzung generativer KI in der qualitativen Forschung*. SSOAR. https://nbn-resolving.org/urn:nbn:de:0168-ssoar-99389-7

Krebs, D., & Menold, N. (2019). Gütekriterien quantitativer Sozialforschung. In N. Baur & J. Blasius (Eds.), *Handbuch Methoden der empirischen Sozialforschung* (2nd ed., pp. 489–504). Springer.

Kreulich, K., Lichtlein, M., Zitzmann, C., Bröcker, T., Schwab, R., & Zinger, B. (2020). Hochschullehre in der Post-Corona-Zeit. Studie der bayerischen Hochschulen für angewandte Wissenschaften. Forschungs- und Innovationslabor Digitale Lehre (FIDL). https://w3-mediapool.hm.edu/mediapool/media/baukasten/img_2/fidl/dokumente_121/FIDLStudiePostCoronaGesamt.pdf.

Krippendorff, K. (2004). *Content analysis: An introduction to its methodology*. Sage.

Kristensen-McLachlan, R. D., Canavan, M., Kardos, M., Jacobsen, M., & Aarøe, L. (2023). Chatbots are not reliable text annotators. *arXiv preprint*. https://doi.org/10.48550/arXiv.2311.05769

Krotov, V., Johnson, L., & Silva, L. (2020). Tutorial: Legality and ethics of web scraping. *Communications of the Association for Information Systems*, *47*(1), 22.

Kuckartz, U. (2014). *Qualitative text analysis: A guide to methods, practice and using software*. Sage.

Kuckartz, U., & Rädiker, S. (2022). *Qualitative text analysis: A guide to methods, practice and using software*. Sage.

Kuhn, T. (1962/1976). *Die Struktur wissenschaftlicher Revolutionen*. Suhrkamp.

Kumar, V., & Garg, M. (2018). Deep learning as a frontier of machine learning: A. *International Journal of Computer Applications*, *975*, 8887.

Lakatos, I. (1970/1997). Falsification and the methodology of scientific research programmes. In I. Lakatos & A. Musgrave (Eds.), *Criticism and the growth of knowledge* (pp. 91–196). Cambridge University Press.

Lamont, Michèle. (2012). Toward a Comparative Sociology of Valuation and Evaluation. *Annual Review of Sociology 38*, 201–221, https://doi.org/10.1146/annurev-soc-070308-120022

Latour, B. & Woolgar, S. (1979). *Laboratory life. The social construction of scientific facts*. Sage Publications.

Lauer, C., Brumberger, E., & Beveridge, A. (2018). Hand collecting and coding versus data-driven methods in technical and professional communication research. *IEEE Transactions on Professional Communication*, *61*(4), 389–408.

Laver, M., Benoit, K., & Garry, J. (2003). Extracting policy positions from political texts using words as data. *The American Political Science Review*, *97*(2), 311–331.

Le Roux, B., & Rouanet, H. (2010). *Multiple correspondence analysis*. Sage Publications.

Le Roux, B., Rouanet, H., Savage, M., & Warde, A. (2008). Class and cultural division in the UK. *Sociology*, *42*(6), 1049-1071.

Le, Q., & Mikolov, T. (2014). Distributed representations of sentences and documents. In *Proceedings of the 31st international conference on machine learning* (pp. 1188–1196). Association of Computing Machinery.

Lebaron, F. (2018). Pierre Bourdieu, geometric data analysis and the analysis of economic spaces and fields. *Forum for Social Economics*, *47*(3–4), 288–304. https://doi.org/10.1080/07360932.2015.1043928

Lebaron, F. (2021). Geometric data analysis as a tool for reflexivity. *Historical Social Research/Historische Sozialforschung*, *46*(2), 126–154.

Lee, S., Lim, S., Han, S., Oh, G., Chae, H., Chung, J., Kim, M., Kwak, B., Lee, Y., Lee, D., Yeo, J., & Yu, Y. (2024). Do LLMs Have Distinct and Consistent Personality? TRAIT: Personality Testset designed for LLMs with Psychometrics (arXiv:2406.14703). *arXiv*. https://doi.org/10.48550/arXiv.2406.14703

Leifeld, P. (2020). Policy debates and discourse network analysis: A research agenda. *Politics and Governance*, *8*(2), 180–183.

Lewandowsky, S., Mann, M., Brown, N., & Friedman, H. (2016). Science and the public: Debate, denial, and skepticism. *Journal of Social and Political Psychology*, *4*(2), 537–553. https://doi.org/10.5964/jspp.v4i2.604

Lieder, F. R., & Schäffer, B. (2024, June 14). Reconstructive social research prompting (RSRP). Distributed interpretation between AI and researchers in qualitative research. *OSF*. https://doi.org/10.31235/osf.io/d6e9m

Liljequist, D., Elfving, B., & Skavberg Roaldsen, K. (2019). Intraclass correlation – A discussion and demonstration of basic features. *PLOS ONE*, *14*(7), e0219854. https://doi.org/10.1371/journal.pone.0219854

Lin, Y., Michel, J. B., Lieberman, E. A., Orwant, J., Brockman, W., & Petrov, S. (2012). Syntactic annotations for the google books ngram corpus. In *Proceedings of the ACL 2012 system demonstrations* (pp. 169–174). Association for Computational Linguistics.

Lincoln, Y. S., & Guba, E. G. (1985). *Naturalistic inquiry*. Sage.

Liu, N. F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F., & Liang, P. (2024). Lost in the middle: How language models use long contexts. In Association for Computational Linguistics (Eds.), *Transactions of the association for computational linguistics*, volume 12 (pp. 157–173). Association for Computational Linguistics. https://doi.org/10.1162/tacl_a_00638

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). Roberta: A robustly optimized Bert pretraining approach. *arXiv preprint* arXiv:1907.11692.

Lixandru, I.-D. (2024). The use of artificial intelligence for qualitative data analysis: ChatGPT. *Informatica Economica*, *28*(1/2024), 57–67. https://doi.org/10.24818/issn14531305/28.1.2024.05

Lopez-Fierro, S., & Nguyen, H. (2024). *Making human-AI contributions transparent in qualitative coding*. https://repository.isls.org//handle/1/10537

Lörz, M., Marczuk, A., Zimmer, L., Multrus, F., & Buchholz, S. (2020). Studieren unter Corona – Bedingungen: Studierende bewerten das erste Digitalsemester. *DZHW Brief*. https://doi.org/10.34878/2020.05.DZHW_BRIEF

Lovitts, B. E. (2006). Making the implicit explicit. Faculty's performance expectations for the dissertation. In P. L. Maki & N. A. Borkowski (Eds.), The assessment of doctoral education. In *Emerging criteria and new models for improving outcomes* (pp. 163–187). Stylus.

Mannheim, K. (1982). *Structures of thinking*. Routledge.

Manning, C. D., & Schütze, H. (1999). *Foundations of statistical natural language processing*. MIT Press.

Mäntylä, M. V., Graziotin, D., & Kuutila, M. (2018). The evolution of sentiment analysis – A review of research topics, venues, and top cited papers. *Computer Science Review*, *27*, 16–32. https://doi.org/10.1016/j.cosrev.2017.10.002

Martín-Martín, A., Thelwall, M., Orduna-Malea, E., & Delgado López-Cózar, E. (2021). Google Scholar, Microsoft Academic, Scopus, Dimensions, Web of Science, and OpenCitations' COCI: A multidisciplinary comparison of coverage via citations. *Scientometrics*, *126*(1), 871–906.

Mayring, P. (2021). *Qualitative content analysis: A step-by-step guide*. Sage.

McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, *5*, 115–133.

McGill, R., Tukey, J. W., & Larsen, W. A. (1978). Variations of box plots. *The American Statistician*, *32*(1), 12–16.

McInnes, L., Healy, J., & Astels, S. (2017). hdbscan: Hierarchical density based clustering. *Journal of Open Source Software*, *2*(11).

McInnes, L., Healy, J., & Melville, J. (2018). UMAP: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint* arXiv:1802.03426.

McKinney, W. (2011). pandas: A foundational Python library for data analysis and statistics. *Python for High Performance and Scientific Computing*, *14*(9), 1–9.

Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., & Galstyan, A. (2021). A survey on bias and fairness in machine learning. *ACM Computing Surveys (CSUR)*, *54*(6), 1–35.

Mercioni, M. A., & Holban, S. (2020). The most used activation functions: Classic versus current. In *2020 international conference on development and application systems (DAS)* (pp. 141–145). IEEE.

Merton, R. (1942/1973). The normative structure of science. In N. Storer (Ed.), *The sociology of science* (pp. 267–278). University of Chicago Press.

Mesec, B. (2023). The language model of artificial inteligence chatGPT – A tool of qualitative analysis of texts. https://doi.org/10.22541/au.168182047.70243364/v1

Meuser, M., & Nagel, U. (2009). The expert interview and changes in knowledge production. In A. Bogner, B. Littig, & W. Menz (Eds.), *Interviewing experts. Research methods series* (pp. 17–42). Palgrave Macmillan. https://doi.org/10.1057/9780230244276_2

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems 26*. NeurIPS Foundation.

Miles, M. B., & Huberman, A. M. (1994). *Qualitative data analysis: An expanded sourcebook* (2nd ed.). Sage.

Miller, G. A. (1995). WordNet. A lexical database for English. *Communications of the ACM*, *38*(11), 39–41.

Mills, C. (1956/2000). *The power elite*. Oxford University Press.

Mills, K. A. (2018). What are the threats and potentials of big data for qualitative research? *Qualitative Research*, *18*(6), 591–603.

Mills, M., van de Bunt, G., & de Bruijn, J. (2006). Comparative research: Persistent problems and promising solutions. *International Sociology*, *21*(5), 619–631. https://doi.org/10.1177/0268580906067833

Minaee, S., Kalchbrenner, N., Cambria, E., Nikzad, N., Chenaghlu, M., & Gao, J. (2021). Deep learning – Based text classification: A comprehensive review. *ACM Computing Surveys (CSUR)*, *54*(3), 1–40.

Mitroff, I. (1974). Norms and counter-norms in a select group of the Apollo moon scientists: A case study of the ambivalence of scientists. *American Sociological Review*, *39*(4), 579–595. https://doi.org/10.2307/2094423.

Mohr, J. W. (1998). Measuring meaning structures. *Annual Review of Sociology*, *24*(1), 345–370.

Mohr, J. W., & Bogdanov, P. (2013). Introduction – Topic models. What they are and why they matter. *Poetics*, *41*(6), 545–569.

Mohr, J. W., Wagner-Pacifici, R., & Breiger, R. L. (2015). Toward a computational hermeneutics. *Big Data & Society*, *2*(2), 1–8.

Molina, M., & Garip, F. (2019). Machine learning for sociology. *Annual Review of Sociology*, *45*, 27–45.

Moretti, F. (2000). Conjectures on world literature. *New Left Review*, *1*, 54–68.

Moretti, F. (2013). *Distant reading*. Verso.

Morin, E. (1956/2005). *The cinema, or the imaginary man*. University of Minnesota Press.

Morris, R. (1994). Computerized content analysis in management research. A demonstration of advantages & limitations. *Journal of Management*, *20*(4), 903–931.

Müller, L., & Braun, E. (2018). Student engagement. *Zeitschrift für Erziehungswissenschaft*, *21*(3), 649–670.

Müller-Hansen, F., Callaghan, M. W., Lee, Y. T., Leipprand, A., Flachsland, C., & Minx, J. C. (2021). Who cares about coal? Analyzing 70 years of German parliamentary debates on coal with dynamic topic modelling. *Energy Research & Social Science*, *72*, 101869.

Münch, R. (2007). *Die akademische Elite*. Zur sozialen Konstruktion wissenschaftlicher Exzellenz. Suhrkamp.

Münch, R. (2014). *Academic capitalism. Universities in the global struggle for excellence*. Routledge.

Munezero, M., Montero, C. S., Sutinen, E., & Pajunen, J. (2014). Are they different? Affect, feeling, emotion, sentiment, and opinion detection in text. *IEEE Transactions on Affective Computing*, *5*(2), 101–111.

Munoz-Najar Galvez, S., Heiberger, R., & McFarland, D. (2019). Paradigm wars revisited. A cartography of graduate research in the field of education (1980–2010). *American Educational Research Journal*, *57*(2), 612–652.

Murphy, K. P. (2012). *Machine learning: A probabilistic perspective*. MIT Press.

Murshed, B. A. H., Mallappa, S., Abawajy, J., Saif, M. A. N., Al-Ariki, H. D. E., & Abdulwahab, H. M. (2023). Short text topic modelling approaches in the context of big data: Taxonomy, survey, and analysis. *Artificial Intelligence Review*, *56*(6), 5133–5260.

Myrtveit, I., Stensrud, E., & Shepperd, M. (2005). Reliability and validity in comparative studies of software prediction models. *IEEE Transactions on Software Engineering*, *31*(5), 380–391.

Nelson, L. K. (2020). Computational grounded theory: A methodological framework. *Sociological Methods & Research*, *49*(1), 3–42.

Neuendorf, K. A. (2002). *The content analysis guidebook*. Sage.

Newman, M. E. (2006). Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, *103*(23), 8577–8582.

Newman, M. L., Pennebaker, J. W., Berry, D. S., & Richards, J. M. (2003). Lying words: Predicting deception from linguistic styles. *Personality and Social Psychology Bulletin*, *29*(5), 665–675.

Nguyen, T. H., & Shirai, K. (2015). Topic modelling based sentiment analysis on social media for stock market prediction. In Y. Matsumoto (Eds.), *Proceedings of the 53rd annual meeting of the Association for Computational Linguistics and the 7th international joint conference on natural language procession* (pp. 1354–1364). Association for Computational Linguistics.

Nguyen-Trung, K. (2024). ChatGPT in thematic analysis: Can AI become a research assistant in qualitative research? *Quality & Quantity*. https://doi.org/10.31219/osf.io/vefwc

Nicolae, S., Endreß, M., Berli, O., & Bischur, D. (Eds.). (2019). *(Be)Werten: Beiträge zur sozialen Konstruktion von Wertigkeit*. Springer.

Niehr, T. (2014). *Einführung in die Politolinguistik*. Gegenstände und Methoden. UTB.

North, D. (1990). *Institutions, institutional change, and economic performance*. Cambridge University Press.

O'Halloran, K. L., Tan, S., Pham, D.-S., Bateman, J., & Vande Moere, A. (2018). A digital mixed methods research design: Integrating multimodal analysis with data mining and information visualization for big data analytics. *Journal of Mixed Methods Research*, *12*(1), 11–30. https://doi.org/10.1177/1558689816651015

Obar, J. A., & Oeldorf-Hirsch, A. (2020). The biggest lie on the internet: Ignoring the privacy policies and terms of service policies of social networking services. *Information, Communication & Society*, *23*(1), 128–147.

Obermeyer, Z., & Emanuel, E. J. (2016). Predicting the future – Big data, machine learning, and clinical medicine. *The New England Journal of Medicine*, *375*(13), 1216–1219.

Oliphant, T. (2006). *A guide to NumPy (Bd. 1)*. https://ecs.wgtn.ac.nz/foswiki/pub/Support/ManualPagesAndDocumentation/numpybook.pdf

Oppenheimer, R. (2021). *Hollywood can take on science denial: 'Don't Look Up' is a great example*. Scientific American. Retrieved November 18, 2024, from https://www.scientificamerican.com/article/hollywood-can-take-on-science-denial-dont-look-up-is-a-great-example/

Orduna-Malea, E., Martín-Martín, A., & Delgado López-Cózar, E. (2017). Google Scholar as a source for scholarly evaluation. A bibliographic review of database errors. *Revista Española de Documentación Científica*, *40*(4), e185.

Östling, J. (2020). Humboldt's University: The history and topicality of a German tradition. In L. Engwall (Hrsg.), *Missions of universities* (pp. 63–80). Springer.

Ovidious Naso, P. (1922). *Metamorphoses*. Cornhill Publishing Co. Retrieved October 31, 2024, from http://data.perseus.org/citations/urn:cts:latinLit:phi0959.phi006.perseus-eng1:4.706

Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and Trends® in Information Retrieval*, *2*(1–2), 1–135.

Parnas, D. L., Shore, J. E., & Weiss, D. (1976). Abstract types defined as classes of variables. *ACM SIGPLAN Notices*, *11*(SI), 149–154. https://doi.org/10.1145/942574.807133

Parsons, T. (1968). *The structure of social action*. Free Press.

Parsons, T., & Platt, G. (1973). *The American university*. Harvard University Press.

Pearson, K. (1901). LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, *2*(11), 559–572. https://doi.org/10.1080/14786440109462720

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., & Dubourg, V. (2011). Scikit-learn: Machine learning in Python. *The Journal of Machine Learning Research*, *12*, 2825–2830.

Peirce, C. S. (1934). *Collected papers of Charles Sanders Peirce (Vol. 5)*. Harvard University Press.

Pennebaker, J. W., Boyd, R. L., Jordan, K., & Blackburn, K. (2015). *The development and psychometric properties of LIWC2015*. University of Texas at Austin.

Perkins, M., & Roe, J. (2024). The use of Generative AI in qualitative analysis: Inductive thematic analysis with ChatGPT. *Journal of Applied Learning & Teaching*, *7*(1). https://doi.org/10.37074/jalt.2024.7.1.22

Petticrew, M., & Roberts, H. (2008). *Systematic reviews in the social sciences. A practical guide*. Wiley.
Phillips, D. (2018). *Python 3 Object-Oriented Programming: Build robust and maintainable software with object-oriented design patterns in Python 3.8*. Packt Publishing Ltd.
Piketty, T. (2020). *Capitalism and ideology*. Harvard University Press.
Pintas, J. T., Fernandes, L. A., & Garcia, A. C. B. (2021). Feature selection methods for text classification: A systematic literature review. *Artificial Intelligence Review*, *54*(8), 6149–6200.
Pisner, D. A., & Schnyer, D. M. (2020). Support vector machine. In A. Mechelli & S. Vieira (Eds.), *Machine learning. Methods and applications to brain disorders* (pp. 101–121). Elsevier.
Plath, W. (1967). *Multiple path analysis and automatic translation*. North-Holland.
Popper, K. (1935/2002). *The logic of scientific discovery*. Routledge.
Popper, K. (1972). *Objective knowledge: An evolutionary approach*. Clarendon.
Popping, R. (2000). *Computer-assisted text analysis*. Sage.
Power, M. (2022). Theorizing the economy of traces: From audit society to surveillance capitalism. *Organization Theory*, *3*(3), 26317877211052296. https://doi.org/10.1177/26317877211052296
Powers, E. (2017). My news feed is filtered? Awareness of news personalization among college students. *Digital Journalism*, *5*(10), 1315–1335.
Prasad, B. D. (2019). Qualitative content analysis: Why is it still a path less taken? Forum Qualitative Sozialforschung/Forum: *Qualitative Social Research*, *20*(3).
Przyborski, A., & Wohlrab-Sahr, M. (2014). *Qualitative Sozialforschung: Ein Arbeitsbuch*. de Gruyter.
Rabe-Hesketh, S., & Skrondal, A. (2012). *Multilevel and longitudinal modeling using Stata (3. Auflage)*. Stata Press Publication.
Rasamoelina, A. D., Adjailia, F., & Sinčák, P. (2020). A review of activation function for artificial neural network. In IEEE (Eds.), *IEEE 18th World Symposium on Applied Machine Intelligence and Informatics (SAMI)* (pp. 281–286). IEEE.
Řehůřek, R., & Sojka, P. (2011). *Gensim – Statistical semantics in python*. genism.org.
Reichenbach, H. (1938). *Experience and prediction: An analysis of the foundations and the structure of knowledge*. University of Chicago Press.
Reichenbach, H. (1951). *The rise of scientific philosophy*. University of California Press.
Reichertz, J. (2013). *Die Abduktion in der qualitativen Sozialforschung: über die Entdeckung des Neuen*. Springer-Verlag.
Rekker, R. (2021). The nature and origins of political polarization over science. *Public Understanding of Science*, *30*(4), 352–368. https://doi.org/10.1177/0963662521989193
Resnik, P., Garron, A., & Resnik, R. (2013). Using topic modelling to improve prediction of neuroticism and depression in college students. In T. Baldwin & A. Korhonen (Eds.), *Proceedings of the 2013 conference on empirical methods in natural language processing* (pp. 1348–1353). Association for Computational Linguistics.
Riebling, J. R. (2018). The medium data problem in social science. In C. M. Stuetzer, M., Welker & M. Egger (Eds.), *Computational social science in the age of big data. Concepts, methodologies, tools, and applications* (pp. 76–100). Halem.
Roberts, M. E., Stewart, B. M., Tingley, D., Lucas, C., Leder-Luis, J., Gadarian, S. K., . . . Rand, D. G. (2014). Structural topic models for open-ended survey responses. *American Journal of Political Science*, *58*(4), 1064–1082.
Rocco, T. S. (2010). Criteria for evaluating qualitative studies. *Human Resource Development International*, *13*(4), 375–378. https://doi.org/10.1080/13678868.2010.501959
Röder, M., Both, A., & Hinneburg, A. (2015). Exploring the space of topic coherence measures. In *Proceedings of the eighth ACM international conference on Web search and data mining* (pp. 399–408). Association for Computing Machinery.
Rosenblatt, F. (1957). *The perceptron, a perceiving and recognizing automaton Project Para*. Cornell Aeronautical Laboratory.

Rosenblatt, M. (1956). Remarks on some nonparametric estimates of a density function. *The Annals of Mathematical Statistics*, *27*(3), 832–837. https://doi.org/10.1214/aoms/1177728190

Rossier, T., Ellersgaard, C. H., Larsen, A. G., & Lunding, J. A. (2022). From integrated to fragmented elites. The core of Swiss elite networks 1910–2015. *The British Journal of Sociology*, *73*(2), 315–335.

Rössler, P. (2010). *Inhaltsanalyse*. UVK.

Roth, C., & Cointet, J.-P. (2010). Social and semantic coevolution in knowledge networks. *Social Networks*, *32*(1), 16–29.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, *323*(6088), 533–536.

Russell, J. A. (1980). A circumplex model of affect. *Journal of Personality and Social Psychology*, *39*(6), 1161.

Ryan, R. M., & Deci, E. L. (1953/2017). *Self-determination theory: Basic psychological needs in motivation, development, and wellness*. Guilford Publishing.

Sainburg, T., McInnes, L., & Gentner, T. Q. (2021). Parametric UMAP embeddings for representation and semisupervised learning. *Neural Computation*, *33*(11), 2881–2907. https://doi.org/10.1162/neco_a_01434

Sandel, M. (2020). The tyranny of merit: What's become of the common good? Penguin UK.

Sandelowski, M., & Barroso, J. (2002). Reading qualitative studies. *International Journal of Qualitative Methods*, *1*(1), 74–108. https://doi.org/10.1177/160940690200100107.

Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). DistilBERT: A distilled version of BERT: Smaller, faster, cheaper and lighter. *arXiv preprint* arXiv:1910.01108.

Sartori, G. (1991). Comparing and miscomparing. *Journal of Theoretical Politics*, *3*(3), 243–257.

Sbalchiero, S., & Eder, M. (2020). Topic modelling, long texts and the best number of topics. Some problems and solutions. *Quality & Quantity*, *54*, 1095–1108.

Schäffer, B., & Lieder, F. R. (2023). Distributed interpretation – Teaching reconstructive methods in the social sciences supported by artificial intelligence. *Journal of Research on Technology in Education*, *55*(1), 111–124. https://doi.org/10.1080/15391523.2022.2148786

Schank, R., & Tesler, L. (1969). A conceptual dependency parser for natural language. In Association for Computing Machinery (Eds.), *International conference on computational linguistics COLING 1969*. Association for Computing Machinery.

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, *61*, 85–117.

Schmidt-Wellenburg, C., & Schmitz, A. (2023). Divided we stand, united we fall? Structure and struggles of contemporary German sociology. *International Review of Sociology*, *33*(3), 512–545.

Schmitz, A., Flemmen, M., & Rosenlund, L. (2018). Social class, symbolic domination, and *Angst*: The example of the Norwegian social space. *The Sociological Review*, *66*(3), 623–644. https://doi.org/10.1177/0038026117738924

Schmitz, A., Witte, D., & Gengnagel, V. (2017). Pluralizing field analysis: Toward a relational understanding of the field of power. *Social Science Information*, *56*(1), 49–73. https://doi.org/10.1177/0539018416675071

Schneijderberg, C. (2023). Konventionen von Gütekriterien empirischer Sozialforschung [Conventions of quality criteria of empirical social research]. *Forum: Qualitative Social Research*, *24*(3), Art 1. https://doi.org/10.17169/fqs-24.3.3994

Schneijderberg, C., & Götze, N. (2020). Organisierte, metrifizierte und exzellente Wissenschaftler*innen. Veränderungen der Arbeits- und Beschäftigungsbedingungen an Fachhochschulen und Universitäten von 1992 über 2007 bis 2018. INCHER Working Paper 13, Kassel. https://doi.org/10.5281/zenodo.3949756

Schneijderberg, C., & Götze, N. (2021). Academics' societal engagement in cross-country perspective: Large-n in small-n comparative case studies. *Higher Education Policy*, *34*(1), 1–17. https://doi.org/10.1057/s41307-021-00227-z

Schneiker, A., Dau, M., Joachim, J., Martin, M., & Lange, H. (2019). How to analyze social media? Assessing the promise of mixed-methods designs for studying the Twitter feeds of PMSCs. *International Studies Perspectives*, *20*(2), 188–200.

Schofield, A., Magnusson, M., Thompson, L., & Mimno, D. (2017). Understanding text pre-processing for latent Dirichlet allocation. In *Proceedings of the 15th conference of the European chapter of the Association for Computational Linguistics* (Vol. 2, pp. 432–436). Association for Computational Linguistics.

Schoonenboom, J. (2018). Designing mixed methods research by mixing and merging methodologies: A 13-step model. *American Behavioral Scientist*, *62*(7), 998–1015. https://doi.org/10.1177/0002764218772674

Schoonenboom, J., & Johnson, R. B. (2017). How to construct a mixed methods research design. *Köllner Zeitschrift für Soziologie und Sozialpsychologie*, *69*(Suppl 2), 107–131. https://doi.org/10.1007/s11577-017-0454-1

Schreier, M. (2012). *Qualitative content analysis in practice*. Sage.

Schreier, M., Stamann, C., Janssen, M., Dahl, T., & Whittal, A. (2019). Qualitative content analysis: Conceptualizations and challenges in research practice – Introduction to the FQS special issue "Qualitative Content Analysis I". *Forum Qualitative Sozialforschung/Forum: Qualitative Social Research*, *20*(3).

Schulz, W. (2009). Kommunikationsprozess. In E. Noelle-Neumann, W. Schulz, & J. Wilke (Eds.), *Fischer Lexikon Publizistik, Massenkommunikation* (2nd ed., pp. 169–199). Fischer Taschenbuch.

Schütz, A. (1962). Common-sense and scientific interpretation of human action. In *Collected papers I: The problem of social reality* (pp. 3–47). Springer Netherlands.

Schwarz-Friesel, M., & Consten, M. (2014). *Einführung in die Textlinguistik*. WBG.

Schweinitz, J. (2006). *Film und Stereotyp. Eine Herausforderung für das Kino und die Filmtheorie*. Akademie.

Schweinitz, J. (2006/2011). *Film and stereotype. A challenge for cinema and theory*. Columbia University Press.

Schwemmer, C., & Wieczorek, O. (2020). The methodological divide of sociology: Evidence from two decades of journal publications. *Sociology*, *54*(1), 3–21. https://doi.org/10.1177/0038038519853146

Seaver, N. (2019). Captivating algorithms: Recommender systems as traps. *Journal of Material Culture*, *24*(4), S. 421–436.

Seawright, J., & Gerring, J. (2008). Case selection techniques in case study research: A menu of qualitative and quantitative options. *Political Research Quarterly*, *61*(2), 294–308.

Shetty, P., Shrishma Rao, V. S., & Muralikrishna, S. N. (2024). Assessing bias in large language models: A comparative study of ChatGPT, Gemini, and Claude. In IEEE (Eds.), *2024 International Conference on Modeling, Simulation & Intelligent Computing (MoSICom)* (pp. 133–137). https://doi.org/10.1109/MoSICom63082.2024.10881888

Short, J. C., & Palmer, T. B. (2008). The application of DICTION to content analysis research in strategic management. *Organizational Research Methods*, *11*(4), 727–752.

Shorten, C., Khoshgoftaar, T. M., & Furht, B. (2021). Text data augmentation for deep learning. *Journal of Big Data*, *8*(1), 101.

Shrout, P. E., & Fleiss, J. L. (1979). Intraclass correlations: Uses in assessing rater reliability. *Psychological Bulletin*, *86*(2), 420.

Sinha, R., Solola, I., Nguyen, H., Swanson, H., & Lawrence, L. (2024). The role of generative AI in qualitative research: GPT-4's contributions to a grounded theory analysis. In Association for Computing Machinery (Eds.), *Proceedings of the 2024 symposium on learning, design and technology* (pp. 17–25). Association for Computing Machinery. https://doi.org/10.1145/3663433.3663456

Smelser, N. J. (1976/2013). *Comparative methods in the social sciences*. Quid pro Quo Books.

Smelser, N. J. (2003). On comparative analysis, interdisciplinarity and internationalization in sociology. *International Sociology*, *18*(4), 643–657.

Solan, Z., Horn, D., Ruppin, E., & Edelman, S. (2005). Unsupervised learning of natural languages. *Proceedings of the National Academy of Sciences*, *102*(33), 11629–11634.

Soydaner, D. (2022). Attention mechanism in neural networks: Where it comes and where it goes. *Neural Computing and Applications*, *34*(16), 13371–13385.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, *15*(1), 1929–1958.

Stamann, C., Janssen, M., & Schreier, M. (2016). Searching for the core: Defining qualitative content analysis. *Forum Qualitative Sozialforschung Forum: Qualitative Social Research*, *17*(3).

Steigleder, S. (2008). *Die strukturierende qualitative Inhaltsanalyse im Praxistest*. Tectum.

Steinhardt, I. (2015). *Lehre stärkt Forschung: Studiengangentwicklung durch ProfessorInnen im Handlungssystem Universität*. Springer.

Steinhardt, I. (2021). Students in the spotlight. Using collaborative autoethnography to build a community of learning in the Corona crisis. *ISA Pedagogy Series*, *1*(1), 42–59.

Steinhardt, I., & İkiz-Akıncı, D. (2020). Digitale Bildungspraktiken von Studierenden (DEPS). Daten- und Methodenbericht zur Studie DEPS. DZHW. https://metadata.fdz.dzhw.eu/public/files/data-packages/stu-dps2018$-1.0.0/attachments/dps2018_Data-Methods_Report.pdf

Steinhardt, I., Fischer, C., Heimstädt, M., Hirsbrunner, S. D., İkiz-Akıncı, D., Kressin, L., Kretzer, S., Möllenkamp, A., Porzelt, M., Rahal, R.-M., Schimmler, S., Wilke, R., & Wünsche, H. (2021). Opening up and sharing data from qualitative research: A primer. RatSWD Working Paper Series, 273a. https://doi.org/10.17620/02671.63

Steinhardt, I., Schneijderberg, C., Götze, N., Baumann, J., & Krücken, G. (2017). Mapping the quality assurance of teaching and learning in higher education. The emergence of a specialty? *Higher Education*, *74*(2), 221–237.

Steinke, I. (2004). Quality criteria in qualitative research. In U. Flick, E.v. Kardorff, & I. Steinke (Eds.), *A companion to qualitative research* (pp. 184–190). Sage.

Stevens, M., Elizabeth, A., & Richard, A. (2008). Sieve, incubator, temple, hub: Empirical and theoretical advances in the sociology of higher education. *Annual Review of Sociology*, 34, 127–151.

Stinchcombe, A. L. (1968/1987). *Constructing social theories*. Brace & World.

Stock, W. G., Dorsch, I., Reichmann, G., & Schlögl, C. (2023). Counting research publications, citations, and topics: A critical assessment of the empirical basis of scientometrics and research evaluation. *Journal of Information Science Theory & Practice (JIStaP)*, *11*(2), 37–66. https://doi.org/10.1633/JISTaP.2023.11.2.4

Stone, P. J., Dunphy, D. C., & Smith, M. S. (1966). *The general inquirer: A computer approach to content analysis*. MIT Press.

Strauss, A. L. (1987). *Qualitative analysis for social scientists*. Cambridge University Press.

Strauss, A., & Corbin, J. M. (1997). *Grounded theory in practice*. Sage.

Strong, C. (2014). The challenge of "Big Data": What does it mean for the qualitative research industry? *Qualitative Market Research*, *17*(4), 336–342.

Strübing, J., Hirschauer, S., Ayaß, R., Krähnke, U., & Scheffer, T. (2018). Gütekriterien qualitativer Sozialforschung. *Ein Diskussionsanstoß. Zeitschrift für Soziologie*, *47*(2), 83–100.

Suissa, O., Elmalech, A., & Zhitomirsky-Geffet, M. (2022). Text analysis using deep neural networks in digital humanities and information science. *Journal of the Association for Information Science and Technology*, *73*(2), 268–287.

Sukumar, S. T., Lung, C.-H., Zaman, M., & Panday, R. (2024). Knowledge graph generation and application for unstructured data using data processing pipeline. *IEEE Access*. https://doi.org/10.1109/ACCESS.2024.3462635

Suri, H. (2011). Purposeful sampling in qualitative research synthesis. *Qualitative Research Journal*, *11*(2), 63–75.

Sutton, R. S. (2018). *Reinforcement learning: An introduction*. A Bradford Book.

Swanborn, P. G. (1996). A common base for quality control criteria in quantitative and qualitative research. *Quality & Quantity*, *30*, 19–35. https://doi.org/10.1007/BF00139833

Swedberg, R. (2012). Theorizing in sociology and social science: Turning to the context of discovery. *Theory and Society*, *41*(1), 1–40.

Swedberg, R. (2016). Before theory comes theorizing or how to make social science more interesting. *British Journal of Sociology*, *67*(1), 5–22.

Tashakkori, A., & Teddlie, C. (2003). *Handbook of mixed methods in social & behavioral research*. Sage.

Tausczik, Y. R., & Pennebaker, J. W. (2010). The psychological meaning of words. LIWC and computerized text analysis methods. *Journal of Language and Social Psychology*, *29*(1), 24–54.

Tennant, J. (2020). Web of Science and Scopus are not global databases of knowledge. *European Science Editing*, *46*, e51987.

Thelwall, M., & Stuart, D. (2006). Web crawling ethics revisited: Cost, privacy, and denial of service. *Journal of the American Society for Information Science and Technology*, *57*(13), 1771–1779.

Theng, D., & Bhoyar, K. K. (2024). Feature selection techniques for machine learning: A survey of more than two decades of research. *Knowledge and Information Systems*, *66*(3), 1575–1637. https://doi.org/10.1007/s10115-023-02010-5

Thorndike, R. L. (1953). Who belongs in the family? *Psychometrika*, *18*(4), 267–276. https://doi.org/10.1007/BF02289263

Tight, M. (2014). Working in separate silos? What citation patterns reveal about higher education research internationally. *Higher Education*, *68*, 379–395.

Tilly, C. (1984). *Big structures, large processes, huge comparisons*. Sage.

Tilly, C. (1998/1999). *Durable inequality*. University of California Press.

Tinto, V. (1975). Dropout from higher education: A theoretical synthesis of recent research. *Review of Educational Research*, *45*(1), 89–125.

Tinto, V. (1997). Classrooms as communities. Exploring the educational character of student persistence. *The Journal of Higher Education*, *68*(6), 599–623.

Traus, A., Höffken, K., Thomas, S., Mangold, K., & Schröer, W. (2020). Stu.diCo. – Studieren Digital in Zeiten von Corona. https://dx.doi.org/10.18442/150

Turobov, A., Coyle, D., & Harding, V. (2024). Using ChatGPT for thematic analysis. *arXiv preprint*. https://doi.org/10.48550/arXiv.2405.08828

van Atteveldt, W., van der Velden, M. A., & Boukes, M. (2021). The validity of sentiment analysis: Comparing manual annotation, crowd-coding, dictionary approaches, and machine learning algorithms. *Communication Methods and Measures*, *15*(2), 121–140.

Vaswani, A. (2017). Attention is all you need. Advances in neural information processing systems. *arXiv preprint*. https://arxiv.org/abs/1706.03762v7.

Vatin, F. (2013). Valuation as evaluating and valorizing. *Valuation Studies*, *1*(1), 31–50.

Vines, T. H., Albert, A. Y. K., Andrew, R. L., Débarre, F., Bock, D. G., Franklin, M. T., Gilbert, K. J., Moore, J.-S., Renaut, S., & Rennison, D. J. (2014). The availability of research data declines rapidly with article age. *Current Biology*, *24*(1), 94–97. https://doi.org/10.1016/j.cub.2013.11.014

Vormbusch, U. (2012). *Die Herrschaft der Zahlen. Zur Kalkulation des Sozialen in der kapitalistischen Moderne*. Campus.

Wachinger, J., Bärnighausen, K., Schäfer, L. N., Scott, K., & McMahon, S. A. (2024). Prompts, pearls, imperfections: Comparing ChatGPT and a human researcher in qualitative data analysis. *Qualitative Health Research*, 10497323241244669. https://doi.org/10.1177/10497323241244669

Wallace, R. (1984). Introduction to blumenberg. *New German Critique*, *32*, 93–108. https://doi.org/10.2307/488157. https://www.jstor.org/stable/488157

Wang, C., & Blei, D. M. (2011). Collaborative topic modelling for recommending scientific articles. In C. Apte, J. Ghosh & P. Smyth (Eds.), *KDD '11: Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 448–456). Association for Computing Machinery.

Wang, Q., Ma, Y., Zhao, K., & Tian, Y. (2020). A comprehensive survey of loss functions in machine learning. *Annals of Data Science*, *9*, 187–212.

Wang, Z., Alsam, A., Morrison, D., & Arne Strand, K. (2021). On the naming recommendations derived from expert python code. In *Proceedings of the 13th international conference on education technology and computers* (pp. 468–476). Association for Computing Machinery.

Wang, Z., Xia, L., Yuan, H., Srinivasan, R. S., & Song, X. (2022). Principles, research status, and prospects of feature engineering for data-driven building energy prediction: A comprehensive review. *Journal of Building Engineering*, *58*, 105028. https://doi.org/10.1016/j.jobe.2022.105028

Wankmüller, S. (2022). Introduction to neural transfer learning with transformers for social science text analysis. *Sociological Methods & Research*, 00491241221134527.

Warrens, M. J. (2015). Five ways to look at Cohen's kappa. *Journal of Psychology & Psychotherapy*, *5*(4), 1000197.

Weber, M. (1981). Some categories of interpretive sociology. *The Sociological Quarterly*, *22*(2), 151–180. https://doi.org/10.1111/j.1533-8525.1981.tb00654.x

Weber, R. (1990). *Basic content analysis*. Sage.

Weingart, P. (2006). Chemists and their craft in fiction film. *HYLE – International Journal for Philosophy of Chemistry*, *12*(1), 31–44.

Weingart, P., Muhl, C., & Pansegrau, P. (2003). Of power maniacs and unethical geniuses: Science and scientists in fiction film. *Public Understanding of Science*, *12*(3), 279–288.

Weisberg, H. F. (2009). *The total survey error approach: A guide to the new science of survey research*. University of Chicago Press.

Weischedel, R., Palmer, M., Marcus, M., Hovy, E., Pradhan, S., Ramshaw, L., Xue, N., Taylor, A., Kaufman, J., Franchini, M., El-Bachouti, M., Belvin, R., & Houston, A. (2013). *OntoNotes release 5.0* (S. 2806280 KB) [Dataset]. Linguistic Data Consortium. https://doi.org/10.35111/XMHB-2B84

Werbos, P. J. (1990). Backpropagation through time: What it does and how to do it. *Proceedings of the IEEE*, *78*(10), 1550–1560. https://doi.org/10.1109/5.58337

Whitley, R. (1984). *The intellectual and social organization of the sciences*. Oxford University Press.

Wieczorek, O., & Malzahn, M. (2024). Exploring an extinct society through the lens of Habitus-Field theory and the Tocharian text corpus. *Humanities and Social Sciences Communications*, *11*(1), 1–10.

Wieczorek, O., Hallonsten, O., & Åström, F. (2024). Is management and organizational studies divided into (micro-)tribes? *Scientometrics*, *129*(7), 3871–3995.

Wieczorek, O., Münch, R., & Schubert, D. (2023). All power to the reviewers: British sociology under two-level supervision of the Research Excellence Framework. *Social Science Information*, *61*(4), 481–528.

Wieczorek, O., Schmitz, A., Volle, J., Bayarkhuu, K., & Münch, R. (2023). Types of collaboration and the consolidation of sociological research evidence from publications in five German sociology journals 2000–2019. *Soziale Welt, Sonderband*, *26*, 239–279.

Wieczorek, O., Steinhardt, I., Schmidt, R., Mauermeister, S., & Schneijderberg, C. (2025). The Bot Delusion. Large language models and anticipated consequences for academics' publication and citation behavior. *Futures*, *166*, 103537.

Wieczorek, O., Unger, S., Riebling, J., Erhard, L., Koß, C., & Heiberger, R. (2021). Mapping the field of psychology: Trends in research topics 1995–2015. *Scientometrics*, *126*(12), 9699–9731. https://doi.org/10.1007/s11192-021-04069-9

Willke, H. (2004). *Einführung in das systemische Wissensmanagement*. Carl-Auer-Systeme.

Wilson, T., Wiebe, J., & Hoffmann, P. (2005). Recognizing contextual polarity in phrase-level sentiment analysis. In Association of Computational Linguistics (Eds.), *Proceedings of human language technology conference and conference on empirical methods in natural language processing* (pp. 347–354). Association of Computational Linguistics.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., & Funtowicz, M. (2020). Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: System demonstrations*. Association of Computational Linguistics.

Wu, J., Li, L., & Wang, W. Y. (2018). Reinforced co-training. *arXiv preprint*. https://arxiv.org/abs/1804.06035

Xiao, Z., Yuan, X., Liao, Q. V., Abdelghani, R., & Oudeyer, P.-Y. (2023). Supporting qualitative analysis with large language models: Combining Codebook with GPT-3 for deductive coding. In Association for Computing Machinery (Eds.), Proceedings of the 28th international conference on intelligent user interfaces (pp. 75–78). Association for Computing Machinery. https://doi.org/10.1145/3581754.3584136

Yao, J., & Shepperd, M. (2020, April). Assessing software defection prediction performance: Why using the Matthews correlation coefficient matters. In Proceedings of the 24th International Conference on Evaluation and Assessment in Software Engineering (pp. 120–129).

Yin, R. K. (1981). The case study crisis. *Administrative Science Quarterly*, *26*(1), 58–65. https://doi.org/10.2307/2392599.

Ylä-Anttila, T., Eranti, V., & Kukkonen, A. (2022). Topic modelling for frame analysis: A study of media debates on climate change in India and USA. *Global Media and Communication*, *18*(1), 91–112.

Young, T., Hazarika, D., Poria, S., & Cambria, E. (2018). Recent trends in deep learning based natural language processing. *IEEE Computational Intelligence Magazine*, *13*(3), 55–75.

Yu, Y., Si, X., Hu, C., & Zhang, J. (2019). A review of recurrent neural networks: LSTM cells and network architectures. *Neural Computation*, *31*(7), 1235–1270.

Zaheer, R., & Shaziya, H. (2019). A study of the optimization algorithms in deep learning. In *2019 third international conference on inventive systems and control (ICISC)* (pp. 536–539). IEEE.

Zainuddin, N., Selamat, A., & Ibrahim, R. (2018). Hybrid sentiment classification on twitter aspect-based sentiment analysis. *Applied Intelligence*, *48*(5), S. 1218–1232.

Zhang, H., Wu, C., Xie, J., Lyu, Y., Cai, J., & Carroll, J. M. (2023, September 19). Redefining qualitative analysis in the AI era: Utilizing ChatGPT for efficient thematic analysis. arXiv preprint. http://arxiv.org/abs/2309.10771

Zhang, Z. (2018). Improved Adam optimizer for deep neural networks. In *2018 IEEE/ACM 26th international symposium on quality of service (IWQoS)* (pp. 1–2). IEEE.

Zimmermann, J., Champagne, L. E., Dickens, J. M., & Hazen, B. T. (2024). Approaches to improve preprocessing for Latent Dirichlet Allocation topic modelling. *Decision Support Systems*, *185*, 114310.

Zitt, M., & Bassecoulard, E. (2006). Delineating complex scientific fields by an hybrid lexical-citation method: An application to nanosciences. *Information Processing & Management*, *42*(6), 1513–1531.

INDEX

Note: Page numbers in *italic* indicate a figure and page numbers in **bold** indicate a table on the corresponding page.

Abbott, A. 120, 123
abduction 17, 18n2, 296n6
Abend, G. 104n12
Abercrombie, G. 63
Abrams, J.J. 127
abstraction 6–8, 15–16, 26, 32–35, 42, 65–68, 75, 88–93, 239–240, 248–250, 262, 268–269, 308–310, 339–340
academia 13, 24–25, 113–114, 117, 120–122, 130–132, 150
accuracy 313–317, 509–516, 521–522
activation function 309–311
active variable (MCA) 408–411, 426–434
Adam optimizer 508
adequacy 102, 570
AEDiL 150, 188
affiliation 239, 245
Akhavan, A. 275–276
Akremi, L, 48
Alexander, J. 30–31, 88
algorithm 5–11, 15–16, 34–40, 52, 63–65, 68, 99–102, 301–313, 333, 356–358, 374–376, 383–384, 460–465, 469–471, 485–486, 495–499, 525–530, 535–537, 563, 566–573, 586–587, 591
Alves, V. 257
Amazon 274, 288, 304, 318
ambiguity 138–139, 202, 288, 356
Anaconda 5, 323–328, *324–327*, 332, 354
analytical 48–50, 91–99, 131–133, 136, 138–139, 177–179
Angst (great fear) 106, 133–134
Ankerbeispiel [exemplary quote] 74, 87, 106, 116, 126, 138, 153, 155, 234
annex 15–16, 86–87, 117, 131, 197, 276, 293
annotation 306–307, 433, 483, 520
anomalies 11, 523
anonymization 54–58, 80, 87, 179–189, 198, 215–217, 225
AntConc 2–4, 18, 213, 218–222, **221**, 224, 250
anthropology 59, 385, 417, 436–442, 475–476
anthropomorphize 272, 285
Apache 18, 505–506
API (Application Programming Interface) 5, 54, 81, 246, 279–280, 288, 318, 398, 572
app 157, 168, 171, 181, 304, 324, 345
Apple 220–221
approach (e.g., method and theory) 2–6, 11–12, 15–18, 21–40, 42–44, 47–50, 61–66, 71–75, 79–85, 88–95, 98–102, 115, 122–127, 136, 154–155, 174–179, 213–216, 236–241, 255–257, 266–267,

283–295, 306–308, 445, 449, 455–456, 482–484, 522, 557, 558, 577–580, 590
appropriateness-transparency convention *see* conventions for research quality evaluation
archive 43–45, 49–55, 318–319
Armageddon 127, 129
Arndt, C. 150
array 345–349, 365, 405
artifact 17–18, 70–71, 113–115, 308–312, 490–492, 508–510, 564–567
artificial neural networks 271–273, 302, 308–312, 566–567
Assarroudi, A. 256
asterisk 220, 223, 249–250, 348
asynchronously 164, 165, 182, 304
atheoretical 125
attention mechanism 491
Atteveldt, W. 65
audience 25–26, **25**, 42–44, 109–110, 175
audio 51–54, 303–305
audiovisual 28
autobiography 232
autocode 213, 223–224, 227–231
autoethnography 3–4, 154–156, 162–163, 172–174, 177–188, **181**–**187**, 190–194, 202–221, 224–229, 232–237
automated 1, 5, 9–12, 15–16, 26–28, 34–40, 63–68, 76–77, 83, 86–87, 139–140, 265–266, 301–308, 312–317, 320–332, 339–340, 346–358, 362–363, 383–384, 550–551

backpropagation algorithm 312, *312*
Badger, T. A. 257
bag-of-words assumption 526
Bakharia, A. 256
Barany, A. 274–275
Barnes, B. 64
Barroso, J. 72
Barth, J. 255
Basov, N. 63
Bassecoulard, E. 250
Bastos, R. 257
batch 293, 479, 503, 507, 577
Batista-Navarro, R. 63
Baumann, J. 242
Baur, N. 45, 49, 65
Becher, T. 64
Beck, C. 52, 356
Beckert, J. 123
Bedeutsamkeit (significance) 137
Bedeutungsüberschuss (surplus of meaning) 134
Bensing, J., 255
Benvenuto, N. 37
Berlin 179
BERTopic 6–7, 307–308, 314–315, 333, 383, 483, 490–495, 500–502, 558–592
betweenness centrality 451–452, 465–468, 475, **476**, 481n9
bias 99, 273–281, 284–291, 295, 443, 503–504
bias (machine learning) 356–358, 392
biblio 223
bibliography 239–240, 243–245, 248, 258–266, *261*
bibliometric 4, 238–270
Bidirectional Encoder Representations from Transformers (BERT) 6–7, 307–308, 314–315, 333, 383, 483, 490–495, 500–502, 558–592
big data 3–6, 44–45, 51, 303
bigram 362, 574
biplot 408–412, 428–435, *437*, 444–445
bivariate statistics 386, 399
Blablador 273
Blei, D. 266, 525
blog 14, 24, 239, 251–253, 304
Blumenberg, H. 135–137, 141
Bohnsack, R. 207n2
Bojo Mambo Shrimp Restaurant 112–113
Boltanski, L. 70, 72
Bosco, F. 65
bot 83, 108, 279
Bourdieu, P. 72, 75, 89, 93, 407, 410, 526, 559, 580
boxplot 386, 391–402
Braun, E. 150
Braxton, J. M. 150
Brodeur, A. 65
Brontaroc 129, 131
Brown University 122
Bruza, P. 256
bullshit 273–274
Burla, L. 255
Burzan, N. 38
buzzwords 271

Caltech (California Institute of Technology) 111

Cambridge University 111
Cameroon179
Campos,C.J.257
Canada58
CAPI71
Carius, A. 275
Carrington, J. 257
category 3–4, 8–10, 26–37, 46, 65–78, 84–106, 109–110, 113–122, 125–133, 145–154, 163–178, 190–209, 213–215, 224–237, 254, 258, 264–265, 274–283, 289–292, 295, 341, 349–350, 389–394, 409, 419–421, 427–429, 482–483, 487–490, 497–500, 509–511
causality 272
CCPA 58
censorship 273, 279, 287, 291, 295–296
Chakraborty, J. 71, 356
Charmaz, K. 139
chat 23, 27, 289–290, 572
chatbot5, 7, 268–269, 271–295, 558–559, 571–572, 591
Chen, P. P.-S. 53
Chetty, R. 122
CHIIR 256
China 58, 551–556
Churchill, R. 301
Churchill, W. 112
citation 18, 56–58, 238–254, 258–259, 262–264, 275–277, 500
Clarivate Analytics 246
Clarke, V. 274
class (Python) 323, 341, 497, 518
classification 11, 35–37, 88–89, 306–307, 312–313, 490–493, 496–502, 515–520
classify 11, 66–68, 111–115, 302, 312–313, 482–483, 487, 492–496, 503–504, 514–517
Claude 272, 281, 293
closeness centrality 450–451, 465
cloud 209–210
cluster 38, 220, 259, 262–269, 306–307, 451–452, 460–463, 466–470, 474–479, 558–559, 567–572, 577–578
code 18, 22, 31–38, 59, 66–69, 74–77, 91–92, 96–98, 100–106, 109–110, 114–122, 136–139, 147–154, 156, 163–177, 190–197, 202–203, 206–210, 213, 222–224, 227–235, 264–265, 275–283, 289–292, 314–316, 319–325, 334–341, 345–354, 359, 362–374, 377–382, 386–388, 391–405, 415–421, 424–427, 431–434, 452–459, 463, 466–475, 486–490, 493–494, 497–498, 501–512, 516–522, 529–567, 571–575, 580–583, 586, 591
coefficient 399–400, 404–405
coherence measure 528, 530, 556, 589
college 124, 131–132, 443
Collier, D. 131–133
Columbia University 112, 122
comet 109–132, 368–369
command line 329, *330*
comma-separated-value file (csv) 348–349, 377, 386–387, 435–436, 465–466, 469, 485, 490, 497, 501–502, 512, 517, 522, 531, 535, 547, 562–563
comment (Python) 335, 338, 347, 377, 391, 411
communication 2–3, 21–28, 38–40, 43–45, 51–52, 59–64, 70, 132–135, 152–153, 166–167, 175, 193–195, 206, 209, 221, 280, 304
community 324, 333, 447, 452–453, 465–479
community detection (network) 6, 449, 455, 460–462
comparison 2–4, 8, 12, 41–53, 63, 74–79, 84–86, 91–96, 100–106, 125–138, 172–173, 180, 197–198, 201–202, 227, 233–236, 264–269, 281–283, 342–343, 379, 480–482, 497–500, 514–518, 553–555, 576, 590–591
complexity 303, 307–309, 407–409, 591
computational 34–37, 163, 218–219, 224, 251–253, 264–268, 281–282, 301–303, 336–337, 456, 491–493, 521–526, 568–573, 591
concept 2–5, 30–33, 61–63, 123–128, 131–133, 255–256, 301–322, 352–354, 449–451, 494–495
conditional statement 339, 349, 351, 354–355
confidentiality 57, 80
confusion matrix 313–314, 515–520
console (Python) 323, 328, 334–338, 342
constraint 40, 140, 450–451, 464–465, 483

constructivism 17, 48, 75, 257, 286, 351
context 13–18, 22–29, 36–38, 42–44, 48–50, 55–56, 71–84, 88–92, 101–109, 120–130, 134–136, 148–155, 175–180, 190–196, 200–202, 212–214, 219–221, 232–234, 268–269, 272–277, 281–296, 306–308, 484–486, 490–492, 557–559
context window 268–269, 281–284, 484, 409, 495
contract 160, 185–186
conundrum 80, 84, 189, 217
conventions for research quality evaluation: appropriateness-transparency convention 32, 79–89, 102, 288; discovery convention 73–79; iteration and distancing conventions 63, 78, 88, 94, 97–98, 100; research quality conventions 2, 61–102, 196–197, 233–235, 276–277
Copilot 272, 274
copyleft 18, 103, 151, 218, 258
copyright 18, 56, 103, 151, 218, 258, 270, 274, 284, 289, 319
Cornell University 112, 122
Corona 224, 231
corpus 4–11, 34–35, 84–87, 208–214, 218–220, 231–233, 305–307, 315–318, 333–336, 339–340, 343–345, 356–358, 362–370, 374–383, 407–409, 412–417, 479–480, 499–500, 525–549, 555–557, 560–565, 572–589
correlation 88–89, 103–104, 288–291, 391–400, 404–405
correlation coefficient 65, 103, 399–400, 523
Corzine, J. 255
Coulmas, F. 34, 211
counterfactual 74, 89
coupling 240, 258–261, 264, 266
Courtois, C. 52
creativity 13, 279–281
Creswell, J. 38, 60, 66, 72
Critchton, M. 109
criteria 9–11, 417–418, 239–240, 251–254
CrossRef 242, 245–246
C-TF-IDF 571, 574, 577, 581, 583–584, 589
Ctrl 219–221, 335–336

Dahm, G. 150
Dartmouth College 122
data analysis 1–6, 10–18, 29–31, 37–38, 44–46, 61–81, 88–91, 94, 96, 104–107, 179–180, 189–190, 208–209, 213–215, 218–220, 273–277, 294–296, 301–322, 362–363, 558–559
database 4, 53–56, 238–254, 304–305, 318–320
data cleaning 65–67, 357–358
data preparation 100–101, 217, 314–315, 350–352, 386–387, 452–453
dataset 53–56, 69–71, 228–232, 245–246, 357–359, 385–406, 408–409, 501, 504–507
data structure (data frame) 315–316, 344–345, 410–411, 578–580, 586–587
data structure (immutable) 344–345
data structure (matrix) 408–412
data structure (mutable) 344–345
data structure (sequential) 30, 315, 344–345, 350, 357
data structure (set) 291–293, 321–323, 344–345, 387–389, 392–393, 408–411
Deci, E. 72
decipher 200
decision 9–12, 79, 117–122, 125–126, 191–199, 205–207, 287–288, 306–308
decoding 67, 97, 116, 176
decontextualized 209
deductive approach or research design 21–40, 75, 81, 90, 94, 136, 213
deduction 1–4, 10–15, 65–67, 71–75, 81–102, 115–117, 131–133, 145–173, 174–176, 213–215, 236–238, 256–257, 289–291, 295–296, 238–270
Deep Impact 127, 129, 141n12
deep learning 301–302, 308–310, 522–524
default 219, 251, 348–349, 359, 362, 399–400, 538, 558, 572, 577, 581, 583, 592
definition 21–40, 46, 48, 51, 64–69, 86–89, 119–121, 151–153, 168–169, 175–176, 191–192, 215–216, 238–239
degree centrality 450, 465
delusion 591
Deng, L. 35, 302
Denkkollektive (thought collectives) 285

Denzin, N. 109, 116
dependency parsing 369
descriptive statistics 251, 385, 389–390
DFG (German Science Foundation) 14
diagram 213, 436
diary 87, 96, 108, 110, 116, 154
Dibiasky, Kate 108–111, 119, 121–127, 130–132
dictionary 333, 358–359, 483–486, 505–507, 516–517, 522–523, 528–530, 533–537, 540–541, 545–547, 589–590
digital trace data 304–305
digitalization 23–24, 56–58, 65–66, 151–152, 164, 168, 171–173, 191–195, 206–208, 304–306
dimension 38–39, 131–132, 408–411, 422–437, 440–448, 480–481, 557–561, 565–568
Dimock, M. 141n9
directed edge (network) 450
Directed Qualitative Content Analysis (DQlCA) 257
Dirichlet distribution 525–526
disambiguate 138
discipline 57–59, 239–241, 265–269, 398–399, 448–449
disclosure 216
discovery 3, 61–64, 72–84, 87–93, 100–104, 117–119, 439–442
discovery convention *see* conventions for research quality evaluation
discretion 111
dispersion estimate 390–392
dissertation 13–16, 55, 73, 178, 198
distancing 3, 61–63, 94–102, 136–139, 232–233
DistilBERT 500–522, 560–565
doctor 55–57, 115–117
documentation 3–14, 26–30, 34–35, 42–58, 61–64, 71, 78–81, 84–87, 92–102, 105–106, 109–110, 116, 146–148, 159–163, 171–177, *172*, 184–187, 190–192, 195–199, 206–209, 217–220, 224–225, 229–235, 238–239, 267–269, 274–277, 282–283, 287–294, 305–307, 314–316, 319–324, 333–338, 343–346, 356–357, 363–369, 374–378, 406–414, 417–419, 422–423, 426–429, 436–437, 447–449, 452–455, 467–468, 477–487, 492–507, 510–512, 522–547, **527**, 555–559, 562–577, 584–591
document-term matrix 363–368, 411–414, 417–419, 422–423, 452–454
DOI (digital object identifier) 239, 246, 249, 259, 270, 318
DokuMet 289, 292
domain 39, 301–303, 314, 333, 376, 531, 557, 591
Don't Look Up 108–132, 364, 368
dragon 112, 364
Dröge, K. 291–292
drones 129
Dropbox 58
dual 211, 584
Duetz, M. 255
duplicate 253–254, 356–361
duplicates (data) 253–254, 356–361
Duriau, V. 265
dynamic topic modeling 578–580

Ebadi, A. 256
Ebbinghaus, B. 8, 47, 126
edge (network) 449–455, 458–459, 469–475, 480–481, 566–568
editor (Python) 323, 334–339
editorial 239
eigenvalue 409, 422–425
eigenvector centrality 450, 465
Eisenhart, M. 66, 72
Eisewicht, P. 72
ELE (extinction level event) 117, 120, 124, 127–132
elif (Python) 338, 349
elites 122–133, 140–141
elbow method 426
Ellis, C. 214
Elmasri, R. 53
else (Python) 338, 349–351
Elsevier Publisher 246
Elster, J. 29, 75
email 155–162, 171, 181–187
embedding 288–290, 558–568, 571–573, 586–593
emojis 23–24, 86, 135, 481, 484, 532, 562
empiricism 29–30, 36, 103, 302
emulation 72
encoding 219
encryption 67, 97, 116, 176
endnote 131
environment (Python) 54, 321–328, 331–332, 335–336

episode 226, 446, 552
epistemology 17–18, 100–102, 285–287, 295–296
epoch 509–513
Epstein, J. 140n4, 141
equally 39, 483, 559
Erfahrungsraum (space of conjunctive experience) 27
Erhard, L. 63, 334
error 65–69, 227–229, 244–245, 306–307, 310–314, 321–322, 335–340, 355–358, 495–496, 508–509
escape sequence 341–342
Escofier, B. 63, 409
essay 281
ethic 54–60, 281–289, 555–556
ethnography 3, 59–63, 68–69, 103–105
Europe 55–58, 219, 242, 273–274, 284, 294, 478, 553, 580
evaluation 12–13, 62–64, 69–70, 102–103, 122–123, 202, 224–232, 313–315, 500–501, 514–515, 521–522
event 109, 161, 176–177, 187, 204, 273, 555
evidence 107–110, 122–123, 126–132, 136–137
Excel 151–153, 203, 228–231, 251
exception 179, 273, 378, 474, 530
excerpt 202, 472, 534, 574–575
exemplary quote (*Ankerbeispiel*) 74–76, 86–87, 153–155, 197–198, 227–228, 233–236, 349–350, 369–370
experiment 67, 156, 181, 202, 296
expert 25, 28, 40–41, 85, 109, 439
explanandum 63
explanans 63
exploration 6–8, 13–14, 48–50, 336–338, 385–406
extrapolation 55, 141, 288
extreme 135, 205

F1-Score 313
Facebook 24, 49, 303
factor 6–8, 63–65, 147–148, 407–409
falsification 296n7
familiar 146–148, 154–162, 191–193, 236–239, 277–279, 294–296
Fawcett, P. 39
feasibility 81, 139, 172
feature extraction 368–370, 452–453, 530–533
feature selection 374–383, 414, 528–533
feedback 293, 308, 483
fieldwork 80, 255
figurations 94
file 14–16, 42–43, 86–87, 163, 187–188, 217–220, 228–231, 258–259, 318–319, 331–332, 335–338, 386–388
film 108–132, 140–141
filter 287–289, 373–384, 414–415, 418–419, 453–455, 531–534
Fink, A. 241
firmware 218
Fleck, L. 64
Fletcher, I. 255
Flick, U. 66, 70, 72
floating point number 341–342, 345, 359, 424, 474, 538, 592n5
flow control statements 349
flowchart 13–15, 145–147, 177–178, 212–213
font 177, 243, 399, 464
for loop 350, 457–458, 461–462, 487, 531–532, 543–545
formal 33, 140, 201, 239, 302, 310
formal languages 302
format 54–56, 86–87, 318–320, 356–357, 391–393, 505–507, 534–536
formation 123, 128
formulas 114–115, 306, 481
frame 64–72, 200–202, 358–361, 364–367, 373–374, 377–382, 393–395, 412–414, 427–428, 452–455, 458–459, 462–471, 485–486, 489–490, 504–505, 510–511, 531–533, 546–549, 574–575, 578–580, 584–587
framework 9, 33, 46, 50, 58, 125, 154, 216, 287, 316, 449, 541, 557, 564, 591
Frank (Dr.) 118
Frankenbach, J. 65
Frankfurt, H. 18n2, 296n6
Franks (Dr.) 112, 132
Frayling, C. 109
freeware 4, 18, 207–208, 218, 220, 222
frequency 7, 34–35, 209–213, 219–224, 229–230, 239–241, 258–259, 363–367
Friese, M. 65
frozenset 345, 471–472
Fruchterman and Reingold algorithm 469

Früh, W. 26
function 24, 92–93, 119–125, 218–220, 308–309, 321–323, 322–323, 331–340, 346–353, 362–368, 383–384, 393–399, 412–416, 426–428, 436, 464–465, 493–498, 501–502, 505–514, 530–538, 541–545
function (Python) 218, 302, 314, 321–323, 328, 331–340, 346–354, 359, 535–538, 560–561
function call (Python) 337–339, 346–348

Gadamer, H. 17, 286
Gao, S. 274–276
gatekeepers 246
Gauthier, R. 275–276
Geertz, C. 48, 202, 316
Gemini 272, 281–282, 293
gender 9–11, 147–148, 180, 211
generality 306–307, 409–410
generative AI 86, 276
genre 109, 254, 356, 384
gensim 332–333, 528–530, 533–542
geometric data analysis 6, 11, 38, 306, 317, 325, 362, 366, 407–449, 567, 577, 592
Germany 48, 150, 159, 169, 179, 184, 188, 214, 232–234, 318
Gerring, J. 47–48
Gieryn, T. 130
Gill, S. 147
Gilroy, T. 127
GitHub 18, 321, 406
Glaser, B. 9, 32–33, 49, 75, 148, 173, 316
Gläser, J. 33
Glassner, B. 255
glean 254
global 450–452, 555–556
GNU 18, 528
Gold, Z. 55
Google 242–247, 317–318
Götze, N. 8, 47, 50, 242
Goyanes, M. 275–276
Graff, P. 45, 49
Graneheim, U. 32–33, 151–153, 167, 255–257, 265
grant 49, 161, 163, 186–187
Greek 134, 438
Grenz, T. 72
grey 91–94, 239, 284, 431
Grimmer, J. 65, 265
Grok 272–273, 279
Grosman, J. 103n2
ground truth 102n2, 307, 314, 500, 504, 515–517, 521–522
Groves, R. 65
Grudniewicz, A. 245
Guba, E. 65, 68, 72
Guetzkow, J. 73
guideline 44–46, 58–59, 195–199
Gupta, A. 65
gut 279

hacking 14, 65
hallucination 83, 273, 277, 282–287, 295
Hamburg 226
Hamilton, L. 274–275
handbook 1, 63, 75
Händel, M. 150
handle 9, 406, 452, 507
Harrington, A. 89
Harvard University 56, 122, 130, 140n6, 318
Harzing, A.-W. 243
hashtag 52, 303
HDBSCAN *see* Hierarchical Density Based Spatial Clustering of Applications with Noise (HDBSCAN)
heatmap 404–406, 518–520, 582–583
Heiberger, R. 23, 63
Hensel, P. 65
Hensleigh, J. 127
hermeneutic 17, 30, 255, 286, 316
heterogeneity 35, 131, 180, 216, 223–225
heuristic 7–12, 140, 233, 293
Heydari, A. 256
Hicks, M. 272–273
hidden layer 308–310, 312, 491–492
Hierarchical Density Based Spatial Clustering of Applications with Noise (HDBSCAN) 6, 568–573, 590–591
HIPAA 58
Hirose, M. 66, 72
hit 220–221, 225–231, 243–245, 250–254
Hoft, N. 257
Hollywood 116, 125
homogeneous 131, 140, 267, 345
horizontal 292, 341, 469
Howe, K. 66, 72
Hsieh, H.-F. 32–33, 265
Huberman, A. 66, 80, 137, 155, 190, 235

humanities 64, 246, 266, 445
Humboldtian 48
hunches 136
hybrid 293, 295, 307
hyperparameter 306–307, 315, 509, 514, 544, 557, 590–591
hypothesis 29–31, 67–68, 74–78, 82–84, 89–95, 100–104, 171–173

icon 163, 174, 218
identifier 319–320, 378–379
idno 180, 188, 207, 225–226, 234
if (Python) 327–329, 333–353
image 41–42, 53–54
impact 127–129, 306–308
impaired 71, 207
implausible 39, 89
implementation 314–316, 419–421
implication 74–76, 92–95, 100–101
implicit 27, 110, 138, 164, 202, 305
import (Python) 321–326, 332–333, 344–349, 352–355, 560
incoherence 293
incomparable 127
inconsistent 44
index 217–219, 344–345, 359–361, 388–390, 411–412, 417–418, 434–436, 467–469, 521–522, 533–534, 543–546
India 58, 478, 480, 532
indicators 33, 60, 268, 281
inductive approach or research design 11, 21–40, 71–75, 89–95, 136
induction 1–4, 10–18, 61–63, 77–90, 97–107, 113–125, 131–133, 139–140, 174–207, 254–256, 289–292, 295–296
inequality 72, 89
inference 14–15, 25–27, 34–36, 74–75, 305–307
informal 149, 239
informed consent 56–57, 80, 289, 294
infringement 103n10, 270, 284
input layer 308–312
insecurity 82
integer 341–345, 502–505
integral 70, 106, 113, 130, 173
interaction 17–18, 61–62, 149–152, 156, 164–168, 303–304, 580–581
interactionism 17, 286, 477
intercoder reliability 69
interdisciplinary 241
interface 272–274, 323–324, 333–334
Internet 37–39, 49–53, 84–86, 188–189, 217–218, 284
interpretation 22–32, 38–43, 61–81, 86–128, 131–141, 164, 167, 173–179, 191–204, 208–209, 229–237, 257–259, 262–269, 272–278, 285–297, 314–316, 333–334, 407–412, 432–433, 436–440, 449–452, 459–460, 474–481, 508–509, 520–528, 536–538, 555–561, 590–591
intersubjectivity 17, 32, 62, 64, 137–140, 286
interval 389–390
interview 8–9, 27–28, 47–49, 61–64, 84–87, 94–97, 146–149, 281–282
interviewee 47–49, 84–85, 179–180
intrinsic 72
intuition 39, 256, 514, 566, 576, 592
invalid 68
inverse 7, 367, 571
IPython 334–335, 338
IRB (institutional review boards) 60
ISA (International Sociological Association) 59
ISBN 239, 270n2
ISSN 239, 270
item 65, 67, 379, 414, 432, 454, 531
iteration 61–63, 88–104, 292–293, 350–353, 371, 378–379, 454–461, 466–472, 510–511, 542–545, 567–568, 589–591
iteration and distancing convention *see* conventions for research quality evaluation
iterator (Python) 353, 481n8
itertools 412–413, 430, 453
Ivy League universities 111, 118–132

Jalali, M. 275–276
Jankowicz, D. 257
jargon 25
Jason 109–112, 118, 130, 364
Johnson, R. 30
Jones, K. 36
journal 13–15, 154–162, 180–199, 213–217, 234–235, 239–240, 245–246, 256–257, 268–270, 500
judgment 70–72, 118, 147, 256, 273, 294
Jungblut, J. 242
justification 47–49, 69–75, 82–86, 117–119, 131–133, 177–178, 284–285, 294–296

Kappa 498–499, 523
KeyBERT representation 561, 571
keyword 11–13, 208–213, 218–235, 249–250, 571–576
Kibiswa, N. K. 257
Kim, M. 278
Kirby, D. 109
Knierim, B. 255
Knorr-Cetina, K. 64, 108
knowledge 3–6, 17–18, 22–24, 27–32, 48–50, 60–61, 71–73, 92–93, 116–135, 231–233, 285–287, 294–296, 439–442, 475–477, 494–495, 525–526, 556–557
Kracauer, S. 24, 27, 40, 139
Krähnke, U. 281, 292–293
Krebs, D. 72
Kreulich, K. 150
Krippendorff, K. 265
Krotov, V. 54–55, 319
Krücken, G. 242
Kuckartz, U. 33, 62, 105, 148, 296
Kuhn, T. 17, 64, 285
KWIC (keywords in context) 213, 220–221
Kyngäs, H. 145, 148, 153, 255, 265

labeled data 306–307, 356, 384
laboratory 129, 164
Lakatos, I. 124, 140n7
Lama 268
Lamont, M. 70, 202
landlord 160, 169, 185
langdetect 332–333, 412–413, 452–453
language 21–25, 34–36, 40–42, 218–219, 227–229, 266–268, 301–302, 343–344, 368–369, 375–377, 397–399, 412–415, 490–491, 500, 530–532, 557–560, 571–573
latent 25–30, 34–35, 38–40, 104–107, 135–139, 174–178, 197–204, 273–275, 288–292, 305–307, 367–369, 447–449
Latent Dirichlet Allocation (LDA) 525–557, 590–591
Latonero, M. 54–55, 319
Latour, B. 64
Laudel, G. 33
Lauer, C. 39
Laurence, A. 218
Lauterbach, O. 150
layout 243–245, 398–402, 425–426, 463–464, 469–475, 488–489, 519–520
learning rate 310–313, 322, 508
LeChat 268
Lee, L. 266
legal 54–58, 284, 319
legitimacy 69, 116–132
Leifeld, P. 40n1
lemma 222–224, 369–371, 375–378, 382–383, 531–535
lemmatization 222–224, 302, 351, 369, 528, 531, 573
Lewandowsky, S. 110
lexical approach (sentiment analysis) 6, 482–484, 523n2
lexicon 36, 333, 483–484
license 18, 151, 164, 274
Lieder, F. 275, 292
Liewald, K. 255
Likert scale 65, 484
limitation 7–8, 36–39, 41–60, 79–80, 84–89, 100–101, 135–136, 193–194, 203–205, 220–222, 279–284, 288–290, 523
limit-concept 135, 136, 141n14
Lincoln, Y. 65, 68, 72
Lindgren, B.-M. 256, 257
linguistics 34–38, 208–211
link 11, 156–157, 181–182, 450–451
Linux 218, 324
list 161–163, 186–187, 219–220, 228–229, 237–239, 251–254, 325–328, 338–339, 359–360, 364–365, 378–381, 388–389, 418–420, 429–433, 453–454, 457–461, 466, 472–474, 486–487, 510–511, 516–517, 522–524, 531–535, 541–543, 546–547, 563–565, 577–581, 586–589
list comprehension 344–345, 351–354, 371, 378–379, 496
literate programming 5, 316–317
Liu, Y. 35, 302, 524n13
Lixandru, I.-D. 274–275
LLM (large language model) 268–297, 561, 572, 591
log 156, 162, 165, 170, 181, 394, 540–541
logic 28, 73, 237, 272, 274, 283, 436
longitude 43
loop **98**–**99**, 175, 308, 336–338, 350–351, 355, 371, 378, 415, 421, 433, 457–458, 461, 467, 468, 487, 514, 531–532, 536, 543–545

Lopez-Fierro, S. 276
Lorenzer, A. 255
Lörz, M. 150
loss function 310–312, 508–509, 513–514, 524
Lovitts, B. 73
lowercase 219, 340, 344, 364–365
Lyberg, L. 65
lyrics 52, 301, 360

machine learning 34–39, 63–65, 301–317, 356–358
Macintosh 218
macOS 163
macro 103n2, 477
Malaysia 179
manifest 2–4, 25–30, 34–35, 40–42, 104–108, 130–131, 135–139, 174–178, 189–193, 197–205, 208–209, 212–213, 219, 229–231, 290–292, 305
manipulating 259
Mannheim, K. 27
manual 222
manuscript 74, 78–81, 90, 270
map 4–6, 247–250, 253–254, 258–259, 262–268, 344–345, 348–350, 394–395, 481–482, 487–490, 505–506, 516–517, 533–536, *583*
mark 97, 115, 188, 216, 279, 341–342, 484
markers 190
Martín-Martín, A. 243, 247
master 73–74, 78–81, 216–218, 278, 289
match 22, 103, 235, 249, 262, 284, 354, 494, 556, 579
material 14–17, 26–33, 69–72, 119–120, 147–155, 167–168, 172–179, 189–194, 197–207, 277–293
mathematic 112–115, 305–307
matplotlib 332–334, 348–349, 385–387, 412–413, 452–453, 529–530
matrix 313, 344–346, 363–368, 409–414, 417–419, 491–492, 515–520
Matthew Effect 111, 114, 117–118
Matthews correlation coefficient 65
MAXQDA 18, 149–153, 163–170, 222–225, 229–232, 288–291
Mayring, P. 33, 66, 69, 105, 151, 243–245
Mazzi, M. 255
McKay, A. 108, 141
meaning 6–8, 21–28, 38–42, 67, 70–71, 94, 96–99, 105–107, 14–116, 134–139, 174–177, 189–190, 193–204, 209–213, 223–225, 286–288, 291–292, 305–308, 344–345, 391–392, 408–409, 449–450, 458–460, 490–491, 541–543
measurement 50, 67, 255–257, 312–313, 389–390, 449–451, 463–465, 528–530, 547–549
mechanism 68, 72, 140, 282–283, 491
mediators 275
Medusa 134, 136
memo 121, 153, 195
Menold, N. 72
menu 325–328,334 –338
meritocratic 448
Merton, R. 108
Mesec, B. 276
messenger 191, 194, 205–206
metadata 6, 11, 56, 62, 93, 101, 245, 309, 315–319, 322, 332, 378, 385–387, 409–412, 419, 436, 467–468, 480, 490, 501, 523, 531–533, 542, 546, 562
metamorphosis 134
metaphor 123, 134–136, 140, 438
method 1–18, 21–23, 28–47, 59–75, 78–85, 100–102, 136–137, 154–155, 174–180, 211–215, 238–241, 247–259, 264–269, 274–278, 283–285, 296–297, 301–306, 317–323, 331–332, 335–336, 339–340, 344–348, 354–362, 365–371, 374–384, 388–395, 398–400, 407–481, 485–490, 497–498, 502–512, 525–526, 532–545, 548, 557–559, 573–591
metric 241–242, 389–394, 408–409, 509–511, 540–541, 567, 570
metric variable 389–390, 392–394, 400, 409
Meuser, M. 28
Michigan State University 111, 118, 121–123, 130
Miller, G. 266
minutes 185–186, 204, 221, 554
missing data 357–359
Mitroff, I. 108
mixed-methods 1–3, 37–40, 60–63, 66, 480, 591

modals 209
mode 281
model 4–7, 10–12, 29–31, 34–37, 53–55, 63–65, 83–86, 93–95, 268–297, 301–303, 306–321, 325–327, 333–336, 356, 362–363, 366–369, 374–377, 383–385, 407–449, 461–464, 490–496, 499–523, 525–591
model evaluation 301, 500–501, 514–522
model prediction 504
modularity 451, 460–469, 481
module (Python) 348, 362, 530
Mohr, J. 266
Moodle 156–170, 181–187, 209–211
Moretti, F. 43, 233, 255, 410
Morin, E. 109
morphology 11, 43, 370, 558, 591
Morris, R. 265
Müller L. 49, 150
multidimensional 345, 355n5, 524n16
multilayer 37
multilevel analysis 18n1
multiple correspondence analysis (MCA) 409–414, 417–419, 435–437, 479–480
multivariate statistics 45
Murphy, K. 36, 302
Myrtveit, I. 65
myth 134–137, 141, 446

Nabavi, F. H. 256
Nagel, U. 28
naive 278
naming convention (Python) 339–340
nano 113, 127, 130
Narayan, B. 256
narrative 40n4, 138, 154, 198, 209, 266, 279
natural language processing 5, 34, 301, 362
natural languages 34, 302
negative pointwise mutual information 455–457, 557n4
nested 49, 193, 364, 379, 415, 421, 487, 559, 563–565, 579, 586, 589
Netflix 113–114, 140
network 36–37, 271–273, 308–313, 317–319, 321–322, 449–456, 459–462, 465–475, 478–483, 557–558, 566–570, 578–580
network graph 449–452
networkx 332, 453, 459, 465, 472–474
Neuendorf, K. 265
neural network 6–7, 271–273, 306–313, 322n3–5, 383–384, 566–567, 591
neutral 231–232, 333–334, 486–489, 498–500
Newman, M. 265, 451, 465, 481n7
n-gram 317, 332, 362, 515, 525–527, 549, 559, 571–575, 581, 584, 589
Nguyen-Trung, K. 275–276
NHMRC (National Health and Medical Research Council) 58
Nicolae, S. 202
Niehr, T. 34, 211–212
node (network) 310, 449–456, 460–461, 464–474
noise 357, 477, 558, 574–576
nominal variable 389
nomological 29
noncoded 67, 97, 116, 176
nondigital 51
none type 341–342
nonmanifest 35
nonverbal 22
normalization 258–259, 486–488
norms 18, 126, 131, 202, 273, 286
North, D. 140n2
noun 209–211, 343, 345, 362, 368–369, 374–375, 380
novelty 38, 73
novice 82, 88, 104, 278
nuances 225, 557, 583, 591
nucleus 81, 280
numeric 41–43, 47–49, 206, 398, 411, 417, 574
numpy 332, 346–349, 387–389, 452–453
nutshell 68
NVivo 149–151, 189, 218, 222, 274, 277, 282, 288

object 28–29, 268–270, 322–323, 353–354, 415–418, 422–424, 533–534
objectivity 17–18, 40–43, 62–72, 491–492
obscure 40n3, 180
observation 13–17, 27–30, 44–49, 60–64, 69–71, 74–75, 79–91, 109–110, 126–127, 212–214, 284–286, 306–307, 357–359, 389–392, 400–404, 407–411, 422–423, 439–444

Oeldorf-Hirsch, A. 55
Oleinik, A. 255
omission 77, 188–189, 216–217, 357, 360
open data 55–56, 319
OpenAI 272–274, 279, 282, 558–561, 571–572
openpyxl 332
operationalization 14–16, 41, 49, 60, 91, 95, 145, 215
operator (arithmetic) 342–345
operator (assignment) 339, 342–343, 351, 419, 487
operator (Boolean) 341–342, 381, 419
operator (comparison) 342–343
operator (logical) 341–343, 351, 380–382
operator (membership) 342–344
opinion 482–484
Oppenheimer, R. 141n9
oral 57
ordinal variable 390, 400
Östling, J. 48
outcome 90, 96, 201, 303, 306–308, 421
outdated 217, 277, 557
output 68, 273–283, 287–297, 302–303, 307–316, 331–337, 342–353, 355n6, 357–373, 379–383, 387–394, 417–420, 423–424, 427–430, 486–491, 494–499, 502–503, 522–524, 534–535, 538–540, 577–578
output layer 308–310, 312, 491, 499, 502
overemphasis 291
overfitting 306–307, 310, 509–511, 549
overlap 51, 66, 92, 103, 115, 146, 176, 191, 207, 240, 264, 291, 312, 357, 433, 441, 470, 480, 527, 582
overload 93, 100
overlook 188, 197, 225
overthink 104n17

package 321–323, 327–336
Palmer, M. 265
Pandas 323, 330–332, 346–349, 358–361, 365–368, 371–395, 398–402, 410–420, 432–433, 452–454, 462–466, 485–488, 497–498, 504–505, 510–511, 527–535, 542–544, 560–562, 574–575, 578–579, 584–587
paper 15–16, 91–97, 121–122, 177–180, 195–197
paradigm 17–18, 285–288, 294–297
parameter 279–281, 306–312, 461–462, 503–504, 508–510
paraphrase 32–33
part-of-speech tagging 369, 528, 531
passage 31–37, 151–153, 164–170, 192–194, 282–283, 289–291
passive variable (MCA) 408–411, 419–422, 426–435, 444–445
patent 274
pattern 15–16, 26–40, 63–66, 93–96, 125–128, 198–200, 203–207, 212–214, 233–234, 272–275, 286–292, 302–310, 340–341, 352–355, 491–492
paywall 248–249, 269
pdf 117–118, 268–270, 282–284
peer-review 119, 126, 130, 246, 270n1
penmanship 223
Pennebaker, J. 265
Perkins, M. 274–275
perplexity 528–530, 536–541, 544–549, 556
persons 23, 215–216, 438
Petticrew, M. 241
PhD 73, 81, 108, 117, 121, 125, 131, 289
phenomenology 17, 75, 286
phenomenon 8–17, 25–32, 35–41, 46–50, 60–61, 66–68, 71–75, 78–79, 82–102, 113–115, 131–133, 136–137, 145–147, 174–179, 270–272, 285–286, 303–304
Philipp, M. 245
Philippe, M. 245
philosophy 436–439, 445–446, 475–478
photo 53–54, 103
phrase 21–22, 249–251
physics 385, 412, 441–442, 446
picture 3, 41–60, 80, 135
Piketty, T. 141n11
pip installs packages (pip) 329–332, 524
pipeline (Python) 333, 493, 496, 560–561, 573
pirated 103n10
pitfalls 137
platform 49–56, 85–86, 174, 208–209, 237–239, 303–305
plausibility 106–107, 198–200, 227–231, 287–290
podcast 14, 160, 185, 552
poetry 24–25, 40n3, 301, 318
point estimate 390

polarization 40n1, 279, 482–488
polymorphous 136
Popper, K., 29–30, 75, 103, 296
population 44–53, 216, 553
position 286–288, 408–411, 427–433, 444–445, 469–474
positive 274–276, 311–314, 333–334, 349–350, 482–489, 498–502, 516–524
positivism 285–286, 295–297
postdoctoral 113
posts 17, 86, 279, 284
pragmatism 7, 34, 75, 196, 206, 239
Prasad, B. 26
precision 279–280, 515–516, 521–522
predatory journal 245, 270
preferences 191, 219, 308, 324, 334, 338
preprint 270n3
preprocessing 5–6, 42–44, 356–384, 397,528–532,557–564
presentation 15–17, 68–70, 74–78, 212–213
presuppositions 31, 174–175
pretrained model 483, 499
prince 412
Princeton University 112–113, 122–124, 130–132, 140, 368–369
prior (LDA) 525–527, 537, 540
privacy 54–55, 57–58, 319
process 1–3, 8–17, 24–25, 30–32, 48–51, 60–81, 85–96, 100–102, 105–110, 120–122, 125–128, 136–139, 148–149, 163, 173–175, 189–192, 196–198, 203–206, 243–244, 254–258, 272–279, 282–283, 288–292, 301–302, 306–309, 312–316, 357–358, 362–363, 377–382, 460–461, 537–538
procrastination 232, 237
profession 93
professional 57, 59–60
professor 107–111, 131–132, 166–167
profile 54, 247, 337
program 154–156, 163, 165, 191–192, 216–221, 289–296, 319–321
project 40n5, 62, 154, 163, 178–179, 214, 314, 319, 338, 490, 566
prompt 86–87, 268–269, 279–284, 292–295
proposal 30, 85
protocol 290, 318
Przyborski, A. 27
pseudonyms 38, 58, 274
pseudoscientific 124
psychology 265–266, 439–441, 446, 531–534
publication 4, 42–44, 64–65, 78–84, 146–147, 178–180, 212–213, 238–259, 262, 265–270, 419–420, 426–428
publish 63, 239, 243–244, 251–253, 346
publisher 243–248, 270, 409
Python 1–2, 35–36, 314–318, 321–355, 405–406, 496–497, 535–538, 560
python-louvain 332–333, 460

QDA (qualitative data analysis) 148–151, 229–231, 288–296
QInsights 282, 289
quadrant 30–31
QualCoder 282, 289, 292
qualitative method 3, 10, 37, 68, 102, 276, 480, 591
quality 39–44, 61–72, 78–79, 103–104, 108–110, 178–180, 196–197, 233–235, 241–248, 276–277
quantification 26–28, 37–40, 508–509, 521–523
quantitative method 10, 29, 37–38, 45, 63
questionnaire 55–57, 60–62, 65–69, 100–101
quota 47, 51
quote 74–76, 86–87, 153–155, 164–165, 180, 191–192, 197–198, 227–228, 233–236, 341, 360

Rabe-Hesketh, S. 18n1
Rädiker, S. 33, 62, 70, 105, 296
Ramakrishna, R. 256
Randall, Mindy 108–115, 117–118, 120, 123, 130–132
random 47–50, 65–67, 228–229, 355–357, 461–464, 504–506, 537–538, 562–567
range 28, 86, 103, 148, 218, 224, 236, 241, 344, 386, 391, 404, 456, 465, 484
rationalism 17, 75, 137, 285–286, 295–297
reader 51–52, 198–200
reality 131–136, 285–287
recall 313, 521–522
recode 104n16, 116, 419

reconstruction 286–288
record 86, 154–156, 214, 222–224, 228–229, 247
reference 90–91, 174–175, 190–192, 200–202, 214–220, 236–239, 245–246, 289–290
reflexivity 16–18, 74–76, 92–96, 174–177, 286–288
register 54, 318
regression plot 386, 400–401
regular expression 331, 339–341, 352–355
Reichenbach, H. 103n9
Reichertz, J. 18n2, 296n6
Reigeluth, T. 103n2
reiterate 96, 104, 235, 284, 411
Rekker, R. 110, 141
relational data 449
relationship 17–18, 286–288, 555–556
reliability 64–70
report 24, 135, 150–152, 155, 164, 168
repository 55–56, 84–86, 318–319
representativeness 49–51, 131, 146, 189, 356, 441, 571, 574–576, 584, 591
reproducibility 65–67, 276, 331, 405, 504–505
researcher 15–18, 23–25, 30–32, 39–44, 48–75, 78–82, 85–86, 94–97, 114–121, 131–132, 137–139, 188–189, 198–199, 215–216, 241–242, 274–278, 285–288, 292–293
ResearchGate 242, 246–247, 270
research ethics committees (REC) 59
research quality conventions *see* conventions for research quality evaluation
restriction 46, 47, 53, 58, 196, 202–204, 214, 234, 243, 319
result 105–106, 115–117, 137–141, 156, 159–161, 184–185, 224–225, 229–232, 291–293, 348–349
reuse 55–56, 178–179, 346–348
review 4–5, 78–76, 90–95, 98–101, 110–111, 145–149, 238–241, 247–254, 268–271, 482–484
Riebling, J. 23
rigor 296n7
Rimondini, M. 255
risk 27, 59, 83, 104, 113, 127, 130, 138, 276, 292, 294
RoBERTa 492–505, 514–522, 562–563
Roberts, H. 241, 266
robustness 89, 97, 126, 225, 228, 408, 492
Rocco, T. 72
role 16–18, 88–90, 116–117, 150–151, 201–202, 268–269
Rössler, P. 26
routine 48, 155, 214, 227, 235, 509
Routledge Publisher 18, 257
rule of thumb 9, 47, 97, 269, 279, 290–293
Russian 36, 127, 302
Ryan, R. 72

sample 3, 7–11, 14–16, 47–51, 62–67, 79–85, 101–103, 146–148, 153–154, 205–207, 228–229, 254–255, 504–505
Sandel, M. 122
Sandelowski, M. 72
sarcasm 107
Sarkane, E. 256
Sartori, G. 127
saturation 97, 195, 411
Saude Coletiva 257
scale 63–65, 470–473, 482–484
scatterplot 400–402, 430–431, 464
Schäffer, B. 275, 292
Schank, R. 36
scheme 78–81, 97–98, 103–105, 193–195, 212–214, 312–314, 467–468, 571–573
Schneijderberg, C. 2, 8, 47, 50, 63, 72, 242, 278
Schneiker, A. 39
scholar 242–247
Schoonenboom, J. 30, 38
Schreier, C. 23, 29, 32–33, 62, 70, 105
Schütz, A. 17, 266, 286
Schütze, H. 266
Schutzer, D. B. 257
Schwarz-Friesel, M. 24–25
science 26–29, 40–42, 53–55, 64–66, 103–104, 107–110, 118–119, 122–133, 245–246, 251–254, 264–268, 301–303, 316–318, 385–386, 404–407, 438–448
scientist 109–113, 127–132
Scopus 242, 246, 249
scraping 54, 247, 318–319
scree plot 411, 423, 425–426
screenplays 44, 53
screenshot 243–244, 316–317, 324

script 112–122
seaborn 332–334, 385–387, 392–402, 488–489, 518–520, 529–530
Seaver, N. 40
Seawright, J. 47–48
secondary data analysis 46, 75–79, 213–215
secrets 55
security 58, 110, 179
segment 96–97, 106–107, 116–117, 121–123, 138–139, 164–170, 195–197, 225–230
Selvi, A. F. 257
semantic 6–8, 309–310, 449–462, 467–475, 490–491, 524–525, 557–558
semantic network 449–462, 467–475, 479–480
semi-structured data 320
semi-supervised machine learning 306–307
sentiment 5–6, 9–12, 34–38, 63–65, 93–95, 98–99, 302–303, 306–317, 332–334, 349–351, 374–375, 562–565
sentiment analysis 5–6, 10–11, 302–303, 307–308, 314–315, 482–523, 564–565
sequence 38–40, 90–91, 96–97, 116–118, 121–122, 132–133, 281–283, 291–293, 314–315, 338–339, 341–345, 349–354, 362–366, 471–472, 490–492, 565
serendipity 96, 137
series 533–535
sexism 273, 277
shallow neural network 306, 308
Shannon, S. 32–33, 255, 265
Sharifi, H. 256
Shepperd, M. 65
Shetty, P. 279
shitstorm 53
shortcuts 83, 277, 334–338
sign 163, 187, 218
significance 45–48, 76–78, 201–202
similarities 268–269, 582–583
Sirota, D. 108
skeptical 112
skill 275–277
sklearn 364–367, 374–378, 412–414, 452–453, 561–562, 574–576
Skrondal, A. 18
slacker 227, 234
Smelser, N. 8, 48, 60, 125–126
smiley 29
Snapchat 52
SocArXiv 270n3
socialization 24, 70, 123, 130
society 22–24, 438–439, 442–443
sociocultural 275, 288
sociolect 175, 195
sociolinguistic 4, 34, 208–211, 218–219
sociology 25–26, 255–257, 267–269, 412–413, 436–451, 475–480, 526–527
sociotechnical 28
software 2–4, 11–14, 34–37, 148–151, 189–192, 229–231, 290–292
Solan, Z. 37
sozmethode 261, 263–265, 269
spaCy 331–333, 362–363, 368–370, 374–378, 412–415, 452–453, 529–531, 560–561, 573
spam 36, 302
spatial 78, 179, 214, 237, 289, 558
speaker 22, 63, 189, 409, 449, 523, 580
specialist 226
spreadsheet 153, 243, 259, 319, 377
Springer 246
Spyder 323–328, 334–338, 354
StackOverflow 321
Stamann, C. 23
standard 57–59, 243–245
standardization 50, 67–68, 357
Stanford University 113, 130–132, 368–369
statistics 34–37, 42–43, 61–65, 68–71, 78–79, 100–103, 228–229, 269–271, 279–280, 283–288, 302–307, 319–321, 385–406
Steigleder 33
Steinhardt, I. 55, 139, 149–150, 173, 188, 228, 242, 250, 258–259
Steinke, I. 66, 72
stemming 134, 222–224, 302–304, 356–357
stereotypes 443
Stevens, M. 93, 120, 123–125, 140–141
Stewart, B. M. 265
stimuli 60n1
Stinchcombe, A. 88, 126, 128, 133, 141
stochastic gradient descent 310–312, 508
Stokes, D. 31
stop word 365–366, 369–384, 573–574
Strauss, A. 9, 32–33, 49, 75, 139, 148, 173, 179, 316
string 249–255, 317–318, 340–347, 352–354, 359–360, 420–421

Strübing, J. 72
structural 13–16, *14–15*, 113, 130, 269, 283, 306
structure 5–6, 26–28, 33, 39–40, 174–176, 189–190, 206, 264, 320–323, 341–345, 387–389, 392–393, 407–412, 421–422, 450–452, 479–480, 565–567
structured data 315–317, 320, 323, 356–357, 411
Stuart, D. 55
student 147–159, 164–173, 180–183, 194–196, 200–204, 214–217, 224–227, 231–237
subcategory 66–67, 84–87, 91–93, 96–100, 113–122, 149–153, 176–177, 192–196, 203–206, 236–237
subcodes 116, 193, 197, 207
subconcept 128–131, **129**
subgroups 10, 384, 450–451
subject 22–23, 27–28, 48–49, 55–58, 146–147, 132–133, 201–202, 248–249, 295–296
subjective 17, 103, 241, 286–288, 294, 482
subset 359–361, 385–386, 412, 483, 499, 531
substantive methods concept 2, 61, 63, 102
subtypes 132
summary 13–14, 32–36, 83–85, 138, 147–150, 176–179, 190–193, 198–200, 213–215, 229–231, 239–241, 253–254, 264–265, 268–272, 279–284, 288–292, 352–354, 432–434, 440–441, 449
superficial 208, 231, 235, 275, 277, 290, 293
supergraph 470–471, 474
supervised machine learning 36–37, 384n1
Suri, H. 48
Surita, F. G. 257
surplus 107, 134–136
survey 3, 28–30, 44–47, 55–57, 60–62, 65–71, 74–75, 79–81, 85–87, 94–95, 102–105
Swanborn, P. 66, 72
Swedberg, R. 88
Switzerland 48
symbol 22–25, 134–135, 201–202
synonym 102n1, 207n1, 220, 224–225, 227, 229
syntax 8, 11, 334, 384, 401–402, 406, 566

Tashakkori, A. 38, 60
Tausczik, Y. 265
Teddlie, C. 38, 60
Teixeira, A. 275
Temple-Hub-Incubator-Triangle of legitimate scientific knowledge *124*
tendency 150, 246, 281
tensor 355n5, 494, 524n16, 565
TensorFlow 327, 332–333, 504–511
term frequency-inverse document frequency *7*, 367
terminology *7*–10, 13–17, 36–40, 52–55, 68–70, *73–75*, 78–81, 126–128, 146–147, 159–161, 178–180, 187–188, 194–198, 213–223, 236–238, 262–268, 271–274, 445–446
test set 504–507, 514–522
tests 64–65, 69, *75*, 101, 228, 282, 381, 461
text corpus 4–11, 34–35, 208–213, 218–220, 231, 305–307, 315–317, 333–336, 344–345, 356–358, 363–364, 374–376, 407–409, 412–414, 479–480, 525–528, 532–537, 540–545, 560–563, 572–579, 586–589
Thelwall, M. 55
theme 274, 437, 526, 556
theorem 8
theory 3–4, 8–10, 13–18, 29–41, 46–51, 60–85, 88–106, 120–128, 133–134, 145–151, 173–179, 196–204, 212–215, 268–270, 285–286, 293–295, 296n7, 303–304, 374–376, 446–449, 475–480, 526–527, 578–580
theory-led 3, 13–15, 29–31, 61–63, *74–77*, 83–84, 88–91, 97–102, 104–105
theory-reflexive 3, 88–94, 98–104, 176–179, 196–198
thesis 13–16, *14–15*, 40, 55, *73*, 147, 172, 178, 199, 205, 213, 240
Thévenot, L. 70, 72
Thompson, L. 528
Thorne, J. 278

threads 276
threshold 259, 455–456
TikTok 279, 303, 318
timeline 40n1
timestamps 116, 197, 304, 412, 420, 578–579, 584
Tinto 171
token 209–213, 222–224, 279–282, 343–345, 357–384, 407–419, 426–429, 433–438, 449–486, 490–492, 495–496, 502–503, 506–507, 523–542, 549–552, 555–566, 571–577, 584–586, 589–591
Tolkin, M. 127
tool 257–258, 294–295
topic 6, 9–11, 34–35, 38–39, 98–99, 187–190, 248–250, 289–292, 302–305, 315–317, 374–376, 412–414, 436–438, 449–452, 479–480, 523–530, 536–552, 555–562, 571–591
topic community 473–474, 476, 478
topic hierarchy 7, 559, 572, 577–578, 584–586
topic loading 467–468, 527–528, 545–546, 555–556
TQDM package 332, 334, 493, 496, 560–563, 586
traditional 24, 141, 590
training set 306–307, 504–510, 513–514
transcript 42–43, 85–87, 94, 96–97, 113–114, 121–123, 179–180, 281–282, 318–320
Transformers 333, 376, 490, 493, 501, 505–506, 515
transitivity 451
transparency 32–34, 43–44, 63–65, 78–89, 97–99, 137–138
treatment 257
triangulation 255, 257–258, 267
Trowler, P. 64
truth 17, 67, 103, 255, 286, 296, 504, 515–517, 521
tuple 344–345, 371–373, 378–379, 430–431, 542–543
Turato, E. R. 257
Turobov, A. 274–276
tutorial 151, 189, 218, 222, 230, 279, 324, 338
Twitter 8–9, 23–24, 145–146, 208, 237, 303, 483–484
txt 217–219, 258–259, 262
typology 1–6, 25–29, 32–36, 41–42, 44–55, 61–63, 69–71, 122–124, 131–133, 145–146, 214–215, 271–273, 281–283, 303–305, 308–310, 316–320, 340–351, 356–358

uncensored 279
uncoded 149, 167
unconscious 30
understand 2–10, 14–18, 21–29, 36–38, 55–57, 62–64, 74–75, 78–79, 82–84, 88–91, 103–108, 125–128, 133–141, 174–177, 190–191, 197–208, 271–279, 285–292, 301–302, 305–308, 312–313, 440–445, 494–495, 555–557
undirected edge (network) 450
unidimensional 141n10
Uniform Manifold Approximation and Projection (UMAP) 558, 566–570, 586–587, 590–591
unintended 235
univariate 6, 420
univariate statistics 386, 389–399
unstandardized 50
unstructured data 320, 356
unsupervised 35–36, 63, 306–308, 460, 525
unsupervised machine learning 36, 63, 306
upload 160–167, 185–187, 282–284, 385–387
uppercase 209, 219, 340
USA 58, 109, 118, 127, 129–132, 134, 136, 555
USB 179
user 38–39, 52–56, 303–304

vaderSentiment 332–333, 484–486, 523n4
Vaismoradi, M. 256
Valence Aware Dictionary and Sentiment Reasoner (VADER) 333, 484–489, 493, 497–499, 501–502, 563
validation set 307, 501, 504–507, 511–514
validity 16, 47, 59, 65–70, 101, 103, 443, 504–514
values 70, 202, 310–311, 339–351, 357–361, 380–382, 389–395, 398–399, 407–409, 418–424,

432–434, 450–451, 454–468, 474–476, 481–491, 498–499, 504–508, 511–514, 517–520, 537–541, 547–549, 582–586
variable 6–8, 306–307, 357–366, 369–371, 377–379, 382–404, 407–436, 444–445, 453–454, 467–469, 511–514, 536–545, 577–579
variable (Python) 335–348, 536–538
variance 408–411, 422–425, 430–434
Vatin, F. 70, 202
verb 64, 209, 223, 362–363, 374, 414–415
verbal 57
verification 145, 251, 277, 296
video 41–60, 303–305, 385–387, 390–391, 398–408, 436–443, 477–479
violin plot 386, 391–392, 396–397
virtual 38
visualization 6–7, 275–276, 314–315, 333–334, 385–386, 391–392, 399–406, 409–411, 430–431, 488–490, 511–520, 528–530, 580–584, 586–590
VosViewer 2, 18, 258–259, 262
Vukasovic, M. 242

Wachinger, J. 275
Wallace, J. R. 275–276
Wallace, R. 134, 137
weak 230
Weber, M. 17, 265, 286
Web of Science 270n6, 270n9
webpages 52–55, 62, 97, 304–305, 318–319
web scraping 54, 247, 318, 319
weight 5–7, 309–313, 367–368, 450–451, 454–456, 459–462, 467–471, 489–491, 508–512, 565–568, 571–573, 589–590, 592n5
Weisberg, H. 65
WhatsApp 156, 164, 165, 181, 194, 207n4, 230
while loop 350–351
Whitley, R. 64
Wieczorek, O. 72, 301, 376, 407–410, 580, 591
Willke, H. 52
Windows 151, 163, 218, 243, 324, 541
winter 154–156, 179–181, 194–195, 205–206
Wohlrab-Sahr, M. 27
woke 160, 170, 185, 221, 227, 234
Wolf, T. 333
Woolgar, S. 64
wording 106–107, 164, 221
workflow 5–6, 12–14, 314–316, 386–391, 410–411, 451–453, 500–501, 528–531, 559–560
workload 12–13, *12*
Wucherpfennig, F. 257

xAI 272–273, 279
Xiao, Z. 276

Yale University 122, 140
Yao 65
Yin, R. 65
YouTube 303–304, 385–387, 390–393, 397–404, 428–429, 483–486, 562–563, 572–574

Zenodo 239, 270
Zeta Psi 140n6
Zhang, H. 275
Zhang, Z. 508
Zimmermann, C. 255
Zimmermann, J. 356
Zitt, M. 250
Zoom 156–157, 165–166, 181–182, 191–192, 195–196, 204–206, 234–235