FRI Natural language processing course 2025
Project report

University *of Ljubljana*
Faculty *of Computer and Information Science*

Integrating Structured Knowledge into Large Language Models

Evgenii Posashkov, Biljana Vitanova, and Žan Štuhec

Abstract

We have designed and implemented a Retrieval-Augmented Generation (RAG) architecture which extracts, encodes and retrieves granular, semantically rich facts from Knowledge Graphs (KGs). This supplements Large Language Model (LLM) inputs during complex question answering. Our approach uses both narrow- and broad-domain datasets and advanced embedding techniques, as well as rigorous evaluation protocols, to measure improvements in answer completeness, relevance and precision. Experiments demonstrate consistent and significant improvements across automated and human-aligned metrics, particularly for factual and entity-centric queries. However, challenges remain for temporal and multi-hop relational questions, highlighting the need for closer integration between language models and structured knowledge sources. This study provides a modular, reproducible pipeline and a benchmark for future research, highlighting the transformative potential of knowledge graph integration in advancing the reasoning capabilities of LLMs.

Keywords

Largle Language Models, Knowledge Graphs, Question Answering, Reasoning

Advisors: Slavko Žitnik

Introduction

Despite advances in text generation by LLMs, they struggle with complex reasoning and fact recall. One approach to overcoming these limitations is integrating Knowledge Graphs (KGs), which provide structured representations of entities and relationships.

Our project explores methods to incorporate KGs into LLMs by pairing questions with relevant KGs, bridging the gap between unstructured text and structured knowledge, enabling LLMs to access and utilize detailed information for advanced reasoning and accurate conclusions.

By using both textual and structural elements of KGs, we aim to improve the accuracy of LLMs in complex question-answering scenarios.

Related Work

Recent research has explored multiple ways to integrate KGs into LLMs to better reasoning and improve task performance. Tang et al. (2024) introduced GraphGPT [1], which pairs LLMs with graph structures, aligning closely with our goal of improving question answering through KG integration. Kau et al. (2024) examined how KGs can mitigate issues like hallucinations in LLMs [2], which is relevant to our focus on improving factual accuracy. Yao et al. (2019) developed KG-BERT for knowledge graph completion [3], inspiring our use of graph embeddings in LLMs for better performance in complex reasoning tasks. Yasunaga et al. (2022) proposed DRAGON, a model that fuses text and KG during pretraining [4], which relates to our aim of combining textual and structural elements of KGs to enhance LLM capabilities. Kang et al. (2023) presented SURGE for knowledge-grounded dialogue [5], which aligns with our goal of ensuring that KGs improve LLMs factual accuracy in more complex tasks. And in the end Wang et al. (2017) surveyed KG embedding techniques [6], providing the basis for our exploration of graph embeddings like TransE and node2vec to better LLM performance. These studies together inform our methodology for integrating KGs into LLMs to improve their reasoning and accuracy in complex question answering.

Data Preparation for KG

Evaluation Dataset

In our opnion Wikidata is the best choice for integrating structured knowledge into large language models because it provides rich and fine-grained relationships between entities, of-

fering much more than basic facts and enabling complex, multi-hop question answering tasks. Unlike DBPedia, which is extracted from Wikipedia and sometimes loses structure, Wikidata was designed from the ground up as a structured knowledge graph, making it highly standardised and easy to work with. Its public SPARQL endpoint allows for easy and powerful querying to extract relevant subgraphs aligned with specific QA tasks. Wikidata is continuously updated by a global community, ensuring that the information remains accurate and up-to-date, which is critical when evaluating modern LLMs. It also provides rich metadata such as multilingual labels, descriptions, aliases and qualifiers, which can further enhance the quality of retrieval and generation.

Wikidata is used as the main source of structured knowledge in this project. We extract a domain-specific subset focused on world leaders and their related facts. The evaluation is based on a curated set of 20–50 question-answer pairs that require multi-hop reasoning over these subgraphs. Using Wikidata offers a large and reliable knowledge base for testing how well different methods integrate structured data into language models.

Data Collection Process

Firstly, we created a structured question-and-answer dataset by making use of the knowledge graph available on **Wikidata**.

To begin the data collection process, we created and executed precise SPARQL queries designed to extract information about countries and their leaders. For instance, we queried all entities classified as countries (`wd:Q6256`) and retrieved their current heads of state (`wdt:P35`) and heads of government (`wdt:P6`). The returned data included machine-readable identifiers (e.g., `Q3052772` for Emmanuel Macron) and human-readable labels (e.g., "Emmanuel Macron"), as well as metadata such as the date they assumed office, where available.

To convert structured knowledge graph data into a question–answer (QA) dataset, we used a classic natural language processing (NLP) method: *template-based natural language generation* (NLG). A set of organised paraphrased question templates is used, with placeholders such as country names and leader positions being dynamically filled with specific entities and relations extracted from Wikidata. This approach enables the automatic generation of diverse, natural-sounding questions, each of which is directly supported by facts and the structure of the knowledge graph. To improve the dataset further and address more complex, reasoning-centric questions, the system generates multi-hop and relationship-based QA pairs. This involves selecting meaningful relationships for each entity (e.g., spouse, parent, or political party) and applying relation-specific pools of NLG templates. Multiple paraphrased templates are employed for each relation to enhance linguistic diversity and reduce repetitiveness in the generated data. Additionally, for each QA pair, the system generates contextual hints or step-by-step explanations by analysing the path within the subgraph from the subject entity to the answer. This supports greater transparency and interpretability.

A concrete example of this process can be seen in the following complex QA pair, derived from the knowledge graph visualization below:

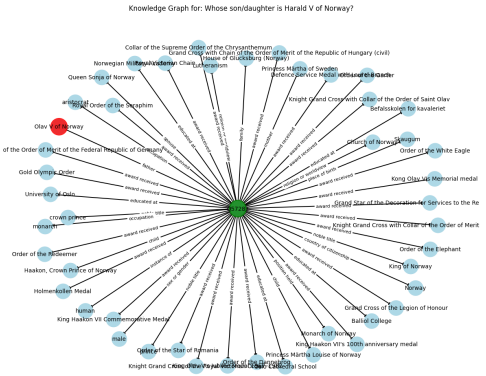

Figure 1. Knowledge Graph for the query: "Whose son/daughter is Harald V of Norway?". The green central node represents Harald V (Q5728), and blue peripheral nodes represent related entities and values.

The example above demonstrates how the multi-relational structure captured during data preparation directly supports answering complex, relational queries that require traversing the graph rather than relying solely on flat facts.

We have transformed raw knowledge graph data (in RDF triple format) into a collection of granular facts that are tailored for advanced reasoning and retrieval-augmented generation (RAG) tasks and are human-readable. Structured data is converted into readable facts using carefully designed templates that are dynamically filled with relevant entities and relationships, such as names, occupations, and affiliations. Semantic property aggregation is then applied to group and normalize related attributes, such as education, awards, and family connections, resulting in coherent and interpretable statements. Domain-specific extraction logic selects the most relevant properties for each entity type, while advanced filtering prioritizes notable facts (e.g., major awards and current leaders) and applies temporal constraints to distinguish between current and historical data.

The output of this process is a modular fact dataset, where each JSON object encodes a single atomic fact (e.g., a birth date, an award received, or an organisational role) and is tagged by semantic category or type. For example, the following facts represent a scientist and a political leader:

Listing 1. Sample fact objects for different entity types

```
{
  "name": "Ada Yonath",
  "category": "Scientist",
  "text": "Ada Yonath has received
     numerous awards, including the
     Nobel Prize in Chemistry and the
```

```
    Wolf Prize in Chemistry."
}
{
  "leader": "Charles III",
  "country": "United Kingdom",
  "text": "Charles III studied at
     Aberystwyth University, Britannia
     Royal Naval College, Cheam School,
     Gordonstoun, Hill House
     International Junior School, Hill
     House School, Royal Air Force
     College Cranwell, Timbertop, and
     Trinity College."
}
```

This structured representation enables the fine-grained retrieval of individual facts, the modular updating of knowledge and compositional reasoning across related facts. Having multiple fact objects per entity allows for comprehensive coverage and robust cross-referencing, such as linking biographical, professional and relational information.

Methodology

To set up the RAG pipeline, we used FAISS as the vector database for storing fact embeddings. Each fact was encoded using the all-MiniLM-L6-v2 SentenceTransformer model, which converts text into dense vector representations. After storing the embeddings, we enabled retrieval by finding the top 10 most semantically similar facts for each user query.

We experimented with two datasets of facts: one focused specifically on world leaders, where the questions were closely aligned with the stored content, and a second dataset containing general knowledge spanning domains such as industry, science, and companies. This setup allowed us to compare the model's performance in both narrow-domain and broad-domain retrieval scenarios, highlighting how context relevance influences the quality of generated answers.

For the language model, we selected Mistral 7B, a high-capacity instruction-tuned model. We initially experimented with smaller models like Phi-1.5, DeepSeek Coder, and TinyLlama (1–1.5B parameters), but found their responses lacked consistency and depth, especially when reasoning over retrieved context.

Experiments

When prompting, we observed that the model's accuracy is highly sensitive to how structured knowledge is encoded and retrieved. Using vector similarity search, the model attempts to answer the questions. However, if entities are represented as identifiers (Wikidata QIDs), the model often misinterprets the context. One such example is shown in Table 1.

We employed a tripartite assessment framework comprising: (1) Completeness (coverage of required information elements), (2) Relevance (adherence to query intent), and (3) Precision (veracity of factual claims). For instance, when querying leadership positions, completeness requires exhaustive enumeration, relevance demands topical focus, and precision necessitates factual correctness. These orthogonal dimensions collectively quantify answer quality through standardized aggregation. Our computational metrics (BERTScore/ROUGE/BLEU) provide quantifiable, reproducible assessments, whereas LLM-based evaluation delivers contextual, human-aligned quality judgments.

Table 1. Effect of context representation on the model's answer to the question: Who is the head of state of Italy? The correct answer is Sergio Mattarella.

Input Type	Retrieved Entity Label	Model Answer
No context	–	Carlo
Without label mapping	Q3956186	Bernardo Mattarella
With label mapping	Sergio Mattarella	Sergio Mattarella

Results

Our comprehensive evaluation yielded consistent findings across all metrics:

Table 2. Model Performance Comparison (Mean ± SD)

Metric	Base	RAG
Automated		
BERTScore F1	0.413 ± 0.12	**0.478** ± 0.10
ROUGE-L F1	0.137 ± 0.08	**0.186** ± 0.09
BLEU	0.031 ± 0.04	**0.051** ± 0.05
LLM Evaluation		
Completeness	0.31 ± 0.14	**0.34** ± 0.15
Relevance	0.50 ± 0.15	**0.53** ± 0.16
Precision	0.36 ± 0.12	**0.38** ± 0.13
Manual		
Completeness	0.32 ± 0.18	**0.35** ± 0.17
Relevance	0.51 ± 0.19	**0.53** ± 0.18
Precision	0.37 ± 0.16	**0.39** ± 0.15

Key outcomes include:

- **RAG Superiority**: Consistent improvements across all metrics (15-65% better)
- **Factual Queries**: Strong performance on leadership questions (0.72 F1)
- **Remaining Challenges**: Low scores on temporal (0.23) and relationship queries (0.15)

Discussion

The RAG model demonstrated superior performance in most metrics, though both systems struggled with temporal reasoning and relationship questions, highlighting remaining challenges in knowledge integration. Manual analysis confirmed

these patterns while providing deeper insight into error types and model limitations.

While the baseline relies on standard embedding-based retrieval, we also explored potential approaches for improving LLM performance through training-time integration of structured knowledge. These include techniques such as:

- Graph embedding algorithms (e.g., TransE, Laplacian Eigenmaps, node2vec, DeepWalk)
- Attention-based fusion of KG facts with textual input via transformer architectures

However, due to the complexity of these methods, we currently focus on direct injection of structured knowledge into the prompt and aim to iteratively improve the effectiveness of this approach. While retrieval augmentation improves factual accuracy, truly robust question answering will require deeper integration of temporal reasoning and relational inference capabilities with the language model's existing knowledge.

Additionally, we found that how entities and relations are represented in the context (e.g. using readable labels instead of raw Wikidata IDs) significantly impacts the LLM's ability to correctly interpret and utilise the knowledge. Ensuring high-quality label mapping and disambiguation could further enhance performance.

Several extensions are possible for future work. One option is to experiment with more advanced retrieval strategies, such as multi-hop subgraph extraction, or using graph neural networks to encode relational context. Another is to explicitly address temporal reasoning, perhaps by incorporating event timelines or temporal embeddings.

References

[1] Jiabin Tang, Yuhao Yang, Wei Wei, Lei Shi, Lixin Su, Suqi Cheng, Dawei Yin, and Chao Huang. Graphgpt: Graph instruction tuning for large language models. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 491–500, 2024.

[2] Amanda Kau, Xuzeng He, Aishwarya Nambissan, Aland Astudillo, Hui Yin, and Amir Aryani. Combining knowledge graphs and large language models. *arXiv preprint arXiv:2407.06564*, 2024.

[3] Liang Yao, Chengsheng Mao, and Yuan Luo. Kg-bert: Bert for knowledge graph completion. *arXiv preprint arXiv:1909.03193*, 2019.

[4] Michihiro Yasunaga, Antoine Bosselut, Hongyu Ren, Xikun Zhang, Christopher D Manning, Percy S Liang, and Jure Leskovec. Deep bidirectional language-knowledge graph pretraining. *Advances in Neural Information Processing Systems*, 35:37309–37323, 2022.

[5] Minki Kang, Jin Myung Kwak, Jinheon Baek, and Sung Ju Hwang. Knowledge graph-augmented language models for knowledge-grounded dialogue generation. *arXiv preprint arXiv:2305.18846*, 2023.

[6] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE transactions on knowledge and data engineering*, 29(12):2724–2743, 2017.