Multilingual Question Answering over Knowledge Graphs

Aleksandr Perevalov

July 15, 2025
Version: 1.0.0

Department of Computer Science
Data Science Group (DICE)

Doctoral Dissertation

Multilingual Question Answering over Knowledge Graphs

A dissertation presented by

Aleksandr Perevalov

to the
Faculty of Computer Science, Electrical Engineering and Mathematics
of
Paderborn University
in partial fulfillment of the requirements for the degree of
Dr. rer. nat.

1. Reviewer Prof. Dr. Axel-Cyrille Ngonga Ngomo
Department of Computer Science
Paderborn University

2. Reviewer Prof. Dr. Andreas Both
Faculty of Computer Science and Media
Leipzig University of Applied Sciences

3. Reviewer Prof. Dr. Mike Scherfner
Department of Computer Science and Languages
Anhalt University of Applied Sciences

Supervisors Prof. Dr. Axel-Cyrille Ngonga Ngomo
Prof. Dr. Andreas Both

July 15, 2025

Aleksandr Perevalov
Multilingual Question Answering over Knowledge Graphs
Doctoral Dissertation, July 15, 2025
Reviewers: Prof. Dr. Axel-Cyrille Ngonga Ngomo, Prof. Dr. Andreas Both
and Prof. Dr. Mike Scherfner
Supervisors: Prof. Dr. Axel-Cyrille Ngonga Ngomo and Prof. Dr. Andreas Both

Paderborn University
Data Science Group (DICE)
Faculty of Computer Science, Electrical Engineering and Mathematics
Department of Computer Science
Warburger Str. 100
33098 and Paderborn

Abstract

Recent statistics on the Web's content language and users' spoken languages suggest that 49.2% of the content on the Web is accessible to a minor share of the Web users (25.9%)—English speakers. In addition, the Web has multiple language barriers e.g., geographically local content tends to be more comprehensive in the native language of the region. For example, information about German cities is often more detailed in sources in German than in other languages. Hence, researching approaches that enable non-English speakers to access Web information using questions in their native languages is highly valuable as it reduces information accessibility barriers on the Web. The machine-readable data on the Web, published according to the Semantic Web standards, forms a giant global *Knowledge Graph*. Consequently, *Knowledge Graph Question Answering systems* often serve as a core component in fulfilling the information needs of search engine users by providing direct answers to questions asked over the data on the Web in multiple languages. However, the multilingual aspect, particularly regarding *low-resource languages*, is often overlooked in the Knowledge Graph Question Answering field. This issue mainly arises due to the lack of reliable evaluation data and insufficient research on how state-of-the-art Knowledge Graph Question Answering systems perform in languages other than English.

Our thesis aims to answer a fundamental research question: how to enable multilingual Knowledge Graph Question Answering systems to deliver equally good quality in multiple languages, including low-resource ones? To answer this question, we systematically review and evaluate existing approaches, create a new multilingual benchmarking dataset, and propose novel methods to improve the quality of the systems while preserving multilingual functionality. Our results contribute to the research field by (a) systematically analyzing existing benchmarking datasets and systems, with a specific focus on multilingual Knowledge Graph Question Answering, (b) creating a novel multilingual benchmarking dataset, (c) implementing individual components for Knowledge Graph Question Answering systems and empirically evaluating them, and, finally, (d) developing a Knowledge Graph Question Answering system that explicitly addresses the outlined challenges, particularly regarding quality improvement in multilingual settings.

This thesis is structured in five parts, each addressing fundamental aspects of multilingual Knowledge Graph Question Answering.

The first part presents a literature review and proposes a taxonomic framework for the systematic categorization and analysis of existing multilingual Knowledge Graph Question Answering systems. This review serves as a foundation for understanding the current landscape of multilingual Knowledge Graph Question Answering and its research gaps. In particular, several key issues have been identified. First, there is a lack of reliable benchmarking datasets. Second, no investigation has been conducted on machine translating multilingual input to the languages supported by systems. Third, there is an absence of approaches that validate output of Knowledge Graph Question Answering systems. Fourth, no *large language model agent-based systems* have been proposed that were evaluated on multiple languages within the task of Knowledge Graph Question Answering.

Considering the aforementioned challenge regarding the lack of benchmarking data, the second part of our thesis introduces a novel multilingual benchmarking dataset for multilingual Knowledge Graph Question Answering—QALD-9-plus, accompanied by a generalizable methodology for creating and extending such benchmarks. The benchmark contains questions in 10 languages: English, German, Spanish, French, Lithuanian, Russian, Ukrainian, Armenian, Belarusian, and Bashkir (the latter two are considered potentially vulnerable by UNESCO). We reuse the QALD-9-plus benchmark in the subsequent parts of this dissertation.

The third part examines the viability of *Machine Translation* as an alternative to dedicated multilingual systems in Knowledge Graph Question Answering and the implications of its usage in such systems. In this part, we connect machine translation tools with Knowledge Graph Question Answering systems to extend their multilingual capabilities and compare the resulting quality. We evaluated this approach on our QALD-9-plus benchmark.

The fourth part proposes a novel approach to improving the quality of multilingual Knowledge Graph Question Answering systems by integrating language models for response validation. The system's response, a ranked list of SPARQL query candidates, must be validated to avoid incorrect queries. Therefore, the incorrect queries are filtered (i.e., excluded) from the ranked SPARQL query candidate list. This approach was evaluated on our QALD-9-plus benchmark.

The fifth part presents a multilingual Knowledge Graph Question Answering system based on a large language model agent, incorporating multiple components. These components include an environmental feedback mechanism, long-term memory

capability, planning functionality, and a tool-calling interface. The components include machine translation and SPARQL query validation from the third and fourth parts. In addition, we provide a detailed analysis of how these components influence response quality and analyze their cost efficiency. This part also uses the QALD-9-plus benchmark for the evaluation.

The dissertation concludes with a comprehensive discussion that identifies emerging challenges and proposes new research gaps and questions, and synthesizes broader insights into the field of multilingual Knowledge Graph Question Answering derived from this body of research.

Kurzfassung

Aktuelle Statistiken über die Sprache der Web-Inhalte sowie über die Sprachen der Web-Nutzer legen nahe, dass (49.2 %) des Inhalts im Web nur von einem geringen Anteil der Web-Nutzer (25.9 %), insbesondere den Englischsprechenden, zugänglich ist. Zudem bestehen im Web viele sprachliche Barrieren, da lokale Inhalte meist umfangreicher in der Muttersprache der jeweiligen Region verfügbar sind. Zum Beispiel sind Informationen über deutsche Städte in deutschsprachigen Quellen oft detaillierter als in anderen Sprachen. Folglich ist die Erforschung von Ansätzen, die es nicht-englischsprachigen Nutzern ermöglichen, durch Fragen in ihrer Muttersprache auf Web-Informationen zuzugreifen, höchst wertvoll, da es Barrieren bei der Informationszugänglichkeit reduziert. Die maschinenlesbaren Daten im Web, die nach den Standards des Semantic Web veröffentlicht sind, stellen einen globalen Wissensgraphen (engl. *Knowledge Graph*) dar. Daher dienen Frage-Antwort-Systeme über Wissensgraphen (sog. *Knowledge-Graph-Question-Answering-Systeme*) oft als zentrale Komponente, um das Informationsbedürfnis von Nutzern von Suchmaschinen zu erfüllen, indem sie direkte Antworten zu Fragen, die in mehreren Sprachen gestellt werden können, bereitstellen, die mit Web-Daten beantwortet werden können. Jedoch wird der mehrsprachige Aspekt, insbesondere hinsichtlich ressourcenarmer Sprachen (engl. *low-resource languages*), im Bereich des Knowledge Graph Question Answering häufig vernachlässigt. Dieses Problem ergibt sich insbesondere aus dem Mangel an verlässlichen Evaluierungsdaten sowie aus der unzureichenden Forschung darüber, wie (gut) heutige Knowledge-Graph-Question-Answering-Systeme in anderen Sprachen als Englisch funktionieren.

Diese Dissertation zielt darauf ab, eine grundlegende Forschungsfrage zu beantworten, nämlich wie mehrsprachige Knowledge-Graph-Question-Answering-Systeme aufgebaut werden können, die über mehrere Sprachen hinweg gleichbleibend gute Ergebnisqualität liefern, einschließlich ressourcenarmer Sprachen. Um diese Frage zu beantworten, analysieren und evaluieren wir systematisch existierende Ansätze, erstellen einen neuen mehrsprachigen Benchmark-Datensatz und schlagen neuartige Methoden vor, die die Qualität der Systeme verbessern und gleichzeitig ihre mehrsprachige Funktionalität erhalten. Diese Dissertation leistet somit folgende Beiträge zur Forschung: (a) eine systematische Analyse existierender Benchmark-Datensätze und Systeme mit besonderem Fokus auf mehrsprachiges Knowledge

Graph Question Answering, (b) die Erstellung eines neuartigen mehrsprachigen Benchmark-Datensatzes, (c) die Implementierung von Komponenten für Knowledge-Graph-Question-Answering-Systeme sowie deren empirische Evaluation und schließlich (d) die Entwicklung eines Knowledge-Graph-Question-Answering-Systems, welches ausdrücklich die genannten Herausforderungen adressiert, insbesondere im Hinblick auf Qualitätsverbesserungen in mehrsprachigen Szenarien.

Diese Dissertation ist in fünf Teile gegliedert, wobei jeder Teil grundlegende Aspekte des mehrsprachigen Knowledge Graph Question Answering behandelt.

Der erste Teil präsentiert eine ausführliche Literaturübersicht und schlägt einen taxonomischen Rahmen zur systematischen Kategorisierung sowie Analyse existierender mehrsprachiger Knowledge-Graph-Question-Answering-Systeme vor. Diese Übersicht dient als Grundlage für das Verständnis der aktuellen Forschungslandschaft im Bereich des mehrsprachigen Knowledge Graph Question Answering sowie zur Identifikation existierender Forschungslücken, insbesondere: (1) der Mangel an zuverlässigen Benchmark-Datensätzen, (2) die fehlende Untersuchung zur maschinellen Übersetzung mehrsprachiger Eingaben in die von Systemen unterstützten Sprachen, (3) das Fehlen von Ansätzen zur Validierung der Ausgabe von Knowledge Graph Question Answering-Systemen und (4) die Abwesenheit von agentengestützten Systemen, die auf großen Sprachmodellen basieren (*large language model agent-based systems*), die im Kontext von mehrsprachigem Knowledge Graph Question Answering systematisch evaluiert wurden.

Daher stellt der zweite Teil der Dissertation einen neuartigen mehrsprachigen Benchmark-Datensatz QALD-9-plus für mehrsprachiges Knowledge Graph Question Answering vor, zusammen mit einer verallgemeinerbaren Methodik zur Erstellung und Erweiterung solcher Benchmarks. Dieser Benchmark umfasst Fragen in zehn Sprachen: Englisch, Deutsch, Spanisch, Französisch, Litauisch, Russisch, Ukrainisch, Armenisch, Belarussisch und Baschkirisch (wobei die beiden letztgenannten von der UNESCO als potenziell gefährdet eingestuft werden). Der QALD-9-plus-Benchmark wird in den anderen Teilen dieser Dissertation wiederverwendet.

Der dritte Teil untersucht die Eignung der maschinellen Übersetzung (engl. *Machine Translation*) als Alternative zu spezialisierten mehrsprachigen Systemen im Bereich des Knowledge Graph Question Answering sowie die Implikationen des Einsatzes in solchen Systemen. In diesem Teil integrieren wir Werkzeuge zur maschinellen Übersetzung in bestehende Knowledge-Graph-Question-Answering-Systeme, um deren mehrsprachige Fähigkeiten zu erweitern, und vergleichen die resultierende Qualität. Diesen Ansatz evaluieren wir anhand unseres QALD-9-plus-Benchmarks.

Der vierte Teil schlägt einen neuartigen Ansatz zur Verbesserung der Qualität von mehrsprachigen Knowledge-Graph-Question-Answering-Systemen vor, indem Sprachmodelle zur Validierung der Antworten integriert werden. Die Antwort eines solchen Systems ist eine sortierte Liste von Anfragekandidaten, welche in der Abfragesprache SPARQL ausgedrückt sind und validiert werden müssen, um falsche Anfragen zu vermeiden. Hierbei werden inkorrekte Anfragen aus der sortierten Liste der SPARQL-Anfragekandidaten herausgefiltert (d. h. ausgeschlossen). Dieser Ansatz wurde mit unserem QALD-9-plus-Benchmark evaluiert.

Der fünfte Teil präsentiert ein mehrsprachiges Knowledge-Graph-Question-Answering-System auf Basis eines agentengestützten großen Sprachmodells (engl. large language model agent), das aus mehreren Komponenten besteht. Diese Komponenten beinhalten einen Feedback-mechanismus mit der Umgebung, eine Long-Term-Memory-Funktion, eine Planungsfunktionalität sowie eine Schnittstelle zur Nutzung externer Werkzeuge (engl. tool-calling interface). Zu diesen Komponenten gehören auch die maschinelle Übersetzung und die SPARQL-Anfragevalidierung aus dem dritten und vierten Teil. Darüber hinaus bieten wir eine detaillierte Analyse darüber, wie diese Komponenten die Antwortqualität beeinflussen, und bewerten ihre Kosteneffizienz. Auch in diesem Teil verwenden wir den QALD-9-plus-Benchmark für die Evaluation.

Die Dissertation schließt mit einer umfassenden Diskussion, in der aufkommende Herausforderungen identifiziert, neue Forschungslücken und -fragen aufgezeigt sowie weiterführende Einsichten in den Bereich des mehrsprachigen Knowledge Graph Question Answering aus den Ergebnissen dieser Forschungsarbeit synthetisiert werden.

Publications

In the following, we list publications of the author of this thesis. All publications went through a peer-review process and were accepted at the respective conferences. Publications on which this thesis is based are marked with the "⋆" symbol.

The conference ranks were derived from the CORE rankings database[1] and Qualis[2], respectively, and refer to the year when a considered publication was submitted and accepted. The journal impact factors were derived from Clarivate Journal Citation Reports[3]. The particular contributions of the thesis author according to the CRediT Taxonomy[4] are listed in the subsequent chapters.

Conference Papers

1. ⋆ **Aleksandr Perevalov**, Dennis Diefenbach, Ricardo Usbeck, and Andreas Both. QALD-9-plus: A multilingual dataset for question answering over DBpedia and Wikidata translated by native speakers. In 2022 IEEE 16th International Conference on Semantic Computing (ICSC), pp. 229-234. IEEE, 2022.
2. ⋆ **Aleksandr Perevalov**, Andreas Both, Dennis Diefenbach, and Axel-Cyrille Ngonga Ngomo. Can Machine Translation be a Reasonable Alternative for Multilingual Question Answering Systems over Knowledge Graphs?. In Proceedings of the ACM Web Conference 2022, pp. 977-986. 2022.
3. ⋆ **Aleksandr Perevalov**, Yan, Xi, Kovriguina, Liubov, Jiang, Longquan, Both, Andreas and Usbeck, Ricardo. Knowledge Graph Question Answering Leaderboard: A Community Resource to Prevent a Replication Crisis. In 2022 Proceedings of the 13th Language Resources and Evaluation Conference, pages 2998–3007, Marseille, France. European Language Resources Association.
4. Aleksandr Gashkov, **Aleksandr Perevalov**, Maria Eltsova, and Andreas Both. Improving question answering quality through language feature-based SPARQL query candidate validation. In European Semantic Web Conference, pp. 217-235. Cham: Springer International Publishing, 2022.

[1] https://portal.core.edu.au/conf-ranks/
[2] https://qualis.ic.ufmt.br/
[3] https://jcr.clarivate.com/jcr/home
[4] https://credit.niso.org/

5. **Aleksandr Perevalov** and Andreas Both. Improving answer type classification quality through combined question answering datasets. In Knowledge Science, Engineering and Management: 14th International Conference, KSEM 2021, Tokyo, Japan, August 14–16, 2021, Proceedings, Part II 14, pp. 191-204. Springer International Publishing, 2021.
6. ⋆ Nikit Srivastava, **Aleksandr Perevalov**, Denis Kuchelev, Diego Moussallem, Axel-Cyrille Ngonga Ngomo, and Andreas Both. "Lingua Franca–Entity-Aware Machine Translation Approach for Question Answering over Knowledge Graphs." In Proceedings of the 12th Knowledge Capture Conference 2023, pp. 122-130. 2023.
7. Aleksandr Gashkov, **Aleksandr Perevalov**, Maria Eltsova, and Andreas Both. "Improving the question answering quality using answer candidate filtering based on natural-language features." In 2021 16th International Conference on Intelligent Systems and Knowledge Engineering (ISKE), pp. 635-642. IEEE, 2021.
8. Annemarie Wittig, **Aleksandr Perevalov**, and Andreas Both. "Towards bridging the gap between knowledge graphs and chatbots." In International Conference on Web Engineering, pp. 315-322. Cham: Springer International Publishing, 2022.
9. ⋆ **Aleksandr Perevalov**, Aleksandr Gashkov, Maria Eltsova, and Andreas Both. Towards Knowledge Graph-Agnostic SPARQL Query Validation for Improving Question Answering. In European Semantic Web Conference, pp. 78-82. Cham: Springer International Publishing, 2022.
10. **Aleksandr Perevalov**, Aleksandr Vysokov, and Andreas Both. Linguistic difference of human-human and human-chatbot dialogues about COVID-19 in the russian language. International Conference on Applied Innovation in IT, 2022.
11. ⋆ **Aleksandr Perevalov**, Aleksandr Gashkov, Maria Eltsova, and Andreas Both. Language Models as SPARQL Query Filtering for Improving the Quality of Multilingual Question Answering over Knowledge Graphs. In International Conference on Web Engineering, pp. 3-18. Cham: Springer Nature Switzerland, 2024.
12. **Aleksandr Perevalov**, Aleksandr Gashkov, Maria Eltsova, and Andreas Both. Understanding SPARQL Queries: Are We Already There? Multilingual Natural Language Generation Based on SPARQL Queries and Large Language Models. In International Semantic Web Conference, pp. 173-191. Cham: Springer Nature Switzerland, 2024.
13. **Aleksandr Perevalov**, Andreas Both, and Mike Scherfner. Towards Automated Semantics-Driven Web Service Composition: Case Study on Question

Answering Systems. In 2024 IEEE 18th International Conference on Semantic Computing (ICSC), pp. 41-48. IEEE, 2024.

14. ⋆ **Aleksandr Perevalov**, Sara Abdollahi, Simon Gottschalk, and Andreas Both. Evaluating Entity Importance in a Cross-National Context using Crowdsourcing and Best–Worst Scaling. Procedia Computer Science 246 (2024): 1479-1487.
15. Dennis Schiese, **Aleksandr Perevalov**, and Andreas Both. Post-Hoc Insights: Natural-Language Explanations for AI-Enhanced/-Integrated Software Systems. In SEMANTiCS (2024).

Journal Articles

- ⋆ Ricardo Usbeck, Xi Yan, **Aleksandr Perevalov**, Longquan Jiang, Julius Schulz, Angelie Kraft, Cedrick Möller, Junbo Huang, Jan Reineke, Axel-Cyrille Ngonga Ngomo, Muhammad Saleem and Andreas Both, QALD-10–The 10th challenge on question answering over linked data. Semantic Web, pp.1-15. 2023.
- ⋆ **Aleksandr Perevalov**, Andreas Both, and Axel-Cyrille Ngonga Ngomo. Multilingual question answering systems for knowledge graphs–a survey. Semantic Web 2024.

Workshop Papers, Posters & Others

1. **Aleksandr Perevalov**, and Andreas Both. Augmentation-based Answer Type Classification of the SMART dataset. In SMART@ International Semantic Web Conference, pp. 1-9. 2020.
2. **Aleksandr Perevalov**, Axel-Cyrille Ngonga Ngomo, and Andreas Both. "Enhancing the accessibility of knowledge graph question answering systems through multilingualization." In 2022 IEEE 16th International Conference on Semantic Computing (ICSC), pp. 251-256. IEEE, 2022.
3. Andreas Both, **Aleksandr Perevalov**, Johannes Richard Bartsch, Paul Heinze, Rostislav Iudin, Johannes Rudolf Herkner, Tim Schrader, Jonas Wunsch, Ann Kristin Falkenhain, and René Gürth. A Question Answering System for retrieving German COVID-19 data driven and quality-controlled by Semantic Technology. In SEMANTiCS (Posters & Demos). 2021.
4. Andreas Both, Paul Heinze, **Aleksandr Perevalov**, Johannes Richard Bartsch, Rostislav Iudin, Johannes Rudolf Herkner, Tim Schrader, Jonas Wunsch, René Gürth, and Ann Kristin Falkenhain. Quality assurance of a german COVID-19 question answering systems using component-based microbenchmarking. In

Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining, pp. 1561-1564. 2022.

5. Kunpeng Guo, Dhairya Khanna, Dennis Diefenbach, **Aleksandr Perevalov**, and Andreas Both. WikidataComplete–An easy-to-use method for rapid validation of text-extracted new facts applied to the Wikidata knowledge graph. In European Semantic Web Conference, pp. 118-122. Cham: Springer International Publishing, 2022.
6. **Aleksandr Perevalov**, and Andreas Both. Hierarchical Expected Answer Type Classification for Question Answering. In International Semantic Web Conference (Posters/Demos/Industry). 2021.
7. **Aleksandr Perevalov**, and Andreas Both. Towards LLM-driven Natural Language Generation based on SPARQL Queries and RDF Knowledge Graphs. 3rd International workshop one knowledge graph generation from text @ Extended Semantic Web Conference (2024).
8. **Aleksandr Perevalov**, Andreas Both, Florian Gudat, Paul Brauning, Johannes Meesters, Lennart Grundel, Marie-Susann Bachmann, and Salem Zin Iden Naser. Qanary Builder: Addressing the Reproducibility Crisis in Question Answering over Knowledge Graphs. In International Semantic Web Conference (Posters/Demos/Industry). 2023.
9. **Aleksandr Perevalov**. Enhancing Multilingual Accessibility of Question Answering over Knowledge Graphs. In Companion Proceedings of the Web Conference 2022, pp. 349-353. 2022.
10. **Aleksandr Perevalov** and Andreas Both. Mutilingual Hierarchical Expected Answer Type Classification using the SMART 2021 Dataset. In SMART@ International Semantic Web Conference, pp. 56-65. 2021.
11. Andreas Both, **Aleksandr Perevalov**, and Aleksandr Gashkov. KinGVisher–Extended Semantic Web Conference, 2024.
12. **Aleksandr Perevalov**, Jiveshwari Chinchghare, Shivam Mouli Krishna, Amal Nimmy Lal Sharma, Aryman Deshwal, Andreas Both, and Axel-Cyrille Ngonga-Ngomo. CLASS MATE: Cross-Lingual Semantic Search for Material Science driven by Knowledge Graphs. Posters & Demos @ Extended Semantic Web Conference 2024.
13. ⋆**Aleksandr Perevalov**, Andreas Both. Text-to-SPARQL Goes Beyond English: Multilingual Question Answering Over Knowledge Graphs through Human-Inspired Reasoning. First International TEXT2SPARQL Challenge Co-Located with Text2KG @ Extended Semantic Web Conference 2025. (Accepted, not published yet).

Acknowledgement

This thesis would never have come to life without the active support of my supervisors, colleagues, family, and friends.

I would like to express my special gratitude to Prof. Dr. Andreas Both, who has been teaching, supporting, supervising, and providing guidance for my work since my master's thesis. Thank you for giving me the opportunity to join the research community!

I would like to thank Prof. Dr. Axel-Cyrille Ngonga Ngomo for agreeing to supervise my thesis and for being willing to provide precise and valuable answers to my questions throughout my PhD journey.

I am grateful to Prof. Dr. Mike Scherfner for his advice and support during my time at Anhalt University of Applied Sciences. Your support ensured that my work did not come to a halt midway.

Special thanks to all the colleagues with whom I have collaborated and worked, in particular: Prof. Dr. Ricardo Usbeck, Dr. Dennis Diefenbach, Dr. Alexander Gashkov, Dr. Maria Eltsova, Nikit Srivastava, Dr. Michael Röder, Jonas Wagner, Dr. Simon Gottschalk, and Sara Abdollahi.

Another milestone in my PhD journey is connected to the Software Campus program. I would like to thank Stefan Jazdzejewski, Susanne Kegler, and Dr. Natalia Sousa Teixeira-Silva for keeping the program running. Thank you as well to the Springer Materials (Springer Nature) team and its former members: Dr. Marcel Karnstedt Hulpuş, Dr. Volha Bryl, Dr. Stefan Scherer, Dr. Alexander Eckl, Dr. Bin Cheng, and Harald Wirsching. Thank you to the colleagues from Holtzbrinck Publishing Group, especially Julia Furtwängler.

I extend my heartfelt gratitude to my wife, Anna, my two sons, Vladimir and Andrei, and my parents and grandparents, whose patience and understanding allowed me the time and focus needed to finish this work.

Contents

1 Introduction

> *People who come from different cultures and use different languages work to make sure that all kinds of languages, whether read from left or right, can be shared and used on the Web.*
>
> **— Tim Berners-Lee**
> (W3C Director and inventor of the WWW)

The search engine landscape, which processes billions of queries daily [@87], is evolving with the emergence of conversational systems powered by *large language models* (LLMs). A notable example is Perplexity AI, which reported handling 100 million weekly search queries in October 2024 [@112], indicating the increasing market presence of these new search technologies. More than half of these queries are *informational*, i.e., are sent by users with an information need related to a certain topic [92]. Informational queries can be formulated as full-fledged *natural language* (NL) questions (e.g., "How old is Angela Merkel?") or as *keyword-based* questions (e.g., "Angela Merkel age"). In both cases, Web users expect a search engine to provide a *direct answer* (or a fact—an assertion that was verified) in a precise way instead of a list of relevant Web pages or documents [189, 160]. An example of a direct answer and the answer box is shown in Figure 1.1. The assertions can be extracted from *unstructured data* (e.g., text or HTML document) or *structured*

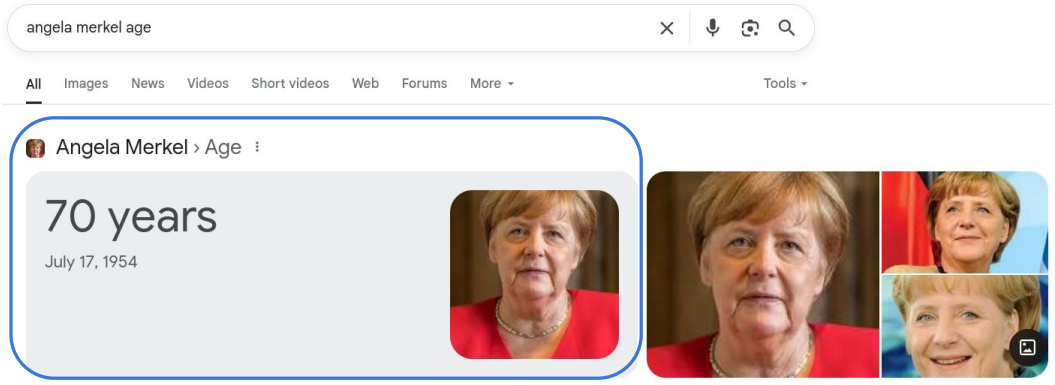


Fig. 1.1.: Direct answer (blue) [140] generated by Google Search.

data (e.g., database, knowledge base, or graph). Thereafter, the assertions are verified and become facts that can be provided to users as direct answers. The systems dealing with direct answer extraction on unstructured data are known as *Information Retrieval-based Question Answering* (IRQA) systems [98]. On the other hand, systems that perform a direct answer extraction over structured data are called *Knowledge-based Question Answering* (KBQA) systems [98]. Our thesis *focuses on the latter class* of systems. *The objective of KBQA* is to identify the answers $\mathcal{A}$ that satisfy the informational need of a user formed as a NL question q, using a KB $\mathcal{K}$ [57]. Recent work on developing KBQA systems focuses on two development paradigms [228, 223, 125]: (1) the information extraction style and (2) the semantic parsing style. The systems following *first development paradigm of KBQA* usually retrieve a set of answer candidates that are ranked or filtered based on a particular feature space. The *second development paradigm of KBQA* focuses on the task of converting a NL question q into a formal query ϕ (e.g., SPARQL) that will be executed on a KB $\mathcal{K}$ to retrieve an answer. Recent advances in the field of *Large Language Models* (LLMs) have demonstrated that both development paradigms can be implemented using their capabilities. In particular, a NL question q can be directly posed to an LLM to receive a direct answer (paradigm 1) [10, 125] or used to create an instruction (prompt) for an LLM to generate a SPARQL query (paradigm 2) [107].

The extraction of direct answers from structured data using KBQA systems is enabled by the introduction of *the Semantic Web* [15], which aims to make Web data machine readable. The machine-readable data on the Web that follows the Semantic Web principles corresponds to the formal definition of a Knowledge Graph (KG) [79] and, therefore, can be considered as a giant decentralized KG. A dominant share of all KGs on the Web, as well as the Semantic Web itself, are described with the *Resource Description Framework* (RDF) [122] (RDF-based KGs). A family of systems that address the challenge of providing direct answers based on KGs is known as *Knowledge Graph Question Answering (KGQA) systems*. *KGQA and KBQA systems share the same objective*—deliver an answer that satisfies a user's information need based on structured data. However, a KB $\mathcal{K}$ is replaced with a knowledge graph $\mathcal{KG}$ (in case of KGQA systems). Although the terms KBQA and KGQA are mainly used interchangeably, we prefer to consider *KGQA systems a subset of KBQA*.

Most research-oriented KGQA systems work on publicly available KGs such as Freebase [19] (now part of Google's KG [@64]), DBpedia [7], Wikidata [211]. The techniques implemented therein can commonly be ported to any KG. KGQA systems and their underlying technologies are serving a major role in providing users with access to structured information available on the Web. This statement is supported by a number of research papers related to KGQA and affiliated with

the organizations responsible for the modern major search engines [12, 129, 162]. KGQA systems find their *industrial application* in any place where the corporate knowledge is stored as a KG. For example, a product or its development process documentation can be stored in structured RDF graphs [159]. Another example is material science data [75] or chemical data [89]. Hence, such data, structured as RDF, enable users to build a KGQA system on top of it and utilize the provenance metadata. Recent work has provided the KGQA community with such *applicability domains* as tourism [2], biomedicine [149], and even "COVID-19 pandemic" [23]. However, we have observed that *providing access to KGs in multiple languages is still a challenge* for such systems [154]. In the following paragraphs, we contextualize this challenge in detail.

1.1 Motivation and Problem Statement

It is well known that the Web is the major source of information for people all over the world [102, 45, 180] in different domains of life. Moreover, the recent advent of LLMs is largely based on the availability of Web data [9]. Despite the large share of the information on the Web (49.2%) [@215] is only accessible to a small fraction of people (25.9%) [@88], that is, English speakers. This information imbalance is also the reason for a "cultural gap" on the Web [132]. Consequently, users who cannot read or write English at a certain level have limited access to a major share of the information available on the Web. This phenomenon in popular science is known as *digital language divide* [@80].

The scope of this thesis is focused on *multilingual and cross-lingual KGQA systems (mKGQA)*. In particular, we consider both *resource-rich* and *low-resource* languages. Such systems extend the standard KGQA functionality by *providing a possibility of processing questions or searching for information in several different languages*. In our case, "several" means *more than one*. Please note that, in the scope of this thesis, the terms "cross-lingual" and "multilingual" are used interchangeably. For convenience, we use the latter. Hence, we regard mKGQA systems as a research topic of particular importance, as *these systems can bridge the gap between users and the information on the Semantic Web*, which is published in different languages.

The analysis of related work, conducted using the systematic review methodology within this thesis, suggests (see Chapter 3) that *currently available systematic surveys on KGQA barely address the aspect of multilinguality and low-resource languages*. In particular, most related surveys mention multilinguality as a challenge for KGQA

but dedicate only a paragraph or less to it. In addition, none of the related works specifically focus on the multilingual aspect of these systems. Although there are KGQA systems that explicitly focus on multilinguality, there is a clear need for a larger study that summarizes this research field and proposes novel methods.

1.2 Research Question and Considered Research Gaps

This thesis covers the broad topic of enhancing the answer quality of mKGQA systems, focusing on multilinguality and low-resource languages. While analyzing the research field considered (see Chapter 3), we identified that the multilingual aspect is often overlooked. In particular, while multilinguality is consistently identified as a crucial challenge in KGQA systems, none of the reviewed surveys specifically focus on this aspect. Furthermore, 88% of the surveyed mKGQA systems predominantly focus on the Indo-European language family, which consists mostly of resource-rich languages. Finally, only a limited set of available KGQA benchmarks incorporate multiple languages. Therefore, this thesis aims to answer the following main *research question*: how to enable multilingual Knowledge Graph Question Answering systems to deliver equally good quality in multiple languages, including low-resource ones?

This work is guided by the following *research gaps*:

- $\mathcal{RG}$**1** Open-source benchmarking datasets for KGQA are limited due to the fact that most benchmarks consider only English or target resource-rich languages (e.g., main European languages such as German or French). Furthermore, even a smaller share of the benchmarks considers low-resource languages. Therefore, it is necessary to identify and implement reproducible approaches to create and extend reliable and diverse benchmarking datasets for mKGQA.
- $\mathcal{RG}$**2** Most of the KGQA systems support only resource-rich languages or only English. Consequently, even with the benchmarking data in a low-resource language, evaluation is impossible. Hence, the research community needs to investigate whether it is possible to handle multilingual support using recent progress in neural machine translation. As a result, one needs to evaluate how uniform the KGQA quality distribution is among different languages. Prior to this work, this approach had not yet been introduced or evaluated. Therefore, we need to study the role of machine translation as a reasonable alternative

for mKGQA systems and conduct the corresponding experiments that include low-resource languages.

RG3 Even though a KGQA system may be able to support a particular language, its output—a SPARQL query—still may appear to be incorrect. When working with users, it is critical to prevent possible misinformation produced by KGQA systems due to incorrect SPARQL queries. Therefore, there is a need to design and implement efficient approaches to validate SPARQL queries produced by mKGQA systems to improve the quality and trustworthiness of the answers. It is crucial that the approach works with multiple languages, including low-resource ones.

RG4 Recent advances in the development of LLM enabled researchers to build systems that simulate human thinking and decision-making processes (e.g., through planning, recapture of prior experience, or iterating based on external feedback). The same is true when applying LLM-based applications to the field of mKGQA. Due to the novelty of this topic, there is a lack of studies investigating how LLM-based KGQA systems answer questions in different languages, including those with limited resources. Therefore, we as researchers need to study how the new LLM-based approach to mKGQA augmented with external capabilities (e.g., planning, long-term memory, feedback performs in multiple languages, how equal is multilingual performance and compare with previous results based on quality, efficiency, and costs.

Based on the research gaps, the *objective of this thesis* is to provide an overview, evaluate, and propose novel methods for improving mKGQA systems in terms of quality while preserving multilingual functionality. This includes the creation of trustworthy benchmarking datasets, new evaluation techniques, conducting experiments to compare different approaches, and introducing new approaches and methods for mKGQA. Note that in this work, we consider publications that describe mKGQA systems, relevant survey articles, and corresponding benchmarking datasets, as our work aims to cover the research field as broadly as possible.

1.3 Thesis Structure

This thesis is based on 9 peer-reviewed research publications, in which the author of this thesis was a major contributor. The thesis introduces the reader to the topic of multilingual knowledge graph question answering and sequentially presents the author's contributions to the research field. The following paragraphs introduce the structure of this thesis.

Chapter 2 introduces the reader to the theoretical and technical foundations of this thesis. The purpose of this section is to provide a sufficient, and complete terminological, methodological, and technical basis for the reader to fully understand the remaining content. In particular, the concepts of natural language processing, semantic web, knowledge graphs, question answering, and information retrieval are considered.

To analyze the related work (see Chapter 3) and generalize the findings, we employ a systematic review methodology to collect and analyze research results in the field of mKGQA. This methodology involves defining scientific literature sources, selecting relevant publications, extracting objective information (e.g., evaluation values, used metrics, etc.), analyzing the information, searching for new insights, and generalizing them in a structured manner (i.e., in the form of a taxonomy). Finally, we summarize our observations and present a general outlook on the investigated research direction, including justifying our research gaps.

As benchmarking data is crucial for this work, the approach for extending and creating trustworthy benchmarking dataset, as well as mKGQA evaluation, is described in Chapter 4. In particular, contributions are presented on the extension of the multilingual benchmark QALD-9-plus (which also includes low-resource languages) and the creation of the QALD-10 benchmark We leverage QALD-9-plus in the subsequent chapters of this thesis to evaluate the proposed approaches.

In Chapter 5 having the multilingual QALD-9-plus benchmark, we present and evaluate an alternative solution to mKGQA—machine translating input questions to a supported language. In particular, an extensive set of experiments is conducted in order to verify whether the machine translation approach connected with a monolingual KGQA may serve as an alternative to mKGQA systems. However, we identified major limitations of this approach, particularly when translating named entities.

Furthermore, we present an approach for improving semantic parsing-based systems by validating their output (i.e., SPARQL queries) in Chapter 6. Using our QALD-9-plus dataset, we employ different language models (e.g., encoder-decoder and decoder-only) that process an input question and a SPARQL query to verify whether this query will deliver a correct result or not. This approach is crucial when optimizing for the answer's trustworthiness (e.g., reducing false information in the responses).

Section 7 puts the previous work together in a multilingual KGQA system. Using the QALD-9-plus dataset and the other aforementioned approaches (machine translation

and query validation) to show how augmented LLMs (LLM agents) are adapted to work in the mKGQA environment. In particular, we focus on the semantic parsing paradigm and use the LLM agents to answer questions in multiple languages. Those agents can access specific tools, including machine translation of an input and query validation for an output, to improve the quality of SPARQL query generation based on multilingual input.

Section 8 revisits the answers to the research gaps and highlights future challenges in the field of mKGQA.

Finally, in Section 9, we summarize our work and conclude the thesis.

To ensure the transparency and reproducibility of this work, we provide all the data, scripts, and documentation collected as an online appendix[1].

Statement on using generative artificial intelligence tools for writing assistance. Generative artificial intelligence tools were used exclusively to support the writing quality of this thesis. Specifically, grammar and spelling checks were conducted using Writefull[2] and Grammarly[3]. The scientific ideas and contributions presented throughout this thesis are solely original works of the author, and generative artificial intelligence was not employed to create, develop, or inform any scientific content or contributions.

[1] https://github.com/Perevalov/phd_thesis
[2] https://www.writefull.com
[3] https://www.grammarly.com

2 Preliminaries

This chapter introduces the theoretical foundations that are central for this thesis. Mostly, this work is based on the terminology, methods, and tools coming from the Natural Language Processing (NLP), Information Retrieval (IR), and Semantic Web research fields. Further subsections will introduce the aforementioned content in detail.

2.1 Natural Language Processing

2.1.1 Language Modeling

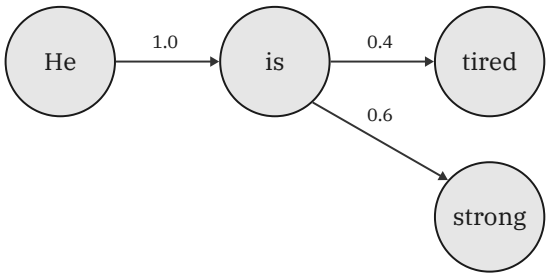
The *language modeling* task is tightly associated with text generation, as language models (LMs) can approximate the likelihood of a next word w_t in a sequence given preceding word context $w_1, \ldots, w_{t-1}$, i.e., $P(w_t|w_1, \ldots, w_{t-1})$ [99]. Generally speaking, LMs assign the probability to upcoming words or sequences of words.

N-gram LMs have been known and used in natural language processing for several decades. Therefore, the probability approximation of a word given its context using an n-gram LM is as follows [99]: $P(w_t|w_{1:t-1}) \approx P(w_t|w_{t-N+1:t-1})$, where t is the number of words in the entire sequence and N is the context size (uni-gram (1), bi-gram (2), etc.). To estimate these n-gram probabilities, the n-gram counts can be utilized (see Equation 2.1).

$$P(w_t|w_{t-N+1:t-1}) \approx \frac{C(w_{t-N+1:t-1}w_n)}{C(w_{t-N+1:t-1})} \tag{2.1}$$

Obviously, the probability estimation of Equation 2.1 may appear naive. Extended methods for probability estimation include smoothing (e.g., *add-k smoothing* [99], *stupid backoff* [25], *Kneser-Ney smoothing* [109]). The high-level schema of the n-gram models is shown in Figure 2.1.

For more than 20 years, research on language modeling has utilized methods based on *neural networks* (NNs). The NN-based LMs are *trained on large text corpora*

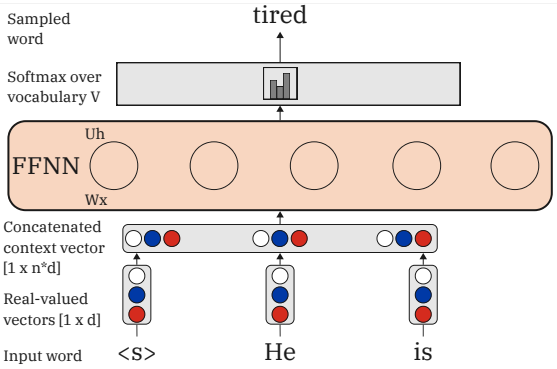


$$P(\text{tired} \mid \text{is}) = \frac{\text{Count(is tired)}}{\text{Count(is)}} = \frac{4}{10}$$

Based on Markov chains and Markov's assumption
Examples (probability estimation): Kneser-Ney smoothing (1995), Stupid backoff (2007).

Fig. 2.1.: A high-level representation of an n-gram LM

in a self-supervised manner (i.e., no explicitly labeled dataset is required). The *feedforward NNs* (FFNNs) for the language modeling task were first proposed by Bengio [13]. Generally speaking, this is an extension of the n-gram LMs where the probability of the word is estimated by NNs. The training process is based on the task of predicting a missing word w in the sequence of context words c: $P(w|c)$, where $|c|$ is the *context window size*. The FFNN LM architecture is demonstrated in Figure 2.2.

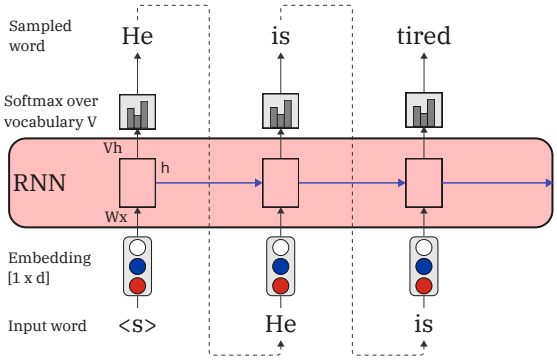


$$P(\text{tired} \mid \text{He is}) = \arg\max_i y_i$$
$$y = \text{softmax}(Uh)$$
$$h = g(Wx)$$

Essentially the same as N-gram models, but the probability is estimated via FF neural network.
Example: Bengio et al. (2000)

Fig. 2.2.: An architecture of a feedforward NN-based LM

Recurrent NNs (RNNs) [59] are also used for constructing LMs [99]. RNNs' advantage is that they are able to capture dependencies of the entire word sequence while adding the hidden layer output from the previous step $(t-1)$ to the input from the current step (t): $h_t = g(Uh_{t-1} + Wx_t)$, where h_j—hidden layer output, $g(x)$—activation function, x_t—input at the current time step, U and W—weight matrices. The RNNs are trained following the self-supervised manner on a large text corpora to predict the next word given a sequence. The text production is then performed via the *autoregressive generation* process, which is also called *causal language modeling* (see Figure 2.3). In practice, canonical RNNs are rarely used for the LM tasks. Instead, there are improved RNN architectures such as *stacked* and *bidirectional* [176], *long short-term memory* (LSTM) [76], and its variations. One of the most remarkable RNN



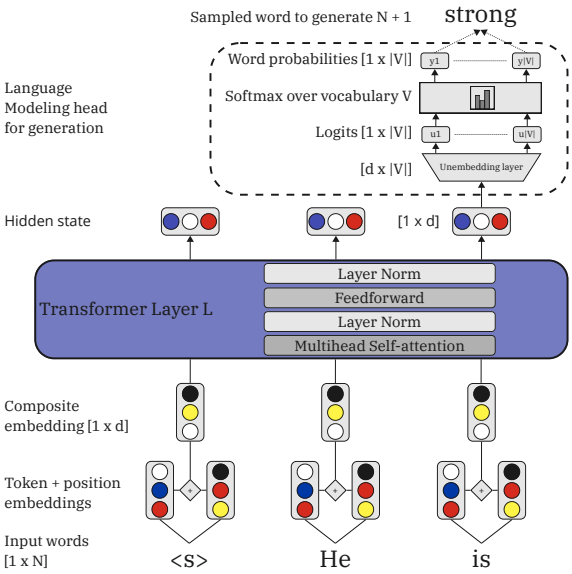
$$P(\text{tired} \mid \text{is}) = \arg\max_i y_{t,i}$$
$$y_t = \text{softmax}(Vh_t)$$
$$h_t = g(Uh_{t-1} + Wx_t)$$

RNN-based LMs by definition take into the account preceeding tokens in a sequence via recurrent connections.
Examples (generation): Seq2seq: Sutskever et al. (2014), Attention: Bahdanau et al. (2015).
Examples (embeddings): ELMo (2018), ULMFiT (2018)

Fig. 2.3.: An example of autoregressive generation with RNN-based LM

architectures was proposed by Sutskever et al. [190] *encoder-decoder* (or seq2seq) LSTM-based architecture. Instead of simple autoregressive generation, the seq2seq model encodes an input sequence to an intermediate representation and then uses autoregressive generation to decode the intermediate representation.

Modern state-of-the-art LMs utilize the *attention mechanism* [11] and, in particular, the *self-attention* [210], which is an integral part of the *transformer* architecture [210]. Figure 2.4 demonstrates the architecture of a transformer LM for text generation.



$$P(\text{strong} \mid \text{He is}) = \arg\max_i y_i$$
$$y = \text{softmax}(u)$$
$$u = h_N^L E^T$$
h_N^L – last hidden state of L-th transformer block
E^T – undembedding matrix

Transformers work in the pre-train and fine-tune paradigm. The pre-trained transformer can solve different tasks with just changing its head (transfer learning).
Examples (encoder): BERT (2018)
Examples (decoder): GPT-2 (2019)
Examples (encoder-decoder): T5 (2020)

Fig. 2.4.: An example of a transformer-based LM set up for text generation

It is worth mentioning different word-sampling strategies when discussing LMs that generate text [99]: *greedy*—select the word with the highest softmax score,

random—select the random word according to the softmax distribution, *top-k*—the same as greedy, but made within a renormalized set of top-k words, *nucleus*—the same as top-k, but instead of k as a number use percentage, *temperature*—divide the logits by a constant to reshape the softmax distribution.

The traditional measure for LM quality is *perplexity* [93]. Perplexity is defined as follows:

$$\text{PP}(p) = 2^{-\frac{1}{N}\sum_{i=1}^{N}\log_2 p(w_i)} \tag{2.2}$$

where p—probability distribution of the words, N—is the total number of words in the sequence, w_i—represents the i-th word. Since Perplexity uses the concept of entropy, the intuition behind it is how uncertain a particular model is about the estimated sequence of words. Hence, a high Perplexity score corresponds to a worse probability estimation of a target token sequence. The instruction-tuned large language models (LLMs) are evaluated on instruction benchmarks, such as Massive Multitask Language Understanding (MMLU) [73], HumanEval for code [33], Mathematical Problem Solving (MATH) [73], and others. Importantly, LLMs are often released with a specified *knowledge cutoff date*, which corresponds to the time when the pre-training dataset for such a model was collected [35].

2.1.2 Sequence Labeling

Sequence labeling is one of the important downstream tasks in NLP. There, for each word (or, generally speaking, for each token) $x_i \in X$, where X is a sequence of words in a text, we assign a label $y_i \in Y$, s.t., $|X| = |Y|$ [99]. In this case, a label y_i can denote, for example, a type of proper noun or a *named entity* (e.g., person, location, chemical compound, etc.). Therefore, *named entity recognition* (NER) is the task of assigning tags like `PERSON`, `LOCATION`, `CHEM_COMPOUND` to words [99]. In practice, the task NER is carried out while finding *spans of text* that contain named entities i.e., not necessarily linked to individual words. The most well-known approaches for sequence labeling are Hidden Markov Models (HMMs) [220, 16], Conditional Random Fields (CRF) [113, 124], RNNs [69, 145], and transformers [185, 6] (the approaches are listed by their historical appearance).

There is a more specific version of the NER task that is called *named entity disambiguation* or *named entity linking* (NEL or NED). This task refers not only to the identification of entities but also to the association of this entity with its unique identifier in a database or a knowledge base [195]. The approaches to solving NEL include traditional search methods on indexes, rule-based methods, and deep learning-based methods [170, 100].

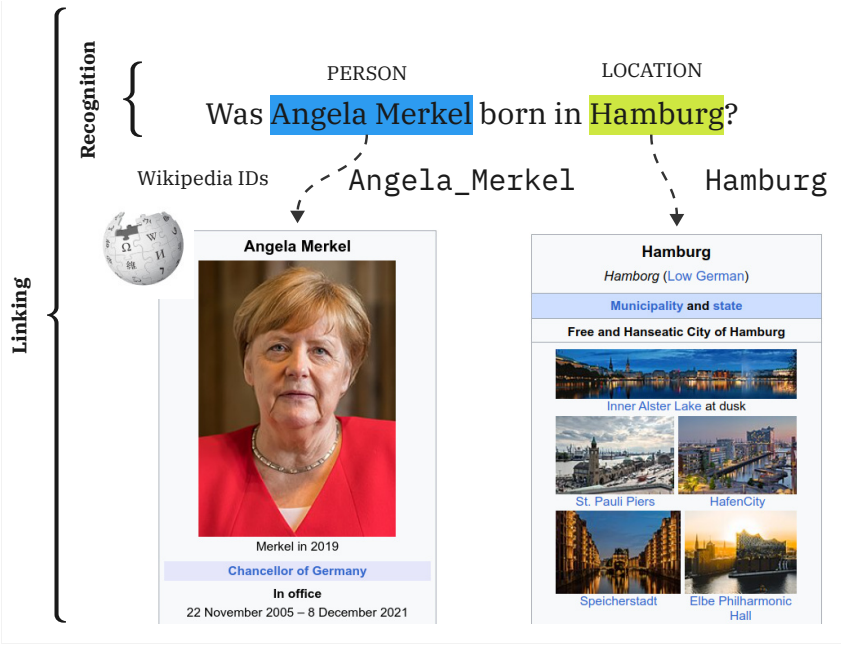


Fig. 2.5.: An example of the relation between named entity *recognition* and *linking*. Here, the linking is done over Wikipedia.

2.1.3 Text Classification

Text classification is the task of assigning a class (or a label) $y_i \in Y$ to an entire text or a document $d_j \in D$ [99]. Hence, the classification algorithm F_{clf} maps an input text d_j to a particular class y_i as follows $F_{clf} : D \to Y$. The class may represent a topic of the given text (e.g., scientific, pop culture), a sentiment (e.g., positive, negative), etc. It is important to mention that there are three types of the classification task:

1. $Y = \{0, 1\}$—binary classification. For example, when labeling whether a scientific article is artificially generated or not.
2. $Y = \{1, \ldots, K\}$—multi-class classification. For example, when classifying a subject area of a scientific article (e.g., computer science, chemistry, mathematics).
3. $Y = \{0, 1\}^K$—multi-label classification (multi-class where multiple classes can be selected). For example, when tagging a scientific article with keywords.

One of the applications of the text classification task is the *relation detection or extraction* (RE) task. Given a predefined *relation set* (each relations is a class), the goal of RE is to determine whether the text indicates any types of relations between the entities or not [226]. Naturally, the RE task can be generalized to such cases where there is more than one relation contained in the text (i.e., multil-label classification). Although this task can also be solved with the means of sequence

labeling, the research suggests that RE is usually formulated as the classification task [226].

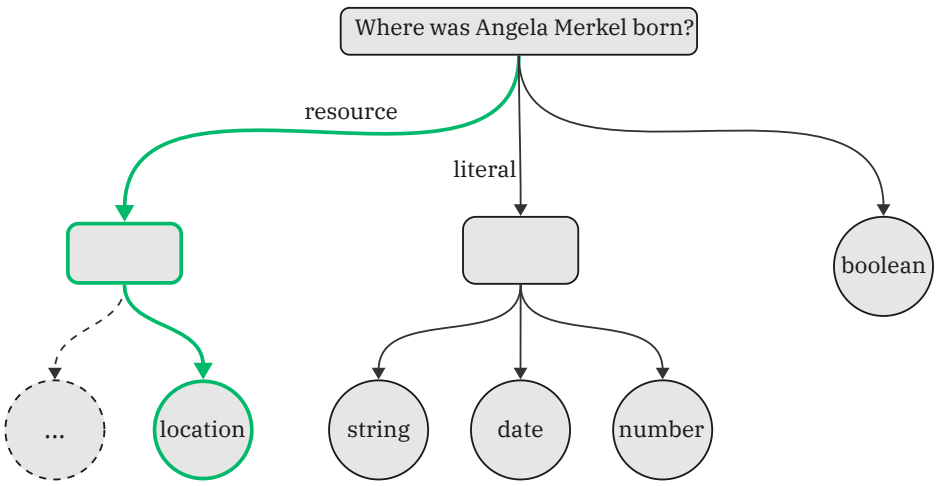


Fig. 2.6.: An example of an expected answer type hierarchy (cf. [152])

As in the classification-formulated RE, the predefined relation set is used, and the individual relations are already associated with a particular numeric identifier according to this set (i.e., label encoded). However, if the relations are associated with a particular dataset or a knowledge base, the RE task is extended to *relation linking* (REL). Hence, within the REL task, the relations are associated with their unique identifiers in a data or knowledge base.

Another important text classification example is the *expected answer type classification* (EAT) task. The goal of the EAT task is to understand what a user expects to see as an answer based on a natural language (NL) question (see Figure 2.6). Therefore, the task can be interpreted as a multi-class classification. Here, the set of classes Y may contain, for example, such labels as `boolean`, `literal`, `resource`, and `list`. The `boolean` answer type corresponds to a "Yes-No" question e.g., "Was Angela Merkel born in Hamburg?". The `resource` answer type corresponds to questions that are intended to be answered with a specific entity, e.g., "Where was Angela Merkel born?". The modern state-of-the-art approaches for text classification are mostly based on the transformer architecture, e.g., EAT [152], REL [222].

2.1.4 Sequence-to-Sequence Conversion

The task of the *sequence-to-sequence* (seq2seq) conversion consists of generating an output sequence of tokens $\boldsymbol{y}$ based on an input sequence $\boldsymbol{x}$. This formulation is

different from the sequence labeling task, as here every component of both $\boldsymbol{x}$ and $\boldsymbol{y}$ corresponds to a token from a predefined *vocabulary* V, whereas the output of sequence labeling is limited to a predefined label set Y. Moreover, the lengths of $\boldsymbol{x}$ and $\boldsymbol{y}$ can differ.

One of the most frequent applications of the seq2seq task is the *machine translation* (MT). The task of MT involves producing a comparable sentence in a *target language* from a given sentence in a *source language* [99]. For instance, a MT system is given a German sentence: "Ich verstehe nur Bahnhof" and must translate it into English: "It is all Greek to me". This particular example is not trivial for MT systems, as the given sentences are idioms. Formally, the MT systems are typically designed to optimize the conditional probability of the target token sequence, $y_1, \ldots, y_m$, given the token sequence in the source language, $x_1, \ldots, x_n$: $P(y_1, \ldots, y_m | x_1, \ldots, x_n)$. The seq2seq tasks are modeled with the *encoder-decoder* NN architecture, such as the state-of-the-art Flan-T5 [37] and Flan-UL2 [193].

2.1.5 Vector Semantics and Embeddings

Let us consider the following classification task: given a word w and a *context word* c (i.e., surrounding word), predict whether c is a real context word of w: $P(c|w)$ [99]. Speaking more generally, the context can contain more than one word. This task formulation is used to model (or learn) the meaning of words in the form of vectors (i.e., *vector semantics*), which originates from *distributional hypothesis* [71] that states the following: words that occur in similar contexts tend to have similar meanings (see Figure 2.7).

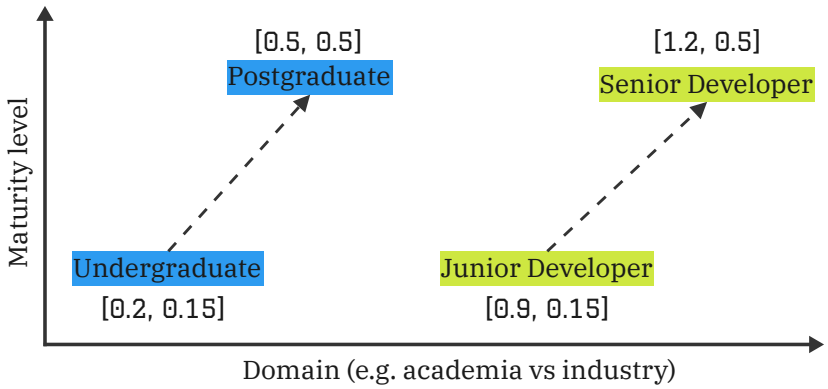


Fig. 2.7.: A simple yet comprehensive example of the idea behind vector semantics projected on two dimensions. The color represents the corresponding domain counterparts.

The models for the vector semantics are divided into two classes: *sparse* and *dense*. The sparse models rely on different textual statistics, for example, computing the *term-document matrix* [172] results in a sparse vector that can represent the meaning of the considered words. Computing co-occurrence matrices (e.g., via *Positive Pointwise Mutual Information (PPMI)* [38]) of words results in sparse word representations as well. The dimension of the vectors produced by the sparse models depends on the size of the vocabulary of a text corpora. Therefore, the dimensionality may become very large (e.g., tens of thousands).

The dense models, as per their name, produce vectors of smaller dimensions, which size frequently lies between 50 and 1000. The dense vectors are also frequently called *embeddings*. The embeddings of words or tokens can be *learned based on a large text corpora* in a self-supervised manner (i.e., no explicitly labeled dataset is required). The first embedding models, such as Word2vec [128], FastText [97], GloVe [148], are based on the task of predicting a missing word (or a token) w in the sequence of the context words (tokens) c: $P(+|w, c)$. Therefore, training such models results in *static embeddings* for the words or tokens in the vocabulary V. The most recent dense models (e.g., BERT [46] or XLM-RoBERTa [40]) produce *dynamic contextual embeddings*, where each word w will be represented by a different vector when it appears in a different context c. This is particularly important when dealing with *homonymy* (multiple words have the same spelling).

2.1.6 Multilinguality and Low-Resource Languages

The *multilinguality*, as a feature of a system that works with NL, is defined as providing the ability to process input or search for information in several different languages $l \in \mathcal{L}$ [158]. We interpret "several languages" as *more than one* according to [82]. Hence, a user can ask input questions in different languages: $q_{l_1}, \ldots, q_{l_n}$, where $n = |\mathcal{L}|$. At the same time, a system may search for the answers in source documents composed in different languages: $\mathcal{D}_{l_1}, \ldots, \mathcal{D}_{l_n}$. Let us assume that a user writes their NL question q in the language l_i and a system finds an answer $\mathcal{A}$ in the $\mathcal{D}_{l_j}$ (instantiated for the language l_j). The *ability to handle more than one language is called multilinguality*, however, *if $l_i \neq l_j$ given the example above, such an ability is called cross-linguality*. In the scope of this work, the terms "cross-lingual" and "multilingual" are interchangeable, and the latter is to be used.

Low-resource languages are those that lack sufficient linguistic resources, such as supervised training data (e.g., shortage of parallel corpora for neural machine

translation), which is essential for the NLP tasks [121, 165, 67]. In the research community, there is no straightforward threshold for separating low-resource languages from *resource-rich languages*. In the context of this thesis, we define low-resource languages as those that meet one or more of the following criteria:

- Limited availability of KGQA benchmarking datasets (≤ 1 for a particular language);
- Scarce annotated datasets for downstream NLP tasks (e.g., text classification, named entity recognition);
- Insufficient linguistic tools and resources (such as parsers, tokenizers, or comprehensive digital dictionaries).

This definition aligns with the common practices in multilingual NLP research and provides clear and quantifiable metrics for our specific research context.

2.1.7 Large Language Model-based Agents

Given the emergence of LLM decision-making and autonomous agents in language models [216, 119], it is essential to establish clear definitions of key concepts, terms, and notation used throughout this thesis. Our definitions are derived primarily from the literature from peer-reviewed surveys in this field [126, 216].

An *LLM agent* (or AI agent) is frequently conceptualized as an *augmented language model*, extending beyond autoregressive natural language generation to incorporate extended reasoning capabilities, external tool utilization and multi-agent collaboration. The foundation of such systems is built on a *backbone LLM*, which serves as a core component. These agent frameworks typically comprise the following essential components:

Profiling A component that establishes the agent's identity and role through explicit directives (e.g., "you are an intelligent assistant specialized in..."). This profiling information is typically embedded within a *system prompt*, which guides the agent's behavior and responses;

Long-term Memory A mechanism that enables the agent to access and utilize historical interactions and outcomes (e.g., "based on previous successful interactions..."—experience pool), enhancing decision-making through learned experiences;

Planning An approach to decompose complex tasks into manageable, sequential sub-tasks (e.g., "First, analyze the input data. Second, apply the transformation. Finally, validate the results..."");

Action (tooling) The execution component that transforms the agent's decisions into concrete operations (e.g., invoking a *tool* such as a RESTful API), bridging the gap between reasoning and real-world effects.

The agent workflow is not always straightforward, sometimes involving additional iteration loops. Typically, such loops are caused by a *feedback* mechanism, which is responsible for refining the ongoing workflow. *Environmental feedback* is obtained from the objective world or virtual environment (e.g., a relational database response when an SQL or SPARQL query was executed). All the components and mechanisms mentioned above can also be enriched with *in-context* examples (i.e., *short-term memory*) for better output. For example, the planning component can be enriched with a successful example from the past agent runs. A general overview of the architecture of an LLM-based agent is presented in Figure 2.8 As the figure suggests,

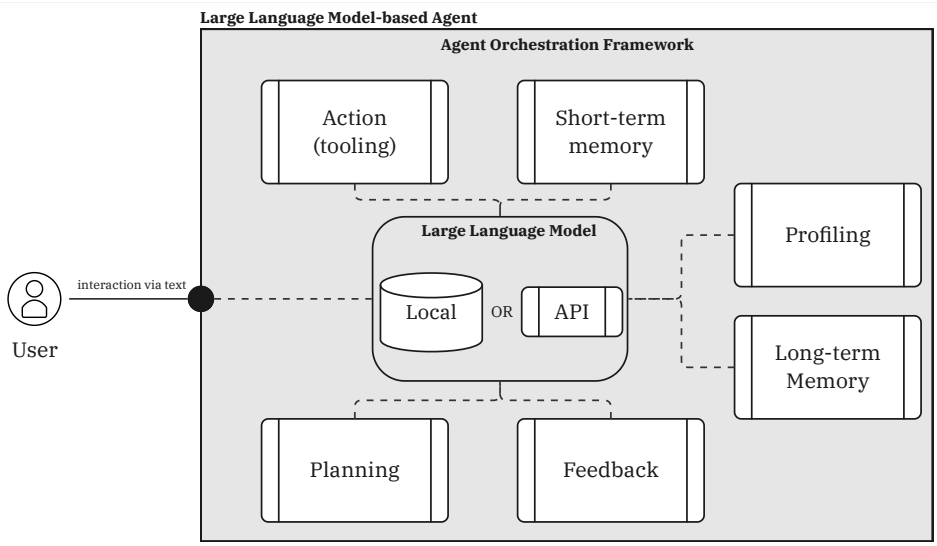


Fig. 2.8.: A general overview of an LLM-based agent's logical architecture

there is an additional layer called *agent orchestration framework*, which is responsible for creating workflows that connect an LLM with the other components. Some of the well-known frameworks are: LangChain[1], LlamaIndex[2], CrewAI[3], etc.

[1] `https://www.langchain.com/`
[2] `https://www.llamaindex.ai/`
[3] `https://www.crewai.com/`

2.2 Semantic Web and Knowledge Graphs

This chapter describes the terms and concepts related to Linked Data, Semantic Web, and Knowledge Graphs (KGs). These are the crucial topics for this thesis since KGs are used as a data layer behind KGQA systems.

2.2.1 Linked Data and Semantic Web

Resource Description Framework (RDF)

RDF is a standardized framework for representing information on the Web [@44]. As may be recalled from the name, *resources* is a central concept within RDF. A resource is a *referent*—a real-world object, an object on the Web, or a set of objects denoted by a sign (or a label). RDF resources are denoted by *Internationalized Resource Identifiers* (IRIs). Often, IRI and URI (*Uniform Resource Identifier*) are used interchangeably in this context. RDF is based on a graph data model, these graphs are sets of *subject-predicate-object triples* (see Figure 2.9).

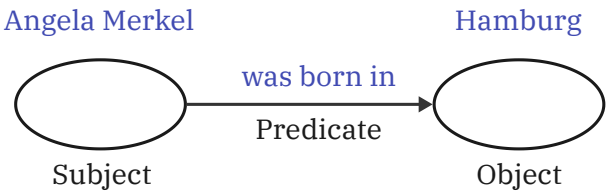


Fig. 2.9.: An example of a triple—an RDF graph with two nodes (subject, object) that are connected via predicate.

An RDF vocabulary is a collection of IRIs intended for use in RDF graphs [@214]. In particular, RDF schema is a data-modeling vocabulary [@212]. RDF enables merging data even when the underlying schemas are different. It is designed to accommodate the evolution of schemas over time without any changes for all data consumers.

The IRIs in an RDF vocabulary often begin with a substring (e.g., "`xsd`")—namespace IRI or a *namespace prefix* (e.g., "`https://www.w3.org/2001/XMLSchema`"). Vocabularies such as OWL [@213] and SKOS [@91] are built on top of RDF, offering an instrument to define structured *ontologies*. Figure 2.10 shows a sample RDF graph data in Turtle (`.ttl`)[4] format. In this work, we use several RDF Terms [@214]:

[4] `https://www.w3.org/TR/turtle/`

```
# define prefixed namespaces
@prefix ex: <http://example.org/> .
@prefix dbo: <http://dbpedia.org/ontology/> .
@prefix dbr: <http://dbpedia.org/resource/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

# define assertions about the entities and relations
ex:AngelaMerkel # define Angela Merkel as a resource
    a ex:Human ; # is a human
    dbo:birthDate "1954-07-17"^^xsd:date ; # has the specified birth date
    dbo:occupation ex:Politician ; # has the following occupation
    dbo:birthPlace ex:Hamburg . # has the following birth place

ex:Hamburg # define Hamburg as a resource
    a dbo:City ; # is a city
    dbo:country dbr:Germany ; # is located in the following country
    dbo:population "1758041"^^xsd:integer . # has the following population
```

Fig. 2.10.: An example of an RDF graph in the Turtle format. Here, the DBpedia [7] namespaces are reused for defining the relations (`dbo`) and for interlinking with other resources (`dbr`).

IRI Internationalized Resource Identifiers, defined in [56].

Literal Literals are used for values such as strings (may be language-tagged e.g., `"Hallo"@de`), numbers (e.g., “xsd:integer”), and dates (e.g., “xsd:date”).

Boolean e.g., “xsd:boolean”.

Semantic Web

The Semantic Web is an initiative of the World Wide Web Consortium (W3C) for an extension of a traditional “static” Web to make its data machine readable [15]. To facilitate *annotating semantics within data*, technologies such as RDF are employed. RDF enables formal and standardized representation of metadata. For example, ontologies can define concepts, relationships between entities, and categories of objects. The embedded semantics provide benefits, such as enabling reasoning over data and enhancing interoperability with diverse data sources. Typically, such data are provided directly within the Web pages, e.g., through RDF in attribute syntax (RDFa) (see Figure 2.11) and are used most frequently for *Search Engine Optimization* (SEO) purposes.

Figure 2.11 represents the usage of `schema.org`—one of the most used ontologies on the Web, mainly for the SEO purposes.

```
<div vocab="https://schema.org/" typeof="Person">
  <!-- specify schema.org vocabulary and Angela Merkel as an entity of type "Person" -->
  <!-- the person's name is defined below -->
  <span property="name">Angela Merkel</span> was born in
  <!-- define the birthPlace relation,
  specifying the birthplace as an entity of type "Place",
  linked to external knowledge (Wikidata) -->
  <span property="birthPlace" typeof="Place" href="https://www.wikidata.org/wiki/Q1055">
    <!-- explicitly provide the place's name property -->
    <span property="name">Hamburg</span>.
  </span>
</div>
```

Fig. 2.11.: An example of annotating semantics via RDFa syntax and `schema.org` vocabulary within a web page. The semantics of this statement are equivalent to Figure 2.9.

Linked Data and Web of Data

Linked Data is a concept in which structured data based on RDF are interlinked with other data [18]. Hence, it complements the Semantic Web with the ability of reusing already existing data, for example, in Figure 2.11 one may recognize the annotated textual span "Hamburg" points to a URI from an external KG—Wikidata [211]. Thus, one does not define a new resource to describe some real-world object but rather *reuses* an existing one. Reusability is provided by the ontologies that rely RDF standards and simplify data merging.

The core idea of the *Web of Data* is to publish content on the Web in formats that machines can process more easily and accurately than human-friendly HTML documents [78]. The Web of Data initiative follows several principles, such as: (1) using a canonical form of expressing factual claim, (2) using one name to refer to one thing, (3) using unambiguous names that refer to one thing, (4) writing queries using a structure analogous to the data, (5) making the semantics of terms explicit, and (6) providing links to discover relevant information in remote locations.

SPARQL

The *SPARQL query language* is designed to query the RDF data [@70]. It enables its users to formulate queries across various data sources, regardless of whether the data is originally stored as RDF or presented as RDF through middleware. SPARQL's features include querying both required and optional graph patterns and their combinations through conjunctions and disjunctions. It also supports functionalities such as aggregation, subqueries, negation, value creation through expressions, extensible value testing, and constraining queries based on the source RDF graph.

```
PREFIX ex: <http://example.org/> # defines prefix "ex" for example.org resources
PREFIX dbo: <http://dbpedia.org/ontology/> # defines prefix "dbo" for DBpedia ontology

SELECT ?birthPlace # selects the birth place of Angela Merkel
WHERE {
    ex:AngelaMerkel dbo:birthPlace ?birthPlace . # retrieves Angela Merkel's birth place
}
```

Fig. 2.12.: A simple SPARQL query that returns the birthplace of Angela Merkel based on the data from Figure 2.10.

The results of SPARQL queries can be either result sets or RDF graphs. Hence, SPARQL is a powerful tool for manipulating RDF data. Figure 2.12 demonstrates a simple SPARQL query over RDF data from Figure 2.10.

2.2.2 Knowledge Graphs

Although knowledge graphs (KGs) were briefly discussed in the previous subsections, their formal definition is provided in this subsection. A *knowledge graph* is a machine-readable representation of real-world knowledge. The nodes in a KG represent entities of interest and the edges indicate various relationships between these entities [79]. Naturally, KGs adhere to a graph-based data model. The most popular forms of the graph data model are *directed edge-labeled graph* and a *property graph*. This work mainly considers directed edge-labeled graphs; however, in further subsections, we describe both in detail. Figure 2.13 demonstrates the "knowledge graph evolution" in terms of the technology foundation.

Directed Edge-labeled Graphs

A *directed edge-labeled (del) graph* (sometimes also a *multi-relational graph* [21]) is defined as a set of *nodes*—such as "Hamburg", "Angela Merkel", "165 cm" (i.e., both resources and literals can represent nodes) and *edges* represent relations between those entities (e.g., Hamburg "has type" City) [79]. Figure 2.14 illustrates the example of a del graph. Typically, del graphs are based on RDF.

Property Graphs

A *property graph* (also *labeled property graph*—LPG) allows a set of property–value tuples and a label to be associated with both nodes and edges [79]. For example, a

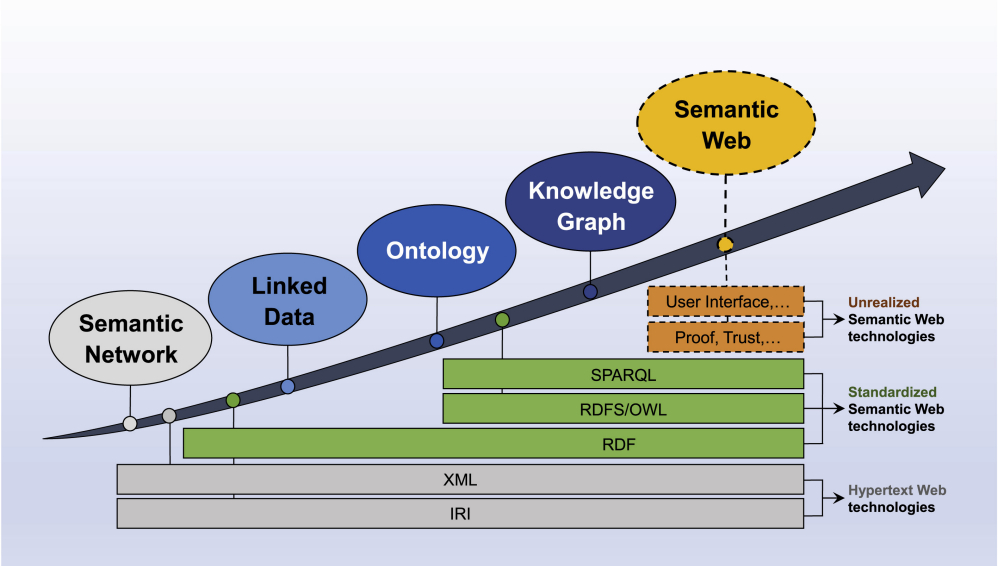


Fig. 2.13.: The RDF-based knowledge graph (directed edge-labeled) evolution (the figure from [131]). According to this figure, our work is based on the standardized Semantic Web technologies and specifically tackles the user interface in terms of natural language access to the data

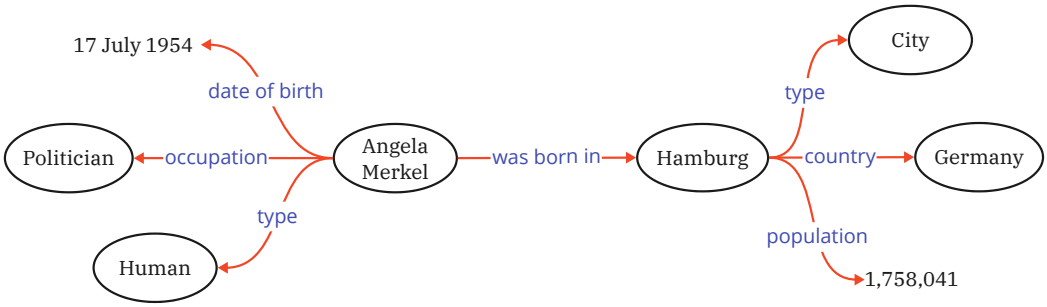


Fig. 2.14.: A conceptual example of a directed edge-labeled (del) graph.

del graph requires a new node and edge when defining the occupation of Angela Merkel, while for a property graph, one only needs to add the corresponding property-value tuple (see Figure 2.15). Property graphs do not follow any standard, in contrast to del graphs, which are typically based on RDF. However, they are implemented in popular graph databases, such as Neo4j [130].

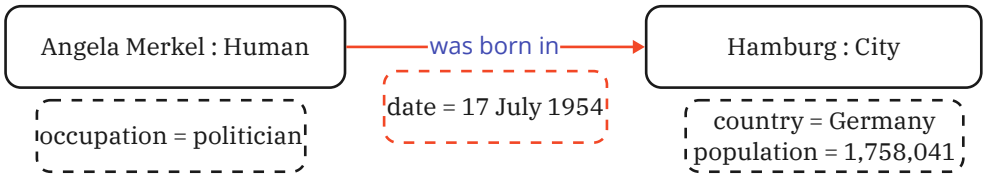


Fig. 2.15.: A conceptual example of a property graph.

2.3 Question Answering and Information Retrieval

This chapter describes the core concepts of question answering and information retrieval. *Information retrieval (IR)*—is the task of returning documents to a user according to their information need, while *question answering (QA)*—is the task of answering a user's questions [99] directly, based on structured or unstructured data.

IR and QA utilize algorithms, methods, and technologies from the previously described research fields, namely, natural language processing, semantic web, and knowledge graphs. Figure 2.16 presents the pipeline of a KG construction: from multi-source heterogeneous data to the applications of KGs.

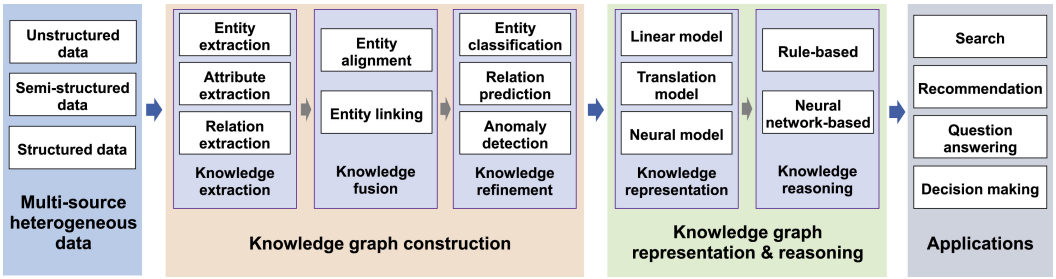


Fig. 2.16.: The pipeline of RDF-based knowledge graph construction, representation, reasoning, and applications [131]. Our work is focused specifically on the applications, namely question answering.

2.3.1 Knowledge Graph Question Answering

Knowledge-based Question Answering (KBQA) systems perform a *direct answer* extraction on structured data, while IR-based QA (or Open Domain QA) operates on unstructured documents [98]. *The objective of KBQA* is to identify the answers $\mathcal{A}$ that satisfy the informational need of a NL question q, using a KB $\mathcal{K}$ [57]. The introduction of the Semantic Web enabled the extraction of direct answers based on structured data. The data published on the Web and made machine-readable via Semantic Web principles corresponds to the formal definition of a Knowledge Graph (KG) [79] and, therefore, can be considered as a global decentralized RDF-based KG. A family of systems that addresses the challenge of providing direct answers based on KGs is named *Knowledge Graph Question Answering (KGQA) systems* (see Figure 2.17). KGQA systems share the same objective as the KBQA ones. However, a KB $\mathcal{K}$ is replaced by a knowledge graph $\mathcal{KG}$. Although the terms KBQA and KGQA are often used interchangeably, we prefer to view *KGQA systems as a subset of KBQA*.

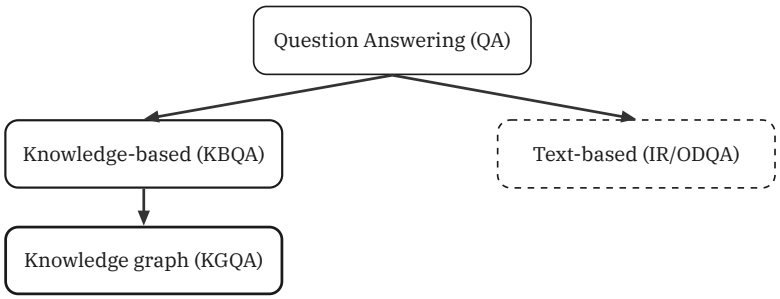


Fig. 2.17.: Hierarchy of the question answering systems

The answer types supported by the majority of the KGQA systems are limited to the following: resource[5] (e.g., a URI of an entity), literal (e.g., a string or a number), boolean, or a list of the previously mentioned ones. These answers can be converted back into NL using natural language generation (NLG) frameworks for KGs [8, 156]. The majority of KGQA systems developed by researchers work on general domain KGs such as Freebase [19] (now part of Google's KG [@64]), DBpedia [7], Wikidata [211], etc. However, the approaches implemented therein can commonly be ported to any KG.

2.3.2 Development Paradigms of Knowledge Graph Question Answering Systems

Classification by output type

Recent KGQA developments show that systems can be divided into two classes by their generated output [228, 223, 125]: (1) information extraction and (2) semantic parsing. KGQA systems following the *information extraction* class usually retrieve a set of answer candidates directly from a KG by traversing the graph. The *semantic parsing* class focuses on the tasks of converting a NL question q into a formal query ϕ (e.g., SPARQL) that must be executed on a KG for the sake of retrieving an answer.

Recent advances in the field of *Large Language Models* (LLMs) have shown that both the aforementioned system classes can be implemented with utilizing LLMs as a core component. In particular, one may directly pose a NL question q to such a model to expect a direct answer (paradigm 1) [10, 125] or create an instruction for an LLM to generate a SPARQL query (paradigm 2) [107]. Furthermore, LLMs enable the so-called *agentic* approach, where a KGQA system is built as an LLM surrounded by external tools, planning and reasoning, and feedback mechanisms [84, 96, 232].

[5] `https://www.w3.org/DesignIssues/Generic.html`

Classification by system design

From the system design point of view, the KGQA systems are classified into the following: (1) end-to-end systems and (2) component-based systems. The *end-to-end* systems are typically represented through a monolithic architecture or are neural models that directly produce the output [182, 183]. The *component-based* systems are a more general class that can incorporate end-to-end systems together with other components or use sequential pipelines of components that perform different tasks (e.g., NED, REL) [53, 201].

2.3.3 Evaluation of Knowledge Graph Question Answering Systems

The specifics of question answering evaluation are that *these systems mostly return one direct answer to a question by design*, in contrast to IR systems that return ranked lists of documents. Although the ranked list metrics are not widely used in the field of KGQA, many information retrieval metrics are still used.

The concept of relevance

Conventionally, the evaluation of KGQA systems is based on the binary judgement of *relevance* [177]. In other words, an answer can be either *relevant* (i.e., correct) or *nonrelevant* (i.e., incorrect). The relevance judgment is based on a *ground truth* answer that is part of a *test dataset*. Hence, an answer is relevant if it fully addresses the stated *information need*. The information need is formulated via an NL question and is reflected via ground truth answers. The binary relevance is formalized in Equation 2.3).

$$\text{Relevance}(\hat{y}, y) = \begin{cases} 1 & \text{if } \hat{y} = y, \\ 0 & \text{else } \hat{y} \neq y \end{cases} \tag{2.3}$$

Where $\hat{y}$—an answer given by a system, y—a ground truth answer.

Based on the binary relevance, one may calculate *Precision* (see Equation 2.4) as the fraction of retrieved answers that are relevant and *Recall* as the fraction of relevant documents that are retrieved [177] (see Equation 2.5).

$$\text{Precision} = \frac{|\text{relevant items retrieved}|}{|\text{retrieved items}|} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2.4}$$

$$\text{Recall} = \frac{|\text{relevant items retrieved}|}{|\text{relevant items}|} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2.5}$$

The confusion matrix in the context of Information Retrieval is demonstrated in Table 2.1.

Tab. 2.1.: Confusion Matrix in the context of Information Retrieval

	Relevant	**Not Relevant**
Retrieved	TP (True Positive)	FP (False Positive)
Not Retrieved	FN (False Negative)	TN (True Negative)

The measure that represents the effectiveness of retrieval with respect to a user who attaches β times as much importance to Recall as Precision is called F score F_β [209] (see Equation 2.6).

$$F_\beta = (1 + \beta^2) \cdot \frac{\text{Precision} \cdot \text{Recall}}{(\beta^2 \cdot \text{Precision}) + \text{Recall}} \tag{2.6}$$

The parameter β controls the weight of the Recall over the Precision (e.g., $\beta > 1$ gives more weight to the Recall. However, the most popular situation is when $\beta = 1$—this is known as the F_1 score (or simply the F1 score). The *F1 score* is the harmonic mean of Precision and Recall and is commonly used for the comparison of different KGQA systems.

When considering such answer types as resource, boolean, or literal (see Figure 2.6 for the idea of answer types), the possible Precision and Recall values are 1 or 0 since they imply only one ground truth answer. The examples with Precision and Recall calculation for such answer types are demonstrated in Figure 2.2.

Tab. 2.2.: Examples for calculation of Precision and Recall for resource, boolean, and literal answer types

Input question	**Answer type**	$\hat{y}$	y	**Precision**	**Recall**
Where was Angela Merkel born?	resource[6]	Hamburg	Hamburg	1	1
Was Angela Merkel born in Hamburg?	boolean	False	True	0	0
When was Angela Merkel born?	literal (date)	“17-07-1954”	“17-07-1954”	1	1

When it comes to the list answer type, which is composed from resources or literals (e.g., “Name all children of Barack Obama”), then the Precision and Recall scores are more diverse (see Figure 2.18).

[6] `https://dbpedia.org/resource/Hamburg`

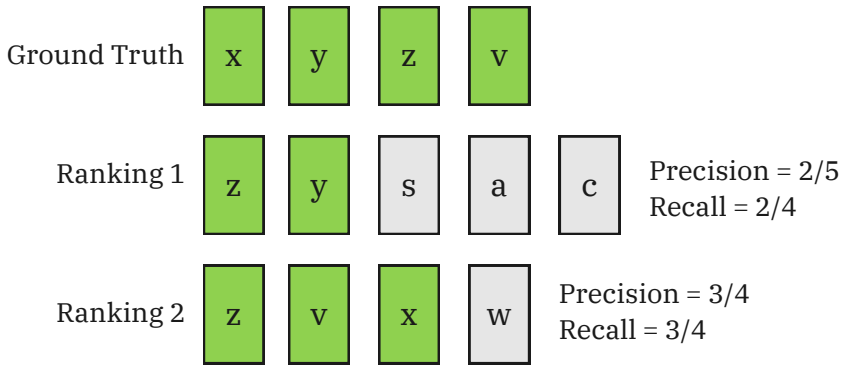


Fig. 2.18.: An example of Precision and Recall calculation for the list answer type. Green boxes represent relevant answers.

Averaging strategies for Precision, Recall and F1 Score

According to the established KGQA community standards [206], the different *averaging strategies* (Micro and Macro) are implemented as follows.

Micro: aggregate all true positives, false positives, and false negatives across all questions (q), computing these metrics only at the end (see Equation 2.7).

$$\begin{aligned} \text{Precision}_{\text{micro}} &= \frac{\sum_q TP_q}{\sum_q TP_q + \sum_q FP_q}, \\ \text{Recall}_{\text{micro}} &= \frac{\sum_q TP_q}{\sum_q TP_q + \sum_q FN_q}, \\ \text{F1}_{\text{micro}} &= \frac{2 \cdot \text{Precision}_{\text{micro}} \cdot \text{Recall}_{\text{micro}}}{\text{Precision}_{\text{micro}} + \text{Recall}_{\text{micro}}} \end{aligned} \tag{2.7}$$

Macro: Precision, Recall, and F1 Score are calculated individually for each question, and these scores are averaged separately at the end. Therefore, this metric emphasizes more strongly the system's ability to perform consistently well across all questions (see Equation 2.8).

$$\begin{aligned} \text{Precision}_{\text{macro}} &= \frac{1}{N} \sum_{q=1}^{N} \frac{TP_q}{TP_q + FP_q} \\ \text{Recall}_{\text{macro}} &= \frac{1}{N} \sum_{q=1}^{N} \frac{TP_q}{TP_q + FN_q} \\ \text{F1}_{\text{macro}} &= \frac{1}{N} \sum_{q=1}^{N} \frac{2 \cdot TP_q}{2 \cdot TP_q + FP_q + FN_q} \end{aligned} \tag{2.8}$$

It is worth noting that, *when using macro averages, the resulting F1 Score may not necessarily fall between Precision and Recall.*

Other metrics

Precision@k This metric is very similar to the standard Precision. However, it cuts off the list of retrieved items by the length of k. Typically, $k = 1$ in the KGQA setup. Hence, the evaluation strategy is to verify whether the answer ranked as top-1 (out of n) is correct.

Hits@k This metric follows the same calculation strategy as Precision@k [34]. In the KGQA interpretations, Hits@1 is defined as follows [178]: (1) for literal, boolean, resource or null answers, this is the same as exact match, (2) for list answers, returns 1 if at least one relevant answer appears in the predicted answer, otherwise 0. Hence, Hits@1 can produce potentially higher values than Precision@1.

Answer Trustworthiness Score (ATS) This metric accounts the unanswerable questions, specifically when $\mathcal{A} = \emptyset$. It was first defined in [63] and is specifically designed to assess the trustworthiness of KGQA systems. It calculates a score for each question q in a dataset $\mathcal{D}$, sums these scores, and normalizes the result within a range of -1 to $+1$:

$$ATS(\mathcal{D}) = \frac{\sum_{q \in \mathcal{D}} f(q)}{|\mathcal{D}|}, \text{ where } f(q) \begin{cases} +1 & \text{if } isCorrect(\mathcal{A}_q, \hat{\mathcal{A}}_q) = correct \\ 0 & \text{if } \hat{\mathcal{A}}_q = \emptyset \\ -1 & \text{otherwise} \end{cases} \tag{2.9}$$

ATS considers correct, incorrect, and empty answer sets. In line with the principle that "no answer is better than a wrong answer" [219], there is no penalty for a KGQA system returning no result (i.e., systems that provide fewer incorrect answers to users achieve a higher score). A KGQA system can achieve an average ATSscore of 0 by simply responding without an answer to all the questions in D. To achieve a positive ATS, the system must provide more correct answers than incorrect ones. Consequently, ATS is more stringent than other common metrics and is an ideal measure for evaluating the quality of KGQA systems.

Boundary cases

The metrics are computed using the following additional rules [206, 169]:

- If the ground truth answer set is empty (i.e., $\mathcal{A} = \{\emptyset\}$) and the system responds correctly with an empty answer, Precision, Recall, and F1 Score are set to 1.
- If the ground truth answer set is empty, but the system returns any non-empty answer, Precision, Recall, and F1 Score are all set to 0.
- If the ground truth answer set is non-empty, but the system provides an empty answer set (meaning it failed to respond), Precision, Recall, and F1 Score are each set to 0.
- In all other cases, Precision, Recall, and F1 Score are calculated in the standard way for each individual question.

3 Systematic Survey on Multilingual Knowledge Graph Question Answering

3.1 Introduction

This chapter presents a systematic survey of the mKGQA field. Our review methodology follows best practices from the prior survey articles [106, 127, 17, 150] to comprehensively analyze research developments in mKGQA. Our methodology encompasses the identification of scientific literature sources, selection of relevant publications, and systematic extraction of quantitative information (e.g., performance metrics, evaluation methodologies, etc.). We then analyze this information to revear novel insights and organize them into a structured taxonomy.

Our analysis is based on 48 publications, filtered from an initial pool of 1875 papers. Through the formal selection criteria, we identified 27 papers discussing mKGQA systems, 14 papers describing benchmarks and datasets, and 7 comprehensive survey articles.

Our review of the past decade's literature reveals that *existing systematic surveys on KGQA significantly underrepresent the multilingual dimension*. Most surveys merely acknowledge multilinguality as a KGQA challenge, typically dedicating no more than a paragraph to this aspect. Furthermore, we found no surveys specifically focused on analyzing multilingual capabilities of these systems. Given the emergence of dedicated KGQA systems addressing multilinguality, a thorough survey of mKGQA has become imperative. *This chapter contributes to all research gaps* we defined in Section 1 and, in particular, addresses three primary *research questions*:

$\mathcal{RQ}$***3.1*** — How to define a systematic review methodology to identify, structure, and characterize mKGQA approaches?

$\mathcal{RQ}$***3.2*** — How does the language distribution patterns, representation of diverse language groups, alphabets, and writing systems in mKGQA look like?

$\mathcal{RQ}$***3.3*** — What are the trends in benchmark dataset sizes and language coverage?

The *contributions of this chapter* are threefold: (*a*) provide a systematic analysis of historical developments and current advances in mKGQA systems, (*b*) develop a comprehensive taxonomy of mKGQA methods and their distinctive characteristics, and (*c*) identify existing research challenges and outline future research directions.

To ensure comprehensive coverage, our analysis encompasses publications describing mKGQA systems, relevant survey articles, and associated benchmarking datasets, thereby providing a holistic view of this emerging research field.

The chapter concludes with a synthesis of our findings and a discussion of future research directions. To facilitate reproducibility and further research, we have made all collected data, analysis scripts, and accompanying documentation publicly available in an online appendix[1].

CRediT Statement about the author's contribution roles This chapter is mainly based on the following peer-reviewed publications:

1. **Perevalov, A.**, Both, A., and Ngonga Ngomo, A.-C. (2024). Multilingual Question Answering Systems for Knowledge Graphs–a Survey. Semantic Web 15(5): 1–36.
2. **Perevalov, A.**, Abdollahi, S., Gottschalk, S., Both, A. (2024). Evaluating Entity Importance in a Cross-National Context using Crowdsourcing and Best–Worst Scaling. KES conference, Procedia Computer Science 246 (2024): 1479-1487.
3. **Perevalov, A.**, Yan, X., Kovriguina, L., Jiang, L., Both, A. and Usbeck, R. (2022). Knowledge Graph Question Answering Leaderboard: A Community Resource to Prevent a Replication Crisis. In 2022 Proceedings of the 13th LREC Conference: 2998–3007.

The author's roles according to the CRediT taxonomy for the publications are: Conceptualization, Data curation, Formal analysis, Methodology, Visualization, and Writing – original draft.

3.2 Systematic Review Methodology

Our survey follows a systematic review methodology based on established guidelines [106, 127, 17, 150] to ensure transparency and reproducibility. This section provides a detailed description of our methodology and its implementation throughout the review process.

[1] `https://github.com/Perevalov/phd_thesis`

The methodology comprises three primary phases: (1) source selection, (2) initial publication screening, and (3) information extraction and systematization. Each phase is detailed in the subsequent subsections. The review process was conducted exclusively by the authors of the article included in the introduction of this chapter, with the first author taking the lead role in executing key steps, including developing automated information extraction scripts (Section 3.2.2) and performing manual information extraction (Section 3.2.3).

3.2.1 Selection of Sources

Our source selection focused on *digital research databases in computer science that provide advanced search capabilities*. In alignment with our multilingual focus, we expanded our literature search beyond English-language sources to include *publications in English, German, and Russian*. The selection of these specific languages was based on the native language proficiency of the authors, ensuring accurate interpretation and analysis of the content.

To discover the digital research databases, we utilized three major search engines—Google[2], Bing[3], Yandex[4]—with a specific search string in different languages. The following search queries were used to find the sources:

English: `("digital library" or "research database")`
`and "computer science";`
German: `("digitale Bibliothek" or "forschungsdatenbank")`
`and "informatik"`[5];
Russian: (“цифровая библиотека” or “индекс цитирования”)
and “информатика”[6].

We filtered the search results using the following *acceptance criteria*:

- articles must be within the computer science domain,
- be available in at least one of our selected languages,
- be accessible through advanced search functionality.

[2] `https://google.com/`
[3] `https://www.bing.com/`
[4] `https://yandex.ru/`
[5] Literally translated as: `("digital library" or "research database") and "informatics"`
[6] Literally translated as: `("digital library" or "citation index") and "informatics"`

Based on these criteria, we identified the *following sources* for subsequent analysis phases: IEEE Xplore[7], ACM DL[8], Springer[9], DBLP[10], ACL Anthology[11], Cyberleninka[12].

3.2.2 Initial Publications Selection

Building upon the digital research databases identified in the previous phase, we leveraged advanced search functionality and constructed complex search queries to systematically identify relevant publications. *Our search strategy specifically targeted three fundamental aspects* of the publications:

System aspect Question Answering systems;
Data aspect RDF-based Knowledge Graphs;
Language aspect Multilinguality and cross-linguality.

Our review encompassed *three categories of publications*: survey papers, system descriptions, and benchmarking datasets. We applied the following *acceptance criteria* (all have to match):

- publications must describe either a QA system, a relevant benchmarking dataset, or present a systematic survey,
- the systems or benchmarks must be designed to operate on RDF-based KGs,
- the systems must support multilingual capabilities (supporting at least two languages).

The review covered publications from 2011 to 2025[13]. To ensure comprehensiveness and reproducibility, we conducted multiple iterations of our review process before finalizing this work, minimizing the risk of overlooking relevant publications. The last time *the publications list was updated is February 15th, 2025*.

We developed comprehensive search queries incorporating all three aforementioned aspects for each database source. Table 3.1 presents the conceptual framework of our search query structure.

[7] `https://ieeexplore.ieee.org`
[8] `https://dl.acm.org`
[9] `https://www.springer.com`
[10] `https://dblp.org`
[11] `https://aclanthology.org`
[12] `https://cyberleninka.ru/`
[13] This work began in 2021, with the initial goal of reviewing the previous decade's publications, hence the 2011 start date.

Tab. 3.1.: The conceptual structure of the search query, with its constituent components joined by the AND operator.

Aspect	Search query part
System	("Semantic search" OR "Question Answer*" OR "Question-Answer*" OR "KBQA" OR "KGQA" OR "KB QA" OR "KB-QA" OR "KG-QA" OR "KG QA" OR "NLI" OR "NLIDB" OR "QA" OR "Natural Language Interface")
Data	("Knowledge Base*" OR "Knowledge Graph*" OR "DBpedia" OR "Wikidata" OR "YAGO" OR "Semantic Web" OR "Linked Data" OR "RDF*" OR "data web" OR "SPARQL" OR "Query Graph" OR "Web data" OR "WWW" OR "web of data" OR "QALD*" OR "SimpleQuestions" OR "WebQuestions" OR "WebQSP" OR "LC-QuAD" OR "RuBQ" OR "SimpleDBpediaQA" OR "ComplexWebQuestions" OR "MCWQ")
Language	("multilingual*" OR "multi-lingual" OR "crosslingual*" OR "cross-lingual" OR "internationalized" OR "multilingualism" OR "multilinguistic" OR "multilanguage" OR "bilingual")

Subsequently, *we developed an automated process to extract key publication metadata*, including: authors, title, abstract, publication year, DOI/URL, source, and publisher. The extraction script and its comprehensive documentation are available in our online appendix[14]. We then conducted a thorough manual assessment of these publications, evaluating them against our previously established acceptance criteria. Table 3.2 presents a statistical breakdown of selected and accepted publications across different sources.

Tab. 3.2.: Distribution of selected and accepted publications categorized by source databases

#	IEEE Xplore	ACM DL	Springer	DBLP	Cyberleninka	ACL Anthology	Related Work	Total
Selected	19	289	1366	140	38	16	12	1880
Accepted	2	7	16	11	1	4	5	46

Based on our expertise in mKGQA, we recognized that *our systematic review methodology demonstrated high specificity*, which inadvertently excluded some relevant publications previously known to us. To address this limitation, we implemented one modification to our paper selection process. In particular, *we incorporated previously known publications that satisfied two conditions*:

- alignment with our three main aspects (as outlined above),
- either having received at least five citations or undergone peer review (refer to the "Related Work" column in Table 3.2).

These additional publications from the "Related Work" category constitute approximately 10% of our total selection.

[14] https://github.com/Perevalov/multilingual-KGQA-survey

3.2.3 Extraction and Systematization of the Information

Following the initial selection phase, we conducted *an in-depth manual analysis of all selected publications*. Beyond the already annotated metadata (authors, title, abstract, publication year, DOI/URL, source, publisher), we systematically extracted relevant factual information to address our research questions. This data was organized into a comprehensive *tabular format* with the following fields:

Paper type categorization as: system paper, dataset paper, or systematic survey;
Problem comprehensive description of the research problem addressed;
Approach authors' proposed methodology for addressing the problem at a conceptual level;
Solution concrete implementation and achievements in solving the stated problem;
Languages comprehensive list of languages utilized in addressing the multilingual aspect;
Knowledge graphs enumeration of knowledge graphs directly utilized in the research;
Datasets compilation of datasets referenced or actively utilized in the study;
Metrics evaluation metrics employed to assess the research outcomes;
Technologies technical tools and frameworks explicitly mentioned in the paper or identified in associated repositories;
Source code & demo URLs direct links to implementation code and/or demonstration applications;
Comment optional notes providing additional context or significant observations about the publication.

Subsequently, we validated all selected publications against the acceptance criteria outlined in Section 3.2.1. Through this validation process, we reduced our final collection to 48 publications, comprising 27 mKGQA system descriptions, 14 benchmarking dataset papers, and 7 survey papers.

3.3 Systematic Surveys about Question Answering over Knowledge Graphs

In this section, we examine relevant survey articles that were selected based on two criteria: their pertinence to our research domain and their alignment with the methodology outlined in Section 3.2. Table 3.3 provides a comprehensive overview of these publications, which are detailed in the following discussion.

Tab. 3.3.: The overview of the survey papers that include the aspect of multilinguality

Authors	Year	Problem	Methodology	# Papers	Multilinguality
Höffner et al. [77]	2016	The SOTA methods are not systematically collected	✓	72	Multilingual systems understanding noisy human natural language input
Diefenbach et al. [49]	2018	Making a vast amount of data in KBs available with KBQA	✓	n/a	The vocab in the user query and the KB vocab are lexicalized in different languages
Dimitrakis et al. [54]	2020	No prior dataset/benchmark surveys were available	✗	n/a	Mentioned as a challenge
Franco da Silva et al. [179]	2020	No surveys available with a focus on Deep Learning	✓	13	Mentioned as a challenge
Zhang et al. [228]	2021	Recent advances in Deep Learning in the KBQA	✗	n/a	Lack of data in languages other than English
Antoniou et al. [5]	2022	No taxonomy of the systems was available	✗	n/a	Mentioned as a challenge
Pereira et al. [150]	2022	KBQA for data accessibility in the biomedical domain	✓	66	Mentioned as a challenge
Our survey [154]	2024	No prior survey articles focusing on multilingual KGQA	✓	48	Comprehensively covered

3.3.1 Overview of the Surveys

Höffner et al. [77] published a survey in 2017 addressing a critical issue in the field: *the frequent redundant development of essential components* rather than collaborative advancement. Their work highlighted *common practices naturally evolve, they lack systematic documentation and organization*. To address these challenges, the authors proposed *a systematic approach to collecting and organizing methodologies for addressing common challenges in the field*. Their methodology employed specific inclusion criteria: publications accessible through Google Scholar[15] using a predefined search query[16] or published in major Semantic Web conferences within a specified timeframe. The study analyzed 72 publications covering 62 systems developed between 2010 and 2015, identifying common challenges and their respective solutions. Their recommendations for future semantic question answering (SQA) systems emphasized the need for enhanced modularization, automatic reuse capabilities, self-wiring mechanisms, and encapsulated modules with dedicated benchmarks and evaluation metrics. The authors also noted *an emerging trend toward multilingual, multi-knowledge source SQA systems* capable of processing natural language input with inherent noise.

Diefenbach et al. [49] published their 2017 survey addressing *the challenge of leveraging question answering systems to access the "enormous amount of information stored in knowledge bases"*. The authors distinguished their work by *emphasizing the technical aspects of existing QA systems, a focus they noted as unique among contemporary surveys*. The survey's primary objective was to systematically analyze,

[15] `https://scholar.google.com/`
[16] The search query: `'question answering' AND ('Semantic Web' OR 'data web')`

categorize, and evaluate techniques employed by QA systems participating in the QALD[17] challenge [203]. The authors implemented a specific methodology for system selection, encompassing both direct QALD challenge participants and systems subsequently evaluated using the QALD framework. Regarding multilingual capabilities, the survey's scope was notably limited, *addressing multilingual functionality only in contexts where user query vocabulary and knowledge base lexicalization existed in different languages*. The treatment of multilingual aspects was minimal, with *only brief mentions of multilingualism as a consideration in QA systems*.

Da Silva et al. [179] conducted a 2020 survey focusing on end-to-end "simple QA systems". They acknowledged that *traditional QA approaches typically involve five distinct steps: question analysis, phrase mapping, disambiguation, query construction, and querying distributed knowledge*. The survey specifically examined deep learning-based QA systems designed for factoid question answering, analyzing how existing systems implement critical features within their end-to-end models. Their systematic review methodology employed three-tiered inclusion criteria: 1) an initial literature search[18] for deep learning-based QA works, 2) narrowing the scope to systems evaluated using the SimpleQuestions benchmark, and 3) selecting works that specifically addressed their predetermined research questions. The review covered publications from 2015 to 2019, initially identifying 59 papers, which was refined to 13 papers after applying their selection criteria. While the survey acknowledged multilingualism as a significant challenge, particularly in bridging the gap between users' local language information needs and culturally-biased semantic data representation, *the discussion of multilingual aspects was limited to a single paragraph*.

Dimitrakis et al. [54] published their comprehensive survey in 2020, distinguishing their work through two key contributions. First, they noted that *while previous surveys up to 2018 focused exclusively on reviewing QA systems, their work provided an extensive compilation of available training and evaluation datasets for QA*. Second, *they explored the potential of integrating various QA system types and information sources into unified pipelines, offering researchers insights into potentially more effective combinatorial approaches*. Their survey presented a comprehensive analysis encompassing text-based, data-based, and hybrid methodologies, along with their corresponding datasets. However, *the survey's treatment of multilingual aspects was limited to a brief paragraph*, suggesting a potential area for future expansion.

[17] Question Answering over Linked Data

[18] The search query: `(("simple questions answering" OR "end to end" OR "user to end" OR "machine learning") and ("natural language" OR "nlp" OR "natural language understand") and (application OR system OR program OR reviews OR techniques OR frameworks OR "practical applications"))`

Zhang et al. [228] published a comprehensive survey in 2021 examining the integration of deep learning approaches in KBQA systems. The survey systematically analyzed *deep learning-based KBQA approaches for simple questions* through two primary frameworks: (1) information extraction methodologies and (2) semantic parsing techniques. The authors then expanded their analysis to examine *neural architectures designed for complex questions requiring multi-hop reasoning capabilities*. Additionally, they *provided a detailed review of prominent KBQA evaluation benchmarks*, including WebQuestions [14], SimpleQuestions [20], and LC-QuAD [199]. The authors identified several persistent challenges in the field: compositional generalizability issues, disparities between natural language and knowledge base representations, insufficient training data, limited knowledge base coverage, and the predominance of English-language resources. While these challenges were thoroughly discussed, it should be noted that the survey did not detail its publication selection methodology, and *multilingual considerations were addressed in only a single sentence*.

Pereira et al. [150] published their 2022 survey on KBQA, addressing critical challenges in KB data accessibility. They identified two significant limitations: *visual navigation's inadequacy for handling complex queries* and *the barrier presented by SPARQL's requirement for formal query language expertise*. Their work primarily focused on applications within the biomedical domain. The survey distinguished itself through its rigorous methodological approach, adhering to PRISMA (Preferred Reporting Items for Systematic Reviews and Meta-Analyses) guidelines [144] for protocol execution and results presentation. Their comprehensive analysis encompassed 66 papers, categorizing KBQA systems by architectural approaches: 25 semantic parsing pipeline systems, 12 subgraph matching implementations, 7 template-based systems, and 22 information extraction architectures. The authors identified two paramount challenges facing the field: the growing demand for systems capable of processing increasingly complex questions and the need to address inherent knowledge base incompleteness. While the survey provided extensive coverage of these aspects, multilingual functionality received limited attention, confined to a single paragraph discussion.

Antoniou et al. [5] presented a survey in 2022 addressing a significant gap in semantic question answering (SQA) systems research. Their survey introduced a novel categorization framework for SQA systems, establishing foundational criteria for systematic classification. The authors developed a multifaceted classification system based on nine key properties: domain specificity, data source characteristics, question types, question analysis approaches, question representation methods, knowledge base characteristics, answer retrieval techniques, user interaction patterns, and

answer properties. This comprehensive framework represented the first systematic attempt to organize and categorize the diverse landscape of SQA systems. While the survey made significant contributions to SQA system classification, it should be noted that the authors did not explicitly detail their methodology for conducting the survey. Additionally, *multilingual functionality received limited coverage, discussed only within a single paragraph*.

3.3.2 Summary

Our analysis of the related survey articles reveals a significant gap: despite multilinguality being consistently identified as a crucial challenge in KGQA systems, none of the reviewed surveys specifically focuses on the multilingual aspects of these systems. *This notable absence of a comprehensive survey dedicated to mKGQA serves as the primary motivation for our current research endeavor*. Furthermore, our analysis highlighted a concerning methodological issue: four of the seven examined survey papers failed to document their review methodology or protocol. This omission raises questions about the reproducibility and systematic nature of their findings. Given this observation, *we strongly advocate for the explicit inclusion of methodological frameworks and review protocols in future survey papers*, as these elements are essential for ensuring research transparency and reproducibility in the field.

3.4 Multilingual Question Answering Systems over Knowledge Graphs

This section presents our comprehensive analysis of 22 mKGQA systems identified through 27 publications[19], which were selected following our methodology outlined in Section 3.2.

Our review is structured in three main parts. First, Section 3.4.1 provides detailed summaries of each publication describing mKGQA systems, systematically documenting key system attributes such as supported languages, knowledge graphs, and code availability. These findings are synthesized into a comprehensive overview presented in Table 3.4.

[19] Some systems are described in multiple publications as they present minor improvements

Second, Section 3.4.2 demonstrates the evaluation results of the selected mKGQA systems. We also summarize other important data, such as language and KG usage, benchmarking datasets and used technologies.

Finally, Section 3.4.3 introduces detailed analysis of the languages used and supported by the selected mKGQA systems.

3.4.1 Review of the Selected Systems

This section presents a methodological classification of the reviewed systems based on their implementation approaches. While detailed descriptions of these method groups can be found in Appendix A, it is important to note that our classification assigns each system to a single dominant method group, even though some systems may incorporate elements from multiple approaches. This classification approach allows us to provide a clear *taxonomic overview* while acknowledging the potential hybrid nature of certain systems.

Systems Predominantly Using Methods Based on Rules and Templates (G1)

The QALL-ME system [62], introduced in 2011, was designed to address the challenge of providing accurate answers to user queries in the context of what the authors describe as "the exponential growth of digital information". The system presents a reusable architectural framework for developing multilingual QA systems capable of processing queries across freely definable domains using structured data sources. While the specific implementation methods are not explicitly detailed, the query generation process utilizes pattern mapping techniques within a Service-Oriented Architecture (SOA) framework. The system *supports four European languages: German, Spanish, English, and Italian*. Performance evaluation showed promising results, with an average accuracy of 72.89% across all languages and an average Recognizing Textual Entailment (RTE) component performance of 86.97%. However, these results should be considered with caution as the evaluation was conducted on a non-public RDF-based knowledge graph and an unpublished benchmark, making direct comparisons with other systems impossible. The authors identified two primary areas requiring further development: the acquisition of minimal question patterns and the enhancement of interactive QA processes. While the system's source code is publicly available in Java, it has become outdated.

The KGQA system developed by Aggarwal [1] was introduced in 2012, addressing the challenge of low accuracy in multilingual natural-language interfaces for Semantic

Web data access. The system architecture parallels QALL-ME's approach, employing a multilingual QA pipeline with three main components: entity search (utilizing exact matching between entity and ontology labels), parse tree generation (implemented via Stanford Parser[20] [123]), and cross-lingual semantic similarity and relatedness computation. *Supporting both English and German queries*, the system operates on the DBpedia knowledge graph. Performance evaluation on the QALD-2 dataset yielded moderate results: Precision (44%), Recall (48%), and F1 score (46%). While these metrics demonstrate the system's basic functionality, the author acknowledges the need for enhanced semantic relatedness measures. Modern approaches could significantly improve performance through the integration of LMs [115] or graph neural networks [218]. The system was implemented in Java, though its current status and availability are not specified.

The QAKiS system [29, 39], introduced in 2013 and extended in 2014 [28], addresses a unique challenge in multilingual knowledge bases: the information asymmetry across different language chapters of DBpedia. The system aims to enhance user interaction with the Web of Data by developing sophisticated query interfaces that facilitate flexible mappings between natural language expressions, concepts, and relations in structured KBs. Its architecture comprises four key multilingual components: Named Entity Recognition (NER) implemented via Stanford Core NLP NER[21], pattern matcher, query generator, and SPARQL package. At its core, QAKiS employs "relational patterns" to capture diverse linguistic expressions of relationships across languages. *The system supports English, French, and German* queries over DBpedia, achieving a precision of 50% when evaluated on the QALD-2 dataset. While innovative in its approach, the system faces a significant limitation: the requirement for language-specific mapping creation hampers its scalability and generalizability.

The SWSNL system [66], introduced in 2013, was designed to enhance form-based search capabilities by enabling natural language and keyword-based queries over domain-specific data. The system, implemented in Java, employs a two-stage query processing pipeline. First, it converts textual questions into knowledge graph-independent queries through preprocessing, NER, and semantic parsing. Subsequently, it transforms these intermediate queries into SPARQL using a rule-based interpretation approach. They also developed a test dataset comprising 68 questions in *English and Czech*. Performance metrics showed mixed results: 66% Precision, 34% Recall, and 45% F1 score. The authors identified several areas for future devel-

[20] `http://nlp.stanford.edu/software/lex-parser.shtml`
[21] `https://stanfordnlp.github.io/CoreNLP/ner.html`

opment, including: expanding the evaluation corpus, incorporating full-text search capabilities, enhancing the NER module, improving overall system performance.

The AMUSE system, introduced in 2017 [68], addresses the challenging "lexical gap" problem in mapping natural language questions to SPARQL queries, particularly in multilingual contexts. The system employs a sophisticated probabilistic inference approach to identify the most likely query interpretation for a given question. Implemented in Java and leveraging Universal Dependencies (UD) [139], AMUSE operates through a two-layer architecture: (1.) The L2KB layer and (2.) The query construction layer. The system ultimately produces SPARQL queries and *supports three languages: English, German, and Spanish*, operating over the DBpedia KG. Performance evaluation on the QALD-6 dataset demonstrated varying effectiveness across languages, with macro F1 score values of 34%, 37%, and 42% respectively.

The WDAqua-core0 system [51], introduced in 2017, addresses the challenge of managing and querying increasingly large volumes of structured Semantic Web data. Built on the Qanary framework [22, 53], WDAqua-core0 demonstrates notable versatility in query processing. *The system operates in four languages: English, French, German, and Italian*, and is compatible with both DBpedia and Wikidata knowledge bases. Performance evaluation on the QALD-7 dataset [205] yielded mixed results: DBpedia (English only): 51.1% F1 score. Wikidata: English: 32.2%, French: 12.7%, German: 24.0%, Italian: 17.3% F1 score. While the original paper provides limited implementation details, subsequent publications on WDAqua-core1 [52] and QAnswer [50] offer comprehensive insights into the framework's architecture and functionality.

The KGQA system UDepLambda [168], introduced in 2017, addresses the notable English-language bias in KGQA research. Following an approach similar to AMUSE, UDepLambda implements a two-stage processing pipeline: first converting natural language questions to logical forms, then transforming these forms into machine-interpretable representations. The system's architecture leverages universal dependencies [138] to achieve near language-independent mapping of natural language to logical forms, effectively capturing predicate-argument structures. *UDepLambda supports English, German, and Spanish queries*, operating over the Freebase knowledge graph. Performance evaluation was conducted on two benchmark datasets, with varying results across languages. On the WebQuestions benchmark [14], the system achieved F1 score values of 49.5%, 46.1%, and 47.5% for English, German, and Spanish respectively. The GraphQuestions benchmark yielded lower performance metrics, with F1 score values of 17.7% for English, 9.5% for German, and 12.8% for Spanish.

The system MuG-QA [231], introduced in 2018, addresses the challenges posed by the rapid expansion of RDF and KGs, particularly focusing on the growing volume of non-English data. The system's core innovation lies in its approach to multilingual question answering through the formation of abstract conceptual grammar from input questions. MuG-QA's grammar framework is built upon the Grammatical Framework (GF) [166] and GF Resource Grammar Library [167], while entity and class linking is accomplished through the "interlanguage-links-dataset" [@86]. Supporting English, French, Italian, and German, MuG-QA demonstrated strong performance in evaluations on the QALD-7 benchmark over DBpedia. The system achieved micro F1 score values of 67.7% for English, 56.6% for French, 65.6% for Italian, and 61.3% for German. The grammar-based approach requires significant expert involvement and higher labor costs for development. Furthermore, despite the language-agnostic nature of the abstract grammar trees, implementing new languages still necessitates the creation of specific language mappings.

The WDAqua-core1 system [52], published in 2018, builds upon its predecessor WDAqua-core0 with significant enhancements. The authors' vision emphasizes that a freely available KGQA solution could catalyze the deployment of question-answering services across diverse data sources, potentially accelerating the publication of new RDF datasets. WDAqua-core1's distinctive approach is founded on the premise that question understanding can be achieved by prioritizing word semantics over syntactic structure. The system implements a sophisticated modular pipeline integrated within the Qanary framework. *The system supports five languages: English, German, French, Italian, and Spanish*, and operates across multiple knowledge graphs including DBpedia, Wikidata, MusicBrainz, and DBLP. Performance evaluation conducted across QALD-{3,...,7} benchmarks showed varying effectiveness, with QALD-7 yielding F1 score values of 42% for English, 25% for German, and 18% each for French, Italian, and Spanish.

The LAMA system [164], introduced in 2018, addresses the fundamental challenge of making RDF data more accessible by providing a natural language interface that eliminates the need for users to learn specialized query languages. The system employs an innovative approach based on lexico-syntactic pattern analysis to generate SPARQL queries, focusing on identifying generalized linguistic structures that represent semantic relationships between concepts. The core processing module then transforms these syntax trees into intermediate representations, which are ultimately converted into triple patterns for SPARQL query generation. The implementation leverages several sophisticated tools: SyntaxNet[22], Penn Treebank [194], OntoNotes [83], and Universal Dependencies [139]. *While LAMA supports both*

[22] `https://ai.googleblog.com/2016/05/announcing-syntaxnet-worlds-most.html`

French and English languages, it relies on the Google Translate API[23] for non-English queries, which may impact performance. Operating on the DBpedia KG, LAMA demonstrated impressive performance in evaluations on QALD-7 and LC-QuAD 1.0 [199] benchmarks, achieving English language F1 score values of 90.5% and 81.6% respectively.

Systems Predominantly Using Statistical Methods (G2)

The UTQA system [161], introduced in 2016, addresses the notable English-language bias in KGQA research. The authors attribute this bias to two primary challenges: the scarcity of multilingual tools and resources, and the "vocabulary gap" between source and target languages. Question processing begins with keyword extraction utilizing a maximum-entropy Markov model, though non-English queries require translation via the Google Translate API. Subsequent steps include keyword type detection through SVM classification, followed by entity linking and ontology type extraction, which employs custom queries across multiple data sources, semantic similarity measures, and the Babelfy tool [133]. The pipeline concludes with answer extraction. *Operating across English, Persian, and Spanish*, UTQA interfaces with the DBpedia KG. The system's evaluation on the QALD-5 benchmark employed two distinct approaches: language-specific processing and machine translation-based processing. The language-specific approach achieved F1 score values of 65.2% for English, 52.4% for Persian, and 54.2% for Spanish, significantly outperforming the machine translation approach, which yielded lower F1 score values of 29.5% for Persian and 32.2% for Spanish.

The Platypus system [147], launched in 2018 and accessible online[24], advances the paradigm of natural language question answering over structured knowledge repositories. Rather than directly generating SPARQL queries, the system introduces an innovative approach using custom logical representations as an intermediate step. The system's architecture comprises two distinct analyzer components working in parallel. A grammatical analyzer employs manually designed rules to transform NL questions into logical representations, while a template-based analyzer identifies the best-matching template for each question and populates logical representation slots accordingly. These intermediate representations are subsequently converted into executable SPARQL queries. The implementation leverages state-of-the-art natural language processing tools, including Stanford Core NLP, SpaCy [81], and Rasa NLU.

[23] `https://cloud.google.com/translate`
[24] `https://qa.askplatyp.us/`

Platypus supports both French and English languages and interfaces with Wikidata as its knowledge base.

The QAnswer system, published in 2019 [50, 48], evolves from its predecessors WDAqua-core0 and WDAqua-core1 to address the limited accessibility of Linked Open Data (LOD) datasets. This limitation stems from the prevalent single-dataset, single-language constraints of existing systems. Building upon WDAqua-core1's multilingual and KG-agnostic foundation, QAnswer implements a four-stage question answering process: question expansion, query construction, query ranking, and answer decision. The system introduces innovative feedback mechanisms and re-training capabilities based on user interactions, enhancing its adaptability and performance over time. *The system demonstrates extensive language support, including English, German, French, Italian, Spanish, Portuguese, Arabic, and Chinese*. Its knowledge base integration encompasses major repositories: Wikidata, DBpedia, MusicBrainz, DBLP, and Freebase. While performance metrics for QALD-{3, ..., 7} [203] and LC-QuAD 1.0 [199] benchmarks are detailed in the WDAqua-core1 analysis, QAnswer achieved impressive F1 score values on specialized benchmarks: 92% for "cocktail", 97% for "HR", and 90% for "EU".

The DeepPavlov system [60], introduced in 2020, specializes in addressing complex questions requiring "logical or comparative reasoning" (i.e., "If Sarah is older than Tom, and Tom is older than Lily, who is the youngest?"). The system's pipeline, powered by deep-learning neural networks, comprises several specialized components: query template prediction, entity detection, entity linking, relation ranking, path ranking, constraint extraction, and final query generation. Each component utilizes state-of-the-art machine learning techniques. Query template classification leverages the BERT [46] large language model's CLS token, while entity detection employs BERT-based sequence labeling. The entity linking process implements fuzzy matching with inverted index, and relation ranking processes question token embeddings through a 2-layer Bi-LSTM, computing dot products with relation embeddings before applying Softmax. Additionally, path ranking utilizes BERT for candidate relation evaluation, while constraint extraction employs regular expressions for modifier identification. *Currently supporting English and Russian languages*, DeepPavlov interfaces with the Wikidata KG. The system demonstrated strong performance on the LC-QuAD 2.0 dataset, achieving 60% Precision, 66% Recall, and 63% F1 score.

The Tiresias system [134], published in 2022, addresses the challenge of multilingual accessibility in KGQA systems. The system innovates by complementing structured DBpedia information with multilingual DBpedia abstracts, expanding the available knowledge base for question answering. Tiresias employs a sequential processing

pipeline that begins with named entity recognition using DBpedia Spotlight, followed by retrieval of relevant DBpedia abstracts through SPARQL queries. The system then translates non-English questions to English using either Bing or Opus MT services, before generating final answers using a pre-trained BERT-like QA model. The system's evaluation was conducted on a custom-designed bilingual dataset supporting *English and Greek*. The authors implemented a manual evaluation framework categorizing responses as correct, partially correct, or wrong.

The *DeepPavlov 2023 system* [201] advances question answering technology by generating complete NL answers through KG triplet verbalization. This updated version significantly improves upon its predecessor [60], achieving state-of-the-art performance on the Russian RuBQ 2.0 benchmark [169]. The system implements a comprehensive pipeline comprising entity detection, entity linking, relation ranking, SPARQL template prediction, SPARQL slot filling, and path ranking. These components culminate in generating executable SPARQL queries for Wikidata. The answer generation phase utilizes the JointGT model [101] to transform query paths and answer URIs into natural language responses. Each component leverages BERT-based models trained on various KGQA datasets, including LC-QuAD 1.0 [199]. While supporting both *English and Russian* languages, the system operates as two separate instances due to its reliance on monolingual neural models. Performance evaluation shows promising results, with the English version achieving 47% F1 score on the LC-QuAD dataset and the Russian version reaching 53.1% F1 score on RuBQ 2.0.

The XSemPLR approach [229] presents a comprehensive investigation into cross-lingual semantic parsing (CLSP), encompassing various meaning representations including SPARQL, SQL, and lambda calculus. While the work's primary contribution is a unified CLSP benchmark synthesized from nine existing datasets, its evaluation of multilingual LLMs on KGQA tasks provides particularly valuable insights for the field. The study focuses on CLSP over SPARQL using the MCWQ benchmark [43], which features questions in *English, Hebrew, Kannada, and Chinese* mapped to Wikidata queries. The evaluation encompasses several state-of-the-art language models: LSTM [76], mBERT+Pointer-based Decoders (PTR) [46], XLM-R+PTR [40], mBART [36], Codex [33], BLOOM [175], and mT5 [224]. These models were tested across various configurations: monolingual, monolingual few-shot, multilingual, cross-lingual zero-shot transfer, and cross-lingual few-shot transfer. The mT5 model demonstrated superior performance in the monolingual setting, achieving *exact match* scores of 39.29%, 33.02%, 23.74%, and 24.56% for the respective languages.

The CLRN system [192] introduces an innovative approach to Cross-lingual KGQA (CLKGQA), challenging the conventional methodology of merging multiple Cross-lingual Knowledge Graphs (CLKGs). To address this challenge, the authors present the Cross-lingual Reasoning Network (CLRN), an advanced multi-hop QA model that enables dynamic transitions between knowledge graphs during reasoning processes. The system implements an iterative framework that synergistically combines CLRN and EA models, extracting potential alignment triple pairs from CLKGs during question answering, thereby enhancing EA model effectiveness. The system's evaluation was conducted on the MLPQ benchmark [191], which encompasses language-specific DBpedia KGs in *English, Chinese, and French*. The results demonstrate CLRN's superior performance compared to baseline systems, with notable improvements in EA model accuracy through iterative enhancement, achieving a significant 1.0% increase in both Hit@1 and Hit@10 metrics.

The MST5 system [187] introduces a streamlined, transformer-based approach to multilingual KGQA, challenging traditional methods that employ separate encoders to integrate auxiliary linguistic context and entity information. To address this, the authors present MST5—a unified framework that concatenates natural language queries with extracted linguistic features and entity links before processing them with a single, pretrained multilingual transformer model. The core component of the MST5 approach is the mT5 [224] model, fine-tuned on the SPARQL query generation task. This design simplifies the SPARQL generation process and also enhances the model's capacity to accurately parse queries into executable graph queries. The system's performance was rigorously evaluated on the QALD-9-Plus [155] (9 languages) and QALD-10 [208] (4 languages) benchmarks, spanning low-resource languages, and demonstrated significant improvements in multilingual KGQA quality over baseline systems. For example, MST5 outperforms the DeepPavlov 2023 system on English and Russian: 41.87% vs 37.16% and 37.61% vs 31.17% respectively.

Systems Predominantly Using Machine Translation Methods (G3)

The system developed by Y. Zhou et al. [230], introduced in 2021, addresses the growing demand for multilingual KGQA capabilities while acknowledging the substantial costs associated with developing language-specific knowledge graphs and annotating QA datasets. The authors identify a significant performance disparity between source and target languages in KGQA tasks, a challenge consistently observed across various natural language processing applications. The system's innovative approach centers on a self-supervised pre-training methodology for multilingual

transformer encoders, followed by fine-tuning on data-rich source languages. The authors adapt this paradigm to KGQA for constructing symbolic logical forms in KG queries, notably replacing traditional machine translation with unsupervised bilingual lexicon induction (BLI) [90] for word-level translation. *The system supports an impressive range of languages: English, Farsi, German, Romanian, Italian, Russian, French, Dutch, Spanish, Hindi, and Portuguese*, operating on the DBpedia KG. Evaluation on LC-QuAD 1.0 (machine-translated to multiple languages via Google Translate) and QALD-9 benchmarks yielded average F1 score values of 75.9% and 63.0% respectively.

The system by A. Perevalov et al. [153], published in 2022, addresses the challenge of unequal language distribution and content accessibility on the Web, particularly within the KGQA domain where multilingual access solutions remain limited. The authors investigate the effectiveness of integrating machine translation (MT) tools with existing KGQA systems, evaluating whether MT could serve as a viable alternative to dedicated multilingual solutions. The research combines three established KGQA systems (QAnswer, DeepPavlov, and Platypus) with two translation services: Yandex Translate API[25] and Helsinki NLP [197]. The evaluation utilizes the QALD-9-plus benchmark [155], testing performance across both DBpedia and Wikidata knowledge graphs for *nine languages: English, German, French, Russian, Ukrainian, Lithuanian, Belarusian, Bashkir, and Armenian*. The evaluation reveals peak performance with QAnswer achieving a 44.59% F1 score on English-language queries without translation. Notably, the results suggest that translating queries to English consistently yields better QA quality compared to using systems' native language support, regardless of the source language.

The *Lingua Franca approach* [188] advances upon Perevalov et al.'s work by introducing a named entity-aware machine translation methodology integrated with mKGQA systems. The system's innovative feature lies in its utilization of Wikidata's symbolic information to ensure accurate named entity translation across languages. The solution implements a two-step processing pipeline: (1) Named entity recognition and linking to identify question entities; (2) Machine translation enhanced with an entity-replacement technique that leverages Wikidata's language-specific entity labels The system's evaluation was conducted using QAnswer and Qanary KGQA systems on the QALD-9-plus dataset, focusing on German, French, and Russian questions. The results demonstrated Lingua Franca's superiority over standard machine translation tools, with improved performance in 19 out of 24 experimental scenarios, validating the effectiveness of entity-aware translation in multilingual KGQA systems.

[25] `https://yandex.com/dev/translate/`

Summary

Table 3.4 presents a comprehensive overview of the reviewed systems, chronologically ordered by publication date. For each system, we analyze the following key characteristics:

- *publication year* of the research;
- *languages* evaluated or supported by the system;
- *knowledge graphs* utilized in the evaluation process;
- *datasets* and benchmarks employed for system assessment;
- *metrics* implemented to evaluate QA performance;
- *technologies* underlying the system implementation;
- *code/demo* public availability status;
- *methods* employed, categorized according to the taxonomy presented in Appendix A.

Appendix A introduces a comprehensive taxonomy of methodologies employed in the development of mKGQA systems that was defined by us in [154].

3.4.2 Evaluation Results of the Reviewed Systems

While addressing *RQ3.2*, we systematically compiled and analyzed evaluation results from the reviewed systems across various benchmarks, focusing on their most recent performance metrics. Although most studies report standard evaluation metrics such as Precision, Recall, and F1 score, some researchers specify additional details like averaging strategies (e.g., micro, macro) or employ alternative metrics (e.g., Accuracy, Hits@k, or Exact match for SPARQL queries). To ensure data consistency, we explicitly document the metrics used in each evaluation (refer to the detailed table in our online appendix[26]).

Based on the performance analysis presented in Section 3.4.1, KGQA systems utilizing multilingual LMs (Group 2) and MT approaches (Group 3) demonstrate superior performance in multilingual contexts compared to traditional methods (Group 1). Notably, our analysis reveals a consistent pattern where English language performance significantly exceeds that of other languages in terms of QA quality.

[26] https://github.com/Perevalov/multilingual-KGQA-survey/blob/main/data/review_tables/evaluation-unified-metric-column.tsv

Tab. 3.4.: The overview on the multilingual KGQA systems published between 2011 and 2025.

System name	Year	Languages	Knowledge Graphs	Datasets	Metrics	Technologies	Code/Demo	Methods (Taxonomy)
QALL-ME [62]	2011	en, de, es, it	Custom KG	Custom data	Accuracy, RTE	Java	✗	M1.4
N. Aggarwal [1]	2012	en, de	DBpedia	QALD-2	Precision, Recall, F1 score	Stanford Parser, Java	✗	M1.1
QAKiS [27, 29, 28, 39]	2013	en, fr, de	DBpedia	QALD-2	Precision, Recall	Stanford Core NLP	✗	M1.4, M2.1
SWSNL [66]	2013	en, cs	Custom KG (accommodation domain)	Custom data, ConnectionsCZ, ATIS	Precision, Recall, F1 score	Prague Dependency Treebank, MaxEntNER, LINGVOParser, OntologyNER, WSim, Java	✗	M1.2, M1.3
UTQA [161]	2016	en, fa, es	DBpedia	QALD-5	Precision, Recall, F1 score	Google Translate	✗	M2.1
AMUSE [68]	2017	en, de, es	DBpedia	QALD-6	Macro F1 score	Universal Dependencies, Java	✓	M1.1, M1.3
WDAqua-core0 [51]	2017	en, fr, de, it	DBpedia, Wikidata	QALD-7	Precision, Recall, F1 Score	Qanary	✗	M1.4, M2.1
UDepLambda [168]	2017	en, de, es	Freebase	WebQuestions, GraphQuestions	F1 Score	Universal Dependencies, Java	✓	M1.3, M1.4
MuG-QA [231]	2018	en, de, it, fr	DBpedia	QALD-7	Precision, Recall, F1 Score	Grammatical Framework (GF), The GF Resource Grammar Library	✗	M1.1, M1.2
WDAqua-core1 [52]	2018	en, de, fr, it, es	DBpedia, Wikidata, MusicBrainz, DBLP	QALD-{3-7}, LC-QuAD 1.0	Precision, Recall, F1 score	Qanary	✗	M1.4, M2.1
LAMA [164, 163]	2018	fr, en	DBpedia	QALD-7, LC-QuAD 1.0	F1 score	Google Translate, SyntaxNet, Penn Treebank, OntoNotes, Universal Dependencies	✗	M1.1, M1.3, M2.1, M3.1
Platypus [147]	2018	fr, en	Wikidata	WikidataSimpleQuestions		Core NLP, Spacy, RasaNLU	✓	M1.2, M1.3, M2.1
QAnswer [50, 48]	2019	en, de, fr, it, es, pt, ar, zh	Wikidata, DBpedia, MusicBrainz, DBLP, Freebase	QALD-{3-7}, LC-QuAD 1.0	Precision, Recall, F1 score, Runtime	Java, HDT	✗	M1.4, M2.1
DeepPavlov [60]	2020	en, ru	Wikidata	LC-QuAD {1.0, 2.0}	Precision, Recall, F1 Score	Python	✓	M1.4, M2.2
Y. Zhou et al. [230]	2021	en, fa, de, ro, it, ru, fr, nl, es, hi, pt	DBpedia	LC-QuAD 1.0, QALD-9	ICA, F1 score	Google Translate	✗	M2.2, M3.2
A. Perevalov et al. [153]	2022	en, de, fr, ru, uk, lt, be, ba, hy	Wikidata, DBpedia	QALD-9-Plus	Precision, Recall, F1 Score	Python, Yandex Translate	✓	M3.1
Tiresias [134]	2022	en, gr	DBpedia	Custom data	Custom metric	Python, Transformers, RDFLib, Spark NLP	✓	M2.2, M3.1
DeepPavlov 2023 [201]	2023	en, ru	Wikidata	RuBQ 2.0	Precision, Recall, F1 Score	Python	✓	M1.4, M2.2
XSemPLR [229]	2023	en, zh, he, kn	Wikidata	MCWQ	Exact Match	Python, OpenAI	✓	M2.2
CLRN [192]	2023	en, zh, fr	Wikidata	MLPQ	Hits@k	Python, Transformers	✓	M2.2
Lingua Franca [188]	2023	de, fr, ru	Wikidata	QALD-9-plus	Macro F1 Score, BLEU, NIST	Python, Transformers	✓	M3.2
MST5 [187]	2024	en, de, fr, ru, uk, lt, be, ba, hy, es, zh, ja	Wikidata	QALD-9-plus, QALD-10	Macro F1 Score	Python, Transformers	✓	M2.2

3.4.3 Language Coverage by the mKGQA Systems

To provide a comprehensive answer to *RQ3.2*, we conducted a detailed analysis of the reviewed mKGQA systems (see Table 3.4), examining their coverage across different languages and language families. The results of this analysis are visually represented in Figure 3.1.

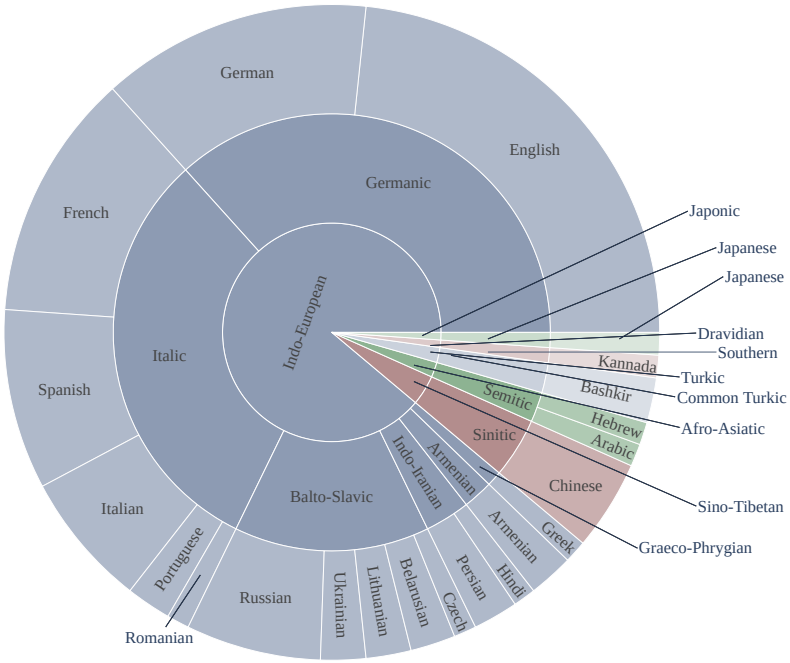
Figure 3.1a reveals a significant dominance of the Indo-European language family in mKGQA research, far surpassing other families (Turkic, Sino-Tibetan, Afro-Asiatic). Our analysis indicates that approximately 88% of the reviewed mKGQA systems target at least one Indo-European language. Figure 3.1b illustrates the distribution of language support across different systems. English emerges as the universally supported language, with all 22 reviewed systems capable of processing English queries. German ranks second in terms of support, with 12 out of 22 systems accommodating German input. The majority of other languages, including Arabic and Romanian, are supported by only a single system. It is important to note that while monolingual KGQA systems exist for languages other than English, they fall outside the scope of our review, which specifically focuses on multilingual capabilities.

3.4.4 Summary

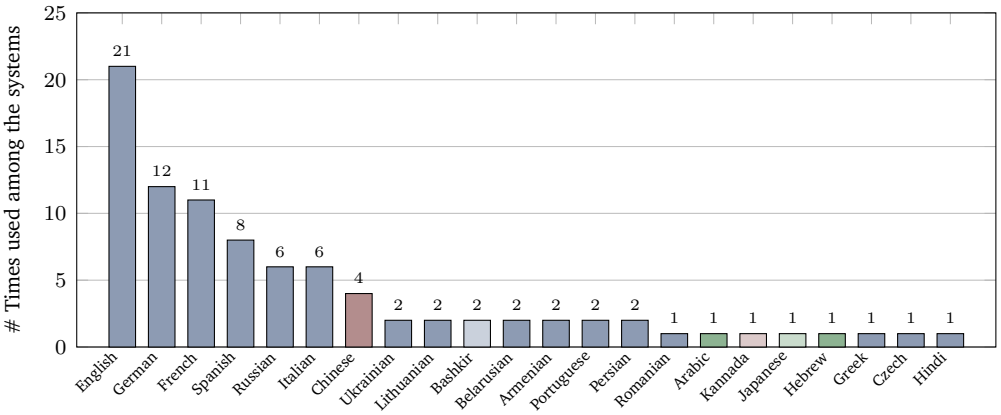
Based on our comprehensive review of the past decade, we identified only 22 mKGQA systems developed within the context of mKGQA. A strong observation is that 88% of these mKGQA systems *predominantly focus on the Indo-European language family*. Consequently, *these systems operate primarily within a single writing system*—Alphabetic (encompassing Latin, Cyrillic, Armenian, or Greek scripts). This limited scope raises significant questions about the true generalizability and scalability of current methodologies across linguistically diverse languages.

Our analysis identifies *three primary methodological approaches in mKGQA* (refer to Appendix A): rules and templates (G1), statistical methods (G2), and machine translation (G3). The taxonomic analysis reveals that *researchers typically adapt existing monolingual methods to other languages rather than developing inherently language-agnostic solutions*. We observed that *systems often resist strict categorization into a single method group, as they frequently integrate multiple approaches*. Furthermore, we have collected and contributed the comprehensive evaluation results of the systems to the KGQA leaderboard [157], which is publicly accessible[27].

[27] `https://kgqa.github.io/leaderboard/`



(a) The sunburst chart represents the number of systems tackling each language family, language branch, and language respectively



(b) The bar chart represents the number of systems supporting a concrete language. The language are color-coded by their language families similarly to Figure 3.1a.

Fig. 3.1.: The visual representation of language and language family coverage among the mKGQA systems

3.5 Benchmarking Datasets for Multilingual Question Answering over Knowledge Graphs

This section examines the benchmarks specifically designed for mKGQA evaluation. Through a comprehensive analysis of these benchmarks, we highlight their distinctive features and contributions, illuminating both the advancements and persistent challenges in the field of mKGQA.

3.5.1 Overview

Research in KGQA is heavily data-dependent, yet a significant *challenge persists in the scarcity of benchmarks for reliable multilingual KGQA system evaluation* [118, 137]. While the field of OpenQA has seen several initiatives in machine-translating existing benchmarks (e.g., [118, 31]), KGQA lacks similar developments. Our review identified *only 6 benchmarks (or benchmark series[28]) that address multiple languages*: Question Answering over Linked Data (QALD) [26, 203, 155], EventQA [186], which focuses on event-centric questions over knowledge graphs, the RuBQ dataset for Question Answering over Wikidata [110, 169], Multilingual Compositional Wikidata Questions (MCWQ) [43], Mintaka [178], and MLPQ (A Dataset for Path Question Answering over Multilingual Knowledge Graphs) [191]. Table 3.5 provides an overview of these benchmarks.

The QALD represents a pioneering benchmark series for mKGQA. Its evolution through multiple versions (QALD-{3,6,7,8,9,9-plus,10}[29]) demonstrates significant progression in multilingual question answering evaluation. The series began with QALD-3, comprising 199 questions and corresponding SPARQL queries for DBpedia and MusicBrainz[30]. QALD-6 expanded to 450 questions over DBpedia, introducing the standardized QALD JSON format that includes multilingual textual representations, SPARQL queries, answer entity URIs, and answer types. QALD-7 incorporated both DBpedia and Wikidata support with 258 questions, while QALD-8 focused on DBpedia with 260 questions. A significant expansion occurred with QALD-9, featuring 558 questions over DBpedia. Its enhanced version, QALD-9-plus [155], introduced improved translations in eight languages and expanded coverage to include Wikidata[31]. These translations were validated through crowd-sourcing by native

[28] Some of the benchmarks (e.g., RuBQ 1-2, or QALD 1-10) exist in multiple versions. We refer to these collectively as *benchmark series*.
[29] The other versions of QALD had only English questions.
[30] `https://musicbrainz.org/`
[31] `https://www.wikidata.org/`

speakers. Further language-specific improvements included the rewordQALD9 [173] for Italian translations and QALD-9-ES [184] for Spanish, which addressed previous translation flaws while also generating English paraphrases. The latest iteration, QALD-10 [208], introduces 402 new questions in English, Chinese, German, and Russian, maintaining the established QALD JSON structure. The benchmark series has become a cornerstone for numerous KGQA research studies (e.g., [77, 53, 181, 47]), establishing itself as a standard in the field.

EventQA specializes in evaluating systems' ability to answer event-centric questions (e.g., "In which tournament, known as major, did Jason Dufner win?"). The benchmark comprises 1000 questions, each available in English, German, and Portuguese, targeting EventKG [65], a comprehensive event-centric KG containing 690,247 events. The dataset employs a custom-designed JSON structure for data representation.

The RuBQ benchmarking series consists of two versions, with RuBQ 2.0 expanding upon its predecessor to include 2910 questions. Like recent QALD versions, RuBQ utilizes Wikidata as its target knowledge base for SPARQL queries. The benchmark's development followed a rigorous semi-automatic methodology: question-answer pairs were first automatically collected, then processed through entity linking tools, followed by crowd-worker verification of the linked entities. The final step involved generating and manually validating SPARQL queries based on the verified question and answer entities. The dataset features questions in native Russian with machine-translated English versions. Each entry is enriched with comprehensive metadata including entities, relations, answer entities, SPARQL queries, answer-bearing paragraphs, and query-type tags, all organized in a custom-designed JSON structure.

MCWQ is a benchmarking dataset that extends the CFQ dataset [104], targeting Wikidata (similar to QALD and RuBQ). The benchmark features questions in Hebrew, Kannada, Chinese, and English, with non-English content generated through machine translation and enhanced by rule-based refinements. The dataset employs a comprehensive structure that includes questions with entity highlighting, original Freebase [19] SPARQL queries (inherited from CFQ), newly created Wikidata SPARQL queries, and multilingual question representations across all four languages. The data is organized in a custom-designed JSON format with additional metadata fields.

Mintaka is a contemporary benchmark featuring 20,000 questions across nine languages: English, Arabic, French, German, Hindi, Italian, Japanese, Portuguese, and Spanish. Like QALD, RuBQ, and MCWQ, it utilizes Wikidata as its knowledge

base. The benchmark's structure incorporates crowd-worker-annotated named entities, gold standard answers, and metadata including question categories (e.g., geography) and complexity levels (e.g., ordinal). Unlike traditional benchmarks, Mintaka doesn't provide gold standard SPARQL queries, instead offering Wikidata entity IDs as answers along with their language-specific labels. The dataset's content, including questions, named entity annotations, and gold standard answers, was created and validated through crowd-sourcing efforts. The data is organized in a custom-designed JSON format.

MLPQ is a benchmark comprising 300,000 questions in English, Chinese, and French. Following the approach of the early QALD versions, MLPQ uses DBpedia as its knowledge base for SPARQL queries. The benchmark employs a unique RDF structure that encapsulates the question text with language tags, ground-truth answer URIs, and corresponding SPARQL queries. Given its automated generation process, the authors have provided transparency by including the query templates used in creation. Furthermore, MLPQ is distributed with relevant DBpedia triples that correspond to all questions in the dataset.

Tab. 3.5.: Overview of the mKGQA benchmarks

Name	Domain	# questions	Format	Languages	KGs	Comment
QALD-3 [26]	General	199	RDF XML & Turtle	en, de, es, it, fr, nl	DBpedia, Musicbrainz	Translations quality is poor
QALD-6 [202]	General	450	QALD JSON	en, fa, de, es, it, fr, nl, ro	DBpedia	
QALD-7 [205]	General	258	QALD JSON	en, pt, de, es, it, fr, nl, hi	DBpedia, Wikidata	
QALD-8 [204]	General	260	QALD JSON	en, de, es, it, fr, nl, hi, ro	DBpedia	
QALD-9 [203]	General	558	QALD JSON	en, de, ru, pt, hi, fa, it, fr, ro, es, nl	DBpedia	
QALD-9-plus [155]	General	558 (507)	QALD JSON	en, de, fr, ru, uk, lt, be, ba, hy, es*	DBpedia, Wikidata	Not all questions are covered by Wikidata
rewordQALD9 [173]	General	558	QALD JSON	en, it	DBpedia	English questions were paraphrased
QALD-10 [208]	General	909	QALD JSON	en, de, zh, ru	Wikidata	Test set has only four languages
EventQA [186]	Events	1,000	EventQA	en, de, pt	EventKG	Lack of event-centric KGQA systems
RuBQ 1.0 [110]	General	1,500	RuBQ	en, ru	Wikidata	Machine-translated questions from Russian to English
RuBQ 2.0 [169]	General	2,910	RuBQ	en, ru	Wikidata	
MCWQ [43]	General	124,187	MCWQ	en, he, kn, zh	Wikidata	Rule-based generation, translations obtained with MT
Mintaka [178]	General	20,000	Mintaka	en, ar, de, ja, hi, pt, es, it, fr	Wikidata	Named entities are annotated in the English questions
MLPQ [191]	General	300,000	MLPQ	en, zh, fr	DBpedia	Uses multilingual DBpedia with inter-language links

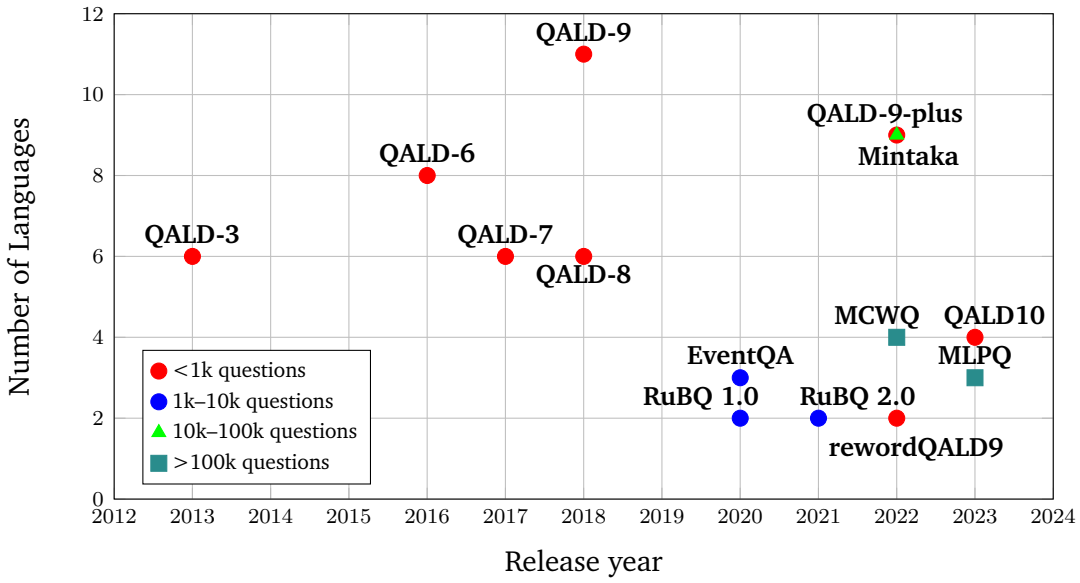


Fig. 3.2.: Temporal progression of the benchmarks with respect to the number of languages.

3.5.2 Summary

Analysis of the reviewed benchmarks reveals limited attention to mKGQA research. Based on our systematic review, 14 papers addresses multilingualism (including various versions that are subsets of newer releases). A notable trend is the increasing adoption of crowd-sourcing methodologies in benchmark creation. These crowd-sourcing efforts range from entity linking verification [110] and question translation/validation [155] to complete question creation and annotation [178]. While addressing $\mathcal{RQ}3.3$, we observed a recent trend toward larger benchmark sizes (starting from 2020), although the language coverage remains variable (as illustrated in Figure 3.2).

Our analysis revealed several significant limitations in existing benchmarks:

- Manually and semi-automatically created benchmarks are limited to a maximum order of magnitude of 10^4 questions (e.g., QALD, RuBQ);
- There is a lack of standardization in benchmark formats across research groups (with QALD JSON being the only exception);
- Most benchmarks are restricted to a single KG, predominantly Wikidata (e.g., RuBQ, Mintaka);
- Benchmarks created through machine translation or automated methods often have questionable quality (RuBQ, MCWQ, MLPQ).

These identified limitations present opportunities for future research directions. In addition, we have contributed the information about the benchmarking datasets to the KGQA leaderboard [157], mentioned earlier.

As it is crucial to have a trustworthy evaluation process and a comparison between different KGQA systems, the priority should be given to developing a multilingual benchmark or extending an existing one in terms of question count, language coverage, and knowledge graph diversity.

3.6 Discussion and Challenges

This section examines the state of mKGQA research through two key perspectives. First, we analyze the critical challenges and limitations researchers encounter when developing and evaluating multilingual KGQA systems. Second, we explore promising future research directions in mKGQA, identifying areas that demand deeper investigation and innovative approaches. Through this comprehensive examination of both challenges and opportunities, we aim to provide strategic insights to guide future advancements in the field of mKGQA.

Our comprehensive review of existing survey papers, mKGQA systems, and associated datasets has revealed several significant challenges in this research domain. The following sections highlight and analyze the most critical challenges currently facing the mKGQA field.

Noisy Human Natural Language Input

A significant challenge in mKGQA is *effectively handling noisy human natural language input*. This challenge becomes particularly complex when dealing with multiple languages due to their diverse structures, grammatical rules, and vocabularies [4]. Input variations can range from well-formed NL questions that follow proper linguistic rules to simplified keyword queries lacking formal sentence structure.

Grammatical and orthographic errors present another layer of complexity in mKGQA systems. Users, especially when interacting in non-native languages, frequently make mistakes in sentence construction, word choice, and grammar usage. These errors significantly *complicate the accurate interpretation of user intent*. Systems employing G1 “Rules and Templates” methods (e.g., SWSNL [66], AMUSE [68], UDepLambda [168]) are particularly vulnerable to such linguistic noise.

Misspellings of named entitites pose an additional challenge in multilingual contexts. While named entities are crucial for understanding question context and semantics [118], users often misspell or incorrectly transliterate them across languages. For example, "Eiffel Tower" might be incorrectly written as "Eiffle Tower" when translating from French to English. This creates significant challenges in *accurately mapping and resolving named entities across different languages*. This issue was initially identified by Perevalov et al. [153] and subsequently addressed in the Lingua Franca system by Srivastava et al. [188].

A User Question and a Knowledge Graph are Expressed in Different Languages

A fundamental challenge identified in the literature is managing the mKGQA process, particularly *when user queries and KGs exist in different languages* [192], a phenomenon known as cross-linguality. Two primary approaches have emerged to address this cross-lingual challenge. The first involves *enriching KGs with multilingual labels*, which enables these knowledge bases to process and understand queries in multiple languages, thus providing a more universal and accessible user experience. The second approach focuses on *leveraging text and graph vector representations* [32]. This strategy is particularly valuable given the frequent scarcity of multilingual training data for direct model training in mKGQA. Vector representations effectively capture cross-lingual semantic similarities and entity relationships, facilitating knowledge retrieval and alignment despite limited multilingual training resources. Although the LAMA system [164] specifically addressed cross-linguality using language-specific DBpedia KGs, it is worth noting that current KG versions, including DBpedia, no longer maintain language-specific divisions.

The Use of Cultural Traits and Country-Specific Aspects

A crucial yet often overlooked challenge in mKGQA is *integration of cultural traits and country-specific considerations in information retrieval*. This challenge comes from the diverse preferences, interests, and expectations that vary significantly between different cultures and geographical regions, as shown in our study [151].

Users' *information seeking behavior and result expectations are heavily influenced by their cultural background and geographical location*. For example, when users from different countries query about popular TV series, their expectations may vary substantially—what is considered popular in the United States might differ markedly

from preferences in Russia or Germany. Such cultural variations in popularity perception and content evaluation create unique challenges for the development of truly effective multilingual systems.

To address this cultural complexity, systems need to incorporate an understanding of cultural nuances and provide contextually appropriate responses to users of diverse backgrounds. The most promising approach would be to *involve collecting and analyzing real-user feedback on entity rankings* for common search topics (e.g., TV series). This could be implemented through structured crowd-sourcing initiatives, systematic result collection, and the development of culturally aware ranking datasets. However, to our knowledge, this significant challenge has not yet been addressed in current mKGQA research.

Quality Discrepancies Among Different Languages

A significant challenge in mKGQA is the *marked disparity in QA quality between English and other languages*. This inequality is particularly evident in evaluation results [153] for systems such as QAnswer, DeepPavlov, and Platypus, which show substantial performance variations across different languages. These findings underscore the urgent need for research initiatives focused on improving multilingual system performance and reducing the quality gap between English and other languages. Several factors contribute to these performance disparities. The scarcity of high-quality training data and resources for non-English languages often results in suboptimal model training. Additionally, the inherent complexities of different linguistic structures and semantic variations across languages further compound the challenge of achieving consistent performance across multiple languages. Addressing these challenges requires a multi-faceted approach. Key priorities include developing enhanced techniques for multilingual KG alignment, creating more comprehensive and diverse benchmarking datasets, and improving training methodologies for multilingual QA models. A particularly promising direction involves leveraging zero- or few-shot inference capabilities of LLMs. These strategies aim to mitigate English-centric bias and achieve more balanced performance across all languages.

The Lack of Language Diversity

The mKGQA field faces significant *challenges when addressing languages from different language families, particularly those using distinct alphabets or writing systems*. Current systems primarily focus on widely spoken languages (e.g., QAKiS [29],

MuG-QA [231]), often neglecting the rich linguistic diversity of low-resource and endangered languages. Notable exceptions include QAnswer [50], Zhou et al. [230], and Perevalov et al. [153], which have demonstrated capabilities across a broader range of languages.

The scarcity of mKGQA systems supporting low-resource and endangered languages is particularly concerning, with only a few systems addressing these linguistic communities (e.g., Bashkir language support in [153]). This limitation significantly impacts the accessibility and inclusivity of KG-based information retrieval for diverse language communities.

Current developments in mKGQA commonly *rely on machine translation approaches or adapt monolingual methods* to achieve multilingual functionality (cf. [153, 188]). However, these adaptations typically prioritize well-resourced language pairs, failing to address the unique challenges presented by low-resource languages.

Size and Translation Quality of Benchmarking Datasets

A significant limitation in the field is that current *benchmarks for mKGQA are relatively small in scale*. Specifically, non-automatically generated benchmarks typically contain only approximately 10^3 instances (e.g., QALD-9-plus [155]).

These size constraints present substantial challenges for comprehensive system evaluation and benchmarking. The restricted scale inherently limits the range and complexity of queries and contexts that can be tested. Consequently, this limitation affects the validity of performance assessments and potentially impedes the advancement of robust mKGQA systems.

Furthermore, translation quality emerges as a critical concern within these benchmarks. Many datasets rely on machine translation (e.g., RuBQ 2.0 [169]), which often results in suboptimal linguistic accuracy. The presence of poor-quality translations (e.g., QALD-9 [203]) introduces significant data inconsistencies, potentially compromising the validity of system evaluations and cross-lingual performance comparisons.

Effective Usage of Large Language Models

The emergence of LLMs has sparked significant interest in their application to mKGQA and broader NLP challenges. However, *empirical evidence from the research*

community indicates a notable disparity between anticipated and actual performance levels.

This performance gap is particularly evident in cross-lingual semantic parsing applications. Zhang et al. [229] documented substantial shortfalls in accuracy and quality metrics when applying LLMs to cross-lingual semantic parsing tasks. Even in monolingual contexts, the research by Klager et al. [107] revealed similarly disappointing results in semantic parsing applications.

A more promising direction has emerged through the integration of structured data directly into the mKGQA pipeline, building on established techniques from monolingual KGQA research [10, 95].

3.7 Conclusion

This chapter introduced the systematic survey that presents a thorough examination of the current research landscape in mKGQA. The work addresses a significant gap in the literature, as no comprehensive surveys specifically focusing on mKGQA previously existed (see Section 6.1). Following rigorous review methodologies (see Section 3.2), we ensured reproducibility and transparency throughout our analysis.

Our initial search yielded 1875 publications based on predefined criteria, with searches conducted across English, German, and Russian literature. The final analysis included 48 publications, comprising 27 system-related papers, 14 benchmark-related papers, and 7 survey papers.

Through systematic review, we developed a comprehensive taxonomy of mKGQA methodologies. We established formal definitions for three critical system characteristics: resource efficiency, multilinguality, and portability. To support future research, we have created an online repository[32] containing structured information about reviewed models and tools, along with source data and analysis scripts.

Our analysis identifies key challenges and future research directions in mKGQA, incorporating recent developments in LLMs. *This work represents the first systematic survey specifically dedicated to mKGQA*, providing a foundation for future research in this rapidly evolving field.

[32] `https://github.com/Perevalov/phd_thesis`

Addressing $\mathcal{RQ}3.1$, we developed a comprehensive taxonomy categorizing mKGQA methods into three primary groups: Rules and Templates, Statistical Methods, and Machine Translation. We introduced a novel evaluation framework using a multi-objective function that considers both average quality across languages and standard deviation, enabling system classification into quality quadrants.

For $\mathcal{RQ}3.2$, our analysis revealed that Precision, Recall, and F1 score are the predominant evaluation metrics, with occasional use of Accuracy, Hits@k, and exact SPARQL query matching. We found a significant language family imbalance, with Indo-European languages representing approximately 88% of studied systems, far overshadowing other families such as Dravidian, Japonic, Turkic, Sino-Tibetan, and Afro-Asiatic.

Regarding $\mathcal{RQ}3.3$, our systematic analysis of mKGQA benchmarks revealed a trend toward larger datasets over time. However, significant variations persist in language coverage and diversity. The lack of standardized benchmark formats creates additional challenges in data creation and access due to undefined requirements and procedures.

This systematic survey demonstrates the significant importance of the research gaps stated in Section 1. It is important to note that our review methodology prioritizes precision over recall in publication selection. Consequently, some relevant work (e.g., [227, 58]) was excluded due to selection criteria mismatches. Common exclusion reasons included absence of peer review (e.g., preprints, see Section 3.2.2) or insufficient citations for preprints (minimum threshold of five, see Section 3.2.2). Furthermore, some studies using multilingual benchmarks were excluded due to their exclusive focus on English [61].

4 From QALD-9 to QALD-9-plus: A Novel Benchmark for mKGQA

4.1 Introduction

One of the most essential features of Web-based systems is the ability to offer the same experience across different user groups [74]. This holds true for Knowledge Graph Question Answering (KGQA) systems, which facilitate access to knowledge graph data through a natural language interface. A persistent challenge in the KGQA domain is the absence of multilingual benchmarks. Most research efforts focus on the English language, despite the fact that there are around 7,000 spoken languages worldwide. Furthermore, numerous languages spoken by millions of people (e.g., Udmurt, Bashkir, Belarusian) remain unaddressed in current KGQA research. From a multilingual perspective, enhanced accessibility allows speakers of various languages, including those of low-resource and potentially vulnerable[1] to effectively use KGQA systems. This chapter focuses on *research gap 1* ($\mathcal{RG}1$) considering the limited availability of benchmarking datasets for mKGQA. In particular, we set the following *research questions*:

$\mathcal{RQ}$**4.1** What is a reproducible strategy for creating and expanding multilingual KGQA benchmarks and what are its key parameters?

$\mathcal{RQ}$**4.2** How can we create a process for SPARQL query migration from DBpedia to Wikidata.

A KGQA benchmarking dataset typically consists of a question written in a specific language, a ground truth SPARQL query and an optional field that holds ground truth answers. Table 4.1 illustrates a sample data model of a benchmark, which is typically used as the foundational structure of any KGQA benchmark framework.

The main *contribution* described in this chapter is that the *QALD-9* benchmark has been extended by adding high-quality translations in eight languages, made and validated by native speakers, and by adapting the SPARQL queries from QALD-9 from

[1] cf. `http://www.unesco.org/languages-atlas/en/atlasmap.html`

Tab. 4.1.: A conceptual data model of a KGQA benchmark. The header represents the data fields within one data point.

NL Question (q)	SPARQL query (ϕ)	ground truth Answers ($\mathcal{A}$)
Language-specific text of a question representing an information need	SPARQL query over a knowledge graph that returns ground truth answers	The result of the SPARQL query execution over a knowledge graph

DBpedia to Wikidata, thereby significantly improving the usability and relevance of the dataset. This enhanced benchmark is named *QALD-9-plus*. According to the related work, five of the languages: Armenian, Ukrainian, Lithuanian, Bashkir, and Belarusian had not previously been considered within the KGQA research community, the latter two are classified as "potentially vulnerable" by UNESCO[2]. The remaining languages in the benchmark: German, French, and Russian were already represented in the research community. Figure 4.1 presents the big picture of our approach.

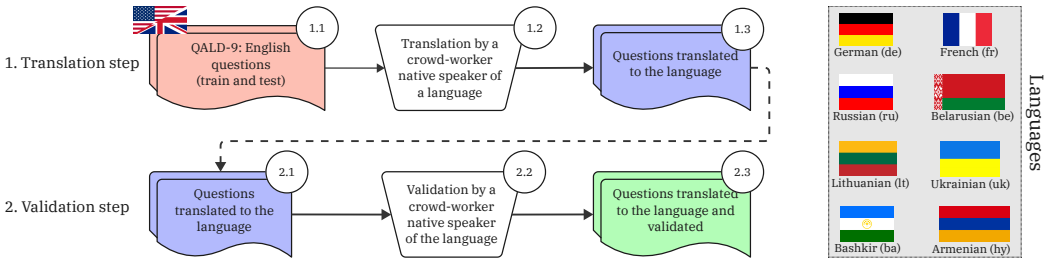


Fig. 4.1.: The two steps for the benchmark extension: (1) translation and (2) validation. The point 2.3 represents the final state of the obtained translations.

CRediT Statement about the author's contribution roles This chapter is mainly based on the following peer-reviewed publication:

1. **Perevalov, A.**, Diefenbach, D., Usbeck, R., and Both, A. (2022, January). QALD-9-plus: A multilingual dataset for question answering over DBpedia and Wikidata translated by native speakers. In 2022 IEEE 16th International Conference on Semantic Computing (ICSC) (pp. 229-234). IEEE.
2. Usbeck, R., Yan, X., **Perevalov, A.**, Jiang, L., Schulz, J., Kraft, A., Möller, C., Huang, J., Reineke, J., Ngonga Ngomo, A.C. and Saleem, M., 2023. QALD-10–The 10th challenge on question answering over linked data. Semantic Web, (pp.1-15).

The author's roles according to the CRediT taxonomy for the publication 1. are: Conceptualization, Data curation, Formal analysis, Methodology, Visualization, and

[2] `https://en.wal.unesco.org/discover/languages`

Writing—original draft. For the publication 2.: Data curation, Writing—review and editing.

4.2 Qualitative Analysis of the Original QALD-9 Benchmarking Dataset

While conducting a qualitative analysis of the original QALD-9 benchmark, several flaws in the data were identified. Firstly, there is notably poor translation quality for most questions in languages other than English. The analysts, who are experts in computer science and fluent in German and Russian, were able to spot translation errors in these languages. For instance, the English question "Which subsidiary of TUI Travel serves both Glasgow and Dublin?" is incorrectly translated into Italian as "Quale società sussidiaria di TUI Travel serve sia Dortmund che Dublino?". Though not Italian speakers, it is evident that two different cities are mentioned in the original (Glasgow) and translated (Dortmund) versions. Notably, the corresponding SPARQL query correctly references Glasgow, matching the English question.

Secondly, some questions lack clarity. For instance, the question "How often did Jane Fonda marry?" is likely intended to mean *how many times*. Another example is "Who is the heaviest player of the Chicago Bulls?", which is ambiguous regarding the time frame—whether it refers to historical players or current ones. Additionally, the ground truth SPARQL query for "Give me all films produced by Steven Spielberg with a budget of at least $80 million." incorrectly uses the property `dbo:director` instead of `dbo:producer`.

In this work, we concentrate on addressing the dataset's primary issue: the poor translation quality. Although the original DBpedia SPARQL queries were preserved, they were converted to Wikidata format. The following sections provide an overview of the translation process and the migration of SPARQL queries from the DBpedia knowledge graph to the Wikidata knowledge graph.

4.3 An Approach to Creating Translations of Questions via Crowdwsourcing

The *translation* process (see Figure 4.2a) was executed in a crowdsourcing manner with the following settings: (1) the crowd workers had to translate the questions

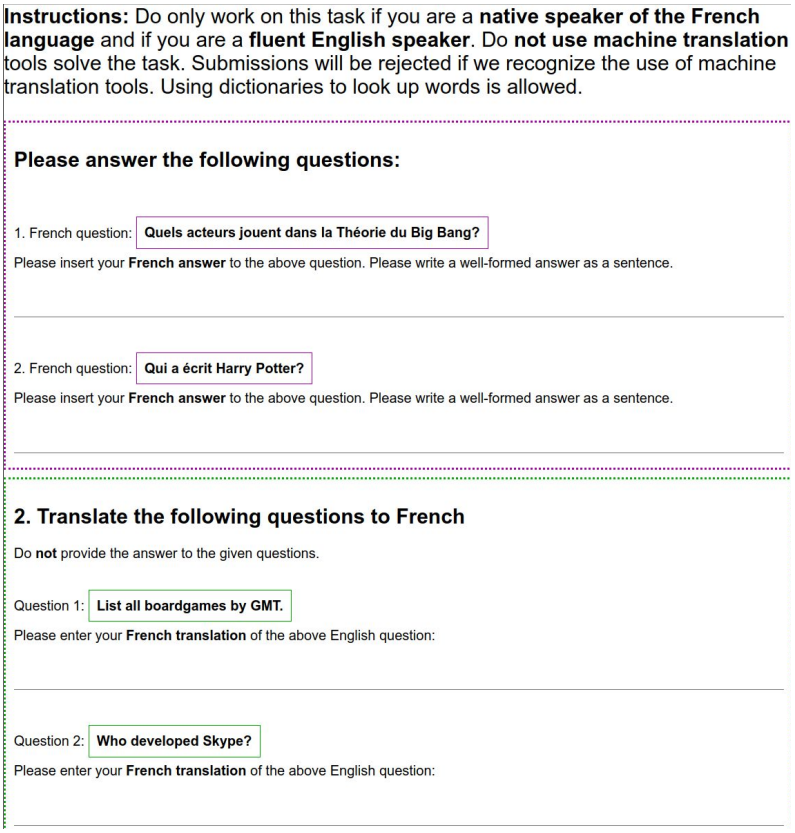
from English to their mother tongue, (2) each crowd worker was provided with a prepared subset of questions selected from different parts of the dataset to avoid possible biases (one crowd worker translated at least 10 questions), (3) the usage of machine translation tools was prohibited (only dictionaries were allowed), (4) the crowd workers were not aware of the existing multilingual representations from the QALD-9 dataset, and (5) the crowd workers were not aware of the other's tasks. In total, 17 volunteers and 290 crowd workers from Amazon Mechanical Turk and Yandex Toloka speaking 8 languages took part in the translation process. The coverage of translations was set to ≥ 2. The volunteers and crowd workers were following the same translation process.

After the translations were obtained, the *validation* process was executed (see Figure 4.2b). During the validation, the crowd workers were provided with the original question and two translation options. A crowd worker had to select one of the following options: (1) no translations are correct, (2) first translation is correct, (3) second translation is correct, or (4) both translations are correct.

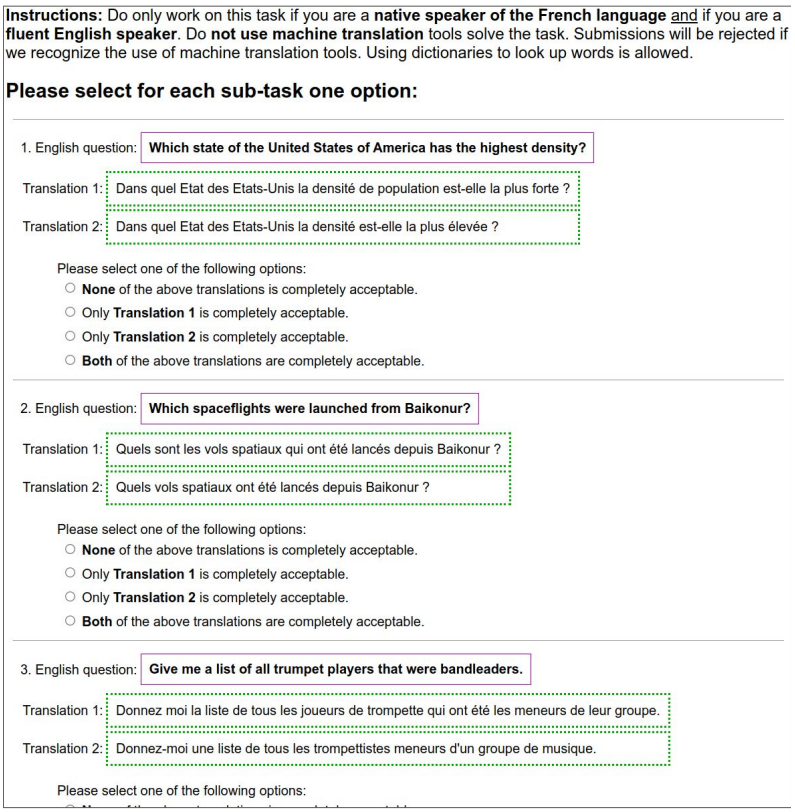
The tasks to the crowd workers were assigned according to the main country of language's origin (e.g., one has to be from France to translate from English to French). Thus, the questions were translated by native speakers of these different languages and additionally multiple times validated.

4.4 Migration from DBpedia to Wikidata Knowledge Graph

To enhance the utility of the dataset, we migrated the corresponding QALD-9 SPARQL queries from DBpedia to the Wikidata knowledge graph. This process was executed and verified internally by a team of three computer science experts. Although semi-automated scripts were partially utilized to accelerate the retrieval of entity and property mappings, the task was still exceedingly labor-intensive. Notably, we discovered that a simple property path (i.e., basic graph pattern) in DBpedia does not always translate to a straightforward path in the Wikidata knowledge graph, which added complexity to the transformation process. For instance, the DBpedia query for the question "When did Finland join the EU?" consists of a single triple in DBpedia, whereas three triples are needed to extract the same information from the Wikidata knowledge graph (see Figure 4.3).



(a) Translation crowd-sourcing task with two control questions



(b) Validation crowd-sourcing task (conducted after translation)

Fig. 4.2.: Crowd-sourcing task interfaces for translation and validation on the Amazon Mechanical Turk platform

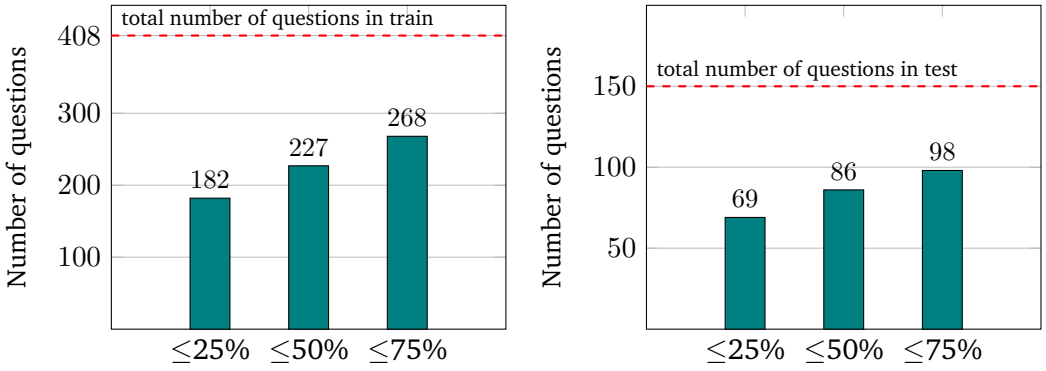
```
# DBpedia
SELECT DISTINCT ?date
WHERE {
  dbr:Finland
    dbp:accessioneudate ?date .
}
```

```
# Wikidata
SELECT DISTINCT ?date
WHERE {
  wd:Q33 p:P463 ?membership .
  ?membership pq:P580 ?date .
  ?membership ps:P463 wd:Q458 .
}
```

Fig. 4.3.: An example of the different basic graph patterns in DBpedia and Wikidata for question: "When did Finland join the EU?". Prefixes are omitted. Note, that the query over DBpedia does not return any answer anymore as of 01 October 2024.

Several DBpedia SPARQL queries could not be adapted to Wikidata due to missing corresponding information (e.g., "How many calories does a baguette have?"). In total, 51 queries could not be transferred to Wikidata because the information available in the DBpedia knowledge graph was not present in the Wikidata knowledge graph. The ground truth answers for these questions over Wikidata were obtained by running the equivalent SPARQL queries. Similarly, the updated ground truth answers over DBpedia were generated using the same approach.

We analyzed the answer sets of the SPARQL queries from the original (QALD-9) dataset and the updated (QALD-9-plus) dataset (see Figure 4.4). According to the figure, more than half of the ground truth answer sets for both training and testing splits showed less than 50% similarity when comparing QALD-9 to QALD-9-plus. This variation is attributed to changes in DBpedia that can occur for various reasons. Factors such as historical changes like a new president or governor (temporal facts), alterations in the data model over time, or updated or added missing information can



(a) Training set of the QALD-9-plus **(b)** Test set of the QALD-9-plus

Fig. 4.4.: The number of ground truth answer sets computed over DBpedia that have less or equal intersection rate while comparing QALD-9 and QALD-9-plus, i.e., how many ground truth answer sets have the intersection rate less than X%? The intersection rate is computed over the results, produced by a SPARQL query. Each question has its own ground truth answer set produced by a SPARQL query.

all contribute. This underscores the significance of the answer sets in KGQA datasets, as they allow comparison of different versions of a knowledge base. Therefore, current information should always be retrieved using the corresponding SPARQL queries.

4.5 QALD-9-plus Dataset Statistics Overview

The statistics of the dataset are summarized in Table 4.2. We have expanded the dataset by adding 4,930 new question translations in various languages i.e., one question may have more than one translation. In doing so, we replaced most of the multilingual representations of original QALD-9 with these new translations. The resulting list of the languages and grouped by the corresponding language branch is as follows:

Western European: English, German, Spanish, French,
Baltic: Lithuanian,
Slavic: Russian, Ukrainian, Belarusian,
Other: Armenian, Bashkir.

It is important to note that for some languages, such as Russian, we were able to gather multiple translations for certain questions (averaging 2.9 Russian translations per question in the training set). This enables the benchmark users to evaluate their systems on different surface forms of the questions. In contrast, for other languages like Armenian, Bashkir, and French, we only managed to provide partial coverage of the original QALD-9 questions. All translations were performed by native speakers.

In terms of SPARQL queries, we enhanced the dataset with 507 new queries over Wikidata. These queries were manually crafted by the authors of this paper.

Tab. 4.2.: Number of questions (including paraphrased ones) for every language within QALD-9-plus.

	en	de	fr	ru	uk	lt	be	ba	hy	DBpedia	Wikidata	# questions
Train	408	543	260	1203	447	468	441	284	80	408	371	408
Test	150	176	26	348	176	186	155	117	20	150	136	150

Tab. 4.3.: Comparison of quantitative text features between QALD-9 and QALD-9-plus (both train and test subsets were considered)

	English (original)		German		Russian		French	
	QALD-9	QALD-9-plus	QALD-9	QALD-9-plus	QALD-9	QALD-9-plus	QALD-9	QALD-9-plus
Avg. syllables per word	1.32		1.86	1.87	2.13	2.31	1.51	1.55
Avg. word length	4.77		5.64	5.68	5.66	6.12	5.07	5.18
Avg. sentence length	33.92		35.65	39.14	19.80	36.66	36.35	39.44
Avg. words per sentence	7.06		6.32	6.89	3.49	5.99	7.18	7.62
Type Token Ratio	0.30		0.33	0.33	0.32	0.49	0.32	0.49

4.6 Quantitative and Qualitative Question Analysis

We used various language-agnostic quantitative text analysis techniques to examine the changes between QALD-9 and QALD-9-plus textual representations concerning descriptive statistics. Consequently, we focused on analyzing only the languages that appeared in both QALD-9 and QALD-9-plus. To facilitate this, the LinguaF library[3] for Python was utilized. The comparison results are presented in Table 4.3.

The data clearly indicate that the QALD-9-plus translations employed longer words across all languages. Additionally, the translations in QALD-9-plus contained more words, leading to an increased average question length. The Type-Token Ratio (TTR) [103], which measures the ratio of unique words to the total number of words, also increased for Russian and French, while remaining unchanged for German. This analysis suggests that the QALD-9-plus multilingual representations are more complex and richer compared to the original QALD-9 translations. Since native speakers performed the translations, we regard these findings as an implicit indication of enhanced translation quality.

4.7 Discussion

In this section we discuss impact and usability of the new QALD-9-plus benchmarking dataset. We also answer to our research questions and revisit the research gap that we covered in this chapter.

Regarding the usability of data, the decision was made to retain the QALD-JSON format[4] for QALD-9-plus, as this allows researchers to reuse their systems for evaluation purposes (e.g., GERBIL [207]). This enables researchers and developers to

[3] https://github.com/Perevalov/LinguaF

[4] https://github.com/dice-group/gerbil/wiki/Question-Answering#web-service-interface

benchmark and compare their KGQA systems in both mono- and multilingual contexts using DBpedia and Wikidata. With multiple alternative translations available, there is potential to develop techniques for paraphrasing and assessing machine translation quality. This also applies to the SPARQL queries across DBpedia and Wikidata.

While answering the *RQ4.1*, a reproducible strategy for multilingual KGQA benchmarking involves a systematically organized translation, query migration, and evaluation. A benchmark structure consists of natural language questions, corresponding SPARQL queries, and optional ground truth answers. Translation follows a controlled crowdsourcing approach: native speakers translate without prior exposure to existing representations, using only dictionaries. Each question is translated at least twice and validated by crowd workers. This translation strategy was also employed while working on creation of the QALD-10 benchmark [208] where the author of this thesis was one of the contributors.

While answering the *RQ4.2*, the SPARQL queries are adapted from DBpedia to Wikidata using semi-automated mapping and manual validation. Structural differences often require additional transformations, and some queries remain untranslatable due to missing properties. Moreover, the automated approach would require significant labor costs and may produce incorrect queries.

4.8 Conclusion

In this chapter, we introduced a new multilingual benchmark for KGQA, known as QALD-9-plus. This benchmark builds upon QALD-9 and includes translations from English to German, French, Russian, Armenian, Belarusian, Lithuanian, Bashkir, and Ukrainian. These translations were carried out by native speakers of each language in a 2-step crowdsourcing setup. Additionally, the DBpedia SPARQL queries from QALD-9 were adapted for Wikidata to enhance data usability. QALD-9-plus offers multiple text representations across several languages and features multilingual questions, forming a parallel corpus. This allows researchers to explore paraphrasing and machine translation tasks. Moreover, paraphrasing can be conducted at the SPARQL query level, converting queries from DBpedia to Wikidata. QALD-9-plus maintains the QALD-JSON format to ensure reusability within the research community. The quantitative and qualitative analyses comparing the multilingual question representations in QALD-9 and QALD-9-plus demonstrated improved translation quality.

5 Machine Translation as an Alternative for mKGQA

5.1 Introduction

Knowledge graphs (KGs) serve for modeling and integrating multilingual data by associating unique entities with labels and descriptions across different languages. Despite KGs' capability to store multilingual information, only a limited number of KGQA systems can process queries in multiple languages. This constraint arises from two primary challenges: (1) multilingual processing presents considerable obstacles for QA systems due to the linguistic differences between languages with distinct typological characteristics [42], and (2) there is an insufficient availability of datasets for training and evaluating such systems.

Machine translation (MT) has been previously explored as an approach to enable multilingual functionality in QA systems [3]. In this chapter, we conduct an analysis to evaluate the effectiveness of MT in extending KGQA systems to support additional languages. Therefore, we cover here the *research gap 2* ($\mathcal{RG}2$) pertaining to the application of MT approaches for extended language support of KGQA systems. Specifically, we investigate the following *research questions:*

$\mathcal{RQ}$**5.1** How does translation affect the performance of KGQA systems?
$\mathcal{RQ}$**5.2** What correlation exists between MT and QA quality?
$\mathcal{RQ}$**5.3** To what extent can MT enable KGQA systems to accommodate new languages through question translation?

For our evaluation, we focused on three multilingual KGQA systems: QAnswer [50], DeepPavlov [60], and Platypus [147]. These systems were selected based on their capability to answer questions over the Wikidata[1] KG [211] and their ability to answer questions in multiple languages (i.e., converting a natural language to a SPARQL query). To assess their performance, we utilized the QALD-9-plus dataset [155], a comprehensive benchmark that provides multilingual natural language representations alongside corresponding SPARQL queries for both DBpedia and Wikidata KGs. We described QALD-9-plus in Chapter 4.

[1] https://www.wikidata.org/

Through extensive experimentation and analysis, this paper makes the following contributions toward improving KGQA system accessibility for non-English users:

- We present a comprehensive comparative study of three multilingual KGQA systems integrated with two distinct MT tools (one proprietary and one open-source);
- Our systematic evaluation of MT-enhanced KGQA systems demonstrates negligible performance degradation in source language QA quality, while revealing substantial variations in target language performance (with English maintaining superior results), meaning that it is always better to translate questions to English;
- We empirically demonstrate that monolingual KGQA systems can be effectively extended to accommodate additional languages through machine translation-based approaches.

CRediT Statement about the author's contribution roles This chapter is mainly based on the following peer-reviewed publication:

1. **Perevalov, A.**, Both, A., Diefenbach, D., and Ngonga Ngomo, A.-C. (2022, April). Can Machine Translation be a Reasonable Alternative for Multilingual Question Answering Systems over Knowledge Graphs?. In Proceedings of the ACM Web Conference 2022 (pp. 977-986). 2023.

The author's roles according to the CRediT taxonomy for the publication 1. are: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Visualization, and Writing—original draft. For the publication 2.: Conceptualization, Data curation, Formal analysis, Investigation, Visualization, and Writing—original draft.

5.2 Machine Translation Approach for KGQA Systems

Our initial approach follows a component-based workflow, utilizing existing QA components (cf. [24, 22, 53]) to address our research questions. To perform a thorough evaluation of KGQA systems extended to new languages through MT tools, we identified the following essential components:

- A diverse set of multilingual KGQA systems with well-documented language support capabilities,

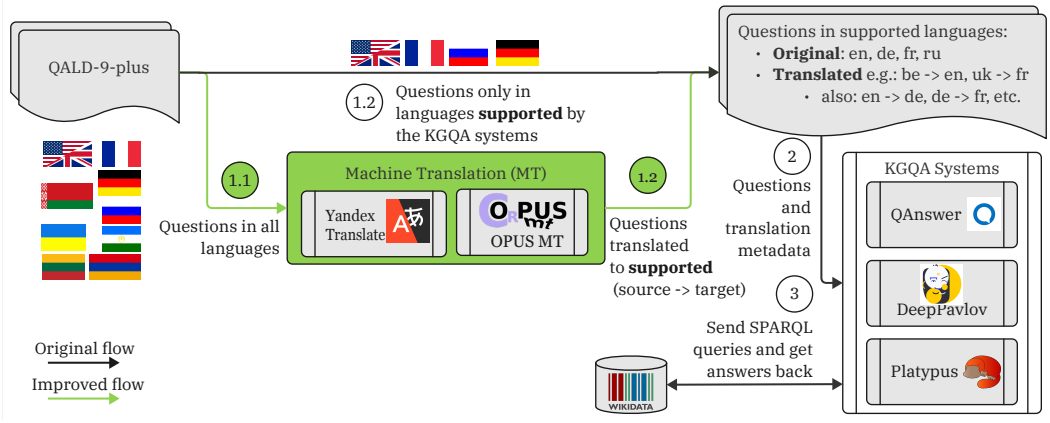


Fig. 5.1.: Overview of the proposed approach. Important is the machine translation step that translates questions to supported languages and returns the metadata such that we can track the source language and the translation tool.

- A comprehensive collection of high-quality questions created by native speakers across multiple languages, divided into two distinct categories:
  - Questions formulated in languages currently supported by the KGQA systems
  - Questions composed in languages not yet supported by the KGQA systems
- Robust MT tools capable of accurately translating questions from unsupported languages into supported ones.

Figure 5.1 illustrates the approach we employ in this work. We begin by establishing baseline performance through evaluating the KGQA systems using questions in their natively supported languages. We then systematically translate questions from each source language into all languages supported by the KGQA systems. For example, Lithuanian questions are translated into English, German, Russian, and French, while English questions are translated in German, Russian, and French. To investigate the correlation between translation quality and QA quality, we utilize two distinct MT tools: an open-source solution and a proprietary system, enabling a comparative analysis of their effectiveness.

Our evaluation methodology utilizes GERBIL [207], a widely-adopted benchmarking platform that ensures reproducible, comparable, and transparent assessment of KGQA systems through standardized evaluation protocols.

5.3 Experimental Setup

5.3.1 Question Answering Systems

For our comprehensive evaluation, we selected three prominent KGQA systems: QAnswer, DeepPavlov, and Platypus. The selection was based on three critical criteria: multilingual support capabilities, compatibility with the Wikidata KG, and demonstrated state-of-the-art performance as validated in recent research [169, 49]. To the best of our knowledge, these systems are unique in satisfying all these essential requirements among published KGQA solutions. In alignment with our component-oriented methodology (detailed in Section 5.2), we treat these systems as black-box components (i.e., we maintain their original configurations without examining or modifying their internal architectures). This system-agnostic approach ensures the broader applicability and generalizability of our methodology across different KGQA implementations.

5.3.2 Machine Translation tools

Our machine translation requirements were fulfilled by two distinct tools: OPUS MT [197] and Yandex Translate [72].

OPUS MT represents a state-of-the-art open-source neural machine translation framework that provides extensive pre-trained models for diverse language pairs. These models are readily accessible through the Huggingface library's Python interface[2]. *Yandex Translate*, a proprietary translation service developed by Yandex, offers comprehensive translation capabilities through a robust RESTful API[3]. We selected Yandex as it was the only service supported all the languages we used[4]. The inclusion of both open-source and proprietary solutions enables a balanced comparison of translation performance across different technological approaches.

5.3.3 Evaluation Process

For our systematic evaluation and comparative analysis, we utilized GERBIL [207], a comprehensive benchmarking framework. GERBIL implements standardized

[2] `https://huggingface.co/models`
[3] `https://yandex.com/dev/translate`
[4] At the time of the experiments (late 2022).

interfaces for KGQA systems and employs a uniform dataset format (QALD-JSON format[5]) for consistent benchmarking. The framework assesses QA quality through Precision, Recall, and F1 score metrics, calculated using both micro and macro averaging strategies. Following our research community practices [198], we adopt the macro averaging strategy for our comparative analysis.

To evaluate translation quality, we leverage the availability of both original questions and their machine-translated counterparts, employing widely-recognized metrics: BLEU [146] and NIST [55]. For transparency and reproducibility, we have made our evaluation code publicly available[6].

5.4 Evaluation and Analysis

For our evaluation, we employed a comprehensive multilingual dataset QALD-9-plus, which test set consists of 136 questions across nine diverse languages: English (en), German (de), Russian (ru), French (fr), Ukrainian (uk), Lithuanian (lt), Belarusian (be), Bashkir (ba), and Armenian (hy). It is important to note that question coverage varies among languages when compared to the English baseline (cf. Table 4.2). To clearly represent language translation processes throughout our analysis, we employ the notation X $\rightarrow$ Y, where X denotes the source language and Y indicates the target language. The subsequent subsections present a detailed analysis of our evaluation findings and their implications.

5.4.1 Evaluation Results for Original Questions

We initiated our evaluation by testing the three KGQA systems using questions in their natively supported languages (complete results presented in Table 5.1). These

Tab. 5.1.: Evaluation Results: Native Language Performance Without Machine Translation

	QAnswer			DeepPavlov			Platypus		
	Precision	**Recall**	**F1 score**	**Precision**	**Recall**	**F1 score**	**Precision**	**Recall**	**F1 score**
en	0.5046	0.4484	0.4459	0.1765	0.1130	0.1240	0.1589	0.1526	0.1503
de	0.3305	0.3244	0.3171	not supported by the system			not supported by the system		
ru	0.2229	0.2223	0.2143	0.0956	0.0833	0.087			
fr	0.2532	0.2535	0.2300	not supported by the system			0.0417	0.0417	0.0417

[5] https://github.com/dice-group/gerbil/wiki/Question-Answering#web-service-interface
[6] https://github.com/Perevalov/phd_thesis

initial measurements establish crucial baseline performance metrics against which we compare the effectiveness of machine-translated questions. The analysis reveals a consistent pattern across all QA quality metrics, with English-language queries demonstrating superior performance. Note: While the performance differences between systems are evident, we deliberately exclude system-to-system comparisons from our analysis due to space limitations and our focused research objectives.

5.4.2 Machine Translation Quality

Before analyzing the QA quality on machine-translated questions, we conducted a thorough evaluation of the selected MT tools' translation capabilities. Table 5.2 presents comprehensive evaluation metrics for each language pair. It is important to note that OPUS MT lacks support for certain language combinations (denoted as "no model provided" in the table). Additionally, the evaluation of MT quality for some language pairs was not possible due to incomplete question coverage in specific languages (e.g., lt, be, hy, fr), resulting in the absence of parallel corpora for combinations such as lt $\rightarrow$ fr and be $\rightarrow$ fr (marked as "n/a" in the table).

Our analysis demonstrates that the Yandex MT tool generally outperforms OPUS MT across most language pairs (e.g., en $\rightarrow$ ru, de $\rightarrow$ fr, etc.). However, this superiority is not universal. For example, OPUS MT exhibits better performance in translating from Lithuanian to Russian (lt $\rightarrow$ ru). This performance variability prompts an intriguing investigation into the potential correlation between translation quality and subsequent QA quality. Specifically, we hypothesize that questions translated using the superior MT tool (predominantly Yandex Translate) should yield enhanced KGQA evaluation results compared to those processed by OPUS MT. We explore this hypothesis and its implications in the following subsection.

5.4.3 Evaluation Results for Machine Translated Questions

Following our methodology outlined in Section 5.2, we conducted comprehensive translations of benchmark questions into languages supported by the KGQA systems using both MT tools. The subsequent evaluation of KGQA systems using these translated questions, compared against native speaker baselines, is presented in Tables 5.3 and 5.4. The table employs several visual indicators for clarity: bold text (excluding headers) represents QA quality metrics for native speakers' questions (row "native"), while green highlighting identifies superior results within each source language group for specific metrics and systems. Target languages achieving optimal

Tab. 5.2.: Evaluation Results for the Machine Translation Tools. “n/a” represents no translation support for the particular language pair. “no model provided” represents no OPUS MT model for such language pair.

(a) Resource-rich languages

Source	Target	MT Tool	BLEU	NIST
en	de	Yandex	0.8142	4.7828
		OPUS MT	0.8140	4.7863
	ru	Yandex	0.8518	4.9791
		OPUS MT	0.7787	4.6061
	fr	Yandex	0.8137	4.7730
		OPUS MT	0.8147	4.7869
de	en	Yandex	0.8108	4.6237
		OPUS MT	0.8015	4.6640
	ru	Yandex	0.8088	4.7559
		OPUS MT	no model provided	
	fr	Yandex	0.7446	4.4700
		OPUS MT	0.7147	4.2586
ru	en	Yandex	0.7387	4.3537
		OPUS MT	0.6786	4.1188
	de	Yandex	0.7013	4.3179
		OPUS MT	no model provided	
	fr	Yandex	0.7465	4.6180
		OPUS MT	0.6494	4.1641
fr	en	Yandex	0.8061	4.7432
		OPUS MT	0.7807	4.6734
	de	Yandex	0.8743	5.1113
		OPUS MT	0.8707	5.1020
	ru	Yandex	0.8759	5.0825
		OPUS MT	0.7432	4.4508

(b) Low-resource languages

Source	Target	MT Tool	BLEU	NIST
lt	en	Yandex	0.7603	4.3235
		OPUS MT	no model provided	
	de	Yandex	0.7568	4.5226
		OPUS MT	0.7434	4.5001
	ru	Yandex	0.4415	3.2701
		OPUS MT	0.5407	3.5498
	fr	Yandex	n/a	
		OPUS MT		
uk	en	Yandex	0.7670	4.5134
		OPUS MT	0.6648	4.0733
	de	Yandex	0.7159	4.4119
		OPUS MT	0.6453	4.1386
	ru	Yandex	0.8521	4.9549
		OPUS MT	0.8168	4.7874
	fr	Yandex	0.7538	4.5770
		OPUS MT	0.6220	3.9215
be	en	Yandex	0.7585	4.5403
		OPUS MT	no model provided	
	de	Yandex	0.7057	4.3809
		OPUS MT	no model provided	
	ru	Yandex	0.8667	5.3427
		OPUS MT	no model provided	
	fr	Yandex	n/a	
		OPUS MT		
ba	en	Yandex	0.5501	3.6762
		OPUS MT	no model provided	
	de	Yandex	0.5088	3.4857
		OPUS MT	no model provided	
	ru	Yandex	0.6274	3.9938
		OPUS MT	no model provided	
	fr	Yandex	0.4894	3.4446
		OPUS MT	no model provided	
hy	en	Yandex	0.7619	4.3769
		OPUS MT	no model provided	
	de	Yandex	0.7894	4.7016
		OPUS MT	no model provided	
	ru	Yandex	0.7363	4.4223
		OPUS MT	no model provided	
	fr	Yandex	0.7493	4.4963
		OPUS MT	no model provided	

Tab. 5.3.: Evaluation results for the machine-translated questions (resource-rich languages). The green color-coded cells correspond to the highest metric value for a given KGQA system and a source language. The "⋆" corresponds to the highest performing target language given the source language.

Source	Target	MT Tool	QAnswer			DeepPavlov			Platypus		
			Precision	Recall	F1 score	Precision	Recall	F1 score	Precision	Recall	F1 score
en	Native⋆		0.5046	0.4484	0.4459	0.1765	0.1130	0.1240	0.1589	0.1526	0.1503
	de	Yandex	0.3491	0.3518	0.3283	not supported by the system			not supported by the system		
		OPUS MT	0.3490	0.3500	0.3374						
	ru	Yandex	0.2284	0.2161	0.2107	0.0735	0.0699	0.0711	not supported by the system		
		OPUS MT	0.2238	0.2063	0.2041	0.0882	0.0735	0.0784			
	fr	Yandex	0.3216	0.2949	0.2886	not supported by the system			0.0894	0.1010	0.0892
		OPUS MT	0.3076	0.2819	0.2757				0.0821	0.0936	0.0819
de	Native		0.3305	0.3244	0.3171	not supported by the system					
	en⋆	Yandex	0.4136	0.3871	0.3649	0.1985	0.1259	0.1392	0.1227	0.1160	0.1144
		OPUS MT	0.4095	0.3811	0.3657	0.1618	0.0983	0.1093	0.1521	0.1434	0.1426
	ru	Yandex	0.2233	0.2014	0.2033	0.0882	0.0737	0.0763	not supported by the system		
		OPUS MT	no model provided								
	fr	Yandex	0.3105	0.2823	0.2771	not supported by the system			0.1227	0.1160	0.1144
		OPUS MT	0.2832	0.2621	0.2521				0.0591	0.0662	0.0593
ru	Native		0.2229	0.2223	0.2143	0.0956	0.0833	0.0870	not supported by the system		
	en⋆	Yandex	0.4409	0.3979	0.3898	0.1691	0.1054	0.1161	0.1613	0.1527	0.1511
		OPUS MT	0.3912	0.3602	0.3463	0.1324	0.0793	0.0879	0.1324	0.0793	0.0879
	de	Yandex	0.2920	0.2988	0.2717	not supported by the system			not supported by the system		
		OPUS MT	no model provided								
	fr	Yandex	0.2982	0.2734	0.2662	not supported by the system			0.0784	0.0919	0.0781
		OPUS MT	0.2160	0.2044	0.1972				0.0664	0.0735	0.0667
fr	Native		0.2532	0.2535	0.2300	not supported by the system			0.0417	0.0417	0.0417
	en⋆	Yandex	0.4211	0.3504	0.3657	0.3158	0.2202	0.2273	0.2105	0.2105	0.2105
		OPUS MT	0.4211	0.3504	0.3657	0.2105	0.1667	0.1729	0.2105	0.2105	0.2105
	de	Yandex	0.3301	0.3582	0.3101	not supported by the system			not supported by the system		
		OPUS MT	0.2774	0.3056	0.2574						
	ru	Yandex	0.1158	0.1574	0.1228	0.1053	0.1053	0.1053	not supported by the system		
		OPUS MT	0	0	0	0.1053	0.1053	0.1053			

quality metrics are designated with a star (⋆). In cases where translation universally diminishes quality metrics across all target languages, we denote this with "Native⋆". For completeness, we indicate unsupported language combinations as "not supported by the system" and missing OPUS MT translation models as "no model provided".

Our analysis reveals a clear pattern of English superiority as a target language for translation (see Figure 5.2). The majority of experimental results demonstrate enhanced QA quality metrics when translating from various source languages into English (e.g., de → en, uk → en). When English serves as the source language, questions from native speakers consistently achieve the highest QA quality, with MT generally resulting in degraded results. An interesting exception emerges in the case of Lithuanian (see Table 5.4), where German proves to be the most effective target language (lt → de), although English translations (lt → en) also demonstrate strong performance. Similarly, while Armenian to German translation (hy → de) shows better results than Armenian to English (hy → en) specifically for the QAnswer

Tab. 5.4.: Evaluation results for the machine-translated questions (low-resource languages). The green color-coded cells correspond to the highest metric value for a given KGQA system and a source language. The "⋆" corresponds to the highest performing target language given the source language.

Source	Target	MT Tool	QAnswer			DeepPavlov			Platypus		
			Precision	Recall	F1 score	Precision	Recall	F1 score	Precision	Recall	F1 score
lt	en	Yandex	0.4196	0.3919	0.3930	0.1739	0.1094	0.1173	0.1739	0.1522	0.1594
		OPUS MT	no model provided								
	de⋆	Yandex	0.3894	0.3660	0.3734	not supported by the system			not supported by the system		
		OPUS MT	0.4573	0.3935	0.3949						
	ru	Yandex	0.1732	0.1739	0.1735	0.0435	0.0435	0.0435	not supported by the system		
		OPUS MT	0.1413	0.1522	0.1449	0.0435	0.0435	0.0435			
	fr	Yandex	0.3326	0.2832	0.2916	not supported by the system			0.1359	0.1522	0.1256
		OPUS MT	0.3036	0.2628	0.2642				0.1304	0.1304	0.1304
uk	en⋆	Yandex	0.4315	0.4135	0.3917	0.1691	0.1018	0.1137	0.1497	0.1435	0.1417
		OPUS MT	0.3267	0.3301	0.3038	0.1691	0.1024	0.1118	0.1422	0.1344	0.1348
	de	Yandex	0.3186	0.3032	0.2882	not supported by the system			not supported by the system		
		OPUS MT	0.2948	0.2732	0.2713						
	ru	Yandex	0.2265	0.2253	0.2202	0.0735	0.0699	0.0711	not supported by the system		
		OPUS MT	0.2464	0.2397	0.2319	0.0662	0.0625	0.0637			
	fr	Yandex	0.3085	0.2807	0.2727	not supported by the system			0.0968	0.1066	0.0953
		OPUS MT	0.2268	0.2352	0.2076				0.0775	0.0846	0.0765
be	en⋆	Yandex	0.5455	0.4554	0.4562	0.0909	0.0455	0.0606	0.0909	0.0909	0.0909
		OPUS MT	no model provided								
	de	Yandex	0.3654	0.2747	0.2746	not supported by the system			not supported by the system		
		OPUS MT	no model provided								
	ru	Yandex	0.1818	0.1818	0.1818	0.0909	0.0909	0.0909	not supported by the system		
		OPUS MT	no model provided								
	fr	Yandex	0.4091	0.2281	0.2441	not supported by the system			0.0909	0.0909	0.0909
		OPUS MT	no model provided						no model provided		
hy	en⋆	Yandex	0.2632	0.2188	0.2253	0.2105	0.1149	0.1221	0.1579	0.1579	0.1579
		OPUS MT	no model provided								
	de	Yandex	0.2774	0.3495	0.2950	not supported by the system			not supported by the system		
		OPUS MT	no model provided								
	ru	Yandex	0.0526	0.0526	0.0526	0.1053	0.1053	0.1053	not supported by the system		
		OPUS MT	no model provided								
	fr	Yandex	0.1684	0.1662	0.1378	not supported by the system			0.0526	0.0526	0.0526
		OPUS MT	no model provided						no model provided		
ba	en⋆	Yandex	0.3142	0.2845	0.2844	0.1562	0.0907	0.0986	0.0877	0.0836	0.0795
		OPUS MT	no model provided								
	de	Yandex	0.2366	0.2319	0.2287	not supported by the system			not supported by the system		
		OPUS MT	no model provided								
	ru	Yandex	0.2073	0.2016	0.1917	0.0938	0.0755	0.0806	not supported by the system		
		OPUS MT	no model provided								
	fr	Yandex	0.1863	0.1723	0.1608	not supported by the system			0.0524	0.0625	0.0528
		OPUS MT	no model provided						no model provided		

system, English maintains its overall advantage when considering performance across all three KGQA systems.

A significant finding of our analysis is that machine translation can substantially improve QA quality for several source languages, particularly German, Russian, and French when translated to English (i.e., de → en, ru → en, fr → en) compared

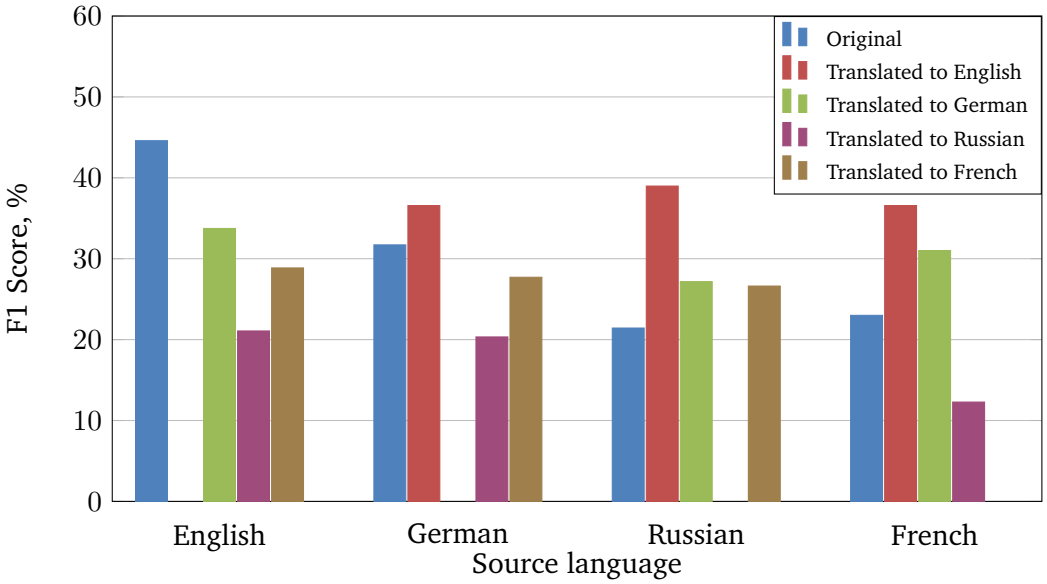


Fig. 5.2.: KGQA quality on major languages considering the original and translated versions of the questions. The superiority of KGQA quality is achieved when translating questions to English from the source language (x-axis). We take the best F1 score among the two MT tools.

to their native performance. The improvements can be dramatic, as evidenced by French-to-English translations, which yielded remarkable F1 score improvements: from 0.2300 to 0.3657 (+59.0% relative improvement) for QAnswer and from 0.0417 to 0.2105 (+504.8% relative improvement) for Platypus (consistent across both Yandex and OPUS MT). However, translations targeting non-English languages frequently result in considerable performance deterioration. This degradation is illustrated by German-to-French translations, where the F1 score drops from 0.3171 (native) to 0.2521 (OPUS MT) on QAnswer (-20.49%). The pattern persists in English-to-Russian translations, showing a substantial decline in F1 score from 0.1240 (native) to 0.0711 (Yandex) on DeepPavlov (-42.66%).

A notable outlier in our findings is the Belarusian to English translation using Yandex, which achieved an exceptional F1 score of 0.4562 on QAnswer. However, this remarkable performance was not consistent across other systems, suggesting it may be an anomalous result. Another outlier appears in the French to Russian translation using OPUS MT on QAnswer, which produced an unusually poor performance with an F1 score of 0.0.

Building on our earlier discussion in Section 5.4.2 regarding Yandex Translate's superior performance over OPUS MT in terms of MT quality, we conducted a deeper analysis of the relationship between translation and question answering performance.

To quantify this relationship, we computed Pearson's correlation coefficients between translation quality metrics (BLEU, NIST) and question answering performance metrics (Precision, Recall, F1 score). The detailed correlation analysis is presented in Table 5.5.

Tab. 5.5.: Analysis of Correlation between MT and QA Quality Metrics

	QAnswer			DeepPavlov			Platypus		
	Precision	**Recall**	**F1 score**	**Precision**	**Recall**	**F1 score**	**Precision**	**Recall**	**F1 score**
BLEU	0.1442	0.1889	0.1604	0.2403	0.3420	0.3383	0.3241	0.4137	0.3826
NIST	0.0800	0.1262	0.0969	0.1353	0.2889	0.2722	0.1821	0.3177	0.2638

The correlation analysis demonstrates that the BLEU metric shows stronger associations with QA quality metrics compared to NIST. Our findings indicate small to moderate positive correlations between translation quality and question answering quality metrics, as detailed in Table 5.5. An interesting pattern emerges where Platypus and DeepPavlov systems exhibit stronger correlation coefficients than QAnswer, though this may be attributed to their more limited experimental scope due to restricted language support.

5.5 Discussion

Drawing from our experimental findings, we now address the research gap and the research questions outlined at the chapter's beginning.

In response to $\mathcal{RQ}5.1$, our results provide strong evidence that machine translation generally improves QA quality when converting questions from various source languages to English, with English consistently delivering superior QA quality across all evaluated systems. While the original language appears to have minimal impact on translation effectiveness, we found that the choice of target language significantly influences quality outcomes. The extent of improvement or deterioration in QA quality is primarily determined by a system's proficiency in processing questions in the selected target language.

Addressing $\mathcal{RQ}5.2$, our experimental analysis reveals small to moderate positive correlations between machine translation and question answering quality metrics. Among the evaluation metrics examined, BLEU shows notably stronger correlation coefficients with Precision, Recall, and F1 score. However, we suggest that traditional MT quality metrics alone may not fully capture the factors influencing QA quality. Future investigations should focus on specific translation aspects critical

for question answering, particularly the accuracy of named entity translation. Our observations highlighted numerous cases where incorrect named entity translations led to processing failures in QA systems.

For $\mathcal{RQ}5.3$, we confirm that machine translation serves as an effective tool for extending KGQA systems to previously unsupported languages, thereby substantially improving web information accessibility for non-English speakers. Currently, the most successful approach involves translating source languages to English, which maintains its position as the dominant language for optimal QA quality. However, further research is necessary to determine the most effective combinations of system components for different language pairs and use cases.

Our study acknowledges the following limitations:

- The evaluation framework is restricted to a single knowledge graph (Wikidata), potentially limiting the generalizability of our findings across different knowledge bases
- The analysis does not differentiate between various question types (e.g., complex questions, keyword-based questions), which may respond differently to translation
- The current test set size, while informative, could be expanded to provide greater statistical confidence in the results

5.6 Conclusion

Our research is motivated by the fundamental goal of making the Web information accessible, particularly for users who may lack English proficiency. This study provides a comprehensive evaluation of current capabilities in multilingual knowledge graph access through KGQA systems enhanced with MT tools.

Although the development of dedicated KGQA systems for each language remains resource intensive, our experimental results demonstrate that the integration of MT components offers a viable and efficient solution to bridge linguistic gaps, particularly for low-resource languages. This is exemplified by the remarkable translation performance from Belarusian to English, which achieved a F1 score of 0.4562, exceeding even the native English QA quality. These results suggest promising opportunities to identify optimal combinations of MT components and QA systems to enhance the accessibility of knowledge for non-English speakers.

Importantly, our approach maintains independence from specific MT tools and KGQA systems, which makes it easily adaptable to various tools, systems, and languages. We conclude that MT presents an effective strategy for enabling multilingual KGQA systems, particularly for certain language combinations, although performance varies significantly between different languages and translation directions.

On the very abstract level, our research has studied how the manipulation of the input of a KGQA system, in particular, machine translation, may impact the resulting quality. Further research may focus on the manipulation of a KGQA system's output (e.g., SPARQL query), in particular, its refinement or validation (see Chapter 6).

6 Improving mKGQA Systems' Quality by Verbalizing and Filtering Incorrect Output

6.1 Introduction

As we mentioned in Chapter 3, current research in KGQA primarily follows two developmental paradigms [228, 223, 125]: (1) The *information extraction paradigm*, which directly retrieves answers from a feature space derived from the KG. (2) The *semantic parsing paradigm*, which transforms NL questions into formal queries or ranked sets of *query candidates* that can be executed against the KG to obtain answers. When examining the semantic parsing paradigm more closely, a significant challenge emerges: incorrect *query candidates* may sometimes receive higher rankings than correct ones. This incorrect ranking can substantially degrade both the quality and trustworthiness of KGQA systems. In this chapter, we address the challenge of query misranking by introducing a SPARQL query filtering method that leverages Language Models (LM) to distinguish between correct and incorrect SPARQL queries (cf. Figure 6.1). Unlike previous research that mainly focused on English, our work encompasses a diverse set of languages, including the following:

Western European: English, German, Spanish, French
Baltic: Lithuanian

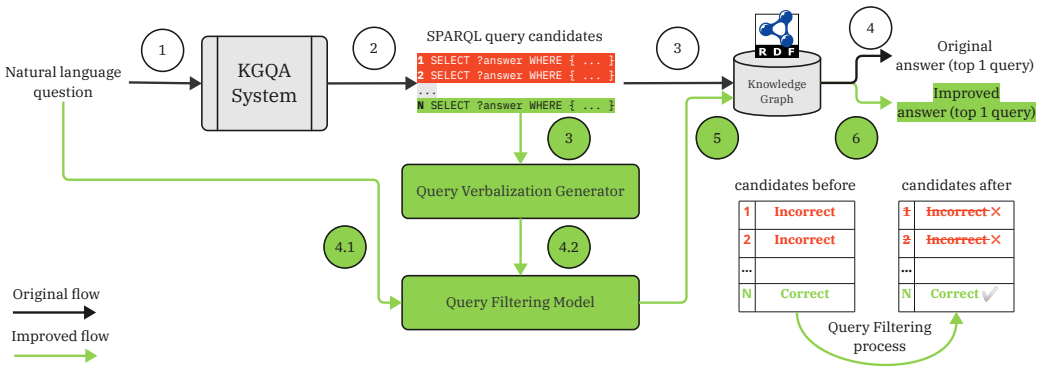


Fig. 6.1.: Big Picture: Query candidate filtering for multilingual KGQA systems.

Slavic: Russian, Ukrainian, Belarusian
Other: Armenian, Bashkir

These languages are drawn from the QALD-9-plus benchmark [155], which serves as the foundation for our experiments (detailed in Chapter 4). This chapter covers the *research gap 2* and the following *research questions*:

$\mathcal{RQ}$**6.1** What is the potential for developing a universal quality improvement approach that enhances answer quality across various KGQA systems?
$\mathcal{RQ}$**6.2** How well can our query filtering approach work in languages other than English, to ensure KGQA systems deliver good results across different languages?

In our experiments, we utilize Language Models (LMs) as binary classifiers to evaluate whether a specific SPARQL query accurately addresses a given natural language (NL) question (*correct*) or fails to do so (*incorrect*). To ensure robust and insightful findings, we employ a diverse set of LMs. This selection includes BERT [46] and DistilBERT [174], open-source instruction-tuned large language models (LLMs) such as ZEPHYR-7B [200] and Mistral-7B [94], as well as proprietary LLMs like GPT-3.5 [@143] and GPT-4 [142], which represent the cutting edge in language model technology.

Our experimental process is structured in two stages:

S_1 In the initial experimental stage, we assess the classification quality of LMs in differentiating between correct and incorrect SPARQL queries by employing the P@1 metric.
S_2 In the subsequent stage, we investigate the impact of our query filtering method on the performance of KGQA systems, specifically QAnswer [50] and MemQA (a new system introduced in this paper to provide baseline reference values).

We measure the quality of question answering (QA) using Precision@1 and ATS [63] both before and after the application of our filtering approach. Our results indicate a substantial enhancement in quality across all metrics and languages evaluated.

CRediT Statement about the author's contribution roles This chapter is mainly based on the following peer-reviewed publication:

1. Gashkov, A., **Perevalov, A.**, Eltsova, M., Both, A. (2022). Improving Question Answering Quality Through Language Feature-Based SPARQL Query Candidate Validation. In: Groth, P., et al. The Semantic Web. ESWC 2022. Lecture Notes in Computer Science, vol 13261. Springer, Cham.

2. **Perevalov, A.**, Gashkov, A., Eltsova, M., Both, A. (2024). Language Models as SPARQL Query Filtering for Improving the Quality of Multilingual Question Answering over Knowledge Graphs. In: Stefanidis, K., Systä, K., Matera, M., Heil, S., Kondylakis, H., Quintarelli, E. (eds) Web Engineering. ICWE 2024. Lecture Notes in Computer Science, vol 14629. Springer, Cham.

The author's roles according to the CRediT taxonomy for the publication 1. are: Data curation, Formal analysis, Investigation, Software, Visualization, and Writing—original draft. For the publication 2.: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Visualization, and Writing—original draft.

6.2 The SPARQL Query Filtering Approach

The approach focuses on filtering out incorrect SPARQL query candidates produced by a KGQA system in response to natural language questions. By considering questions in multiple languages in our evaluation, we enhance the generalizability of our method. The central component of our approach involves using fine-tuned or instruction-tuned language models for binary classification tasks. These models act as filters to remove incorrect SPARQL queries (see Figure 6.1). Let QAS represent a KGQA system such that $QAS : Q \rightarrow \Phi$, where:

- **Input:** $q \in Q$ denotes a natural language question written in a specific language (e.g., German), where q is a question within a dataset.
- **Output:** $\hat{\Phi} = \{\hat{\phi}_1, \hat{\phi}_2, \ldots, \hat{\phi}_k\}$ is the output of the KGQA system for question q. $\hat{\Phi}$ is an ordered set (i.e., a list) of SPARQL query candidates produced by a system. This set can be empty, contain one or more correct queries, or consist entirely of incorrect queries. Six examples are illustrated in Figure 6.2.

Each question q has an associated list of *ground truth answers* $\mathcal{A}$, as defined by the dataset. This list may be empty. For any given q, a SPARQL query generated by the QAS produces a different list of answers, $\mathcal{A}'$, which are considered as *predicted* answers.

To assess the *correctness* of a query, we use a function called $isCorrect$. This function operates as follows: (1) It takes the answers generated by a specific $\hat{\phi}_i$ query, denoted as $\mathcal{A}'_i$, along with the corresponding ground truth answers $\mathcal{A}_i$ as input.

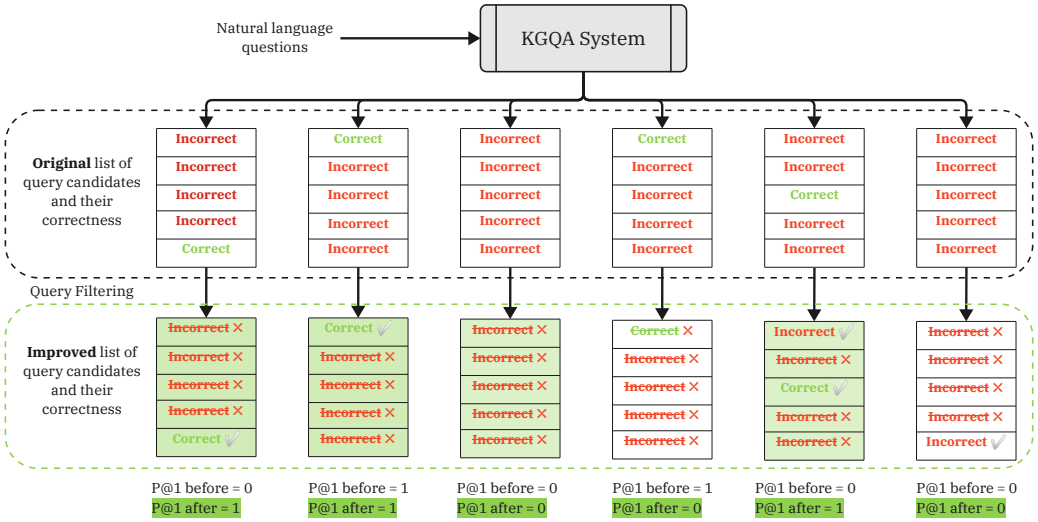


Fig. 6.2.: The impact of query filtering (QF) on six examples (E1 to E6), each consisting of lists of five candidates, is evaluated using Precision@1 (P@1). The examples, where QF eliminated *all incorrect* and *no correct* were eliminated are highlighted with green. In example E1, the application of QF optimizes the result to perfection. In E2, all incorrect candidates are removed, although this does not alter the final result. In E3 and E4, all candidates are filtered out, which is beneficial for E3 as it eliminates all incorrect results but damaging for E4, where correct results are also removed. In examples E5 and E6, despite optimization, the quality remains unchanged because an incorrect query candidate remains in the top position. While the optimized results maintain the same P@1 score, their trustworthiness is significantly enhanced.

(2) It computes the F1 score based on the provided answer sets. (3) It assigns a $label = \{\text{correct}, \text{incorrect}\}$ to indicate whether the query's answers are correct.

$$isCorrect(\mathcal{A}_i, \mathcal{A}'_i) = \begin{cases} correct, & \text{if F1 score}(\mathcal{A}_i, \mathcal{A}'_i) = 1.0 \\ incorrect, & otherwise \end{cases}$$

To enhance the quality of the QA process by filtering SPARQL query candidates, we need to develop a function F that acts as a binary classifier, such that $F : (q_i, \hat{\phi}_i) \rightarrow$ label. This filtering function F *does not reorder the list but marks queries as correct or incorrect*. Consequently, a correct query can only be positioned at the top of the list once all preceding incorrect queries have been eliminated.

Verbalization and Binary Classification of SPARQL Queries. To develop the filtering function F, we leverage language models that have been fine-tuned or instruction-tuned to serve as binary classifiers. Since many knowledge graphs do not offer human-readable URIs for their entities (for instance, Angela Merkel is represented as `Q567`[1] in Wikidata), we propose to verbalize the SPARQL queries

[1] `https://www.wikidata.org/wiki/Q567`

for such knowledge graphs. This means transforming the queries into a natural language-like representation by using the labels of the corresponding entities from a given knowledge graph (e.g., Wikidata).

We differentiate between pre-trained encoder-only LMs that require fine-tuning for specific downstream tasks (e.g., BERT) and decoder-only instruction-tuned LLMs that generate output based on prompts (e.g., Mistral or GPT-4). Task-specific LMs need SPARQL queries to be verbalized for input, alongside the natural language question. In contrast, instruction-tuned LLMs can employ knowledge injection techniques within their prompts to establish connections between a URI and its corresponding label.

KGQA Systems. The effectiveness of the approach is to be assessed using KGQA systems. The criteria used to select these systems are: (a) support for multilingual input; (b) the ability to answer questions using the Wikidata knowledge graph; and (c) the ability to respond with an ordered list of SPARQL query candidates.

To establish reference values that thoroughly illustrate the potential of our method, we also implemented a KGQA system that memorizes correct SPARQL queries for questions from KGQA benchmarks. We refer to this system as *MemQA*[2]. When given a natural language question, MemQA returns a list of SPARQL query candidates. This list contains one memorized correct query, while the rest are randomly selected from other questions (i.e., they are incorrect for the given question). The list's length can be adjusted parametrically, and the order of the SPARQL query candidates is random. As a result, all SPARQL query candidates generated are technically valid and sourced from human-curated benchmarks, ensuring their accuracy. Since a correct query candidate is guaranteed to be included, a flawless query validation using a binary classifier would lead to impeccable QA quality. This scenario corresponds to a KGQA system that can furnish reference values for evaluating our approach.

Evaluation. To assess the impact of SPARQL query filtering on QA quality, we utilize the Precision@1 (P@1) metric. This metric is evaluated both before and after applying our approach. We adhere to the precision definition recommended by [206] (see Chapter 2), which is designed to address the common division-by-zero error that arises when the sum of true positives and true negatives is 0.0. In this specific scenario, if the true positives, false positives, and false negatives are all 0.0, the precision, recall, and F1 score are defined to be 1.0 (as per [207]). We calculate P@1 in accordance with this modification.

[2] `https://github.com/WSE-research/memorized-question-answering-system`

In situations where all candidates are removed, resulting in a confusion matrix of all zeroes, calculating precision directly is not feasible due to division by zero. In such cases, we define P@1 to be zero if any correct candidate was eliminated during the filtering process; otherwise, it is set to one. To account for unanswerable questions, specifically when $\mathcal{A} = \emptyset$, we also utilize the ATS. This metric, as originally defined in [63] (see Chapter 2), is specifically designed to assess the trustworthiness of KGQA systems.

6.3 Experimental Setup

Our experiments are divided into two major stages. In the *first stage* (S_1), we perform binary classification experiments to ascertain whether a verbalized SPARQL query can correctly answer a given NL question (i.e., determine if it is correct or incorrect). In the *second stage* (S_2), we apply these binary classifiers to the outputs of two KGQA systems, MemQA and QAnswer, to validate the SPARQL query candidates produced (i.e., filter out incorrect queries). For both stages, we utilize the QALD-9-plus dataset, which includes both train and test splits.

We employ three groups of LMs:

MG_1 comprises BERT-like models,
MG_2 includes open-source instruction-tuned LLMs,
MG_3 consists of commercial instruction-tuned LLMs.

Specifically, MG_1 includes multilingual BERT[3] and multilingual DistilBERT[4]. MG_2 features Mistral 7B[5] and Zephyr 7B[6], while MG_3 includes GPT-3.5[7] and GPT-4[8]. The detailed experimental setup for S_1 and S_2 is described in the following subsections.

6.3.1 S_1—Classification

We evaluate the binary classification task as follows. The training and testing data from QALD-9-plus were prepared as follows: Each question q with ID i in the dataset has its own ground truth (i.e., correct) SPARQL query. We randomly assign a SPARQL

[3] https://huggingface.co/bert-base-multilingual-cased
[4] https://huggingface.co/distilbert-base-multilingual-cased
[5] https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.1
[6] https://huggingface.co/HuggingFaceH4/zephyr-7b-beta
[7] https://platform.openai.com/docs/models/gpt-3-5
[8] https://platform.openai.com/docs/models/gpt-4-and-gpt-4-turbo

```
SPARQL candidate: SELECT ?uri WHERE { ?uri wdt:P31 wd:Q131436 . }
Input: List all boardgames by GMT.[SEP]{ ?uri instance of board game }
Label: Correct
```

Fig. 6.3.: Here is an example of a SPARQL query candidate along with its input tuple and corresponding label, which is used to fine-tune and evaluate MG_1 (BERT-like) models. This example is based on the question with "id=1" from the training split of QALD-9-plus (RDF prefixes are omitted).

```
There is a pair of a question and a SPARQL query:
question: List all boardgames by GMT.
query: SELECT ?uri WHERE { ?uri wdt:P31 wd:Q131436 . }
Label for wdt:P31 is instance of.
Label for wd:Q131436 is board game.
Are the question and the query similar? Answer yes or no.
```

Fig. 6.4.: Example of a prompt in English to MG_2 and MG_3 models based on the question with "id=1" from train split of QALD-9-plus.

query from another question with ID j ($i \neq j$) in the same dataset to create an incorrect candidate (cf. *negative sampling*). Consequently, each question results in two data examples: $(q_i, \hat{\phi}_i), 1$, representing a correct or positive example, and $(q_i, \hat{\phi}_j), 0$, representing an incorrect or negative example. This approach ensures a balanced distribution of classes within the dataset.

The models from MG_1 were fine-tuned for the binary classification task. Following our approach (see Section 6.2), we verbalized SPARQL queries using Wikidata labels, representing them in an NL-like surface form. The input tuple for the model, q_i and $\hat{\phi}_i$, is connected using a `SEP` token (see Figure 6.3). The target $label$ values are encoded as a set $\{1, 0\}$, respectively.

Both models from MG_1 were loaded and trained using the `transformers` [221] library on the HuggingFace[9] model hub. Through a grid search procedure for epoch tuning, we empirically determined that both BERT and DistilBERT require 4 epochs to achieve optimal quality on our data. Training was conducted with the Adam optimizer [105] and a batch size of 16. The hardware setup included 64 AMD EPYC 7502P CPUs, 96 GB RAM, and no GPU.

The models from MG_2 and MG_3 were employed "as-is" using zero-shot prompts with the knowledge injection technique. These prompts include q_i, a raw $\hat{\phi}_i$, and a set of $(URI,\ label)$ tuples, which serve as the knowledge injection component retrieved from Wikidata (see Figure 6.4). Using this information, the models in MG_2 and MG_3 are instructed to produce "yes" or "no", indicating a correct or incorrect result, respectively.

[9] `https://huggingface.co/models`

```
Input (German): Liste die Brettspiele von GMT auf.
Query candidates:
1: SELECT ?name WHERE { wd:Q23215 wdt:P1477 ?name.}
2: SELECT ?uri WHERE { ?uri wdt:P31 wd:Q131436 . }
```

Fig. 6.5.: MemQA: A list of SPARQL query candidates, featuring two candidates, for the German translation of the question shown in Figure 6.4.

```
Input (German): Liste die Brettspiele von GMT auf.
Query candidates:
1: SELECT DISTINCT ?o1 WHERE { wd:Q131436 wdt:P2354> ?o1 . } LIMIT 1000
2: SELECT ?s0  WHERE { VALUES ?s0 { wd:Q12139612> }}
```

Fig. 6.6.: QAnswer: A list of query candidates, containing two candidates, for the German translation of the question in Figure 6.4 (response is simplified, prefixes are omitted).

The models from MG_2 were accessed via the HuggingFace inference endpoint[10] powered by an NVIDIA A10G GPU. The models from MG_3 were utilized through the official OpenAI Python library[11]. Specifically, we used the `gpt-3.5-turbo-1106` and `gpt-4` models. The `temperature` parameter was set to 0, while all other parameters were maintained at their default values.

6.3.2 S_2—Question Answering

To evaluate the QA quality and the effect of SPARQL query filtering, we calculate metrics such as P@1 and ATS@1both before and after the SPARQL query candidate validation. We generate query candidates for each question by posing NL questions to the test data splits of QALD-9-plus to two systems: MemQA (our system for reference values) and QAnswer (a real-world system).

We deploy the MemQA system locally and configure it to produce a predetermined number of query candidates for each experiment: 2, 3, 5, 8, 13, 21, 34, and 55, following the Fibonacci sequence. This configuration is intended to provide reference values while ensuring diverse sets of SPARQL query candidates with varying lengths. It is important to note that MemQA supports every input language, as it is based on the aforementioned QA datasets. Examples of input and output from MemQA are shown in Figure 6.5 and Figure 6.6.

The QAnswer system generates varying numbers of query candidates for questions, ranging from 0 to 60 as observed empirically. Additionally, QAnswer does not support all languages present in the test datasets. Therefore, we used four languages that are both included in the dataset and supported by QAnswer—English, German,

[10] `https://huggingface.co/inference-endpoints`
[11] `https://github.com/openai/openai-python`

Russian, and Spanish—each of which has sufficient data for model training. The QAnswer system was accessed through its public API[12].

6.4 Evaluation and Analysis

6.4.1 S_1—Classification

When evaluating multilingual capabilities, we aim to identify the best-performing model with two primary objectives: maximizing the average F1 score and minimizing the standard deviation (stdev) of the F1 score across all languages [154]. The joint objective is, therefore, to achieve the highest average F1 score with the lowest standard deviation in F1 score values (Pareto front). Upon analyzing the results across different model groups, MG_2 exhibits significantly poorer performance compared to MG_1 and MG_3. MG_1 and MG_3 display comparable quality, although GPT-4 consistently attains a high F1 score across most languages, with the exception of Bashkir. The BERT, DistilBERT, and GPT-3.5 models succeed in high-resource languages (English, German, Russian, French, and Spanish), while their performance drops for other languages (see Table 6.1). Consequently, BERT-like models and proprietary GPT models significantly outperform open-source LLMs in our binary classification task setting. For the Pareto efficient solutions, see Figure 6.7.

Tab. 6.1.: S_1—classification results based on the QALD-9-plus, F1 Score, %

Language	BERT	DistilBERT	Zephyr 7B	Mistral 7B	GPT-3.5	GPT-4
en	97.4	95.6	60.6	75.8	95.9	95.4
de	95.6	92.4	57.8	56.0	92.8	93.5
ru	95.9	93.2	67.3	48.1	95.4	87.5
fr	92.0	77.2	33.3	33.3	96.1	89.1
es	96.3	94.4	50.0	54.3	92.3	93.1
uk	92.0	90.6	33.3	50.0	85.4	96.3
lt	85.7	84.3	50.6	51.7	85.2	94.6
be	89.4	84.1	33.3	32.9	66.1	95.9
ba	74.1	72.9	33.3	60.4	52.1	52.0
hy	64.1	78.3	33.3	33.3	71.3	94.9
μ (mean)	88.3	86.3	45.3	49.6	83.3	89.2
σ (std)	11.0	8.1	13.5	13.7	15.2	13.4

[12] `https://backend.app.qanswer.ai/swagger-ui/index.html`

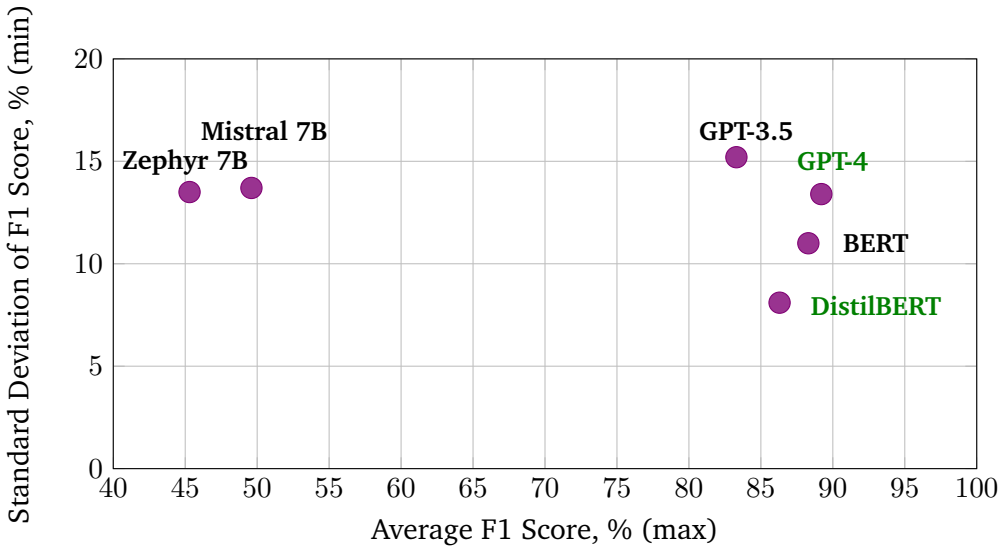


Fig. 6.7.: S_1—multi-objective classification performance of the considered LMs. The green-colored titles correspond to the models at the Pareto front.

6.4.2 S_2—Question Answering

In this subsection, we present the evaluation results of the two QA systems, MemQA and QAnswer[13] We analyze the impact of SPARQL query filtering on QA quality. The *MemQA* system simulates an almost "ideal" KGQA system by ensuring at least one correct SPARQL query in every list of query candidates. Therefore, we use its results as reference values to demonstrate the potential impact of SPARQL query filtering on QA under ideal conditions. Table 6.2 presents the results for ATS@1 and P@1 calculated using our approach on MemQA.

The results after filtering demonstrate substantial improvements, indicating that our approach significantly enhances the trustworthiness of the QA system, particularly when using the reference KGQA system, MemQA. However, questions in the Bashkir language do not fully benefit from the filtering process. The P@1 results shown in Table 6.2 highlight a notable improvement in MG_1 and GPT-4 models, with the exception of Armenian for BERT and French for DistilBERT. The GPT-3.5 model enhances quality for major languages but shows a decline for low-resource languages such as Belarusian, Bashkir, and Armenian. Since these values represent averages from the experiments, we can conclude that the approach generally performs well. It is worth mentioning that many of the results for the GPT-4 model considering ATS@1 and P@1 values are the same. This is caused by the ATS@1 specifics since

[13]The raw data is available in the online appendix at: `https://github.com/Perevalov/phd_thesis`

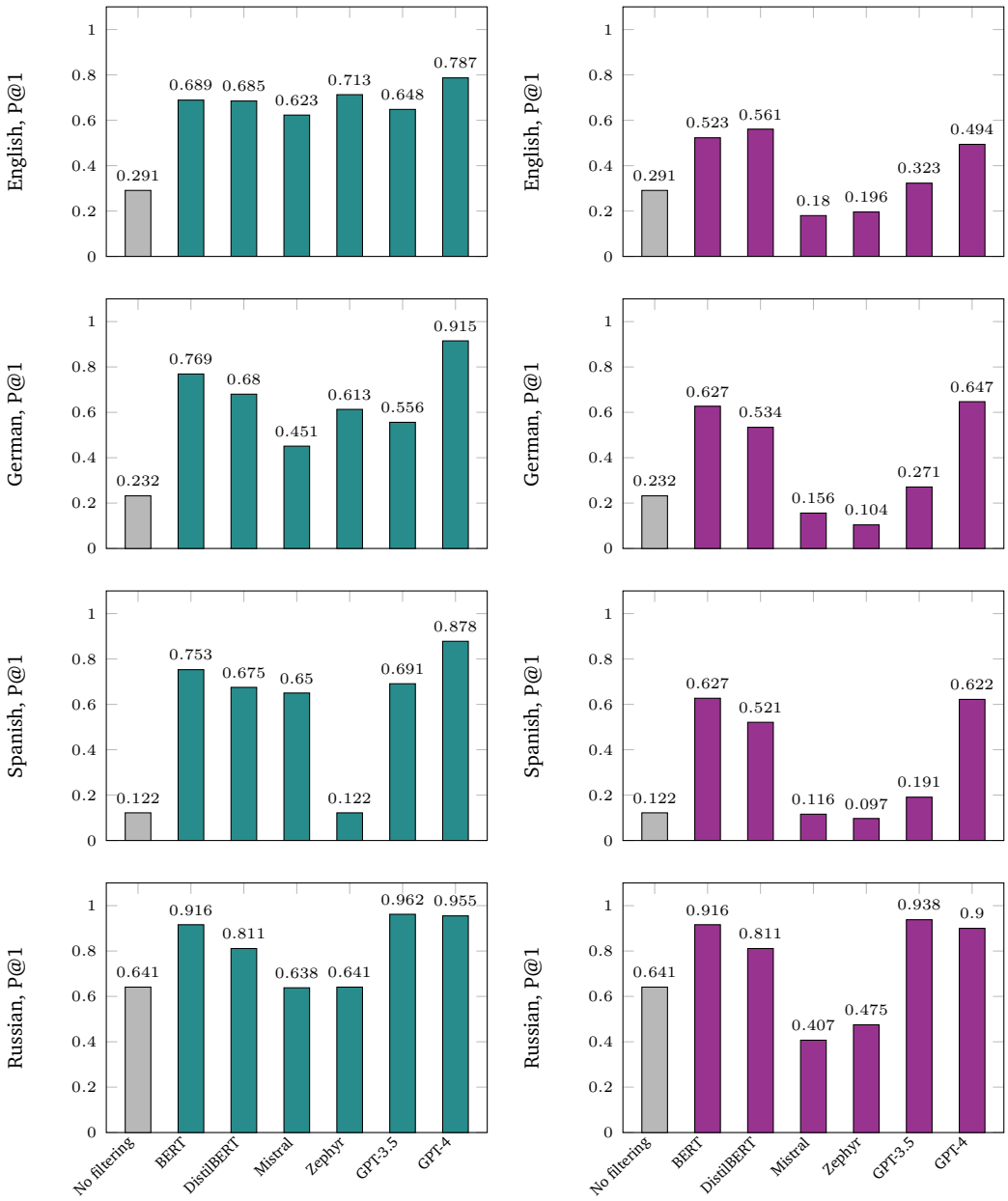
Tab. 6.2.: Results of our filtering method on the MemQA system

Language	No filtering	BERT	DistilBERT	Mistral	Zephyr	GPT 3.5	GPT-4
			ATS@1				
en	-0.580	0.719	0.720	0.362	-0.264	0.318	**0.904**
de	-0.603	0.524	0.605	-0.640	-0.241	0.330	**0.862**
es	-0.608	0.791	0.736	-0.079	-0.110	0.073	**0.853**
ru	-0.555	0.337	0.338	-0.535	0.018	0.092	**0.783**
fr	-0.574	**0.800**	-0.200	0.113	-0.296	0.427	0.760
be	-0.615	0.137	0.343	-0.025	-0.032	-0.209	**0.883**
uk	-0.654	0.248	0.398	0.156	-0.005	-0.276	**0.922**
ba	-0.597	-0.453	-0.056	-0.015	-0.097	-0.555	**0.000**
lt	-0.579	0.491	0.346	-0.298	-0.181	-0.178	**0.882**
hy	-0.496	0.053	0.305	-0.021	0.095	-0.226	**0.832**
			Precision@1				
en	0.210	0.854	0.830	0.415	0.209	0.365	**0.904**
de	0.198	0.751	0.747	0.167	0.174	0.520	**0.862**
es	0.196	**0.880**	0.819	0.029	0.006	0.247	0.853
ru	0.223	**0.895**	0.878	0.225	0.458	0.706	**0.895**
fr	0.213	**0.827**	0.393	0.141	0.191	0.453	0.760
be	0.193	0.548	0.559	0.000	0.122	0.106	**0.901**
uk	0.173	0.615	0.648	0.458	0.080	0.226	**0.923**
ba	0.201	0.271	**0.294**	0.002	0.064	0.078	0.000
lt	0.211	0.634	0.542	0.043	0.197	0.410	**0.884**
hy	0.252	0.053	0.505	0.263	0.147	0.137	**0.863**

it returns exactly the same values if there are no incorrect and nonempty answer sets.

The P@1 results in Figure 6.8 indicate that the MG_1 and MG_3 models show an improvement in P@1 after applying the filtering approach in half of the experimental scenarios on the QALD-9-plus dataset. There are two reasons for the outlier results for Russian, even before filtering, the P@1 is notably high. Firstly, while QAnswer produces up to 60 candidates for other languages, it generates only 3 candidates for Russian in most cases. Secondly, distinguishing between correct and incorrect candidates usually becomes straightforward for Russian. For example, in response to the question "What is the time zone of Salt Lake City?", the candidates are available in Figure 6.9. The first one is correct, while the two others are just nonsense (the label for `wdt:P31` is "instance of").

Table 6.4 presents the average results for Answer Trustworthiness Score achieved using *QAnswer*. The figures may seem inconclusive, but this is due to a decrease in GPT-4's quality when the number of query candidates exceeds 6 (see the online appendix). Surprisingly, the Russian language demonstrates superior results over English, German, and French, although, it is clear that the English capabilities of the used models are inherently better. This happens due to the fact that the QAnswer



(a) Only one SPARQL query candidate in a response

(b) Average from one to 60 SPARQL query candidates in a response

Fig. 6.8.: Precision@1 values for the QAnswer system are depicted. In the figure, the left-hand side illustrates the results when the lists of query candidates are truncated to just 1 (i.e., no second candidate can advance to the top of the list), while the right-hand side presents the average values for candidate lists of sizes ranging from 1 to 60. Each bar represents the value for a specific model. The column labeled "No filtering" shows the metric value without applying our approach.

Tab. 6.3.: Average number of query candidates (μ) returned by QAnswer

	English	German	Spanish	Russian
μ candidates	34.123	29.739	17.915	2.449

Tab. 6.4.: The results of the filtering method applied to the QAnswer system, aggregated across all different lengths of query candidate lists (ranging from 1 to 60) (see Section 6.3.2).

Language	No filtering	BERT	DistilBERT	Mistral	Zephyr	GPT 3.5	GPT-4
Answer Trustworthiness Score @ 1							
en	-0.418	-0.160	**-0.119**	-0.648	-0.627	-0.375	-0.169
de	-0.535	-0.123	-0.223	-0.694	-0.814	-0.502	**-0.080**
es	-0.756	**-0.206**	-0.339	-0.791	-0.861	-0.684	-0.208
ru	0.282	0.592	0.487	-0.176	-0.057	**0.613**	0.571
Precision @ 1							
en	0.291	0.523	**0.561**	0.180	0.196	0.323	0.494
de	0.232	0.627	0.534	0.156	0.104	0.271	**0.647**
es	0.122	**0.627**	0.521	0.116	0.097	0.191	0.622
ru	0.641	**0.916**	0.811	0.407	0.475	0.938	0.900

system consistently returns less query candidates for Russian language (see Table 6.3. This behavior simplifies the task for the query filtering method and increases the final quality results.

Figure 6.8 illustrates the P@1 values for various datasets and models, both before and after applying SPARQL query filtering.

6.5 Discussion

Our results demonstrate a significant impact of our approach on questions in all languages. However, there are some exceptional cases of improvement with languages that are infrequently used, such as Belarusian, Lithuanian, Armenian, Bashkir, and to a lesser extent, Ukrainian. A post-experiment analysis revealed that many questions could not be processed using our approach because labels for the resources were unavailable. This situation led to the automatic acceptance of the question, meaning the filtering method was not applied. This observation highlights a critical issue in striving for accessible information from the Web of Data for all humans (cf. [153]).

```
1: SELECT DISTINCT ?o1 WHERE { wd:Q23337 wdt:P421 ?o1 . } LIMIT 1000
2: SELECT DISTINCT ?o1 WHERE { ?s1 wdt:P31  ?o1 . }  LIMIT 1000
3: SELECT DISTINCT ?s1 ?o1 WHERE { ?s1 wdt:P31 ?o1 . }  LIMIT 1000
```

Fig. 6.9.: Query candidates for the question “What is the time zone of Salt Lake City?”

Consequently, it underscores the need to complete the Linked Open Data Cloud, at least with regard to resource labels, to support broader information accessibility.

Regarding the suboptimal performance of MG_2, one might argue that the prompt used—despite being straightforward—contributed to the problems observed with these models. Based on additional manual experiments, we tentatively assume that LLM-specific prompt optimization would not significantly alter the results. A similar argument could be made for using only English prompts; a language-specific prompt might lead to quality improvements. However, these topics likely require further evaluation beyond the scope of this paper.

We answer *$\mathcal{RQ}$6.1* as follows, by employing a SPARQL query filtering method that integrates LMs as binary classifiers, our study effectively addresses the challenge of incorrect query rankings that degrade the trustworthiness of KGQA systems. This approach is notably system-agnostic, enabling it to function as a universal extension across different KGQA systems without necessitating modifications to their core algorithms. The experimental results show substantial enhancements in answer trustworthiness and precision metrics, underscoring the potential of this method to significantly improve the quality and trustworthiness of KGQA systems across multiple languages while offering a flexible integration framework adaptable to existing architectures.

The *$\mathcal{RQ}$6.2* is targeted by leveraging a diverse set of language models, including BERT, DistilBERT, and newer large language models like GPT-4, exhibiting improved performance across Western European languages such as German and Spanish, as well as Slavic and other languages like Russian and Lithuanian. The experimental results indicate significant precision and trustworthiness gains in question answering across these diverse linguistic contexts. Although the approach shows robust applicability, minor challenges are noted, particularly concerning resources with incomplete labels in less-resourced languages like Bashkir and Belarusian (potentially vulnerable languages).

6.6 Conclusion

In this chapter, we introduced a general approach to enhance the quality of KGQA systems. Unlike other research efforts, we did not develop a new KGQA algorithm but focused on a method to improve answer quality. Specifically, our approach effectively removes incorrect query candidates, thereby significantly reducing the number of incorrect results shown to users. This improvement greatly enhances

the trustworthiness of such systems. Furthermore, our work is applicable to non-English questions, addressing the need for people to access information from KGQA systems in their native languages—most of which are not English—without relying on machine translation.

Future work could involve experimenting with language-specific and LLM-specific prompts. The proposed approach could also be extended by utilizing additional KG properties. A promising direction for further improvement would be to address the issue of unavailable resource labels.

7 Large Language Model-driven Agent for mKGQA

7.1 Introduction

Previous approaches to multilingual KGQA have employed both rule-based and neural methods [50, 201] to address downstream tasks (e.g., named entity recognition, relation detection, and query template classification) required for constructing SPARQL queries that represent the users' information need. More recent methods (e.g., [187]) leverage Large Language Models (LLMs) to generate such structured queries directly from non-English input. The application of newly introduced *LLM agents* (or *augmented* language models) to KGQA has demonstrated significantly improved performance compared to LLMs that rely solely on standard prompting techniques [232, 96, 84]. However, the multilingual aspect of these systems remains largely unexplored within the research community. To the best of our knowledge, *no KGQA systems based on LLM agents have previously been proposed, implemented, or evaluated across multiple languages*. Furthermore, existing research on LLM agents focuses mainly on proprietary closed models or fine-tuning open-source models, i.e., modifying the model's weights to enhance performance on a specific task. This fine-tuning approach poses the risk of "forgetting" previously acquired knowledge (i.e., catastrophic forgetting) while optimizing for the target task, thus reducing the model's reusability in other applications [120].

One of the key advantages of LLM agents is their ability to simulate human-like reasoning processes [116]. When solving complex problems, humans typically break them down into simpler subtasks [85, 41], effectively creating a step-by-step *plan* to solve the problem. While writing a SPARQL query, this decomposition is essential: one must not only break down the task, but also *look up* query language syntax, find relevant entity identifiers in the target KG, and analyze *feedback* from the triplestore upon executing the query. To simulate the human-like process of writing SPARQL queries, we introduce mKGQAgent—an LLM-based agent framework designed as a KGQA system that follows a semantic parsing approach. Specifically, given a user's question (multiple languages are supported), it generates a SPARQL query to fulfill the information need.

Our experiments investigate the effectiveness of both proprietary and open-source LLMs as the backbone of the agent while ensuring model reusability without fine-tuning, thereby avoiding the risk of catastrophic forgetting. We use our findings from Chapter 6 to validate SPARQL queries via the feedback step, which is a part of the mKGQAgent described in Section 7.2.2. In addition, according to our findings in Chapter 5, we include additional experiments in which we use machine translation (MT) as an alternative to asking questions in the native language. To further improve the straightforward MT approach, we use our results from the preliminary study [188] and integrate named entity-aware machine translation (NEAMT).

This chapter addresses research gap 4 and aims to answer the following *research questions*:

- $\mathcal{RQ}$***7.1*** How do different LLM agent workflow steps (e.g., plan, action, tool calling, feedback, etc.) impact the generation of SPARQL queries from natural language?
- $\mathcal{RQ}$***7.2*** How efficient are these LLM agent steps regarding computation time and the number of additional calls required?
- $\mathcal{RQ}$***7.3*** How does the quality of SPARQL query generation vary when prompting LLM agents in non-English languages (especially low-resource ones)?
- $\mathcal{RQ}$***7.4*** How does translating non-English questions into English affect the quality of KGQA?

We conducted experiments on the widely used KGQA benchmark with multilingual support—QALD-9-plus [155]. Our evaluation focuses on 10 languages from the QALD-9-plus benchmark. The experimental results demonstrate the effectiveness of the mKGQAgent architecture, achieving superior performance even in non-English settings. The key *contributions* of our work are as follows: (a) we propose a novel LLM agent architecture for the KGQA task—mKGQAgent—which surpasses previous approaches (in addition, it took first place in the Text2SPARQL challenge), (b) to the best of our knowledge, mKGQAgent is the first initiative that applies LLM agents to multilingual KGQA, and (c) we perform a comprehensive efficiency analysis of our architecture, outlining the associated computational costs when using mKGQAgent for SPARQL query generation.

CRediT Statement about the author's contribution roles This chapter is mainly based on the following peer-reviewed publication:

1. Srivastava N., **Perevalov A.**, Kuchelev D., Moussallem D., Ngonga Ngomo, A.-C., and Both A. Lingua Franca–Entity-Aware Machine Translation Approach

for Question Answering over Knowledge Graphs. In Proceedings of the 12th Knowledge Capture Conference 2023, pp. 122-130.

2. **Perevalov A.**, Both. A. Text-to-SPARQL Goes Beyond English: Multilingual Question Answering Over Knowledge Graphs through Human-Inspired Reasoning. First International TEXT2SPARQL Challenge Co-Located with Text2KG @ Extended Semantic Web Conference 2025. (Accepted, not published yet).

According to the CRediT taxonomy for publication 1, the author's roles are Formal analysis, Investigation, Visualization, and Writing—original draft.

7.2 The mKGQAgent Architecture

Inspired by related work, we introduce the architecture and the workflow of our mKGQAgent framework for SPARQL query generation. Given a natural language (NL) question, it generates a corresponding SPARQL query to retrieve the required information encapsulated within the NL query. The mKGQAgent architecture is *language-agnostic* and consists of several key steps (see Figure 7.1b for an overview).

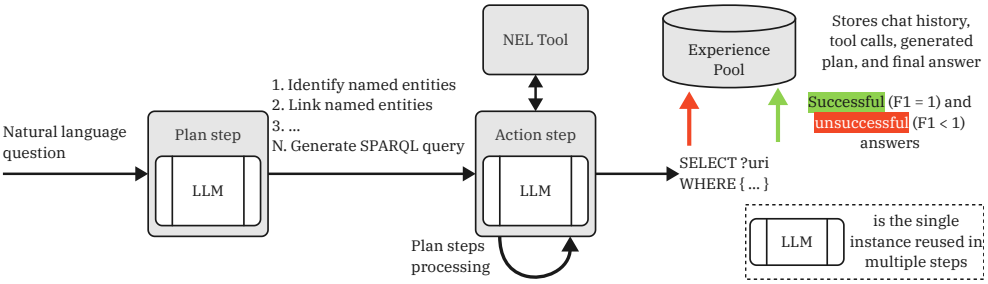
Our approach follows the terminology established in recent survey articles on LLM agents [126, 216]. The framework operates in two main phases: the *offline phase* and the *evaluation phase*. The offline phase is essential for preparing the experience pool (see Section 7.2.1). During the offline phase, we employ the *simple agent* ($\mathcal{SA}gent$) to gather intermediate processing steps for the experience pool (see Figure 7.1a). $\mathcal{SA}gent$ uses the *plan step* (cf. Section 7.2.1) to generate a structured step-by-step plan and the *action step*, that either calls the LLM or the *named entity linking (NEL) tool* (cf. Section 7.2.1) ultimately leading to the SPARQL query generation.

In the *evaluation phase*, the mKGQAgent uses the plan step and the action step with the experience pool and the NEL tool and the feedback step with access to the triplestore.

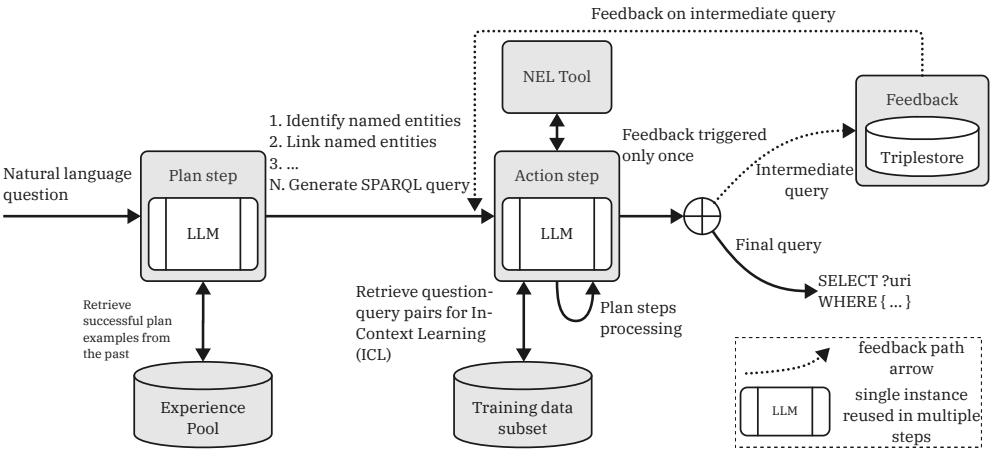
7.2.1 Offline Phase of the mKGQAgent

Named Entity Linking (NEL) Tool

Likewise humans look up a resource URI in a KG, the TOOL component interacts with the environment (i.e., KG) and retrieves labels from there. Assuming that an



(a) $\mathcal{SA}gent$ workflow demonstration (offline phase). In the offline phase, the $\mathcal{SA}gent$ is used to gather the experience pool of positive and negative answers over the train subsets.



(b) mKGQAgent workflow demonstration (online phase). In the evaluation phase, the mKGQAgent uses the experience pool examples to improve planning, the in-context learning training examples to improve SPARQL query generation awareness, and the feedback to correct possible errors.

Fig. 7.1.: General overview on the mKGQAgent framework. The offline phase, which is required for gathering experience pool. The evaluation phase—the routing of the mKGQAgent's components and their integral modules.

LLM was not given the URI-label mappings of a particular resource, the SPARQL query generation would not be possible. Importantly, we do not propose a novel NEL algorithm while introducing the NEL tool in our agent architecture. In contrast, we demonstrate how to utilize an existing NEL service in the LLM agent workflow (see Algorithm 1).

The entity and relation candidates are proposed by the backbone LLM within the tool calling process in the action step (see Sections 7.2.1 and 7.2.2). This tool calling is a crucial part of the agent that is enabled by the instruction-following capabilities of LLMs. Overall, the NEL Tool contains entity and relation linking tools that map recognized entities and relations to identifiers from a KG. Entity and relation linking

Algorithm 1 NEL Tool

Require: Entity candidates E, Relation candidates R, NEL service $\mathcal{NEL}$
Ensure: Dictionary with linked entities: linkedEntities, Dictionary with linked relations: linkedRelations

Initialize empty dictionaries linkedEntities and linkedRelations
for each entity $e_i \in E$ **do**
    $e_i^{\text{URI}} \leftarrow \mathcal{NEL}(e_i)$
    if e_i^{URI} is not empty **then**
        linkedEntities$[e_i] \leftarrow e_i^{\text{URI}}$
    end if
end for
for each relation $r_j \in R$ **do**
    $r_j^{\text{URI}} \leftarrow \mathcal{NEL}(e_j)$
    if r_j^{URI} is not empty **then**
        linkedEntities$[r_j] \leftarrow r_j^{\text{URI}}$
    end if
end for
return (linkedEntities, linkedRelations)

is crucial for the semantic parsing process since the URIs representing resources in a KG may not reflect the label of an entity[1].

Plan step

The plan step leverages the backbone LLM to generate a step-by-step list of tasks to develop a SPARQL query given a question. The intuition behind the plan step is that it simplifies the task for the model such that it does not need to handle the whole complexity at once. For example, such tasks as entity recognition and linking, query refinement, etc. Thus, following the human-like behavior [85, 41], the plan step intends to break down the complex task of writing a SPARQL query into a combination of simpler subtasks. Hence, the action step deals with a simple subtask at one point in time, which has the results of the previous steps in its conversation history. See Algorithm 2 for details regarding the plan step.

Algorithm 2 Plan step without experience pool

Require: Natural language question q_i, system prompt S_{plan}, model $\mathcal{LLM}$
Ensure: Step-by-step plan p_i ▷ List of textual tasks

$p_i \leftarrow \mathcal{LLM}(S_{\text{plan}}, q_i)$ ▷ Query LLM with system prompt and question
return p_i

[1] e.g., in Wikidata [211], Q567 (`https://www.wikidata.org/wiki/Q567`) for "Angela Merkel"

Action Step without Experience Pool

Once the plan is generated, the action step executes each plan task sequentially, leveraging the NEL Tool for the entity linking.

Algorithm 3 Action Step without Experience Pool

Require: Step-by-step plan p_i, model $\mathcal{LLM}$, tool NEL, system prompt S_{action}
Ensure: Generated SPARQL query $\hat{\phi}_i$

```
Bind NEL to LLM
Initialize empty chat history H_i
for each step s_j ∈ p_i do
    h_j ← LLM(S_action, s_j)        ▷ LLM may call tool or just itself
    Append h_j to H_i
end for
φ̂_i ← lastElementOf(H_i)
return φ̂_i
```

This approach ensures the agent follows the structured plan, interacting with necessary tools to refine and complete the SPARQL query generation process.

Experience Pool Construction

During the offline phase, we utilize $\mathcal{SA}gent$ to collect the *experience pool*. This involves evaluating the correctness of the generated SPARQL queries (based on the training data) and storing them together with the intermediate steps (i.e., plan, chat history) in a vector database. Therefore, each natural language question from the training subset is converted into a vector representation associated with metadata, including the corresponding plan, intermediate steps of the action step, and the final results. The experience pool is a non-parametric memory of our agent that contains both successful (F1 score $= 1.0$) and unsuccessful (F1 score < 1.0) SPARQL query generation attempts.

Algorithm 4 Add Example to the Experience Pool

Require: Training set example $d_i \in \mathcal{D}_{\text{train}}$, step-by-step plan p_i, chat history $\mathcal{H}_i$, Experience pool $\mathcal{E}$, Text embedding model $\mathcal{EMB}$
Ensure: Updated experience pool $\mathcal{E}'$

```
q_i, φ_i ← d_i        ▷ Unpack training example (question and ground truth SPARQL)
φ̂_i ← lastElementOf(H)        ▷ Get the SPARQL generated by SAgent
F1_i ← F1_score(φ_i, φ̂_i)        ▷ Compute F1 score
v_{q_i} ← EMB(q_i)        ▷ Convert question to a vector
E' ← E + {q_i, v_{q_i}, φ_i, φ̂_i, p_i, H, F1_i}
return E'
```

Therefore, the experience pool holds the information about the quality of the generated SPARQL queries ($F1_i$), the step-by-step plan (p_i) that was used to generate this particular query, and other metadata (e.g., ground truth SPARQL query). This information can further be accessed by the mKGQAgent using vector similarity over the corresponding embeddings v_q stored in the experience pool $\mathcal{E}$. It is important to note that *only the training or development subsets of the corresponding benchmarks can be used to collect the experience pool*. This ensures the fact that no test data is leaked to the model.

7.2.2 Evaluation Phase of the mKGQAgent

Plan step with the Experience Pool

In the evaluation phase, the plan step leverages the experience pool to find relevant plan examples for better planning quality. The plan examples are included in the system prompt S_{plan} (see Algorithm 5).

Algorithm 5 Plan step with Experience Pool

Require: Natural language question q_i, system prompt S_{plan}, model $\mathcal{LLM}$, experience pool $\mathcal{E}$, text embedding model $\mathcal{EMB}$
Ensure: Step-by-step plan p_i ▷ List of textual tasks

1: $v_{q_i} \leftarrow \mathcal{EMB}(q_i)$
2: $P \leftarrow findTopNPlans(\mathcal{E}, v_{q_i})$ ▷ Finds top-N similar plans with F1 $= 1.0$
3: $S_{\text{plan}}^{\text{experience}} \leftarrow S_{\text{plan}} + P$ ▷ The plans are included to the prompt
4: $p_i \leftarrow \mathcal{LLM}(S_{\text{plan}}^{experience}, q_i)$ ▷ Query LLM with system prompt and question
5: **return** p_i

Hence, plan step benefits from the prior successful planning examples while using them in the system prompt for in-context learning.

Action step with the Experience Pool

Once the plan is generated, the action step executes each of the plan tasks sequentially, leveraging the NEL Tool for the entity linking.

The usage of the experience pool ensures that the LLM benefits from the in-context SPARQL query examples from the training subset. It is important to note that the plan p_i can be populated with the result of the feedback step (in case the feedback is triggered).

Algorithm 6 Action Step with the Experience Pool

Require: Step-by-step plan p_i, model $\mathcal{LLM}$, tool NEL, system prompt S_{action}, experience pool $\mathcal{E}$, text embedding model $\mathcal{EMB}$
Ensure: Generated SPARQL query $\hat{\phi}_i$

Bind NEL to $\mathcal{LLM}$
Initialize empty chat history $\mathcal{H}_i$
$v_{q_i} \leftarrow \mathcal{EMB}(q_i)$
$P \leftarrow findTopNQueries(\mathcal{E}, v_{q_i})$ ▷ Finds top-N similar SPARQL queries
$S_{\text{action}}^{experience} \leftarrow S_{\text{action}} + P$ ▷ The queries are included to the prompt
for each step $s_j \in p_i$ **do**
    $h_j \leftarrow \mathcal{LLM}(S_{\text{action}}^{experience}, s_j)$ ▷ LLM may call tool or just itself
    Append h_j to $\mathcal{H}_i$
end for
$\hat{\phi}_i \leftarrow$ lastElementOf($\mathcal{H}_i$)
return $\hat{\phi}_i$

Feedback Step

The feedback step is inspired by our SPARQL query validation approach using LLMs, which is described in Chapter 6. The feedback executes the generated SPARQL query ϕ on a triplestore, collects the response, and integrates it into a pre-defined prompt template for the action step. Once the first version of a SPARQL query is generated (i.e., the result of the last planning step executed at the action step), it is redirected to the feedback step. The feedback is formulated only once per input question, i.e., there are no multiple feedback options intended to avoid infinite loops. The detailed feedback step workflow is defined in Algorithm 7.

Algorithm 7 Feedback Step

Require: Intermediate query ϕ_i, prompt template S_{feedback}, triplestore $\mathcal{KG}$
Ensure: Feedback prompt S'_{feedback}

$\mathcal{A}_i \leftarrow \mathcal{KG}(\phi_i)$ ▷ Query the triplestore and get the response
$S'_{\text{feedback}} \leftarrow S_{\text{feedback}} + \mathcal{A}_i$ ▷ Populate prompt with the response
return S'_{feedback}

After that, the feedback S'_{feedback} is redirected to the action step. The action step executes the feedback to refine the SPARQL query and delivers the final query as the result.

7.3 Experimental Setup

We conduct our experiments on the commonly used KGQA benchmark: QALD-9-plus [155]. QALD-9-plus contains 558 questions in multiple languages and queries over DBpedia and Wikidata. We consider all available languages from QALD-9-plus, in addition, we also take the Spanish questions, which were contributed to this dataset later [184]. The structure of QALD-9-plus includes question texts and the corresponding ground truth SPARQL queries that return the expected answer to a question. For the evaluation of the KGQA quality, we employ the Precision, Recall, and F1 score metrics.

7.3.1 Large Language Models and Text Embedding Models

In this work, we use both open-source and proprietary LLMs. The proprietary ones are provided by OpenAI[2], namely, GPT-3.5 (`gpt-3.5-turbo-0125`), and GPT-4o (`gpt-4o-2024-05-13`). The models are accessed via the official Python SDK[3] with `temperature=0`, and other hyperparameters are set to default.

The open-source LLMs are: Qwen2.5 72B Instruct[4] and Meta Llama 3.1 70B Instruct[5]. Both models were used with the AWQ [117] quantization (4-bit) to fit into the memory. The models were deployed via the vLLM [111] framework. The maximal context size of the models was set to 16384 tokens to avoid out-of-memory exceptions. The other hyperparameters were set to default. For the open-source LLMs, we used a server with two Nvidia L40S GPUs (each 48GB vRAM).

For creating text embeddings for the experience pool we used a specific model trained for producing high-quality text embeddings for multilingual input—`multilingual e5 large`[6] [217]. According to the MTEB leaderboard[7] [136], the model is listed among the top-3 on different languages (we considered embedding models with the size smaller than 1 billion parameters).

[2] `https://platform.openai.com/docs/models`
[3] `https://github.com/openai/openai-python`
[4] `https://huggingface.co/Qwen/Qwen2-72B-Instruct-AWQ`
[5] `https://huggingface.co/hugging-quants/Meta-Llama-3.1-70B-Instruct-AWQ-INT4`
[6] `https://huggingface.co/intfloat/multilingual-e5-large`
[7] `https://huggingface.co/spaces/mteb/leaderboard`

7.3.2 Implementation of mKGQAgent

The mKGQAgent *architecture* is implemented within the LangChain framework[8] in Python. This framework facilitates the integration of various components required for the agent's functionality.

The *entity linking* within the NEL tool is implemented via the Wikidata official public entity lookup endpoint[9]. This endpoint is capable of handling input in multiple languages. The NEL tool also uses an *external relation linker*—Falcon 2.0 [171].

The *routing* between the plan, action, and feedback is implemented within the LangGraph framework[10], which is part of the LangChain ecosystem.

The *prompts* used within the mKGQAgent are written in different languages so they match the input question language. The prompts in English, German, and Russian were written by native speakers. The other prompts were acquired via machine translation and further structure validation. The English version of the prompts is demonstrated in Figure 7.2. The system prompt instructs the agent that it is an "intelligent KGQA system" (profiling) and that it can use tools for named entity linking. The planning prompt guides the agent to produce a step-by-step plan to write a SPARQL query. The feedback prompt gives the agent feedback from the triplestore execution and asks it to rework its query if necessary. The SPARQL queries generated by the mKGQAgent are executed on the official Wikidata SPARQL endpoint[11].

7.3.3 Baselines

To compare the performance of the mKGQAgent we select both "pre-LLM era" KGQA systems and the ones that use different prompting techniques with LLMs. In particular, the following approaches are selected for comparison with ours:

QAnswer uses classical approaches like query templates, rules indexes, and no language models behind [50];

Platypus uses grammar rules to transform natural language into SPARQL [147];

DeepPavlov uses a pipeline of fine-tuned language models to generate a SPARQL query [201];

[8] `https://python.langchain.com`
[9] `https://www.wikidata.org/w/api.php?action=wbsearchentities`
[10] `https://langchain-ai.github.io/langgraph/`
[11] `https://query.wikidata.org/bigdata/namespace/wdq/sparql`

```
You are an intelligent Knowledge Graph-based Question Answering system.

You can use the tools to help yourself only if you \
DON't have this information in chat history:
- 'wikidata_el' for named entity linking  \
(e.g. "Person name" -> "URI" or "is child of" -> "URI")
to determine URIs in the Wikidata KG

{QUESTION_QUERY_EXAMPLE} # comes from the experience pool
```

(a) The system prompt ($S_{action}^{\text{experience}}$) for the action step with the usage of the experience pool $\mathcal{E}$

```
For the given objective, come up with a simple step by step plan to write a SPARQL query.
This plan should involve individual tasks (e.g., named entity linking, relation linking,
expected answer type classification), that if executed correctly \
will yield the correct SPARQL.
Do not add any superfluous steps.
The result of the final step should be the final  SPARQL query.
Don't propose to execute the query.
At the end step you MUST output exactly **ONE** SPARQL query string \
**without extra text or markdown**.

{USER_QUESTION}
{PLAN_EXPERIENCE_EXAMPLE} # comes from the experience pool
```

(b) The planning prompt ($S_{\text{plan}}^{\text{experience}}$) for the plan step with the usage of the experience pool $\mathcal{E}$

```
This is feedback to your generated SPARQL query produced by executing it on a triplestore.
Please rework your query if neccessary.

Initial question: {USER_QUESTION}
Your query: {GENERATED_SPARQL} # intermediate SPARQL query

--- Start triplestore response ---
{FEEDBACK} # comes from the query execution on a triplestore
--- End triplestore response ---

Make sure that the query is formatted correctly.
No extra text. No markdown. Just plain SPARQL.
Determine whether to output a URI (SELECT ?uri), number (COUNT), date, \
boolean (ASK), string (SELECT ?label).
DON'T USE "SERVICE wikibase:label"
```

(c) The feedback prompt (S_{feedback}) for the feedback step

Fig. 7.2.: The English versions of the prompts used within the mKGQAgent. Placeholders are color-coded with blue. Comments are color-coded with red. Important: we use language-specific prompts for every considered language.

KGQAN uses a set of components like answer type prediction and triple pattern generator to come up with a SPARQL query [141];

Triad an LLM agent architecture that uses three multiple roles for the KGQA task [232];

MST5 a system that is based on the mT5-XL model and fine-tuned specifically for generating SPARQL queries [187];

HQA follows human-like reasoning process using an LLM augmented with tools [114].

The baselines were selected according to the reported results in the KGQA leaderboard [157] and our systematic review from the Chapter 3. We reuse the reported results in our paper for comparison with our mKGQAgent approach.

Fair Comparison Considerations

As some of the baselines were implemented in the "pre-LLM era", one may assume that the comparison of such baselines with the most advanced language models is unfair. Therefore, we would like to emphasize the following statements: (1) some of the baselines are not designed to use LLMs, and (2) we explicitly highlight the size of LLMs and their effect on the end quality, including costs.

7.3.4 Machine Translation of the Input

To evaluate how well machine translation to English serves as an alternative to processing non-English questions natively, we use the OPUS MT [197] models similarly to Chapter 5.

In addition, we use a named entity-aware machine translation (NEAMT) approach, which we first introduced in [188]. The NEAMT approach consists of 3 main steps: (1) Named Entity Recognition (NER) and Named Entity Linking (NEL) of a question, (2) replacing the identified Named Entities (NEs) with the corresponding numerical placeholders, and (3) using an MT tool for obtaining text in a target language. For the NER task, we used Davlan[12] for German and Spanish and Babelscape [196] for Russian. For the NEL task, the mGENRE [30] tool was used for all languages. Finally, for the MT step, we also used OPUS MT. The comparison of the MT and NEAMT approaches is demonstrated in Figure 7.3.

Our machine translation experiments are limited to German, Russian, and Spanish because no NER and NEL tools support a larger set of diverse languages. We also selected these languages because they all represent different language branches: Germanic, Slavic, and Romance, respectively.

[12] `https://huggingface.co/Davlan/bert-base-multilingual-cased-ner-hrl`

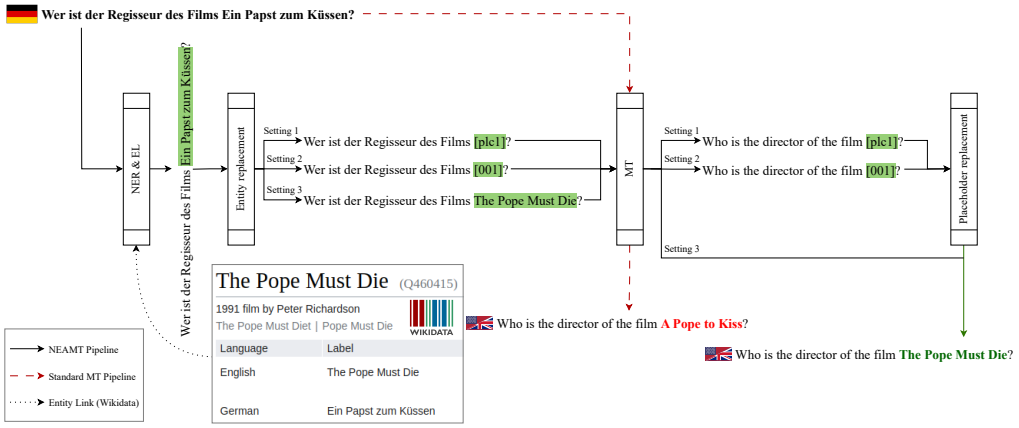


Fig. 7.3.: The detailed illustration of the NEAMT and MT workflows (left to right) based on an example question in German: "Wer ist der Regisseur des Films Ein Papst zum Küssen?" (Who is the director of the film The Pope Must Die?). The NEAMT workflow is depicted with the solid black arrows while the MT is with the dashed red arrows (cf. [188]).

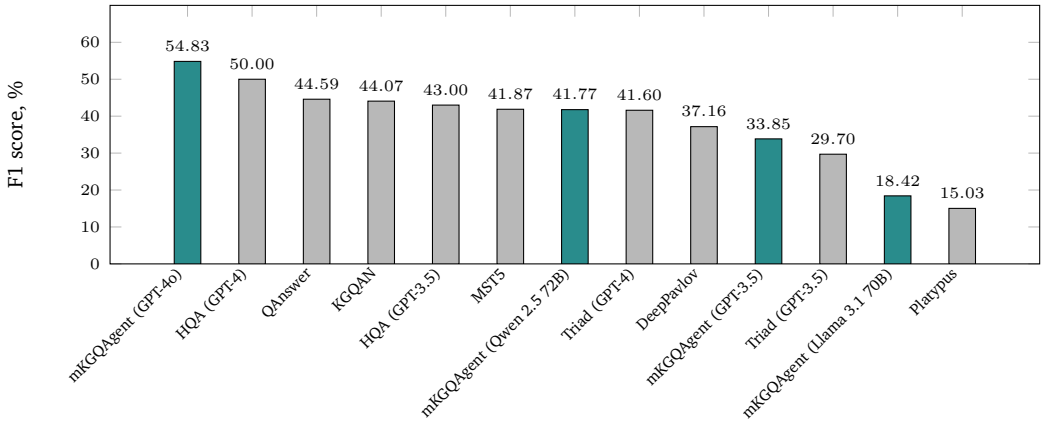


Fig. 7.4.: Comparison between our mKGQAgent approach (teal) and the baselines (grey) on English questions of QALD-9-plus.

7.4 Evaluation and Analysis

7.4.1 English-only Comparison with the Baselines

The results presented in Figure 7.4 illustrate a comparative analysis between our mKGQAgent approach (highlighted in teal) and various baseline methods (depicted in grey) on the English questions from the QALD-9-plus benchmark. Our mKGQAgent (GPT-4o) achieves the highest F1 score of 54.83%, surpassing all baselines, including HQA (GPT-4), which attains 50.00%. This demonstrates the effectiveness of our

approach in leveraging structured planning and retrieval mechanisms to enhance semantic parsing performance.

Among the baselines, QAnswer (44.59%) and KGQAN (44.07%) show competitive results but still fall short of our top-performing model. Interestingly, HQA (GPT-3.5) achieves 43.00%, indicating that the transition to GPT-4 has significantly improved query generation capabilities. The performance of mKGQAgent (Qwen 2.5 72B: 41.77%) and Triad (GPT-4: 41.60%) suggests that smaller models benefit from agentic architecture surpassing bigger ones. Notably, our mKGQAgent (GPT-3.5) variant scores just 33.85%, still outperforming several baselines but trailing behind its GPT-4o counterpart.

Lower-ranked models, such as mKGQAgent (Llama 3.1 70B) (18.42%) and Platypus (15.03%), suggest that certain model architectures struggle with SPARQL query generation in English.

Overall, these findings underscore the advantages of our agentic approach, particularly when paired with state-of-the-art LLMs like GPT-4o. The significant gap between mKGQAgent (GPT-4o) and other models highlights the effectiveness of experience pooling and refined execution strategies in improving semantic parsing for KGQA.

7.4.2 Multilingual Comparison with the Baselines

In Table 7.1, we present the evaluation results of our approach in comparison to the selected baselines that support multiple languages (see Section 7.3.3).

Tab. 7.1.: Evaluation results of mKGQAgent (our approach) compared to the baselines that support multiple languages. The evaluation was conducted on the test subset of the QALD-9-plus. The best results per language are highlighted in bold.

	English	German	Spanish	French	Russian	Belarusian	Ukrainian	Lithuanian	Bashkir	Armenian
QAnswer	44.59	31.71	16.8	**23.00**	21.43	N/A				
Platypus	15.03	N/A		4.17		N/A				
DeepPavlov 2023	37.16	N/A			31.17	N/A				
MST5	41.87	41.26	N/A	41.67	**37.61**	29.07	**34.67**	**31.15**	18.42	N/A
mKGQAgent (GPT-3.5)	33.85	25.85	22.08	22.87	11.75	16.05	12.36	16.27	6.63	8.88
mKGQAgent (GPT-4o)	**54.83**	**43.08**	**38.28**	22.76	31.67	**31.56**	28.54	25.54	**40.48**	9.09
mKGQAgent (Qwen 2.5 72B)	41.77	21.86	17.96	17.60	12.92	10.81	10.86	9.56	10.40	0.00
mKGQAgent (Llama 3.1 70B)	18.42	19.64	5.23	17.70	5.22	5.73	6.70	9.52	4.65	15.07

The experimental results demonstrate mKGQAgent's robust performance across multiple languages on the QALD-9-plus benchmark. When implemented with GPT-4o, the system achieves state-of-the-art results with F1 scores of 54.83%, 43.08%, 38.28%, 31.56%, and 40.48%, respectively, for English, German, Spanish, Belarusian, and Bashkir. The other models, including the open-source ones, demonstrate similar

patterns in quality values for different languages adjusted by the backbone LLM size. For example, the languages using Cyrillic-based scripts (Russian, Belarusian, Ukrainian, Bashkir) generally yield poorer results in comparison to the languages using Latin-based scripts.

While comparing mKGQAgent to the other baselines, we see that QAnswer outperforms mKGQAgent (GPT-3.5) on French. However, the difference is not substantial (23.00% vs 22.87%). The MST5 system significantly outperforms mKGQAgent (GPT-4o) on Russian (37.61% vs 31.67%), Ukrainian (34.67% vs 28.54%), and Lithuanian (25.54% vs 31.15%). It is worth mentioning that the authors leveraged supervised fine-tuning while creating the MST5 system. Our approach does not perform weight updates of the backbone LLMs while preserving their generalizability and reusability in other tasks.

Among the different models tested within the mKGQAgent, GPT-4o consistently demonstrates superior performance across all metrics and languages. While GPT-3.5 shows competitive results, there is a clear performance gap compared to GPT-4o. The Qwen and Llama models, though performing adequately, indicate that model choice significantly impacts system effectiveness. However, the Qwen 2.5 model (72B), being more than 2 times smaller than GPT-3.5 (175B), outperforms it on English questions (41.77% vs 33.85%) and other languages, such as Russian (12.92% vs 11.75%) or Bashkir (10.40% vs 6.63%). Nevertheless, the pattern suggests that the quality of the underlying language model plays a crucial role in the overall KGQA performance.

There are also several outliers in the results of the mKGQAgent evaluation. In particular, the mKGQAgent using the GPT-3.5 model outperforms the one with GPT-4o (22.87% vs 22.76%), while the difference is not substantial, the results on other languages show a completely different pattern where mKGQAgent with GPT-4o significantly outperforms the older GPT-3.5 model. This may be influenced by the incomplete set of French questions in the QALD-9-plus benchmark. Another outlier on the Bashkir language, where mKGQAgent (GPT-4o) demonstrates an unusually high F1 score of 40.48% when compared to GPT-3.5 (6.63%), Qwen 2.5 (10.40%), or Llama 3.1 (4.65%). This can be caused either by the *knowledge cutoff* factor. The knowledge cutoff for GPT-3.5 remains in September 2021, and the initial commits of the QALD-9-plus dataset were also created in late September 2021. Hence, there is a chance that the superior performance of GPT-4o is achieved not only by the technical improvements but also influenced by the knowledge cutoff date. The Qwen 2.5 and Llama 3.1 models do not demonstrate substantial deviations in the Bashkir language.

7.4.3 Machine Translation for non-English Questions

Table 7.2 presents the evaluation results of mKGQAgent in models when handling questions originally formulated in German, Russian, and Spanish. We selected these languages because they are supported by most of the NEAMT pipelines and also represent different language branches. The evaluation compares the models' performance on native-language questions and those translated into English using both standard machine translation (MT) and the named entity-aware machine translation (NEAMT) approach.

Tab. 7.2.: Evaluation results of the mKGQAgent: A comparison between the quality (F1 score, %) of original language questions and the same questions translated into English using standard machine translation (MT) and named entity-aware (NEAMT) approaches.

	German			Russian			Spanish		
	Native	**MT**	**NEAMT**	**Native**	**MT**	**NEAMT**	**Native**	**MT**	**NEAMT**
mKGQAgent (GPT-3.5)	25.85	28.23	**33.86**	11.75	27.93	**31.38**	22.08	27.67	**28.44**
mKGQAgent (GPT-4o)	**43.08**	35.66	38.40	31.67	**35.66**	35.49	38.28	**44.18**	39.92
mKGQAgent (Qwen 2.5 72B)	21.86	**30.95**	28.36	12.92	**32.97**	32.22	17.96	31.35	**32.04**
mKGQAgent (Llama 3.1 70B)	**19.64**	17.29	15.35	5.22	17.29	**18.58**	5.23	**9.63**	7.18

Across all models and languages, mKGQAgent's performance is generally higher for translated questions than for native-language questions. This suggests that translating non-English questions into English before processing yields better results. This also confirms our findings in Chapter 5. Among the translation approaches, NEAMT consistently outperforms MT across the majority of the models and languages, indicating that entity-aware translations provide more effective input for the agent.

The mKGQAgent based on GPT-4o achieves the highest performance across all settings, particularly with MT and NEAMT (for Russian and Spanish), demonstrating the superior comprehension and reasoning capabilities of this model. Qwen 2.5 72B exhibits strong performance in translation-based settings but falls behind GPT-4o. The variance in performance between languages suggests that translation quality and linguistic characteristics play a role in how effectively mKGQAgent can process and answer questions. In general, this comparison demonstrates that translating non-English questions into English consistently improves their quality and underlines the unequal quality distribution among the languages.

Table 7.3 presents a comparative assessment of the NEAMT approach against the MT method in terms of KGQA quality.

Tab. 7.3.: A comparison between the NEAMT and MT performance the the relative improvement (▲) or deterioration (▼) in terms of KGQA quality.

	Was NEAMT better than MT (NEAMT >MT)?		
	German	Russian	Spanish
mKGQAgent (GPT-3.5)	True (+19.94% ▲)	True (+12.35% ▲)	True (+2.78% ▲)
mKGQAgent (GPT-4o)	True (+7.68% ▲)	False (-0.48% ▼)	False (-9.64% ▼)
mKGQAgent (Qwen 2.5 72B)	False (-8.37% ▼)	False (-2.27% ▼)	True (+2.2% ▲)
mKGQAgent (Llama 3.1 70B)	False (-11.22% ▼)	True (+7.46% ▲)	False (-25.34% ▼)

From an overall perspective, NEAMT delivers nearly the same performance as MT, as seen by the equal number of "True" and "False" entries. Notably, the GPT-3.5-based mKGQAgent exhibits substantial improvements with NEAMT in all languages, suggesting that some language models may be better at leveraging entity-aware translations to enhance question-answering outcomes.

Nevertheless, performance is not universally improved by NEAMT. Certain models, such as Qwen 2.5 72B, show mixed results. NEAMT outperforms MT in Spanish but also underperforms in Russian and German. Meanwhile, LLaMA 3.1 70B experiences more pronounced drops in Spanish and German, despite some gains in Russian. These variations suggest that the effectiveness of NEAMT can depend on both the specific language pair and the underlying language model's architecture or training data. Therefore, there is a systematic approach required to study the performance of NEAMT against MT. Hence, the proposed mKGQAgent architecture and our evaluation approach can be reused in further systematic studies.

Table 7.4 presents a comparative evaluation of the performance of MT and NEAMT against native-language questions within the KGQA task.

Tab. 7.4.: A comparison between the NEAMT or MT performance versus the native questions (original language in the first header row). We demonstrate the relative improvement (▲) or deterioration (▼) in terms of KGQA quality.

	German		Russian		Spanish	
	MT >Native	NEAMT >Native	MT >Native	NEAMT >Native	MT >Native	NEAMT >Native
mKGQAgent (GPT-3.5)	True (9.21% ▲)	True (30.99% ▲)	True (137.69% ▲)	True (167.05% ▲)	True (25.31% ▲)	True (28.79% ▲)
mKGQAgent (GPT-4o)	False (-17.22% ▼)	False (-10.86% ▼)	True (12.59% ▲)	True (12.06% ▲)	True (15.42% ▲)	True (4.29% ▲)
mKGQAgent (Qwen 2.5 72B)	True (41.58% ▲)	True (29.73% ▲)	True (155.21% ▲)	True (149.41% ▲)	True (74.52% ▲)	True (78.36% ▲)
mKGQAgent (Llama 3.1 70B)	False (-11.97% ▼)	False (-21.84% ▼)	True (231.46% ▲)	True (256.19% ▲)	True (84.03% ▲)	True (37.4% ▲)

The results demonstrate that, for most models and languages, both MT and NEAMT yield improvements over native-language question answering. This effect is particularly pronounced in Russian and Spanish, where NEAMT provides significant gains. For example, Qwen 2.5 72B achieves a substantial improvement when using NEAMT over native processing in Russian (+149.41%) and Spanish (+78.36%), highlighting the potential of entity-aware translations to enhance KGQA performance.

However, the impact of translation varies across models. GPT-4o, despite its strong overall performance, exhibits slight performance degradation when using MT or NEAMT for German (-17.22% and -10.86%, respectively), suggesting that this model may already be well optimized for handling German-language queries natively. Additionally, LLaMA 3.1 70B shows a decline when using either MT (-11.97%) or NEAMT (-21.84%) in German, further emphasizing that the effectiveness of translation-based approaches is model-dependent.

Overall, these findings highlight the advantages of translation in multilingual KGQA, particularly when leveraging entity-aware techniques. While NEAMT generally provides improvements over native-language processing, the effectiveness of translation varies across models and languages. Our findings in Table 7.4 confirm the results presented in Chapter 5

7.4.4 LLM Calls and Costs

An important aspect of using LLM agents is the number of model calls within one task solution, i.e., in our case, we report the number of calls per generated SPARQL query for an input question. In addition, we report the estimated number of tokens per question and the underlying costs of using LLMs.

Costs calculation for the OpenAI models

According to our calculations, the mKGQAgent *requires 13.03 LLM calls on average* to generate a SPARQL query for an input question (cf. Table 7.5). Consequently, every LLM call consumes 144 input tokens and 199 output tokens on average—this includes chat history that gradually grows during agent execution. The pricing strategy of OpenAI is based on token consumption. Therefore, we calculate the token-based price (TBP) as in Equation 7.1.

$$\text{TBP} = \left[(n_{input} \times p_{input}) + (n_{output} \times p_{output})\right] \times n_{calls} \times n_{questions} \tag{7.1}$$

Where: n_{input} represents the number of input tokens, p_{input} represents the price per input token, n_{output} represents the number of output tokens, p_{output} represents the price per output token, n_{calls} represents the number of LLM calls per question, $n_{questions}$ represents the number of questions. This results in USD 0.48 per 100

questions for the GPT-3.5 and USD 3.06 per 100 questions for the GPT-4o, respectively (prices as of December 01, 2024). For the costs of the different $\mathcal{SA}gent$ configurations, see Table 7.5.

Costs calculation for the open source models

For the open-source LLMs, we use the same values regarding the average number of LLM calls to generate a SPARQL query (13.03) and the average number of tokens per call—144 input tokens and 199 output tokens. The pricing of open source models relies on cloud providers' GPU hours and model efficiency measured in tokens per second (tok/sec). Therefore, we calculate the GPU hours-based price (GBP) as in Equation 7.2.

$$\text{GBP} = \frac{n_{questions} \times n_{calls} \times n_{output}}{r_{\text{tok/sec}}} \times p_{\text{GPU/sec}} \tag{7.2}$$

Where: $r_{\text{tok/sec}}$ represents model efficiency rate (tok/sec), $p_{\text{GPU/sec}}$ price per GPU-second. We estimated the market prices of our GPU experimental setup (2x Nvidia L40S GPU) according to one of the well-known cloud providers[13]. The model performance (tok/sec) was retrieved from the official documentation of the respective models[14], taking into account the usage of the vLLM framework [111] for deployment and the size of the context window—16384 tokens. Hence, for processing 100 questions, mKGQAgent requires 1.96 GPU hours when using the Qwen model and 0.97 GPU hours when using the Llama model. Therefore, the prices per 100 questions are USD 4.05 and 2.01, respectively. For the costs of the different $\mathcal{SA}gent$ configurations, see Table 7.5. Please note that the pricing of the open source models highly depends on the deployment parameters (e.g., number of GPUs, context window size, GPU model) and on the cloud provider fares.

7.4.5 Impact of Individual Components on the Quality

To understand the contribution of each architectural component to the overall system performance, we conducted an ablation study using the English questions from the QALD-9-plus dataset. Our baseline system, implementing only the $\mathcal{SA}gent$ with plan step and NEL tool components, achieved varying results across different language models, with GPT-4o showing the strongest initial performance at 34.37%

[13] `https://www.runpod.io/pricing`

[14] Qwen: `https://qwen.readthedocs.io/en/latest/benchmark/speed_benchmark.html`, Llama: `https://artificialanalysis.ai/models/llama-3-1-instruct-70b`

Tab. 7.5.: Impact (regarding the baseline evaluation—$\mathcal{SA}gent$ (Plan step + NEL tool) of individual components on the QA quality (F1 score) measured on English questions of QALD-9-plus dataset. The strategy of pricing calculation is described in Section 7.4.4. ▲ – increases (the higher, the better), ▲ – increases (the higher the worse).

Backbone LLM	F1 score, %	Impact	LLM Calls	Price per 100 questions	Price impact
$\mathcal{SA}gent$ (Plan step + NEL tool)					
GPT-3.5	23.15	N/A	8.87 (avg.)	USD 0.33	N/A
GPT-4o	34.37			USD 2.08	
Qwen 2.5 72B Instruct	24.87			USD 2.75	
Llama 3.1 70B Instruct	8.12			USD 1.37	
$\mathcal{SA}gent$ + Experience Pool for Plan Step					
GPT-3.5	31.17	▲34.64%	9.71 (avg.)	USD 0.36	9.31% ▲ (avg.)
GPT-4o	46.48	▲35.23%		USD 2.28	
Qwen 2.5 72B Instruct	29.45	▲18.42%		USD 3.02	
Llama 3.1 70B Instruct	17.06	▲110.09%		USD 1.49	
$\mathcal{SA}gent$ + Experience Pool for Action Step					
GPT-3.5	26.97	▲16.50%	9.02 (avg.)	USD 0.33	1.73% ▲ (avg.)
GPT-4o	52.68	▲53.27%		USD 2.12	
Qwen 2.5 72B Instruct	32.87	▲32.17%		USD 2.80	
Llama 3.1 70B Instruct	15.58	▲91.87%		USD 1.39	
$\mathcal{SA}gent$ + Feedback step					
GPT-3.5	26.91	▲16.24%	10.93 (avg.)	USD 0.40	22.65% ▲ (avg.)
GPT-4o	40.47	▲17.74%		USD 2.57	
Qwen 2.5 72B Instruct	25.72	▲3.42%		USD 3.39	
Llama 3.1 70B Instruct	9.48	▲4.43%		USD 1.68	
mKGQAgent (Plan step + NEL tool + Experience Pool for Plan and Action + Feedback step)					
GPT-3.5	33.85	▲46.22%	13.03 (avg.)	USD 0.48	46.62% ▲ (avg.)
GPT-4o	54.83	▲59.53%		USD 3.06	
Qwen 2.5 72B Instruct	41.77	▲67.95%		USD 4.05	
Llama 3.1 70B Instruct	20.42	▲151.47%		USD 2.01	

F1 score, followed by Qwen 2.5 72B (24.87%), GPT-3.5 (23.15%), and Llama 3.1 70B (8.12%)—see Table 7.5.

The integration of the experience pool for the plan step demonstrated substantial improvements across all models. Most notably, GPT-4o's performance increased to 46.48%, representing a 35.23% relative improvement over the baseline. Similar positive trends were observed with other models, with GPT-3.5 and Qwen showing moderate gains, while Llama exhibited the most substantial relative improvement of 110.09%.

The addition of the experience pool to the action step proved particularly effective, especially when combined with GPT-4o, pushing the F1 score to 52.68% (a 53.27% relative improvement over baseline). This component's impact was consistently positive across all models, suggesting its crucial role in enhancing the system's question-answering capabilities.

The feedback step introduced more modest but still significant improvements. GPT-4o's performance with this component reached 40.47%, representing a 17.74%

relative increase over the baseline. Other models showed similar patterns of improvement, though with varying magnitudes (see Table 7.5).

The full integration of all components in mKGQAgent yielded the most impressive results, with GPT-4o achieving a peak F1 score of 54.83%, marking a 59.53% improvement over the baseline. This comprehensive integration demonstrated synergistic effects across all models, with even the initially lower-performing Llama showing a remarkable 151.47% improvement. These results strongly indicate that the combination of all components creates a more robust and effective KGQA system, particularly when powered by advanced language models like GPT-4o (see Table 7.5).

7.5 Discussion

Despite we were able to prove that our proposed architecture mKGQAgent successfully handles the semantic parsing task and outperforms other approaches, this work has several limitations. First of all, we used the LLMs that do not have any significant difference in their model architecture. As these models represent current state-of-the-art and have established a widespread ecosystem, it is possible to achieve relatively high results even with not sophisticated agent architectures (cf. $\mathcal{SA}gent$). In our evaluation, we used only one KGQA benchmark that provides SPARQL queries over one knowledge graph (Wikidata). The evaluation was conducted using prompts written in European languages only.

Below, we are going to answer the research questions addressed in the study. $\mathcal{RQ}7.1$ examines how different mKGQAgent steps affect SPARQL query generation from natural language. The analysis reveals that the proposed agent architecture enables more accurate SPARQL query generation. In particular, mKGQAgent achieved state-of-the-art results (54.83% F1 score) on English and demonstrated superior quality in German, Spanish, Belarusian, and Bashkir. The planning step provides a structured approach to breaking down complex queries into manageable steps, improving the overall SPARQL query generation process. The experience pool for the plan and action steps enhances query generation by leveraging examples and past interactions, while the feedback step offers crucial validation and refinement mechanisms. Overall, the experience pool has the biggest impact on quality improvement. The feedback step aims at refining the SPARQL query after its first version is generated. The feedback brings a reasonable query improvement to the query generation process. Additionally, the NEL tool is a crucial part of the action step as it provides a piece

of external information regarding the mapping of labels to their URIs in a KG. When compared to standalone LLM approaches, the integration of these components demonstrates superior capabilities in generating accurate SPARQL queries.

RQ7.2 investigates the efficiency of these components in terms of computational resources. The evaluation results indicate that while the full mKGQAgent setup with all components achieved substantially better quality, it required additional computational resources through increased LLM calls. For example, the $\mathcal{SA}gent$ requires 8.87 LLM calls on average to achieve the end goal, while the final mKGQAgent requires 13.03 LLM calls on average, which impacts the price per SPARQL query generation by 46.62%. This highlights an important trade-off between quality improvements and computational costs. While the planning and feedback components added computational overhead through additional calls, they proved valuable in improving overall accuracy.

RQ7.3 explores the differences in SPARQL generation quality when working with non-English languages. Our work indicates that multilingual SPARQL generation presents significant challenges even to state-of-the-art LLMs. In particular, even among European languages, the quality of SPARQL query generation may degrade by more than a factor of three. For instance, when considering mKGQAgent with Qwen 2.5 72B model and English (41.77% F1 score) versus Lithuanian (9.56% F1 score). This suggests that while LLM agents can reasonably generate SPARQL queries across major languages, maintaining consistent quality among the low-resource ones requires careful consideration of language-specific challenges, appropriate agent design or model adaptation.

RQ7.4 Our results indicate that both standard machine translation (MT) and Named Entity-Aware Machine Translation (NEAMT) generally lead to higher KGQA performance compared to processing questions in their native languages. This effect is particularly strong for Russian and Spanish, where translated questions yield substantial gains, suggesting that models are better optimized for processing English input. NEAMT further improves KGQA quality by preserving entity integrity, reducing ambiguity, and enhancing contextual understanding. However, the effectiveness of translation-based approaches varies by language and model. Certain models, such as GPT-4o, exhibit strong native-language capabilities that minimize the benefits of translation. Overall, these findings confirm that translating non-English questions into English before processing is a robust strategy for improving KGQA accuracy, particularly when leveraging entity-aware translation techniques.

7.6 Conclusion

The chapter introduces a novel LLM agent framework called mKGQAgent for the Knowledge Graph Question Answering (KGQA) task. This framework aims to mimic a human-like approach by breaking down complex questions into simpler sub-tasks to arrive at an answer—SPARQL query that fulfills information needs of a user. The important feature of our LLM agent framework is that it does not require supervised fine-tuning of the LLMs, which significantly decreases the computation costs and preserves the generalizability of the original LLMs that may be utilized in our agent. The experiments carried out have shown that each step of the mKGQAgent workflow contributes positively to the quality of the results. The mKGQAgent substantially outperforms previous systems in the English, German, Spanish, Belarusian, and Bashkir questions of the QALD-9-plus data set. In addition, mKGQAgent outperformed other solutions in the overall ranking of the Text2SPARQL Challenge 2025[15]. Our approach has shown superior results in the Spanish question answering over DBpedia ranking.

We highlighted significant challenges when LLMs deal with non-English languages, especially low-resource ones, as our experimental results report significant quality degradation for some of the languages. Our experiments demonstrated that the latter challenge can be partially addressed by using MT and NEAMT techniques. However, the use of different translation techniques requires further systematic study to identify settings where each of them performs best. Despite this, the mKGQAgent framework demonstrates a promising approach to KGQA by adopting the LLM agent paradigm. While it shows its ability to work with multiple languages of reasonably good quality, we also demonstrated the trade-off between quality and computational costs that increase with the adoption of the agent paradigm.

[15] `https://text2sparql.aksw.org/results/`

8 Discussion

In this section, we present and interpret the results and findings of this thesis by responding to our main research question and addressing the identified research gaps. Additionally, we outline potential directions for future research in the field of multilingual KGQA.

8.1 Research Question and Research Gaps

This section revisits the research gaps formulated in the introduction by describing how this thesis addresses them.

Research Question Our primary research question—how to ensure that multilingual Knowledge Graph Question Answering systems provide equally high-quality performance across multiple languages, including those with limited resources—can be addressed as follows. To enable multilingual KGQA systems to deliver uniformly high-quality answers across both resource-rich and low-resource languages, this thesis proposes a stepwise methodology: (1) establishing a multilingually diverse benchmark (QALD-9-plus) for trustworthy evaluation, (2) leveraging extrinsic methods, such as machine translation of input questions and the resulting SPARQL query candidates validation for enhancing quality without changing a KGQA system's internals, and (3) combining the latter methods in a LLM-driven KGQA system that follows agentic framework to answer questions over knowledge graphs for having a multilingual system by design thanks to the LLMs capabilities.

Research Gap 1 The $\mathcal{RG}1$, which tackles the problem of the limited availability of the open source mKGQA benchmarking datasets, is covered in Chapter 3 (Related Work) and Chapter 4 (mKGQA Benchmark Extension Approach). In particular, Chapter 3 identifies a significant shortage of benchmarking datasets for mKGQA systems by systematically reviewing the related work from the past decade. In Chapter 4, we established the novel benchmarking dataset that includes translations in eight languages and 507 new SPARQL queries. Two of those languages, Bashkir

and Belarusian, are considered endangered. Quantitative text analysis reveals increased linguistic richness, and gold standard answers are updated to reflect knowledge graph evolution. Since the publication of our benchmarking dataset, it has become widely used within the mKGQA community. Therefore, we consider this a valuable contribution that covers this research gap.

Research Gap 2 The $\mathcal{RG}2$ focuses on utilizing machine translation tools to extend the multilingual capabilities of KGQA systems. Chapter 5 addresses the $\mathcal{RG}2$ by presenting machine translation (MT) as a solution for extending KGQA systems to multiple languages, including low-resource ones. Since most KGQA systems primarily support English or other resource-rich languages, users of underrepresented languages face limited access to structured knowledge. This research empirically assesses the feasibility of using MT to bridge this gap by systematically evaluating three KGQA systems (QAnswer, DeepPavlov, Platypus) with both machine-translated questions from nine languages and the original questions. Key findings reveal that translating queries into English yields the best QA performance, while translations into other languages produce varied results. The study establishes a measurable correlation between MT quality (BLEU, NIST) and QA accuracy (Precision, Recall, F1-score) and demonstrates that MT can effectively enable KGQA in unsupported languages. Using the QALD-9-plus dataset and GERBIL benchmarking platform, the research provides a reproducible framework for assessing MT-enhanced KGQA.

Research Gap 3 The $\mathcal{RG}3$, which targets the potential misinformation due to incorrect SPARQL query outputs, is tackled in Section 6 (validating the KGQA output). Our approach emphasizes the validation and verbalization of these queries. By integrating LMs for binary classification, our method filters out inaccuracies, enhancing the overall reliability of KGQA systems across multiple languages. This system-agnostic approach employs language models to assess the validity of SPARQL queries, ensuring that only the correct queries are presented to users. We ensure a comprehensive assessment that moves beyond the common English focus by utilizing both open-source instruction-tuned and proprietary LLMs across diverse languages. Consequently, our approach significantly mitigates misinformation risks associated with SPARQL errors by improving the trust and accuracy of KGQA systems' outputs, as evidenced by enhanced Precision@1 scores and Answer Trustworthiness Scores in our experiments.

Research Gap 4 The $\mathcal{RG}4$ formulates the need to investigate how LLM-based mKGQA systems generate SPARQL queries and answer questions. Chapter 7 addresses the $\mathcal{RG}4$ by implementing and evaluating mKGQAgent—a novel LLM-based autonomous agent framework for multilingual Knowledge Graph Question Answering. The agent incorporates advanced capabilities that simulate human reasoning processes and provides a comprehensive evaluation of how these components affect SPARQL query generation quality across different languages, including low-resource ones. The analysis includes detailed efficiency metrics regarding computational costs and the number of LLM calls required, enabling direct comparison with previous approaches. The evaluation demonstrates that mKGQAgent achieves state-of-the-art results while highlighting important tradeoffs between performance and computational resources. The experiments systematically compare both proprietary (GPT-3.5, GPT-4), and open-source (Qwen, Llama) LLMs across English, German, and Spanish languages, providing valuable insights into the viability of LLM agents for multilingual KGQA. This chapter significantly contributes to understanding how LLM agents can be effectively leveraged for multilingual KGQA while considering practical constraints around efficiency and costs through ablation studies of individual components and comprehensive performance analysis.

8.2 Future Research Directions

Building upon the findings of this thesis, we identify several future research directions for advancing mKGQA. Addressing these challenges will contribute to developing more robust and inclusive multilingual KGQA systems.

Benchmark Enhancement Extending current mKGQA evaluation frameworks by incorporating high-quality, multilingual question datasets with the means of crowd workers. This expansion will facilitate more comprehensive and fair system assessments across different linguistic and contextual settings. In addition, it is worth integrating the LLM-as-a-judge approach in the translations' validation phase of benchmark creation.

Investigating LLMs' Trustworthiness in Multilingual Context Finding out patterns in how users construct questions across different languages (L1, L2, etc.) and connecting these patterns to the performance of language models in KGQA. This includes analyzing syntactic structures, common linguistic errors, and spelling variations in multilingual web queries to enhance system adaptability.

Cultural Context Integration Enhancing answer ranking mechanisms by incorporating cultural sensitivity. Adapting mKGQA systems to account for users'

cultural backgrounds and contextual preferences will improve response relevance and user satisfaction. Especially, the LLM alignment approach appears to be one of the most promising solutions.

Reproducibility and Standardization Establishing rigorous frameworks for repeatable, reproducible, and replicable results' verification. Standardizing evaluation metrics, sharing open-source implementations, and maintaining accessible multilingual datasets are essential steps toward ensuring research integrity.

We strongly encourage the research community to pursue these directions to advance capabilities of the mKGQA systems, fostering more effective and equitable multilingual information access solutions.

9 Conclusion

This thesis addresses the challenge of improving answer quality in mKGQA systems, with particular emphasis on multilinguality and support for low-resource languages. Through our analysis of the literature (see Chapter 3), we observed that the multilingual dimension is frequently neglected. While multilingual support is widely recognized as a key challenge in KGQA, none of the examined surveys focus explicitly on this aspect. Notably, 88% of surveyed mKGQA systems concentrate on the Indo-European language family, which predominantly covers resource-rich languages. In addition, existing KGQA benchmarks with multilingual coverage remain limited. In light of these gaps, this thesis has set out to address the main *research question*: how can we enable multilingual Knowledge Graph Question Answering systems to achieve equally high answer quality across diverse languages, including low-resource ones? The research question splits into four research gaps introduced in Chapter 1.

A significant contribution of this research was the creation of the QALD-9-plus dataset (see Chapter 4), an extension of the existing QALD-9 benchmark, which introduced translations in eight languages, including underrepresented languages such as Armenian, Bashkir, and Belarusian (the latter two are considered as potentially vulnerable by UNESCO). By leveraging crowdsourcing with native speakers, this dataset ensured high-quality multilingual representations and improved linguistic diversity. Adapting SPARQL queries from DBpedia to Wikidata enhanced the dataset's applicability to a broader range of KGQA systems, offering a more comprehensive benchmarking resource. Since its publication, QALD-9-plus has become a renowned dataset in the research community.

Another key contribution was investigating the feasibility of using Machine Translation (MT) as an alternative to dedicated mKGQA systems (see Chapter 5). The empirical analysis demonstrated that translating questions into English yielded superior results due to the dominance of English in KGQA research. However, performance varied significantly across target languages, emphasizing the need for advancements in named entity-aware MT and more robust translation techniques for low-resource languages.

This thesis also introduced a novel approach for improving mKGQA systems by verbalizing and validating their output (see Chapter 6). The study explored methods to enhance SPARQL query validation and answer quality through language models trained for filtering incorrect results. In addition, we introduced the novel metric, called Answer Trustworthiness Score, that favors no answer rather than a wrong one. By integrating SPARQL query verbalization techniques together with the questions asked by a user, the research demonstrated that both encoder-only and decoder-only language models are capable to filter out incorrect KGQA system outputs, which improves correctness and trustworthiness of the answers.

Furthermore, this research introduced mKGQAgent, a novel Large Language Model (LLM)-driven agentic framework designed to model human-like behavior in writing SPARQL queries (see Chapter 7). By integrating essential components such as planning, in-context learning, experience pool, and environmental feedback, mKGQAgent demonstrated superior quality to previous KGQA systems across multiple languages on our QALD-9-plus benchmark. The framework highlighted the potential of LLM agents in structuring complex multilingual queries and balancing computational efficiency with performance gains.

Despite the advancements presented in this thesis, several challenges and opportunities for future research remain (see Chapter 8). The need for investigating LLMs' trustworthiness in multilingual context for KGQA persists, requiring further expansion of datasets that cover a wider range of languages and linguistic structures. Additionally, aligning the backbone LLMs that are used within KGQA systems to handle language and-specific nuances and cultural contexts better will enhance the accessibility and relevance of these systems around the world. Finally, establishing common frameworks for repeatable, reproducible, and replicable mKGQA results validation generally plays an important role in the whole research community.

In conclusion, this research makes significant contributions toward bridging the gap in mKGQA research by introducing novel datasets, evaluating translation-based approaches, introducing output validation, and proposing an innovative LLM-based agent framework. The findings and contributions of this thesis provide a foundation for future advancements in the field, ultimately fostering more inclusive and equitable access to structured knowledge across languages.

Bibliography

[1] Nitish Aggarwal. “Cross Lingual Semantic Search by Improving Semantic Similarity and Relatedness Measures”. In: *The Semantic Web – ISWC 2012*. Ed. by Philippe Cudré-Mauroux, Jeff Heflin, Evren Sirin, et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 375–382 (cit. on pp. 41, 51).

[2] Sareh Aghaei, Elie Raad, and Anna Fensel. “Question answering over knowledge graphs: A case study in tourism”. In: *IEEE Access* 10 (2022), pp. 69788–69801 (cit. on p. 3).

[3] Kisuh Ahn, Beatrice Alex, Johan Bos, et al. “Cross-lingual question answering using off-the-shelf machine translation”. In: *Workshop of the Cross-Language Evaluation Forum for European Languages*. Springer, 2004, pp. 446–457 (cit. on p. 75).

[4] Khetam Al Sharou, Zhenhao Li, and Lucia Specia. “Towards a better understanding of noise in natural language processing”. In: *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*. 2021, pp. 53–62 (cit. on p. 58).

[5] Christina Antoniou and Nick Bassiliades. “A survey on semantic question answering systems”. In: *The Knowledge Engineering Review* 37 (2022) (cit. on pp. 37, 39).

[6] Mikhail Arkhipov, Maria Trofimova, Yuri Kuratov, and Alexey Sorokin. “Tuning Multilingual Transformers for Language-Specific Named Entity Recognition”. In: *Proceedings of the 7th Workshop on Balto-Slavic Natural Language Processing*. Ed. by Tomaž Erjavec, Michał Marcińczuk, Preslav Nakov, et al. Florence, Italy: Association for Computational Linguistics, Aug. 2019, pp. 89–93 (cit. on p. 12).

[7] Sören Auer, Christian Bizer, Georgi Kobilarov, et al. “DBpedia: A nucleus for a web of open data”. In: *The Semantic Web*. Springer, 2007, pp. 722–735 (cit. on pp. 2, 20, 25).

[8] Diego Moussallem Axel-Cyrille Ngonga Ngomo and Lorenz Bühman. “A Holistic Natural Language Generation Framework for the Semantic Web”. In: *Proceedings of the International Conference Recent Advances in Natural Language Processing*. ACL (Association for Computational Linguistics). 2019, p. 8 (cit. on p. 25).

[9] Stefan Baack. “A Critical Analysis of the Largest Source for Generative AI Training Data: Common Crawl”. In: *Proceedings of the 2024 ACM Conference on Fairness, Accountability, and Transparency* (2024) (cit. on p. 3).

[10] Jinheon Baek, Alham Fikri Aji, and Amir Saffari. “Knowledge-augmented language model prompting for zero-shot knowledge graph question answering”. In: *ACL 2023 Workshop on Matching Entities*. 2023 (cit. on pp. 2, 25, 62).

[11]Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. “Neural machine translation by jointly learning to align and translate”. In: *arXiv preprint arXiv:1409.0473* (2014) (cit. on p. 11).

[12]Krisztian Balog, Jeffrey Dalton, Antoine Doucet, and Yusra Ibrahim. “Report on the Eighth Workshop on Exploiting Semantic Annotations in Information Retrieval (ESAIR ’15)”. In: *SIGIR Forum* 50.1 (2016), 49–57 (cit. on p. 3).

[13]Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. “A neural probabilistic language model”. In: *Advances in neural information processing systems* 13 (2000) (cit. on p. 10).

[14]Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. “Semantic Parsing on Freebase from Question-Answer Pairs”. In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Seattle, Washington, USA: Association for Computational Linguistics, Oct. 2013, pp. 1533–1544 (cit. on pp. 39, 43).

[15]Tim Berners-Lee, James Hendler, and Ora Lassila. “The Semantic Web”. In: *Scientific American* 284.5 (2001), pp. 34–43 (cit. on pp. 2, 20).

[16]Daniel M Bikel, Richard Schwartz, and Ralph M Weischedel. “An algorithm that learns what’s in a name”. In: *Machine learning* 34 (1999), pp. 211–231 (cit. on p. 12).

[17]Jorge Biolchini, Paula Gomes Mian, Ana Candida Cruz Natali, and Guilherme Horta Travassos. “Systematic review in software engineering”. In: *System engineering and computer science department COPPE/UFRJ, Technical Report ES* 679.05 (2005), p. 45 (cit. on pp. 31, 32).

[18]Christian Bizer, Tom Heath, and Tim Berners-Lee. “Linked data: Principles and state of the art”. In: *World wide web conference*. Vol. 1. Citeseer. 2008, p. 40 (cit. on p. 21).

[19]Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. “Freebase: A Collaboratively Created Graph Database for Structuring Human Knowledge”. In: *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*. SIGMOD ’08. Vancouver, Canada: Association for Computing Machinery, 2008, 1247–1250 (cit. on pp. 2, 25, 55).

[20]Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. “Large-scale simple question answering with memory networks”. In: *arXiv preprint arXiv:1506.02075* (2015) (cit. on p. 39).

[21]Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. “Translating embeddings for modeling multi-relational data”. In: *Advances in neural information processing systems* 26 (2013) (cit. on p. 22).

[22]Andreas Both, Dennis Diefenbach, Kuldeep Singh, et al. “Qanary–a methodology for vocabulary-driven open question answering systems”. In: *European Semantic Web Conference*. Cham: Springer International Publishing, 2016, pp. 625–641 (cit. on pp. 43, 76, 161).

[23]Andreas Both, Paul Heinze, Aleksandr Perevalov, et al. “Quality Assurance of a German COVID-19 Question Answering Systems Using Component-Based Microbenchmarking”. In: *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*. WSDM ’22. Virtual Event, AZ, USA: Association for Computing Machinery, 2022, 1561–1564 (cit. on p. 3).

[24]Andreas Both, Axel-Cyrille Ngonga Ngomo, Ricardo Usbeck, et al. “A Service-Oriented Search Framework for Full Text, Geospatial and Semantic Search”. In: *Proceedings of the 10th International Conference on Semantic Systems*. SEM ’14. Leipzig, Germany: Association for Computing Machinery, 2014, 65–72 (cit. on p. 76).

[25]Thorsten Brants, Ashok C. Popat, Peng Xu, Franz J. Och, and Jeffrey Dean. “Large Language Models in Machine Translation”. In: *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*. Ed. by Jason Eisner. Prague, Czech Republic: Association for Computational Linguistics, June 2007, pp. 858–867 (cit. on p. 9).

[26]Elena Cabrio, Philipp Cimiano, Vanessa Lopez, et al. “QALD-3: Multilingual Question Answering over Linked Data.” In: *CLEF (Working Notes)* 38 (2013) (cit. on pp. 54, 56).

[27]Elena Cabrio, Julien Cojan, Alessio Palmero Aprosio, et al. “QAKiS: an open domain QA system based on relational patterns”. In: *International Semantic Web Conference, ISWC 2012*. 2012 (cit. on p. 51).

[28]Elena Cabrio, Julien Cojan, and Fabien Gandon. “Mind the Cultural Gap: Bridging Language-Specific DBpedia Chapters for Question Answering”. In: *Towards the Multilingual Semantic Web: Principles, Methods and Applications*. Ed. by Paul Buitelaar and Philipp Cimiano. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 137–154 (cit. on pp. 42, 51).

[29]Elena Cabrio, Julien Cojan, Fabien Gandon, and Amine Hallili. “Querying Multilingual DBpedia with QAKiS”. In: *The Semantic Web: ESWC 2013 Satellite Events*. Ed. by Philipp Cimiano, Miriam Fernández, Vanessa Lopez, Stefan Schlobach, and Johanna Völker. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 194–198 (cit. on pp. 42, 51, 60).

[30]Nicola De Cao, Ledell Wu, Kashyap Popat, et al. “Multilingual Autoregressive Entity Linking”. In: *CoRR* abs/2103.12528 (2021). arXiv: 2103.12528 (cit. on p. 116).

[31]Casimiro Pio Carrino, Marta Ruiz Costa-Jussà, and José Adrián Rodríguez Fonollosa. “Automatic Spanish translation of SQuAD dataset for multi-lingual question answering”. In: *LREC 2020: 12th International Conference on Language Resources and Evaluation: Marseílle, France: May 13-15, 2020: conference proceedings*. European Language Resources Association (ELRA), 2020, pp. 5515–5523 (cit. on p. 54).

[32]Nilesh Chakraborty, Denis Lukovnikov, Gaurav Maheshwari, et al. “Introduction to neural network-based question answering over knowledge graphs”. In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 11.3 (2021), e1389 (cit. on pp. 59, 161).

[33]Mark Chen, Jerry Tworek, Heewoo Jun, et al. “Evaluating large language models trained on code”. In: *arXiv preprint arXiv:2107.03374* (2021) (cit. on pp. 12, 47).

[34]Zhe Chen, Yuehan Wang, Bin Zhao, et al. “Knowledge Graph Completion: A Review”. In: *IEEE Access* 8 (2020), pp. 192435–192456 (cit. on p. 29).

[35]Jeffrey Cheng, Marc Marone, Orion Weller, et al. “Dated Data: Tracing Knowledge Cutoffs in Large Language Models”. In: *ArXiv* abs/2403.12958 (2024) (cit. on p. 12).

[36]Hugh A Chipman, Edward I George, Robert E McCulloch, and Thomas S Shively. “mBART: multidimensional monotone BART”. In: *Bayesian Analysis* 17.2 (2022), pp. 515–544 (cit. on p. 47).

[37]Hyung Won Chung, Le Hou, Shayne Longpre, et al. “Scaling instruction-finetuned language models”. In: *Journal of Machine Learning Research* 25.70 (2024), pp. 1–53 (cit. on p. 15).

[38]Kenneth Church and Patrick Hanks. “Word association norms, mutual information, and lexicography”. In: *Computational linguistics* 16.1 (1990), pp. 22–29 (cit. on p. 16).

[39]Julien Cojan, Elena Cabrio, and Fabien Gandon. “Filling the Gaps among DBpedia Multilingual Chapters for Question Answering”. In: *Proceedings of the 5th Annual ACM Web Science Conference*. WebSci ’13. Paris, France: Association for Computing Machinery, 2013, 33–42 (cit. on pp. 42, 51).

[40]Alexis Conneau, Kartikay Khandelwal, Naman Goyal, et al. “Unsupervised Cross-lingual Representation Learning at Scale”. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Ed. by Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault. Online: Association for Computational Linguistics, July 2020, pp. 8440–8451 (cit. on pp. 16, 47).

[41]Carlos G Correa, Mark K Ho, Fred Callaway, and Thomas L Griffiths. “Resource-rational task decomposition to minimize planning costs”. In: *Proceedings of the 42th Annual Meeting of the Cognitive Science Society - Developing a Mind: Learning in Humans, Animals, and Machines, CogSci 2020, virtual, July 29 - August 1, 2020*. cognitivesciencesociety.org, 2020 (cit. on pp. 105, 109).

[42]Ryan Cotterell, Sabrina J. Mielke, Jason Eisner, and Brian Roark. “Are All Languages Equally Hard to Language-Model?” In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, June 2018, pp. 536–541 (cit. on p. 75).

[43]Ruixiang Cui, Rahul Aralikatte, Heather Lent, and Daniel Hershcovich. “Compositional Generalization in Multilingual Semantic Parsing over Wikidata”. In: *Transactions of the Association for Computational Linguistics* 10 (Sept. 2022), pp. 937–955 (cit. on pp. 47, 54, 56).

[45]Harini Navoda De Zoysa. “The Internet as an accessible source of knowledge with special reference to undergraduates of the University of Kelaniya”. In: Faculty of Humanities, University of Kelaniya, Sri Lanka, 2016 (cit. on p. 3).

[46]Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "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)*. Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 4171–4186 (cit. on pp. 16, 46, 47, 90).

[47]Dennis Diefenbach, Andreas Both, Kamal Singh, and Pierre Maret. "Towards a question answering system over the semantic web". In: *Semantic Web* 11.3 (2020), pp. 421–439 (cit. on p. 55).

[48]Dennis Diefenbach, José Giménez-García, Andreas Both, Kamal Singh, and Pierre Maret. "QAnswer KG: Designing a Portable Question Answering System over RDF Data". In: *The Semantic Web*. Ed. by Andreas Harth, Sabrina Kirrane, Axel-Cyrille Ngonga Ngomo, et al. Cham: Springer International Publishing, 2020, pp. 429–445 (cit. on pp. 46, 51).

[49]Dennis Diefenbach, Vanessa Lopez, Kamal Singh, and Pierre Maret. "Core techniques of question answering systems over knowledge bases: a survey". In: *Knowledge and Information systems* 55.3 (2018), pp. 529–569 (cit. on pp. 37, 78).

[50]Dennis Diefenbach, Pedro Henrique Migliatti, Omar Qawasmeh, et al. "QAnswer: A Question Answering Prototype Bridging the Gap between a Considerable Part of the LOD Cloud and End-Users". In: *The World Wide Web Conference*. WWW '19. San Francisco, CA, USA: Association for Computing Machinery, 2019, 3507–3510 (cit. on pp. 43, 46, 51, 61, 75, 90, 105, 114).

[51]Dennis Diefenbach, Kamal Singh, and Pierre Maret. "WDAqua-core0: A Question Answering Component for the Research Community". In: *Semantic Web Challenges*. Ed. by Mauro Dragoni, Monika Solanki, and Eva Blomqvist. Cham: Springer International Publishing, 2017, pp. 84–89 (cit. on pp. 43, 51).

[52]Dennis Diefenbach, Kamal Singh, and Pierre Maret. "WDAqua-Core1: A Question Answering Service for RDF Knowledge Bases". In: WWW '18. Lyon, France: International World Wide Web Conferences Steering Committee, 2018, 1087–1091 (cit. on pp. 43, 44, 51).

[53]Dennis Diefenbach, Kuldeep Singh, Andreas Both, et al. "The Qanary Ecosystem: Getting New Insights by Composing Question Answering Pipelines". In: *Web Engineering - 17th International Conference, ICWE 2017, Rome, Italy, June 5-8, 2017, Proceedings*. Ed. by Jordi Cabot, Roberto De Virgilio, and Riccardo Torlone. Vol. 10360. Lecture Notes in Computer Science. Springer, 2017, pp. 171–189 (cit. on pp. 26, 43, 55, 76).

[54]Eleftherios Dimitrakis, Konstantinos Sgontzos, and Yannis Tzitzikas. "A survey on question answering systems over linked data and documents". In: *Journal of intelligent information systems* 55.2 (2020), pp. 233–259 (cit. on pp. 37, 38).

[55]George Doddington. "Automatic Evaluation of Machine Translation Quality Using N-Gram Co-Occurrence Statistics". In: *Proceedings of the Second International Conference on Human Language Technology Research*. HLT '02. San Diego, California: Morgan Kaufmann Publishers Inc., 2002, 138–145 (cit. on p. 79).

[56]Martin Dürst and Michel Suignard. *Internationalized resource identifiers (IRIs)*. Tech. rep. 2005 (cit. on p. 20).

[57]Ritam Dutt, Sopan Khosla, Vinayshekhar Bannihatti Kumar, and Rashmi Gangadharaiah. "Designing harder benchmarks for evaluating zero-shot generalizability in question answering over knowledge bases". In: *ACL 2023 Workshop on Natural Language Reasoning and Structured Explanations*. 2023 (cit. on pp. 2, 24).

[58]Mohammad Fazleh Elahi, Basil Ell, Gennaro Nolano, and Philipp Cimiano. "Multilingual Question Answering over Linked Data building on a model of the lexicon-ontology interface". In: *Semantic Web Journal* (2023) (cit. on p. 63).

[59]Jeffrey L Elman. "Finding structure in time". In: *Cognitive science* 14.2 (1990), pp. 179–211 (cit. on p. 10).

[60]Dmitriy Evseev and Mikhail Arkhipov. "SPARQL query generation for complex question answering with Bert and BiLSTM-based model". In: *Computational Linguistics and Intellectual Technologies: Proceedings of the International Conference Dialogue 2020*. 2020, pp. 270–282 (cit. on pp. 46, 47, 51, 75).

[61]Bruno Faria, Dylan Perdigão, and Hugo Gonçalo Oliveira. "Question Answering over Linked Data with GPT-3". In: *12th Symposium on Languages, Applications and Technologies (SLATE 2023)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik. 2023 (cit. on p. 63).

[62]Óscar Ferrández, Christian Spurk, Milen Kouylekov, et al. "The QALL-ME Framework: A specifiable-domain multilingual Question Answering architecture $*$". In: *Journal of Web Semantics* 9.2 (2011). Provenance in the Semantic Web, pp. 137–145 (cit. on pp. 41, 51).

[63]Aleksandr Gashkov, Aleksandr Perevalov, Maria Eltsova, and Andreas Both. "Improving Question Answering Quality Through Language Feature-Based SPARQL Query Candidate Validation". In: vol. 13261. Lecture Notes in Computer Science. Springer, 2022, pp. 217–235 (cit. on pp. 29, 90, 94).

[65]Simon Gottschalk and Elena Demidova. "EventKG–the hub of event knowledge on the web–and biographical timeline generation". In: *Semantic Web* 10.6 (2019), pp. 1039–1070 (cit. on p. 55).

[66]Ivan Habernal and Miloslav Konopík. "SWSNL: Semantic Web Search Using Natural Language". In: *Expert Systems with Applications* 40.9 (2013), pp. 3649–3664 (cit. on pp. 42, 51, 58).

[67]Barry Haddow, Rachel Bawden, Antonio Valerio Miceli Barone, Jindřich Helcl, and Alexandra Birch. "Survey of Low-Resource Machine Translation". In: *Computational Linguistics* 48.3 (Sept. 2022), pp. 673–732 (cit. on p. 17).

[68]Sherzod Hakimov, Soufian Jebbara, and Philipp Cimiano. "AMUSE: Multilingual Semantic Parsing for Question Answering over Linked Data". In: *The Semantic Web – ISWC 2017*. Ed. by Claudia d'Amato, Miriam Fernandez, Valentina Tamma, et al. Cham: Springer International Publishing, 2017, pp. 329–346 (cit. on pp. 43, 51, 58).

[69]James Hammerton. “Named entity recognition with long short-term memory”. In: *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003*. 2003, pp. 172–175 (cit. on p. 12).

[71]Zellig S Harris. “Distributional structure”. In: *Word* 10.2-3 (1954), pp. 146–162 (cit. on p. 15).

[72]Martin van Hees. “Web-based automatic translation: the Yandex.Translate API”. In: 2015 (cit. on p. 78).

[73]Dan Hendrycks, Collin Burns, Saurav Kadavath, et al. “Measuring Mathematical Problem Solving With the MATH Dataset”. In: *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*. Ed. by J. Vanschoren and S. Yeung. Vol. 1. 2021 (cit. on p. 12).

[74]Shawn Lawton Henry, Shadi Abou-Zahra, and Judy Brewer. “The role of accessibility in a universal web”. In: *Proceedings of the 11th Web for all Conference*. 2014, pp. 1–4 (cit. on p. 65).

[75]Lauri Himanen, Amber Geurts, Adam Stuart Foster, and Patrick Rinke. “Data-driven materials science: status, challenges, and perspectives”. In: *Advanced Science* 6.21 (2019), p. 1900808 (cit. on p. 3).

[76]Sepp Hochreiter and Jürgen Schmidhuber. “Long Short-Term Memory”. In: *Neural Computation* 9.8 (1997), 1735–1780 (cit. on pp. 10, 47).

[77]Konrad Höffner, Sebastian Walter, Edgard Marx, et al. “Survey on challenges of question answering in the semantic web”. In: *Semantic Web* 8.6 (2017), pp. 895–920 (cit. on pp. 37, 55, 161).

[78]Aidan Hogan. “Web of Data”. In: *The Web of Data*. Cham: Springer International Publishing, 2020, pp. 15–57 (cit. on p. 21).

[79]Aidan Hogan, Eva Blomqvist, Michael Cochez, et al. “Knowledge Graphs”. In: *ACM Computing Surveys* 54.4 (July 2021) (cit. on pp. 2, 22, 24).

[81]Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. “spaCy: Industrial-strength Natural Language Processing in Python”. In: (2020) (cit. on p. 45).

[82]Albert Sydney Hornby and Anthony Paul Cowie. “Oxford advanced learner’s dictionary of current English”. In: (1977) (cit. on p. 16).

[83]Eduard Hovy, Mitch Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. “OntoNotes: the 90% solution”. In: *Proceedings of the human language technology conference of the NAACL, Companion Volume: Short Papers*. 2006, pp. 57–60 (cit. on p. 44).

[84]Xiang Huang, Sitao Cheng, Shanshan Huang, et al. “QueryAgent: A Reliable and Efficient Reasoning Framework with Environmental Feedback based Self-Correction”. In: *arXiv preprint arXiv:2403.11886* (2024) (cit. on pp. 25, 105).

[85]Quentin JM Huys, Níall Lally, Paul Faulkner, et al. “Interplay of approximate planning strategies”. In: *Proceedings of the National Academy of Sciences* 112.10 (2015), pp. 3098–3103 (cit. on pp. 105, 109).

[89]Matthias Irmer, Claudia Bobach, Timo Böhme, Anett Püschel, and Lutz Weber. “Using a chemical ontology for detecting and classifying chemical terms mentioned in texts”. In: *Proceedings of Bio-Ontologies 2013* (2013) (cit. on p. 3).

[90]Ann Irvine and Chris Callison-Burch. “A comprehensive analysis of bilingual lexicon induction”. In: *Computational Linguistics* 43.2 (2017), pp. 273–310 (cit. on p. 49).

[92]B. Jansen, Danielle L. Booth, and A. Spink. “Determining the informational, navigational, and transactional intent of Web queries”. In: *Inf. Process. Manag.* 44 (2008), pp. 1251–1266 (cit. on p. 1).

[93]Fred Jelinek, Robert L Mercer, Lalit R Bahl, and James K Baker. “Perplexity—a measure of the difficulty of speech recognition tasks”. In: *The Journal of the Acoustical Society of America* 62.S1 (1977), S63–S63 (cit. on p. 12).

[94]Albert Q. Jiang and et al. “Mistral 7B”. In: *arXiv preprint arXiv:2310.06825* (2023). arXiv: 2310.06825 [cs.CL] (cit. on p. 90).

[95]Jinhao Jiang, Kun Zhou, Zican Dong, et al. *StructGPT: A General Framework for Large Language Model to Reason over Structured Data*. 2023. arXiv: 2305.09645 [cs.CL] (cit. on p. 62).

[96]Jinhao Jiang, Kun Zhou, Wayne Xin Zhao, et al. “KG-Agent: An Efficient Autonomous Agent Framework for Complex Reasoning over Knowledge Graph”. In: *arXiv preprint arXiv:2402.11163* (2024) (cit. on pp. 25, 105).

[97]Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. “Bag of Tricks for Efficient Text Classification”. In: *arXiv preprint arXiv:1607.01759* (2016) (cit. on p. 16).

[98]Daniel Jurafsky and James H. Martin. “Speech and Language Processing (3rd Edition draft).” In: chap. Question Answering and Information Retrieval (cit. on pp. 2, 24).

[99]Daniel Jurafsky and James H. Martin. *Speech and Language Processing (3rd Edition draft)*. USA: Prentice-Hall, Inc., 2020 (cit. on pp. 9–13, 15, 24, 163).

[100]Manoj Prabhakar Kannan Ravi, Kuldeep Singh, Isaiah Onando Mulang’, et al. “CHOLAN: A Modular Approach for Neural Entity Linking on Wikipedia and Wikidata”. In: *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. Ed. by Paola Merlo, Jorg Tiedemann, and Reut Tsarfaty. Online: Association for Computational Linguistics, Apr. 2021, pp. 504–514 (cit. on p. 12).

[101]Pei Ke, Haozhe Ji, Yu Ran, et al. “Jointgt: Graph-text joint representation learning for text generation from knowledge graphs”. In: *arXiv preprint arXiv:2106.10502* (2021) (cit. on p. 47).

[102] Maria Keskenidou, Argyris Kyridis, Lina P. Valsamidou, and Alexandra-Helen Soulani. "The Internet as a source of information. The social role of blogs and their reliability". In: *Observatorio (OBS*)* (2014) (cit. on p. 3).

[103] Kimmo Kettunen. "Can type-token ratio be used to show morphological complexity of languages?" In: *Journal of Quantitative Linguistics* 21.3 (2014), pp. 223–245 (cit. on p. 72).

[104] Daniel Keysers, Nathanael Schärli, Nathan Scales, et al. "Measuring Compositional Generalization: A Comprehensive Method on Realistic Data". In: *International Conference on Learning Representations (ICLR)*. 2020 (cit. on p. 55).

[105] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014) (cit. on p. 95).

[106] Barbara Kitchenham. "Procedures for performing systematic reviews". In: *Keele, UK, Keele University* 33.2004 (2004), pp. 1–26 (cit. on pp. 31, 32).

[107] Gerhard Georg Klager and Axel Polleres. "Is GPT fit for KGQA?–Preliminary Results". In: *Joint Proceedings of TEXT2KG 2023 and BiKE 2023*. Ed. by Sanju Tiwari, Nandana Mihindukulasooriya, Francesco Osborne, et al. 2023, pp. 171–191 (cit. on pp. 2, 25, 62).

[109] Reinhard Kneser and Hermann Ney. "Improved backing-off for M-gram language modeling". In: *1995 International Conference on Acoustics, Speech, and Signal Processing*. Vol. 1. 1995, 181–184 vol.1 (cit. on p. 9).

[110] Vladislav Korablinov and Pavel Braslavski. "RuBQ: a Russian dataset for question answering over Wikidata". In: *International Semantic Web Conference*. Springer. 2020, pp. 97–110 (cit. on pp. 54, 56, 57).

[111] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, et al. "Efficient Memory Management for Large Language Model Serving with PagedAttention". In: *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*. 2023 (cit. on pp. 113, 123).

[113] John Lafferty, Andrew McCallum, Fernando Pereira, et al. "Conditional random fields: Probabilistic models for segmenting and labeling sequence data". In: *Icml*. Vol. 1. 2. Williamstown, MA. 2001, p. 3 (cit. on p. 12).

[114] Jens Lehmann, Dhananjay Bhandiwad, Preetam Gattogi, and Sahar Vahdati. "Beyond boundaries: A human-like approach for question answering over structured and unstructured information sources". In: *Transactions of the Association for Computational Linguistics* 12 (2024), pp. 786–802 (cit. on p. 115).

[115] Bohan Li, Hao Zhou, Junxian He, et al. "On the Sentence Embeddings from Pretrained Language Models". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, Nov. 2020, pp. 9119–9130 (cit. on p. 42).

[116]Yuan Li, Yixuan Zhang, and Lichao Sun. “MetaAgents: Simulating interactions of human behaviors for LLM-based task-oriented coordination via collaborative generative agents”. In: *CoRR* abs/2310.06500 (2023). arXiv: 2310.06500 (cit. on p. 105).

[117]Ji Lin, Jiaming Tang, Haotian Tang, et al. “AWQ: Activation-aware Weight Quantization for On-Device LLM Compression and Acceleration”. In: *Proceedings of Machine Learning and Systems*. Ed. by P. Gibbons, G. Pekhimenko, and C. De Sa. Vol. 6. 2024, pp. 87–100 (cit. on p. 113).

[118]Ekaterina Loginova, Stalin Varanasi, and Günter Neumann. “Towards End-to-End Multilingual Question Answering”. In: *Information Systems Frontiers (ISF)* 22 (Mar. 2020), pp. 1–14 (cit. on pp. 54, 59).

[119]Yikang Lu, A. Aleta, Chunpeng Du, Lei Shi, and Yamir Moreno. “LLMs and generative agent-based models for complex systems research.” In: *Physics of life reviews* 51 (2024), pp. 283–293 (cit. on p. 17).

[120]Yun Luo, Zhen Yang, Fandong Meng, et al. “An empirical study of catastrophic forgetting in large language models during continual fine-tuning”. In: *arXiv preprint arXiv:2308.08747* (2023) (cit. on p. 105).

[121]Alexandre Magueresse, Vincent Carles, and Evan Heetderks. “Low-resource Languages: A Review of Past Work and Future Challenges”. In: *ArXiv* abs/2006.07264 (2020) (cit. on p. 17).

[122]Frank Manola, Eric Miller, Brian McBride, et al. “RDF primer”. In: *W3C recommendation* 10.1-107 (2004), p. 6 (cit. on p. 2).

[123]Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. “Generating Typed Dependency Parses from Phrase Structure Parses”. In: *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*. Ed. by Nicoletta Calzolari, Khalid Choukri, Aldo Gangemi, et al. Genoa, Italy: European Language Resources Association (ELRA), May 2006 (cit. on p. 42).

[124]Andrew McCallum and Wei Li. “Early results for Named Entity Recognition with Conditional Random Fields, Feature Induction and Web-Enhanced Lexicons”. In: *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*. 2003, pp. 188–191 (cit. on p. 12).

[125]Nick McKenna and Priyanka Sen. “KGQA without retraining”. In: *ACL 2023 Workshop on SustaiNLP*. 2023 (cit. on pp. 2, 25, 89).

[126]Grégoire Mialon, Roberto Dessi, Maria Lomeli, et al. “Augmented Language Models: a Survey”. In: *Transactions on Machine Learning Research* (2023) (cit. on pp. 17, 107).

[127]Paula Mian, Tayana Conte, Ana Natali, Jorge Biolchini, and Guilherme Travassos. “A systematic review process for software engineering”. In: *ESELAW'05: 2nd Experimental Software Engineering Latin American Workshop*. 2005 (cit. on pp. 31, 32).

[128]Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. “Efficient Estimation of Word Representations in Vector Space”. In: *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*. 2013 (cit. on p. 16).

[129]Iris Miliaraki, Roi Blanco, and Mounia Lalmas. “From "Selena Gomez" to "Marlon Brando": Understanding Explorative Entity Search”. In: *Proceedings of the 24th International Conference on World Wide Web*. WWW ’15. Florence, Italy: International World Wide Web Conferences Steering Committee, 2015, 765–775 (cit. on p. 3).

[130]Justin J Miller. “Graph database applications and concepts with Neo4j”. In: *Proceedings of the southern association for information systems conference, Atlanta, GA, USA*. Vol. 2324. 36. 2013, pp. 141–147 (cit. on p. 23).

[131]Weiqing Min, Chunlin Liu, Leyi Xu, and Shuqiang Jiang. “Applications of knowledge graphs for food science and industry”. In: *Patterns* 3.5 (2022) (cit. on pp. 23, 24).

[132]Marc Miquel-Ribé and David Laniado. “Wikipedia culture gap: quantifying content imbalances across 40 language editions”. In: *Frontiers in physics* 6 (2018), p. 54 (cit. on p. 3).

[133]Andrea Moro, Alessandro Raganato, and Roberto Navigli. “Entity Linking meets Word Sense Disambiguation: a Unified Approach”. In: *Transactions of the Association for Computational Linguistics* 2 (2014), pp. 231–244 (cit. on p. 45).

[134]Michalis Mountantonakis, Michalis Bastakis, Loukas Mertzanis, and Yannis Tzitzikas. “Tiresias: Bilingual Question Answering over DBpedia”. In: *Workshop at ISWC 2022 on Deep Learning for Knowledge Graphs*. CEUR. 2022 (cit. on pp. 46, 51).

[135]Diego Moussallem, Mihael Arčan, Axel-Cyrille Ngonga Ngomo, and Paul Buitelaar. “Augmenting neural machine translation with knowledge graphs”. In: *arXiv preprint arXiv:1902.08816* (2019) (cit. on p. 162).

[136]Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. “MTEB: Massive Text Embedding Benchmark”. In: *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*. Ed. by Andreas Vlachos and Isabelle Augenstein. Dubrovnik, Croatia: Association for Computational Linguistics, May 2023, pp. 2014–2037 (cit. on p. 113).

[137]A. Neves, Andre Lamurias, and F. Couto. “Biomedical Question Answering using Extreme Multi-Label Classification and Ontologies in the Multilingual Panorama”. In: *Semantic Indexing and Information Retrieval for Health Held in conjunction with the 42nd European Conference on Information Retrieval (SIIRH@ECIR)*. 2020 (cit. on p. 54).

[138]Joakim Nivre, Marie-Catherine De Marneffe, Filip Ginter, et al. “Universal dependencies v1: A multilingual treebank collection”. In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC’16)*. 2016, pp. 1659–1666 (cit. on p. 43).

[139]Joakim Nivre, Daniel Zeman, Filip Ginter, and Francis Tyers. “Universal Dependencies”. In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Tutorial Abstracts*. Valencia, Spain: Association for Computational Linguistics, Apr. 2017 (cit. on pp. 43, 44).

[140]Eran Ofek, Benedict A. Gomes, Tal Cohen, et al. “Providing answer boxes based on query results”. Pat. United States Patent. Mar. 2017 (cit. on p. 1).

[141]Reham Omar, Ishika Dhall, Panos Kalnis, and Essam Mansour. “A universal question-answering platform for knowledge graphs”. In: *Proceedings of the ACM on Management of Data* 1.1 (2023), pp. 1–25 (cit. on p. 115).

[142]OpenAI. “GPT-4 Technical Report”. In: *arXiv preprint arXiv:2303.08774* (2023). arXiv: 2303.08774 [cs.CL] (cit. on p. 90).

[144]Matthew J Page, Joanne E McKenzie, Patrick M Bossuyt, et al. “The PRISMA 2020 statement: an updated guideline for reporting systematic reviews”. In: *Systematic reviews* 10.1 (2021), pp. 1–11 (cit. on p. 39).

[145]Rrubaa Panchendrarajan and Aravindh Amaresan. “Bidirectional LSTM-CRF for named entity recognition”. In: 32nd Pacific Asia Conference on Language, Information and Computation. 2018 (cit. on p. 12).

[146]Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. “BLEU: a method for automatic evaluation of machine translation”. In: *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. 2002, pp. 311–318 (cit. on p. 79).

[147]Thomas Pellissier Tanon, Marcos Dias de Assunção, Eddy Caron, and Fabian M. Suchanek. “Demoing Platypus – A Multilingual Question Answering Platform for Wikidata”. In: *The Semantic Web: ESWC 2018 Satellite Events*. Ed. by Aldo Gangemi, Anna Lisa Gentile, Andrea Giovanni Nuzzolese, et al. Cham: Springer International Publishing, 2018, pp. 111–116 (cit. on pp. 45, 51, 75, 114).

[148]Jeffrey Pennington, Richard Socher, and Christopher Manning. “GloVe: Global Vectors for Word Representation”. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Ed. by Alessandro Moschitti, Bo Pang, and Walter Daelemans. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1532–1543 (cit. on p. 16).

[149]Arnaldo Pereira, João Rafael Almeida, Rui Pedro Lopes, and José Luís Oliveira. “Querying semantic catalogues of biomedical databases”. In: *Journal of Biomedical Informatics* 137 (2023), p. 104272 (cit. on p. 3).

[150]Arnaldo Pereira, Alina Trifan, Rui Pedro Lopes, and José Luís Oliveira. “Systematic review of question answering over knowledge bases”. In: *IET Software* 16.1 (2022), pp. 1–13 (cit. on pp. 31, 32, 37, 39, 161).

[151]Aleksandr Perevalov, Sara Abdollahi, Simon Gottschalk, and Andreas Both. “Evaluating Entity Importance in a Cross-National Context using Crowdsourcing and Best–Worst Scaling”. In: *Procedia Computer Science* 246 (2024), pp. 1479–1487 (cit. on p. 59).

[152] Aleksandr Perevalov and Andreas Both. “Augmentation-based Answer Type Classification of the SMART dataset.” In: *SMART@ ISWC*. 2020, pp. 1–9 (cit. on p. 14).

[153] Aleksandr Perevalov, Andreas Both, Dennis Diefenbach, and Axel-Cyrille Ngonga Ngomo. “Can Machine Translation Be a Reasonable Alternative for Multilingual Question Answering Systems over Knowledge Graphs?” In: *Proceedings of the ACM Web Conference 2022*. WWW ’22. Virtual Event, Lyon, France: Association for Computing Machinery, 2022, 977–986 (cit. on pp. 49, 51, 59–61, 101).

[154] Aleksandr Perevalov, Andreas Both, and Axel-Cyrille Ngonga Ngomo. “Multilingual question answering systems for knowledge graphs–a survey”. In: *Semantic Web* 15.5 (2024), pp. 2089–2124 (cit. on pp. 3, 37, 50, 97).

[155] Aleksandr Perevalov, Dennis Diefenbach, Ricardo Usbeck, and Andreas Both. “QALD-9-plus: A Multilingual Dataset for Question Answering over DBpedia and Wikidata Translated by Native Speakers”. In: *2022 IEEE 16th International Conference on Semantic Computing (ICSC)*. IEEE. 2022, pp. 229–234 (cit. on pp. 48, 49, 54, 56, 57, 61, 75, 90, 106, 113).

[156] Aleksandr Perevalov, Aleksandr Gashkov, Maria Eltsova, and Andreas Both. “Understanding SPARQL Queries: Are We Already There? Multilingual Natural Language Generation Based on SPARQL Queries and Large Language Models”. In: *The Semantic Web – ISWC 2024*. Ed. by Gianluca Demartini, Katja Hose, Maribel Acosta, et al. Cham: Springer Nature Switzerland, 2025, pp. 173–191 (cit. on p. 25).

[157] Aleksandr Perevalov, Xi Yan, Liubov Kovriguina, et al. “Knowledge Graph Question Answering Leaderboard: A Community Resource to Prevent a Replication Crisis”. In: *Proceedings of the Language Resources and Evaluation Conference*. Marseille, France: European Language Resources Association, 2022, pp. 2998–3007 (cit. on pp. 52, 58, 116).

[158] Carol Peters, Martin Braschler, and Paul Clough. *Multilingual information retrieval: From research to practice*. Springer, 2012, pp. I–XVII, 1–217 (cit. on p. 16).

[159] Yevgen Pikus, Norbert Weißenberg, Bernhard Holtkamp, and Boris Otto. “Semi-Automatic Ontology-Driven Development Documentation: Generating Documents from RDF Data and DITA Templates”. In: *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*. SAC ’19. Limassol, Cyprus: Association for Computing Machinery, 2019, 2293–2302 (cit. on p. 3).

[160] Martin Potthast, Matthias Hagen, and Benno Stein. “The dilemma of the direct answer”. In: *ACM SIGIR Forum*. Vol. 54. 1. ACM New York, NY, USA. 2021, pp. 1–12 (cit. on p. 1).

[161] Amir Pouran Ben Veyseh. “Cross-Lingual Question Answering Using Common Semantic Space”. In: *Proceedings of TextGraphs-10: the Workshop on Graph-based Methods for Natural Language Processing*. San Diego, CA, USA: Association for Computational Linguistics, June 2016, pp. 15–19 (cit. on pp. 45, 51).

[162] Chen Qiu, Guangyou Zhou, Zhihua Cai, and Anders Søgaard. “A Global–Local Attentive Relation Detection Model for Knowledge-Based Question Answering”. In: *IEEE Transactions on Artificial Intelligence* 2.2 (2021), pp. 200–212 (cit. on p. 3).

[163]Nikolay Radoev, Amal Zouaq, and Michel Gagnon. *French and English Question Answering using Lexico-Syntactic Patterns* (cit. on p. 51).

[164]Nikolay Radoev, Amal Zouaq, Mathieu Tremblay, and Michel Gagnon. "A Language Adaptive Method for Question Answering on French and English". In: *Semantic Web Challenges*. Ed. by Davide Buscaldi, Aldo Gangemi, and Diego Reforgiato Recupero. Cham: Springer International Publishing, 2018, pp. 98–113 (cit. on pp. 44, 51, 59).

[165]Surangika Ranathunga, E. Lee, Marjana Prifti Skenduli, et al. "Neural Machine Translation for Low-resource Languages: A Survey". In: *ACM Computing Surveys* 55 (2021), pp. 1 –37 (cit. on p. 17).

[166]Aarne Ranta. "Grammatical framework". In: *Journal of Functional Programming* 14.2 (2004), pp. 145–189 (cit. on p. 44).

[167]Aarne Ranta. "The GF resource grammar library". In: *Linguistic Issues in Language Technology* 2 (2009) (cit. on p. 44).

[168]Siva Reddy, Oscar Täckström, Slav Petrov, Mark Steedman, and Mirella Lapata. "Universal Semantic Parsing". In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark: Association for Computational Linguistics, Sept. 2017, pp. 89–101 (cit. on pp. 43, 51, 58).

[169]Ivan Rybin, Vladislav Korablinov, Pavel Efimov, and Pavel Braslavski. "RuBQ 2.0: an innovated Russian question answering dataset". In: *European Semantic Web Conference*. Springer. 2021, pp. 532–547 (cit. on pp. 30, 47, 54, 56, 61, 78).

[170]Ahmad Sakor, Kuldeep Singh, Anery Patel, and Maria-Esther Vidal. "Falcon 2.0: An entity and relation linking tool over wikidata". In: *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2020, pp. 3141–3148 (cit. on p. 12).

[171]Ahmad Sakor, Kuldeep Singh, Anery Patel, and Maria-Esther Vidal. "Falcon 2.0: An Entity and Relation Linking Tool over Wikidata". In: *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. CIKM '20. Virtual Event, Ireland: Association for Computing Machinery, 2020, 3141–3148 (cit. on p. 114).

[172]Gerard Salton. *The SMART retrieval system—experiments in automatic document processing*. Prentice-Hall, Inc., 1971 (cit. on p. 16).

[173]M Sanguinetti, Maurizio Atzori, N Puddu, et al. "rewordQALD9: A Bilingual Benchmark with Alternative Rewordings of QALD Questions". In: *CEUR WORKSHOP PROCEEDINGS*. Vol. 3235. CEUR-WS. 2022 (cit. on pp. 55, 56).

[174]Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter". In: *arXiv preprint arXiv:1910.01108* (2019) (cit. on p. 90).

[175]Teven Le Scao, Angela Fan, Christopher Akiki, et al. "BLOOM: A 176B-parameter open-access multilingual language model". In: *arXiv preprint arXiv:2211.05100* (2022) (cit. on p. 47).

[176]M. Schuster and K.K. Paliwal. “Bidirectional recurrent neural networks”. In: *IEEE Transactions on Signal Processing* 45.11 (1997), pp. 2673–2681 (cit. on p. 10).

[177]Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. *Introduction to information retrieval*. Vol. 39. Cambridge University Press Cambridge, 2008 (cit. on p. 26).

[178]Priyanka Sen, Alham Fikri Aji, and Amir Saffari. “Mintaka: A Complex, Natural, and Multilingual Dataset for End-to-End Question Answering”. In: *Proceedings of the 29th International Conference on Computational Linguistics*. Gyeongju, Republic of Korea: International Committee on Computational Linguistics, Oct. 2022, pp. 1604–1619 (cit. on pp. 29, 54, 56, 57).

[179]José Wellington Franco da Silva, Amanda Drielly Pires Venceslau, Juliano Efson Sales, et al. “A short survey on end-to-end simple question answering systems”. In: *Artificial Intelligence Review* 53.7 (2020), pp. 5429–5453 (cit. on pp. 37, 38, 161).

[180]Justine Slomian, Olivier Bruyère, Jean-Yves Reginster, and Patrick Emonts. “The Internet as a source of information used by women after childbirth to meet their need for information: A web-based survey”. In: *Midwifery* 48 (2017), pp. 46–52 (cit. on p. 3).

[181]Daniil Sorokin and Iryna Gurevych. “Modeling semantics with gated graph neural networks for knowledge base question answering”. In: *arXiv preprint arXiv:1808.04126* (2018) (cit. on p. 55).

[182]Tommaso Soru, Edgard Marx, Diego Moussallem, et al. “SPARQL as a Foreign Language.” In: *SEMANTiCS (Posters & Demos)*. 2017 (cit. on p. 26).

[183]Tommaso Soru, Edgard Marx, André Valdestilhas, et al. “Neural machine translation for query construction and composition”. In: *arXiv preprint arXiv:1806.10478* (2018) (cit. on p. 26).

[184]Javier Soruco, Diego Collarana, Andreas Both, and Ricardo Usbeck. “QALD-9-ES: A Spanish Dataset for Question Answering Systems”. In: *Knowledge Graphs: Semantics, Machine Learning, and Languages*. IOS Press, 2023, pp. 38–52 (cit. on pp. 55, 113).

[185]Fábio Souza, Rodrigo Nogueira, and Roberto Lotufo. “Portuguese named entity recognition using BERT-CRF”. In: *arXiv preprint arXiv:1909.10649* (2019) (cit. on p. 12).

[186]Tarcísio Souza Costa, Simon Gottschalk, and Elena Demidova. “Event-QA: A Dataset for Event-Centric Question Answering over Knowledge Graphs”. In: *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. CIKM ’20. Virtual Event, Ireland: Association for Computing Machinery, 2020, 3157–3164 (cit. on pp. 54, 56).

[187]Nikit Srivastava, Mengshi Ma, Daniel Vollmers, et al. “MST5–Multilingual Question Answering over Knowledge Graphs”. In: *arXiv preprint arXiv:2407.06041* (2024) (cit. on pp. 48, 51, 105, 115).

[188]Nikit Srivastava, Aleksandr Perevalov, Denis Kuchelev, et al. “Lingua Franca – Entity-Aware Machine Translation Approach for Question Answering over Knowledge Graphs”. In: *Proceedings of the 12th Knowledge Capture Conference 2023*. K-CAP ’23. Pensacola, FL, USA: Association for Computing Machinery, 2023, 122–130 (cit. on pp. 49, 51, 59, 61, 106, 116, 117).

[189]Artur Strzelecki and Paulina Rutecka. “Direct answers in Google search results”. In: *IEEE Access* 8 (2020), pp. 103642–103654 (cit. on p. 1).

[190]Ilya Sutskever, Oriol Vinyals, and Quoc V Le. “Sequence to sequence learning with neural networks”. In: *Advances in neural information processing systems* 27 (2014) (cit. on p. 11).

[191]Yiming Tan, Yongrui Chen, Guilin Qi, Weizhuo Li, and Meng Wang. “MLPQ: A Dataset for Path Question Answering over Multilingual Knowledge Graphs”. In: *Big Data Research* 32 (2023), p. 100381 (cit. on pp. 48, 54, 56).

[192]Yiming Tan, Xinyu Zhang, Yongrui Chen, et al. “CLRN: A reasoning network for multi-relation question answering over Cross-lingual Knowledge Graphs”. In: *Expert Systems with Applications* 231 (2023), p. 120721 (cit. on pp. 48, 51, 59).

[193]Yi Tay, Mostafa Dehghani, Vinh Q Tran, et al. “Ul2: Unifying language learning paradigms”. In: *arXiv preprint arXiv:2205.05131* (2022) (cit. on p. 15).

[194]Ann Taylor, Mitchell Marcus, and Beatrice Santorini. “The Penn treebank: an overview”. In: *Treebanks* (2003), pp. 5–22 (cit. on p. 44).

[195]Simone Tedeschi, Simone Conia, Francesco Cecconi, and Roberto Navigli. “Named Entity Recognition for Entity Linking: What works and what’s next”. In: *Findings of the Association for Computational Linguistics: EMNLP 2021*. 2021, pp. 2584–2596 (cit. on p. 12).

[196]Simone Tedeschi, Valentino Maiorca, Niccolò Campolungo, Francesco Cecconi, and Roberto Navigli. “WikiNEuRal: Combined Neural and Knowledge-based Silver Data Creation for Multilingual NER”. In: *Findings of the Association for Computational Linguistics: EMNLP 2021*. Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 2521–2533 (cit. on p. 116).

[197]Jörg Tiedemann and Santhosh Thottingal. “OPUS-MT — Building open translation services for the World”. In: *Proceedings of the 22nd Annual Conferenec of the European Association for Machine Translation (EAMT)*. Lisbon, Portugal, 2020 (cit. on pp. 49, 78, 116, 163).

[198]Kai Ming Ting. “Precision and Recall”. In: *Encyclopedia of Machine Learning*. Ed. by Claude Sammut and Geoffrey I. Webb. Boston, MA: Springer, 2010, pp. 781–781 (cit. on p. 79).

[199]Priyansh Trivedi, Gaurav Maheshwari, Mohnish Dubey, and Jens Lehmann. “LC-QuAD: A corpus for complex question answering over knowledge graphs”. In: *International Semantic Web Conference*. Springer. 2017, pp. 210–218 (cit. on pp. 39, 45–47).

[200] Lewis Tunstall and et al. “Zephyr: Direct distillation of LM alignment”. In: *arXiv preprint arXiv:2310.16944* (2023) (cit. on p. 90).

[201] Raushan Turganbay, Viacheslav Surkov, Dmitry Evseev, and Mikhail Drobyshevskiy. “Generative Question Answering Systems over Knowledge Graphs and Text”. In: vol. 22. 2023, pp. 1112–1126 (cit. on pp. 26, 47, 51, 105, 114).

[202] Christina Unger, Axel-Cyrille Ngonga Ngomo, and Elena Cabrio. “6th open challenge on question answering over linked data (QALD-6)”. In: *Semantic Web Challenges: Third SemWebEval Challenge at ESWC 2016, Heraklion, Crete, Greece, May 29-June 2, 2016, Revised Selected Papers 3*. Springer. 2016, pp. 171–177 (cit. on p. 56).

[203] Ricardo Usbeck, Ria Hari Gusmita, Axel-Cyrille Ngonga Ngomo, and Muhammad Saleem. “9th Challenge on Question Answering over Linked Data (QALD-9)”. In: *Joint proceedings of the 4th Workshop on Semantic Deep Learning (SemDeep-4) and NLIWoD4: Natural Language Interfaces for the Web of Data (NLIWOD-4) and 9th Question Answering over Linked Data challenge (QALD-9) co-located with 17th International Semantic Web Conference (ISWC 2018), Monterey, California, United States of America, October 8th - 9th, 2018*. 2018, pp. 58–64 (cit. on pp. 38, 46, 54, 56, 61).

[204] Ricardo Usbeck, Axel-Cyrille Ngonga Ngomo, Felix Conrads, Michael Röder, and Giulio Napolitano. “8th challenge on question answering over linked data (QALD-8)”. In: *language* 7.1 (2018), pp. 51–57 (cit. on p. 56).

[205] Ricardo Usbeck, Axel-Cyrille Ngonga Ngomo, Bastian Haarmann, et al. “7th open challenge on question answering over linked data (QALD-7)”. In: *Semantic web evaluation challenge*. Springer. 2017, pp. 59–69 (cit. on pp. 43, 56).

[206] Ricardo Usbeck, Michael Röder, Michael Hoffmann, et al. “Benchmarking question answering systems”. In: *Semantic Web* 10.2 (2019), pp. 293–304 (cit. on pp. 28, 30, 93).

[207] Ricardo Usbeck, Michael Röder, Axel-Cyrille Ngonga Ngomo, et al. “GERBIL: General Entity Annotator Benchmarking Framework”. In: *Proceedings of the 24th International Conference on World Wide Web*. WWW ’15. Florence, Italy: International World Wide Web Conferences Steering Committee, 2015, 1133–1143 (cit. on pp. 72, 77, 78, 93).

[208] Ricardo Usbeck, Xi Yan, Aleksandr Perevalov, et al. “QALD-10 — The 10th Challenge on Question Answering over Linked Data”. In: *Under review in the Semantic Web Journal* (Feb. 2023) (cit. on pp. 48, 55, 56, 73).

[209] Cornelis J Van Rijsbergen. *Information Retrieval, 2nd edn. Newton, MA*. 1979 (cit. on p. 27).

[210] Ashish Vaswani, Noam Shazeer, Niki Parmar, et al. “Attention is all you need”. In: *Advances in neural information processing systems* 30 (2017) (cit. on p. 11).

[211] Denny Vrandečić and Markus Krötzsch. “Wikidata: A Free Collaborative Knowledgebase”. In: *Commun. ACM* 57.10 (Sept. 2014), 78–85 (cit. on pp. 2, 21, 25, 75, 109).

[216]Lei Wang, Chen Ma, Xueyang Feng, et al. “A survey on large language model based autonomous agents”. In: *Frontiers of Computer Science* 18.6 (2024), p. 186345 (cit. on pp. 17, 107).

[217]Liang Wang, Nan Yang, Xiaolong Huang, et al. “Multilingual e5 text embeddings: A technical report”. In: *arXiv preprint arXiv:2402.05672* (2024) (cit. on p. 113).

[218]Wenhan Wang, Ge Li, Bo Ma, Xin Xia, and Zhi Jin. “Detecting Code Clones with Graph Neural Network and Flow-Augmented Abstract Syntax Tree”. In: *2020 IEEE 27th International Conference on Software Analysis, Evolution and Reengineering (SANER)*. 2020, pp. 261–271 (cit. on p. 42).

[219]Xuguang Wang, Linjun Shou, Ming Gong, Nan Duan, and Daxin Jiang. “No Answer is Better Than Wrong Answer: A Reflection Model for Document Level Machine Reading Comprehension”. In: *Findings of the Association for Computational Linguistics: EMNLP 2020*. Ed. by Trevor Cohn, Yulan He, and Yang Liu. Online: Association for Computational Linguistics, Nov. 2020, pp. 4141–4150 (cit. on p. 29).

[220]Ralph Weischedel, Marie Meteer, Richard Schwartz, Lance Ramshaw, and Jeff Palmucci. “Coping with Ambiguity and Unknown Words through Probabilistic Models”. In: *Computational Linguistics* 19.2 (1993). Ed. by Julia Hirschberg, pp. 359–382 (cit. on p. 12).

[221]Thomas Wolf and et al. “Transformers: State-of-the-Art Natural Language Processing”. In: *Empirical Methods in NLP: System Demonstrations*. ACL, 2020, pp. 38–45 (cit. on p. 95).

[222]Shanchan Wu and Yifan He. “Enriching pre-trained language model with entity information for relation classification”. In: *Proceedings of the 28th ACM international conference on information and knowledge management*. 2019, pp. 2361–2364 (cit. on p. 14).

[223]Silei Xu, Theo Culhane, Meng-Hsi Wu, Sina J Semnani, and Monica S Lam. “Complementing GPT-3 with Few-Shot Sequence-to-Sequence Semantic Parsing over Wikidata”. In: *arXiv preprint arXiv:2305.14202* (2023) (cit. on pp. 2, 25, 89).

[224]Linting Xue, Noah Constant, Adam Roberts, et al. “mT5: A massively multilingual pre-trained text-to-text transformer”. In: *arXiv preprint arXiv:2010.11934* (2020) (cit. on pp. 47, 48).

[225]Xiaoyu Yin, Dagmar Gromann, and Sebastian Rudolph. “Neural machine translating from natural language to SPARQL”. In: *Future Generation Computer Systems* 117 (2021), pp. 510–519 (cit. on p. 161).

[226]Mo Yu, Wenpeng Yin, Kazi Saidul Hasan, et al. “Improved neural relation detection for knowledge base question answering”. In: *arXiv preprint arXiv:1704.06194* (2017) (cit. on pp. 13, 14).

[227]Yirui Zhan, Yanzeng Li, Minhao Zhang, and Lei Zou. “ADMUS: A Progressive Question Answering Framework Adaptable to Multiple Knowledge Sources”. In: *arXiv preprint arXiv:2308.04800* (2023) (cit. on p. 63).

[228] Chen Zhang, Yuxuan Lai, Yansong Feng, and Dongyan Zhao. “A review of deep learning in question answering over knowledge bases”. In: *AI Open* (2021) (cit. on pp. 2, 25, 37, 39, 89).

[229] Yusen Zhang, Jun Wang, Zhiguo Wang, and Rui Zhang. “XSemPLR: Cross-Lingual Semantic Parsing in Multiple Natural Languages and Meaning Representations”. In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Toronto, Canada: Association for Computational Linguistics, July 2023, pp. 15918–15947 (cit. on pp. 47, 51, 62).

[230] Yucheng Zhou, Xiubo Geng, Tao Shen, Wenqiang Zhang, and Daxin Jiang. “Improving Zero-Shot Cross-lingual Transfer for Multilingual Question Answering over Knowledge Graph”. In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Online: Association for Computational Linguistics, June 2021, pp. 5822–5834 (cit. on pp. 48, 51, 61).

[231] Elizaveta Zimina, Jyrki Nummenmaa, Kalervo Jarvelin, Jaakko Peltonen, and Kostas Stefanidis. “MuG-QA: Multilingual Grammatical Question Answering for RDF Data”. In: *2018 IEEE International Conference on Progress in Informatics and Computing (PIC)*. 2018, pp. 57–61 (cit. on pp. 44, 51, 61).

[232] Chang Zong, Yuchen Yan, Weiming Lu, et al. “Triad: A Framework Leveraging a Multi-Role LLM-based Agent to Solve Knowledge Base Question Answering”. In: *arXiv preprint arXiv:2402.14320* (2024) (cit. on pp. 25, 105, 115).

Webpages

[@44] Richard Cyganiak, David Wood, and Markus Lanthaler. *RDF 1.1 Concepts and Abstract Syntax*. Ed. by Richard Cyganiak, David Wood, and Markus Lanthaler. W3C Recommendation 25 February 2014. 2014. URL: http://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/ (cit. on p. 19).

[@64] Google. *Freebase Data Dumps*. 2022. URL: https://developers.google.com/freebase/data (cit. on pp. 2, 25).

[@70] Steve Harris and Andy Seaborne. *SPARQL 1.1 Query Language*. Ed. by Eric Prud’hommeaux. W3C Recommendation. W3C. 2013 (cit. on p. 21).

[@80] Holly Young. *The digital language divide*. 2023. URL: https://web.archive.org/web/20230602205007/http://labs.theguardian.com/digital-language-divide/ (cit. on p. 3).

[@86] *Interlanguage-links dataset*. 2021. URL: https://databus.dbpedia.org/dbpedia/generic/interlanguage-links/2021.12.01 (cit. on p. 44).

[@87] Internet Live Stats. *Google Search Statistics*. 2025. URL: https://www.internetlivestats.com/google-search-statistics/ (cit. on p. 1).

[@88]Internet World Stats. *Top Ten Languages Used in the Web*. 2025. URL: https://web.archive.org/web/20240425090921/https://www.internetworldstats.com/stats7.htm (cit. on p. 3).

[@91]Antoine Isaac and Ed Summers, eds. *SKOS Simple Knowledge Organization System Primer*. Document Status Update 2023-10-30. 2009. URL: https://www.w3.org/TR/2009/NOTE-skos-primer-20090818/ (cit. on p. 19).

[@108]Graham Klyne and Jeremy J. Carroll. *Resource Description Framework (RDF): Concepts and Abstract Syntax*. W3C. 2004. URL: http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/ (visited on Mar. 15, 2015) (cit. on p. 163).

[@112]Kyle Wiggers. *Perplexity says it's now serving 100M search queries a week*. 2024. URL: https://techcrunch.com/2024/10/25/perplexity-says-its-now-serving-100m-search-queries-a-week/ (cit. on p. 1).

[@143]OpenAI. *Introducing ChatGPT*. 2022. URL: https://openai.com/blog/chatGPT (cit. on p. 90).

[@212]W3C. *RDF Schema 1.1*. 2014. URL: https://www.w3.org/TR/rdf-schema/ (visited on June 6, 2025) (cit. on p. 19).

[@213]W3C OWL Working Group. *OWL 2 Web Ontology Language Document Overview (Second Edition)*. 2012. URL: http://www.w3.org/TR/2012/REC-owl2-overview-20121211/ (cit. on p. 19).

[@214]W3C Working Group. *RDF 1.2 Concepts and Abstract Syntax)*. Available online: https://www.w3.org/TR/rdf12-concepts/. 2025. URL: https://www.w3.org/TR/rdf12-concepts/ (cit. on p. 19).

[@215]W3Techs. *Usage statistics of content languages for websites*. 2025. URL: https://w3techs.com/technologies/overview/content_language (cit. on p. 3).

List of Figures

List of Tables

A Taxonomy of the Methods to mKGQA

A

We synthesize the previously presented information to provide broader insights into both the systems and their underlying methodologies. This synthesis has led to the development of a comprehensive taxonomy of methods employed in mKGQA system development. While these methodological approaches are applicable to monolingual KGQA systems as well, our focus here specifically emphasizes their multilingual capabilities and applications.

Our taxonomy draws from both our comprehensive review and insights from previous survey articles [77, 179, 150]. It's important to note that not all methods presented operate in an end-to-end fashion - that is, not all directly generate answers or SPARQL queries. Several methods require integration with complementary approaches to form a complete and functional mKGQA system.

The development of KGQA systems fundamentally follows two distinct paradigms: *The first paradigm, known as "Semantic Parsing" or "QA pipelines" [22], employs a sequence of specialized components*. In this approach, questions undergo multi-step processing through various components such as NER, Relation Prediction (REL), Query Builder (QB), and Query Executor (QE). The ultimate objective is to transform natural language questions into executable SPARQL queries. *The second paradigm, termed "end-to-end KGQA", aims to generate answers in a single processing step*. These systems predominantly leverage neural network architectures [32], either for answer candidate ranking or direct question-to-query translation (end-to-end semantic parsing) [225]. Notably, systems following either paradigm may incorporate one or multiple methods from the taxonomy detailed below.

The taxonomy is structured hierarchically, with high-level general groups (designated as "G") encompassing specific low-level methods (marked as "M"):

G1 — *methods based on rules and templates*:

- **M1.1** — *syntax tree parsing* is used to convert NL to a machine-readable syntax tree;
- **M1.2** — *grammar rules* are used to extract structured information from NL with manually defined rules;

M1.3 — *logical representations* are used as a machine-readable intermediate form to represent the semantics of a given NL text;

M1.4 — *dictionaries, indexes, and templates* are used for generating queries or matching entities and relations;

G2 — *statistical methods*:

M2.1 — *classical machine learning and statistical methods* are used for the downstream tasks of KGQA (e.g., NER, REL, etc.);

M2.2 — *deep learning methods* are mainly used in the context of language modeling, graph embedding models, and encoder-decoder architectures;

G3 — *machine translation methods*:

M3.1 — *end-to-end machine translation methods* are used for direct translation of a source language to the target one that is supported by the system;

M3.2 — *enhanced machine translation methods* are used for machine translation with intermediate improvements (e.g., KG enriched [135]) of a source language to the target one that is supported by the system.

The taxonomy is demonstrated in Figure A.1. Importantly, these methods are not mutually exclusive within a single system implementation. Systems often combine multiple methods to achieve optimal performance - for instance, the QAnswer system utilizes both M1.4 and M2.2. For a comprehensive overview of which methods are employed by specific systems, please refer to Table 3.4.

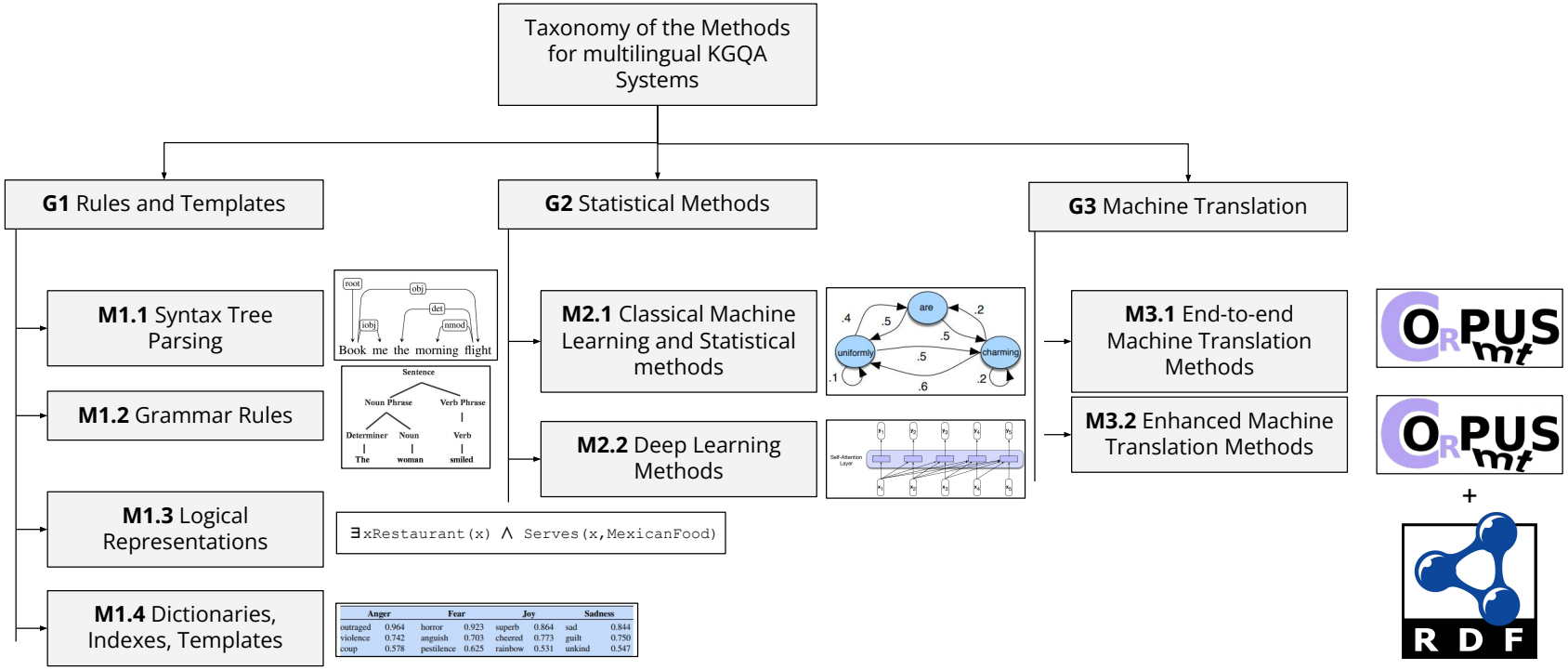


Fig. A.1.: A Comprehensive Taxonomy of Methods for Developing Multilingual KGQA Systems. The method example pictures are taken from [99, @108, 197]. The surveyed systems are classified according to this taxonomy in Table 3.4.