Performance and Limitations of Fine-Tuned LLMs in SPARQL Query Generation

Thamer Mecharnia
LORIA, Université de Lorraine
CNRS, INRIA 54506
Vandœuvre-lès-Nancy, France
thamer.mecharnia@loria.fr

Mathieu d'Aquin
LORIA, Université de Lorraine
CNRS, INRIA 54506
Vandœuvre-lès-Nancy, France
mathieu.daquin@loria.fr

Abstract

Generative AI has simplified information access by enabling natural language-driven interactions between users and automated systems. In particular, Question Answering (QA) has emerged as a key application of AI, facilitating efficient access to complex information through dialogue systems and virtual assistants. The Large Language Models (LLMs) combined with Knowledge Graphs (KGs) have further enhanced QA systems, allowing them to not only correctly interpret natural language but also retrieve precise answers from structured data sources such as Wikidata and DBpedia. However, enabling LLMs to generate machine-readable SPARQL queries from natural language questions (NLQs) remains challenging, particularly for complex questions.

In this study, we present experiments in fine-tuning LLMs for the task of NLQ-to-SPARQL transformation. We rely on benchmark datasets for training and testing the fine-tuned models, generating queries directly from questions written in English (without further processing of the input or output). By conducting an analytical study, we examine the effectiveness of each model, as well as the limitations associated with using fine-tuned LLMs to generate SPARQL.

1 Introduction

In recent years, the interaction between users and automated systems across various domains has become the center of Artificial Intelligence (AI) research. One of the main challenges in this interaction is translating human-written questions into machine-readable formats. This is specifically true for knowledge graphs represented in RDF format. Those contain vast amounts of information on a wide variety of topics, but would generally require a user to write a query in the SPARQL language to use them to answer specific questions. Such queries provide accurate answers from reliable and structured data sources. However, they are accessible only to people with SPARQL knowledge and the time to formulate such queries. This is a significant barrier to the accessibility of information embedded in KGs.

This study explores how LLMs, when fine-tuned, can generate accurate SPARQL queries from NLQ, allowing direct human-friendly interaction with KGs such as DBpedia[1] and Wikidata[2], thus giving access to complicated systems to a wider range of people, including non-specialists. By investigating a range of LLMs, we aim to identify the conditions under which these models produce accurate SPARQL queries, while highlighting their limitations. These limitations include syntactic errors in generated queries, hallucinated identifiers, etc. Our objective is to better understand the capabilities of different LLMs in this task, and to identify potential areas of improvement for future research.

The remainder of this paper is organized as follows. In Section 2, we present related works on QA and LLM development. Then, in Section 3, we present our analytic study and the fine-tuning process of various Meta's LLaMA models to get a new model specialized on transforming NLQ to SPARQL queries, called Llama-KGQA (Llama based model for Knowledge Graph Question Answering). Section 4 presents our results, comparing the fine-tuned models with each other, as well as with other existing QA systems. Finally, Section 5 summarizes our findings and outlines future perspectives to enhance Llama-KGQA's capabilities.

2 Related Works

Question-answering is a branch of Natural Language Processing (NLP) that aims to automatically respond to user questions asked in natural language.

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

Proceedings of the Generative AI and Knowledge Graph Workshop (GenAIK), pages 69–77
January 19, 2025.

The goal of QA systems is to provide precise and contextually relevant answers to a wide range of questions by accessing various data sources, such as unstructured text (see, for example, (Nassiri and Akhloufi, 2023)), structured databases (see, for example, (Khanam and Subbareddy, 2017)), or knowledge graphs (see, for example, (Pramanik et al., 2024)). QA systems have become central to many applications, including search engines, intelligent virtual assistants (e.g. Siri, Alexa, etc.), and customer support chatbots.

QA systems are typically classified according to the type of data on which they rely to answer questions: text-based QA systems (TBQA) and knowledge-based QA systems (KBQA). The TBQA systems extract answers from large collections of unstructured or semi-structured text, such as documents, web pages, or research papers. They involve tasks such as document retrieval and answer extraction, relying heavily on NLP techniques such as information retrieval (see, for example, (Arbaaeen and Shah, 2020; Abbasiantaeb and Momtazi, 2021; Otegi et al., 2022), text classification (see, for example, (Fields et al., 2024)), and semantic matching (see, for example, (Zhang et al., 2019)). KBQA systems utilize structured data sources, such as knowledge bases or knowledge graphs, where information is stored in a highly organized manner (such as RDF data). These systems interpret user queries given as NLQ and translate them into formal queries (e.g. SPARQL for RDF-based knowledge graphs) that can directly retrieve factual answers from the knowledge base.

In this paper, we focus specifically on KBQA systems that rely on knowledge graphs. Those systems are discussed later in this section.

Large Language Models (LLMs) have significantly advanced the field of QA by enhancing the ability of AI assistance and chatbots to comprehend and generate natural language responses. LLMs such as GPT (Achiam et al., 2023), BERT (Devlin et al., 2019), Mixtral (Jiang et al., 2024), and Meta-Llama Models (AI@Meta, 2024) are pre-trained on massive datasets that include diverse text sources such as books, web pages, and scientific articles. This extensive pre-training enables LLMs to internalize vast amounts of general knowledge and linguistic structures, allowing them to respond to open-domain questions across various fields with minimal task-specific training. Unlike traditional QA systems that rely on explicit query-to-answer mappings or structured knowledge bases, LLMs can generate nuanced, context-aware responses by leveraging their pre-trained language understanding models.

However, LLMs also face challenges, such as actual generated errors or "hallucination" (Min et al., 2023), where the models generate plausible but incorrect answers due to their reliance on learned patterns rather than factual verification. Despite these challenges, LLM-based QA systems are at the forefront of NLP, offering robust capabilities for applications in virtual assistants, search engines, and more.

Among the research questions that have arisen in recent years is the possibility for LLMs to efficiently generate machine-readable queries from questions written in natural language to interrogate information sources. For knowledge graphs, this involves transforming a question posed (for example) in English into a valid SPARQL queries to a specified KG (Khorashadizadeh et al., 2024). This could significantly improve linking human language with machine-readable data stores, such as KGs. This capability is crucial because KGs, such as DBpedia and Wikidata, store vast amounts of structured information that can be accessed through SPARQL queries. By enabling LLMs to automatically convert user questions into SPARQL, QA systems can provide accurate and rich responses by tapping directly into these vast repositories. Furthermore, automating this process would reduce the need for defining the query manually, which will improve the accessibility to complex data for non-expert users.

This line of research contributes to the development of Knowledge Graph Question Answering (KGQA) systems, which utilize KG to retrieve answers from structured data repositories. To assess the performance of these systems, a KGQA leaderboard has been established, as presented by the authors in (Perevalov et al., 2022c), which allows the evaluation of KGQA systems using benchmark datasets. In this context, several benchmarking frameworks have been proposed, such as GERBIL QA (Usbeck et al., 2019), which is designed to evaluate KGQA systems in a comprehensive way. Among the widely used datasets for KGQA system evaluation, the Question Answering over Linked Data (QALD) dataset series is considered a standard. In particular, the QALD

challenge[3] was launched to compare KGQA systems on various benchmarks, including QALD-9-plus (Perevalov et al., 2022b) and QALD-10 (Usbeck et al., 2023). Furthermore, detailed evaluations tracking the progress of KGQA systems are available through the QALD leaderboard[4], providing valuable insights into the evolution of these systems.

3 The analytic study

In this analytic study, we fine-tuned several LLMs, all based on the Llama architecture, including Llama-3-8b[5], Llama-2-7b[6], Llama-3-70b[7], and Mixtral-8x7b[8], to evaluate their performance in generating SPARQL queries from NLQ and compare the results with existing similar systems reported in the KGQA leaderboard[9] and in (Perevalov et al., 2022b). The reason to choose Llama-based LLMs is both their availability, so they could be downloaded and fine-tuned locally, and their relative high performance in NLP related tasks.

The models were trained and tested against two KGs, DBpedia and Wikidata, providing a comprehensive comparison of their capabilities. The latest KGQA benchmark datasets for these KGs are provided in QALD-9-plus[10] and QALD-10[11] respectively.

The proposed method fine-tunes an LLM such as Meta-Llama and MistralAI-Mixtral to transform NLQ into SPARQL queries. The objective is to create a robust NLQ-to-SPARQL transformation system capable of querying complex KGs accurately with minimal human input.

Since we, at this stage, only focus on questions in English, the first step involves filtering training datasets that pair NLQs with their corresponding SPARQL queries, retaining only English-language entries. These datasets, including benchmarks like QALD-9-plus and QALD-10, offer rich annotations for NLQ-to-SPARQL transformations.

Since these LLMs are pre-trained on general language modeling tasks, the next step involves fine-tuning them on the filtered NLQ-SPARQL dataset. The fine-tuning follows a sequence-to-sequence learning paradigm, where the model takes a natural language question as input and generates the corresponding SPARQL query as output. The model learns to align the structure of the NLQ with the syntax of SPARQL queries. To enhance the efficiency of the fine-tuning process, we used the Parameter-Efficient Fine-Tuning[12] (PEFT) library, along with the LoRA technique (Hu et al., 2021), which adapts pre-trained models by fine-tuning only an additional subset of parameters (the adapters). In this LoRA configuration, we have set the lower-rank matrices of the adapter at 16 to save memory and reduce computational cost by training fewer parameters. We also have set the scaling factor for the low-rank matrices at 32 to scale up the impact of adapters to help the model learn the task-specific adjustments more effectively. In order to prevent overfitting, the dropout probability for LoRA layers is set at 0.05. We have also specified the task of the model fine-tuning as "causal language modeling", so the model is trained to predict the next word in a sequence. In this configuration, LoRA is enabled only to adapt attention mechanisms ("k_proj", "q_proj", "v_proj", and "o_proj") and feed-forward layers ("up_proj" and "down_proj"). By selectively adapting only these modules, we focus on the parts of the model most relevant to language generation while preserving computational efficiency. In this configuration, no additional bias terms are learned in the adapters, i.e. the bias is set at "none". This simplifies the structure of the model.

This approach reduces memory consumption and accelerates the fine-tuning process without compromising on model performance.

Once trained, the model performance is validated on a testing set of NLQ-SPARQL pairs. During the testing phase, an execution correctness cycle is applied over 10 attempts, i.e. if a generated SPARQL query contains syntactic errors, the same NLQ is re-processed to generate a different query, thereby improving the chances of generating a valid query.

[3] https://www.nliwod.org/challenge

[4] https://github.com/KGQA/leaderboard?tab=readme-ov-file

[5] https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct

[6] https://huggingface.co/togethercomputer/Llama-2-7B-32K-Instruct

[7] https://huggingface.co/meta-llama/Meta-Llama-3-70B-Instruct

[8] https://huggingface.co/mistralai/Mixtral-8x7B-Instruct-v0.1

[9] https://github.com/KGQA/leaderboard?tab=readme-ov-file

[10] https://github.com/KGQA/QALD_9_plus/tree/main/data

[11] https://github.com/KGQA/QALD-10

[12] https://huggingface.co/docs/peft/main/en/index

4 Experiment

All codes were written in Python using the Huggingface Transformer library[13]. The models were trained and tested on two NVIDIA RTX A6000 GPUs with 48 GB GDDR6 memory. During the fine-tuning and the testing phases, we used DBpedia and Wikidata QALD datasets, which both include a training set and a testing set. DBpedia benchmark version is provided in QALD-9-plus as a set of question-query pairs. The question is formulated in many languages, including English that we have used, and the query is the SPARQL translation of the question. QALD-9-plus DBpedia training set contains 408 question-query pairs, and its testing set contains 150 pairs. Wikidata is provided in QALD-10 with a training set that contains 412 pairs and a testing set that contains 395 pairs.

The experimental results, including detailed outputs for various models tested on QALD-9-plus and QALD-10 datasets, are available in the Llama-KGQA GitHub repository[14].

Comparison between LLMs

In this comparison, we evaluate the performance of four LLMs: Llama-3-8b, Llama-2-7b, Mixtral-7b, and Llama-3-70b on the latest DBpedia benchmark version provided in the QALD-9-plus dataset; this benchmark is designed for question-answering over DBpedia KG. We used GERBIL QA metrics[15] to evaluate the accuracy of the models. In this evaluation, we used the micro as well as the macro version of precision (Equation 1), recall (Equation 2), and F-measure (Equation 3), in addition of QALD-specific Macro F1 metric. In all of those metrics, the items considered are the individual responses to SPARQL queries. In other words, the best result is obtained when the generated query gives exactly the same set of answers as the one in the gold standard.

$$Precision = \frac{TP}{TP + FP} \quad (1)$$

$$Recall = \frac{TP}{TP + FN} \quad (2)$$

$$\text{F1-measure} = 2 \cdot \frac{Precision \times Recall}{Precision + Recall} \quad (3)$$

For micro measures, they are calculated based on the overall counts of true positives (TP), false negatives (FN), and false positives (FP) across all queries, without considering the individual predictions for each query. However, the macro measures are calculated for each query, and returns the average.

The additional Macro F1 QALD metric, which is usually used to compare the models in QALD challenge, is calculated differently. The Macro F1 QALD metric builds upon Equation 3, incorporating additional semantic information as described in (Usbeck et al., 2019); if the golden answer is not empty but the generated query retrieves an empty answer, it is assumed that the model cannot generate a correct query. So, the precision of this query is set to 1 and its recall and F-measure are set to 0.

We fine-tuned each model with different epoch settings: 2, 4, 6, 8, and 10. During the testing phase, we made 5 runs per epoch upon the same testing set, i.e. we asked the model to generate SPARQL queries from the same NLQs in the testing set 5 times. Therefore, we performed 25 runs and used the GERBIL QA tool[16] version 0.2.5 to evaluate the results of each run and calculate the average Macro F1 QALD of the 5 runs per epoch. Table 1 shows that Llama-3-8b with 6 epochs achieves the highest Macro F1 score, demonstrating superior performance in generating SPARQL queries for DBpedia KG. In most other epoch settings (4, 8, and 10), Llama-3-70b has obtained slightly better results than Llama-3-8b. Although this can simply be attributed to Llama-3-70b being a larger model, which enables better handling of intricate language structures and knowledge graph queries, the differences in performance between the two remain small. Llama-2-7b and Mixtral-7b show competitive performance, although with slightly lower Macro F1 scores than Llama-3-8b, indicating an effective yet more limited capacity for precision and recall in this task. This comparison highlights the trade-off between model size and performance, particularly in knowledge-intensive tasks such as KGQA. As a result, we chose Llama-3-8b with 6 epochs as the base model to fine-tune for Llama-KGQA.

Table 2 shows the detailed results obtained by the representative run (the run with the closest Macro F1 QALD to the average Macro F1 QALD) of the Llama-KGQA model that obtained the best results

[13] https://huggingface.co/docs/transformers/

[14] https://github.com/ThamerMECHARNIA/Llama-KGQA.

[15] https://github.com/dice-group/gerbil/wiki/Precision,-Recall-and-F1-measure

[16] https://gerbil-qa.aksw.org/gerbil/

	Epoch	Llama-3-8b	Llama-2-7b	Mixtral-7b	Llama-3-70b
Average Macro F1 QALD	2	**52.89%**	49.78%	50.64%	52.57%
	4	56.34%	55.66%	54.73%	**57.75%**
	6	**60.65%**	57.19%	54.88%	58.78%
	8	57.64%	55.95%	55.64%	**59.86%**
	10	57.79%	58.03%	57.78%	**58.38%**

Table 1: Average Macro F1 QALD of Llama-3-8b, Llama-2-7b, Mixtral-7b, and Llama-3-70b on QALD-9-plus DBpedia dataset.

(fine-tuned Llama-3-8b with 6 epochs), as trained and tested on the DBpedia KG. These results are published in GERBIL QA[17].

The experiments indicated that, on average, the model generated SPARQL queries with syntactic errors for approximately 3 questions in this run (the one shown in Table 2) out of 150 questions asked. This represents a 2% error rate. However, prompting the model with the same question again led to successful error correction, yielding a valid SPARQL query within just one additional attempt. This suggests that such errors are rare cases where the model randomly failed to generate a valid SPARQL query, since additional attempts consistently led to correct queries. This iterative querying approach therefore offers a practical solution to improving the accuracy of the LLM-based NLQ-to-SPARQL models.

Comparison with other QA models

We have also compared our results to existing KGQA systems using DBpedia and Wikidata as KG. The results of these models are performed using QALD-9-plus dataset benchmarks for DBpedia and QALD-10 for Wikidata. All results are reported in the QALD leaderboard. The training and the testing sets are both using English questions only for all systems.

Table 3 compares the results obtained by Llama-KGQA, with QAnswer (Diefenbach et al., 2020), DeepPavlov (Burtsev et al., 2018), and Platypus (Pellissier Tanon et al., 2018) using the QALD-9-plus DBpedia dataset. The results of these models are reported in (Perevalov et al., 2022a). We notice that our fine-tuned Llama-3-8b significantly outperforms the top systems of the leaderboard, obtaining 60.68% vs 30.39% of QAnswer. This is a surprising result considering the relatively low effort required to fine-tune Llama3-8b to achieve it. However, this is probably explained by the fact that DBpedia is a well-known resource derived from Wikipedia and using human-readable identifiers. In other words, Llama3-8b likely already had a strong ability to relate to the content of DBpedia from its pre-training, on which the fine-tuning process could rely.

Table 4 compares Llama-KGQA that is fine-tuned this time with QALD-10[18], with the results reported in (Usbeck et al., 2023) that were obtained by (Borroto et al., 2022), QAnswer (Shivashankar et al., 2022), (Baramiia et al., 2022), Gavrilev et al.[19]. This comparison uses the Wikidata dataset in QALD-10. This table shows that Llama-KGQA struggled with Wikidata and only obtained 13.36% Macro F1 QALD. This low performance refers to Wikidata queries in the dataset that use entity identifiers instead of named entities (property and individual names). In other words, our model not having access to or a way to actually query the KG, it could not accurately generate SPARQL queries with valid identifiers in DBpedia. In fact, it would often hallucinate them.

Runtime analysis

This study aims to assess not only accuracy but also the trade-offs in computational efficiency and scalability. To evaluate the efficiency of model fine-tuning, we tracked the training process with Weights & Biases[20] in order to generate the run history and summary. The performance metrics of the model are shown in Figures 1, 2, and 3.

Figure 1 shows the training loss graph of the model with the run that fine-tunes Llama-KGQA. Since the model is configured for causal language model task, it is fine-tuned with the cross-entropy loss function. The X-axis of this graph represents the training steps, each step on this axis reflects a single update to the model parameters during train-

[17] https://gerbil-qa.aksw.org/gerbil/experiment?id=202410290002

[18] https://gerbil-qa.aksw.org/gerbil/experiment?id=202410290003

[19] Their findings are reported in (Usbeck et al., 2023)

[20] https://wandb.ai/site

	Micro F1	Micro Precision	Micro Recall	Macro F1	Macro Precision	Macro Recall	Macro F1 QALD
Llama-KGQA	18.97%	20.00%	18.04%	45.34%	45.82%	46.93%	60.68%

Table 2: Detailed GERBIL QA results for Llama-KGQA.

Model / System	Year	Macro Precision	Macro Recall	Macro F1 QALD
Llama-KGQA	2024	**45.82%**	**46.93%**	**60.68%**
QAnswer	2022	-	-	30.39%
DeepPavlov	2022	-	-	12.40%
Platypus	2022	-	-	15.03%

Table 3: Comparison between Llama-KGQA and QAnswer, DeepPavlov, and Platypus using QALD-9-plus DBpedia benchmarking dataset.

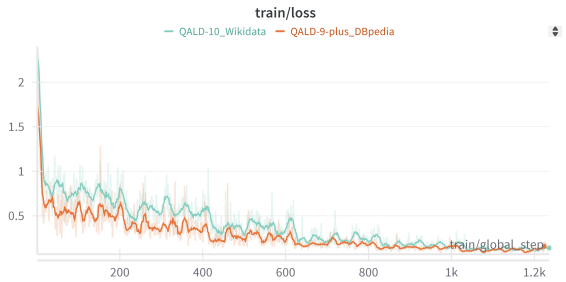

Figure 1: The training loss of Llama-KGQA on QALD-9-plus and QALD-10.

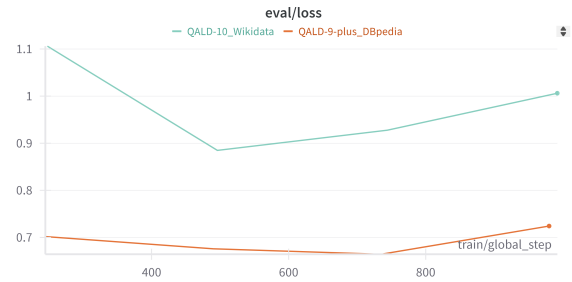

Figure 2: The evaluation loss of Llama-KGQA on QALD-9-plus and QALD-10.

ing, so as the number of steps increases, the model iteratively learns from the training data. The Loss is used in this graph to measure the performance of the model at each step of the training process, i.e. it quantifies the difference between the model's predictions and the actual target values in the training dataset. This graph contains 2 curves for training the model on QALD-9-plus and QALD-10. We notice that in both cases, both curves show a consistent downward trend in losses, suggesting that the model is learning effectively.

Figure 2 shows the evaluation loss (test loss) graph of the same model as Figure 1 (Llama-KGQA) to evaluate it. The evaluation loss is calculated on a separate validation dataset (the testing set) in order to indicate how well the model generalizes to new inputs and to help in monitoring overfitting. We notice in the QALD-10 curve that the loss starts to increase while the training loss in Figure 1 continues to decrease, which means that the model struggles to generate good predictions for the testing data. This overfitting is explained by the fact that the model cannot find the correct entity identifiers from Wikidata because it has no context that incorporates the KG.

The curves of QALD-9-plus in both graphs (training loss and evaluation loss) have a smaller loss than the curves of QALD-10, and the margin becomes bigger in the evaluation loss, which refers to better performance in both training and -especially- testing. This is explained by the model struggling with Wikidata identifiers used in the queries in the training set and the testing set.

Figure 3 shows the utilization of the GPU process and its allocated memory graphs during model fine-tuning. We notice that the memory allocation and the GPU utilization were higher when fine-tuning the model using QALD-9-plus compared to QALD-10. It also takes longer for the model to train with QALD-9-plus (1171s) compared to QALD-10 (1005s).

Limitations of the Approach

While our approach demonstrates promising results in generating SPARQL queries from NLQ, two main limitations warrant discussion. The first limitation is that the performance of our fine-tuned model, Llama-KGQA, is notably lower for Wikidata KG, where content is not transparent due to

Model / System	Year	Macro Precision	Macro Recall	Macro F1 QALD
(Borroto et al., 2022)	2022	45.38%	45.74%	**59.47%**
QAnswer	2022	**50.68%**	**52.38%**	57.76%
(Shivashankar et al., 2022)	2022	32.06%	33.12%	49.09%
(Baramiia et al., 2022)	2022	42.89%	42.72%	42.81%
Gavrilev et al.	2022	14.21%	14.00%	19.48%
Llama-KGQA	2024	7.46%	7.43%	13.36%

Table 4: Comparison between Llama-KGQA and (Borroto et al., 2022), QAnswer, (Shivashankar et al., 2022), (Baramiia et al., 2022), Gavrilev et al. using QALD-10 Wikidata benchmarking dataset.

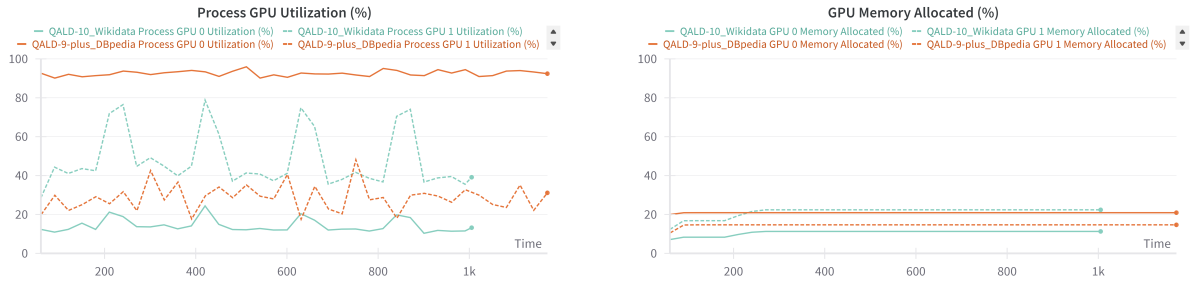

Figure 3: A: GPU process utilization during Llama-KGQA fine-tuning. B: GPU process allocated memory during Llama-KGQA fine-tuning.

the use of non-human-readable identifiers. This limitation underscores the difficulty of interpreting the KG data that was not explicitly available during training. For example, to generate a SPARQL query for the following question in the QALD-10 testing set: "After whom is the Riemannian geometry named?", Llama-KGQA has generated the following SPARQL query:

```
SELECT DISTINCT?uri WHERE {
    <http://www.wikidata.org/
       entity/Q160544> <http://
       www.wikidata.org/prop/
       direct/P31>?uri.
}
```

While the golden query is:

```
...
PREFIX wd: <http://www.wikidata.
   org/entity/>
PREFIX wdt: <http://www.wikidata.
   org/prop/direct/>
SELECT DISTINCT ?result WHERE {
    wd:Q761383 wdt:P138 ?result.
}
```

This example highlights the issue of the model using incorrect identifiers.

Another limitation is that the model occasionally generates SPARQL queries that use incorrect URIs for properties or individuals, leading to inaccurate or invalid results. This issue arises because the model has no access to the target KG, and therefore may not correctly represent the mappings between natural language expressions and the corresponding KG entities. For example, to answer the following question in the QALD-9-plus testing set: "What is the profession of Frank Herbert?", Llama-KGQA has generated the following SPARQL query:

```
PREFIX dbo: <http://dbpedia.org/
   ontology/>
PREFIX res: <http://dbpedia.org/
   resource/>
SELECT DISTINCT?uri WHERE {
    res:Frank_Herbert dbo:
       profession?uri
}
```

While the golden query is:

```
PREFIX dbpedia2: <http://dbpedia.
   org/property/>
PREFIX res: <http://dbpedia.org/
   resource/>
SELECT DISTINCT ?string WHERE {
  res:Frank_Herbert dbpedia2:
     occupation ?string
}
```

This example demonstrates the model's difficulty

in identifying the correct property name used in this KG.

Such limitations highlight the need for improved mechanisms to ensure the correct association between natural language input and the appropriate identifiers or URIs in the target knowledge graph.

5 Conclusion

This study conducted an analysis that compared several Llama-based LLMs for their ability to generate SPARQL queries from NLQ. Our results reveal that Llama-KGQA, the fine-tuned version of Llama-3-8b, has obtained a higher accuracy than larger models like Llama-3-70b, while remaining efficient and scalable for real-world applications. The fine-tuning process using QALD question-answering datasets has shown potential in enhancing the overall effectiveness and adaptability of our new QA model, Llama-KGQA, marking a significant step forward in the application of LLMs within knowledge-driven AI. However, we also showed that for a KG (namely wikidata) which content would not have been transparent to the LLM from its pretraining, especially due to non-human-readable identifiers, the performance Llama-KGQA is dramatically lower.

Future work should therefore further explore incorporating the KG context into LLM fine-tuning, which could improve the model's ability to interpret and generate more accurate queries. This perspective will target the challenges of using fine-tuned LLMs in efficient QA systems powered by knowledge graphs, in particular by enabling the LLM to make use of information about relevant content in the knowledge graph during generation of the SPARQL query.

References

Zahra Abbasiantaeb and Saeedeh Momtazi. 2021. Text-based question answering from information retrieval and deep neural network perspectives: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 11(6):e1412.

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

AI@Meta. 2024. Llama 3 model card.

Ammar Arbaaeen and Asadullah Shah. 2020. Natural language processing based question answering techniques: A survey. In *2020 IEEE 7th International Conference on Engineering Technologies and Applied Sciences (ICETAS)*, pages 1–8. IEEE.

Nikita Baramiia, Alina Rogulina, Sergey Petrakov, Valerii Kornilov, and Anton Razzhigaev. 2022. Ranking approach to monolingual question answering over knowledge graphs. In *NLIWoD@ ESWC*, pages 32–37.

Manuel Borroto, Francesco Ricca, Bernardo Cuteri, and Vito Barbara. 2022. Sparql-qa enters the qald challenge. In *Proceedings of the 7th Natural Language Interfaces for the Web of Data (NLIWoD) co-located with the 19th European Semantic Web Conference, Hersonissos, Greece*, volume 3196, pages 25–31.

Mikhail Burtsev, Alexander Seliverstov, Rafael Airapetyan, Mikhail Arkhipov, Dilyara Baymurzina, Nickolay Bushkov, Olga Gureenkova, Taras Khakhulin, Yurii Kuratov, Denis Kuznetsov, et al. 2018. Deeppavlov: Open-source library for dialogue systems. In *Proceedings of ACL 2018, system demonstrations*, pages 122–127.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Dennis Diefenbach, Andreas Both, Kamal Singh, and Pierre Maret. 2020. Towards a question answering system over the semantic web. *Semantic Web*, 11(3):421–439.

John Fields, Kevin Chovanec, and Praveen Madiraju. 2024. A survey of text classification with transformers: How wide? how large? how long? how accurate? how expensive? how safe? *IEEE Access*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

M Humera Khanam and S Venkata Subbareddy. 2017. Question answering system with natural language interface to database. *International journal of management, IT and engineering*, 7(4):38–48.

Hanieh Khorashadizadeh, Fatima Zahra Amara, Morteza Ezzabady, Frédéric Ieng, Sanju Tiwari, Nandana Mihindukulasooriya, Jinghua Groppe, Soror

Sahri, Farah Benamara, and Sven Groppe. 2024. Research trends for the interplay between large language models and knowledge graphs. *arXiv preprint arXiv:2406.08223*.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Wei Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. Factscore: Fine-grained atomic evaluation of factual precision in long form text generation. *arXiv preprint arXiv:2305.14251*.

Khalid Nassiri and Moulay Akhloufi. 2023. Transformer models used for text-based question answering systems. *Applied Intelligence*, 53(9):10602–10635.

Arantxa Otegi, Iñaki San Vicente, Xabier Saralegi, Anselmo Peñas, Borja Lozano, and Eneko Agirre. 2022. Information retrieval and question answering: A case study on covid-19 scientific literature. *Knowledge-Based Systems*, 240:108072.

Thomas Pellissier Tanon, Marcos Dias de Assunção, Eddy Caron, and Fabian M Suchanek. 2018. Demoing platypus–a multilingual question answering platform for wikidata. In *The Semantic Web: ESWC 2018 Satellite Events: ESWC 2018 Satellite Events, Heraklion, Crete, Greece, June 3-7, 2018, Revised Selected Papers 15*, pages 111–116. Springer.

Aleksandr Perevalov, Andreas Both, Dennis Diefenbach, and Axel-Cyrille Ngonga Ngomo. 2022a. Can machine translation be a reasonable alternative for multilingual question answering systems over knowledge graphs? In *Proceedings of the ACM Web Conference 2022*, pages 977–986.

Aleksandr Perevalov, Dennis Diefenbach, Ricardo Usbeck, and Andreas Both. 2022b. 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)*, pages 229–234.

Aleksandr Perevalov, Xi Yan, Liubov Kovriguina, Longquan Jiang, Andreas Both, and Ricardo Usbeck. 2022c. Knowledge graph question answering leaderboard: A community resource to prevent a replication crisis. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 2998–3007, Marseille, France. European Language Resources Association.

Soumajit Pramanik, Jesujoba Alabi, Rishiraj Saha Roy, and Gerhard Weikum. 2024. Uniqorn: unified question answering over rdf knowledge graphs and natural language text. *Journal of Web Semantics*, page 100833.

Kanchan Shivashankar, Khaoula Benmaarouf, and Nadine Steinmetz. 2022. From graph to graph: Amr to sparql. In *Proceedings of the 7th Natural Language Interfaces for the Web of Data (NLIWoD) co-located with the 19th European Semantic Web Conference (ESWC 2022), Hersonissos, Greece, 29th May*.

Ricardo Usbeck, Michael Röder, Michael Hoffmann, Felix Conrads, Jonathan Huthmann, Axel-Cyrille Ngonga-Ngomo, Christian Demmler, and Christina Unger. 2019. Benchmarking question answering systems. *Semantic Web*, 10(2):293–304.

Ricardo Usbeck, Xi Yan, Aleksandr Perevalov, Longquan Jiang, Julius Schulz, Angelie Kraft, Cedric Möller, Junbo Huang, Jan Reineke, Axel-Cyrille Ngonga Ngomo, et al. 2023. Qald-10–the 10th challenge on question answering over linked data. *Semantic Web*, (Preprint):1–15.

Xu Zhang, Wenpeng Lu, Fangfang Li, Xueping Peng, and Ruoyu Zhang. 2019. Deep feature fusion model for sentence semantic matching. *Computers, Materials and Continua*.