Semantic Web Project : Zero-shot prompting of LLM for question answer task with Knowledge Graphs

Karan Gupta
2021258
karan21258@iiitd.ac.in

Shivesh Gulati
2021286
shivesh21286@iiitd.ac.in

1. Motivation

Training or even fine-tuning a Large-Language Model (LLM) with millions of parameters from scratch takes a lot of resources. The main object of this project is to harness the zero-shot capabilities of large-language models by augmenting the prompt with relevant triples from the knowledge graph to provide context and generate better-suited responses to the input questions from the LLMs. Through this project, we aim to analyse the improvement in the responses generated from the LLMs after providing the context using various triple retrieval techniques. In this project, we also try to develop our triple retriever from the knowledge graph and analyse and compare its performance to pre-existing machine learning and NLP-based retrievers.

2. Literature Review

[1] Knowledge-Augmented Language Model Prompting for Zero-Shot Knowledge Graph Question Answering This study aims to develop a framework for improving the performance of large language models for question-answering tasks in a zero-shot setting. The paper achieves this objective by prepending top k-most relevant triples from a knowledge graph extracted from the knowledge graph using a triple extractor and then finding the most relevant triples after creating embeddings from natural language using sentence transformers like **(mpnet-base-v2)** and then analysing the performance jump achieved on the said task before and after adding the context.[1]

3. Current Progress

Our current methodology follows three steps:-

3.1. Extracting the named entities from the prompts

For this project, we have chosen the Mintaka dataset **Mintaka Dataset**, which already has named entities for each of the questions extracted with their wiki database id alongside the answer entity. It consists of general questions from a wide variety of domains.

3.2. Retrieving triples from the knowledge graph using graph search algorithms

After receiving the relevant named entities from the question, we apply our triple retrieval process. This process consists of executing a SPARQL query on the wiki data knowledge base, which retrieves all the triples in which the predicates belong to the **wdt: namespace**, which is the namespace for all the direct properties of a given entity. This filters all the identifiers and id triples, which may not be relevant for answering the question. Then, for each of the retrieved triples, a dfs call is made on the object retrieved, and at each call, the SPARQL query described is executed again to retrieve the relevant triples at further levels. This way, our retriever can extract the triples from the knowledge graph at a distance of k-hops from a given entity. For now, we have specified (k=2), i.e. for each of the entities extracted from the question text, we are retrieving triples at a distance of k=2 hops away.

3.3. Verbalizing the triples

The subject, object and predicate of the triples extracted were concatenated and saved in a text and CSV file to convert them to natural language form for semantic retrieval. This may not be the best possible method, but it can be further improved to get many triples in a better-suited natural language form.

3.4. Getting the relevant triples

The triples extracted from the retriever in step 2 were very large and were much beyond the word limit of prompts for many existing LLMs; hence, we decided to reduce the number of triples by extracting the most relevant triples from the ones obtained. For this purpose, we created a pipeline using **Lang Chain** in combination with a pre-trained **Instructor Embedding Model** which is publicly available on Hugging Face. The lang-chain pipeline splits the text file containing the triples into sub-files. It was converted into a **FAISS (Facebook AI Similarity Search)** vector database, and a semantic search was conducted based on the question provided to the retriever (Instructor Embedding Model). Cosine-similarity scores were calculated based on the embedding vectors created by the Instructor Embedding Model to evaluate the relevance of the triples, and then the final triples were retrieved.

3.5. Feeding the context into the Large Language Model

After extracting the most relevant triples (3-sub documents) in our case, the triples were concatenated with the question statement from the dataset to generate the final prompt of the form:
Context: Extracted_Triples Question: Question_Statement
The question is fed into the LLM once without the context and once with the context, and the final responses in both scenarios are recorded.

4. Individual Contributions

Task	Member Name
Researching Ideas and Resources	Both Members
SPARQL Query Formation for triple extraction	Both Members
K-Hop on the Knowledge Graph and verbalisation	Karan Gupta
Semantic Similarity Search	Shivesh Gulati

Table 1. Contributions Table

5. Future Proposed Tasks

- Doing backwards hops on the knowledge graph in addition to the forward hops to extract the triples in which the question entity acts as the object of the triple.
- Using semantic search techniques over simple graph traversal techniques to reduce the number of triples produced in the first place.
- Quantifying the improvement in question-answering tasks with and without prompts.
- Creating a summary of the top-k triples using a Small Language Model to increase the number of triples so that more context can fed into the LLM if possible.
- Running the task on all the triples in the test split of the dataset.

References

[1] Jinheon Baek, Alham Fikri Aji, and Amir Saffari. Knowledge-augmented language model prompting for zero-shot knowledge graph question answering. 2023.