

# Supplementary Material

## 1. Exploration with Large Language Models

An alternative approach for generating entity aspects from the KG was explored by leveraging Large Language Models (LLMs). This section presents the details of this approach, along with its limitations, which restricted its application within the *InterpretE* process. We also present an alternative approach for estimating the semantic similarity of *InterpretE* vectors.

### 1.1. Feature Selection

LLMs excel at summarizing data and identifying specific patterns that may elude human analysts. Given a prompt  $\mathbf{q}$  formulated in natural language, an LLM  $\mathbf{f}$  generates a response  $a = f(\mathbf{q}) = \max_x \mathbb{P}(x|\mathbf{q})$ , using the most probable tokens based on the context. However, the LLMs' knowledge strongly depends on the data it has been trained on. LLMs are also prone to hallucinations and can invent convincing facts. To address these limitations, various strategies have been developed. Some approaches involve retraining the LLM (fine tuning) to enhance performance on specific tasks. But these methods can sometimes need important computational resources.

Alternatively, certain techniques bypass the need for fine tuning the model. One notable approach is Retrieval-Augmented Generation (RAG) [2], which constructs a database from a collection of documents chunks. These documents chunks are embedded into a vector space using an embedding model. This built database allows us to enrich the initial context prompt by retrieving the documents chunk similar to the user prompt in terms of vectors. This equips the prompt with the necessary contextual knowledge in order for the LLM to respond accurately, effectively mitigating hallucinations. The objective in this study was to utilize a KG to create the vector database, thereby capturing semantically meaningful dimensions of the conceptual space directly through LLM queries. For this purpose, LlamaIndex (v0.10.28) <sup>1</sup> was employed, a framework that facilitates the construction of this pipeline. The Mistral-7B LLM [1] was selected due to its robust performance and minimal local resource requirements. This framework offers flexibility in the methods used to build the vector database, typically guided by the input data format. Since knowledge graphs can be represented as structured graphs, they provide a key advantage over plain text due to their inherent structure.

**Building the database.** The LlamaIndex framework includes various classes for constructing the database. Focus was placed on two primary storage types: the *VectorStoreIndex* and the *KnowledgeGraphIndex*. All these storage types expect as input a plain text (document). Hence, we need to find a way to express our KG in a text format.

With the *VectorStoreIndex*, documents are divided into smaller chunks, with each chunk associated with a corresponding vector that is stored in the database. As a result, the constructed database consists of vectors representing the various document chunks. In this approach, the expected type of input is a text. One way to express the KG as plain text, which has been used is by creating one line per triple without any prefix, only the entities and relations names. These documents can be seen as a list of the graph's edges. Hence, the triples are expressed in simplified, human-understandable terms. Due to the computational efficiency of this technique, it was valuable for gaining initial insights into the overall approach. However, a major drawback was the loss of the graph's inherent structure.

With the *KnowledgeGraphIndex*, a KG is constructed from the documents chunks, resulting in a graph database. This approach addresses the previous limitation by utilizing the inherent structure of a graph during the retrieval part. However, it assumes input files that are not already in KG format (e.g., text files), as KG construction is embedded within this index. As with the *VectorStoreIndex*, our KG-described as plain text with one triple per line. The KG generated by the pipeline closely resembled the original, with minor variations. While effective for smaller KGs, this approach demanded substantial computational resources for larger KGs, such as Yago3-10 or FB15K-237.

<sup>1</sup><https://www.llamaindex.ai/>

To overcome this limitation and preserve the structure of the KG, an attempt was made to manually rebuild the KG using LlamaIndex’s coded methods and the provided set of triples. However, the approaches tested were unable to construct the KG as expected, often resulting in an empty output. This is probably due to a specific embedded implicitly used by the KnowledgeGraphIndex which has been omitted by the manual approach due to a lack of documentation. Ultimately, focus was shifted to the VectorStoreIndex approach, which allowed for the extraction of all KG knowledge, though at the expense of disregarding the graph’s structural information.

*Answer Generation.* Once the database was built, the goal was to retrieve relevant information to augment the user prompt with the appropriate knowledge. After several trials, the *tree\_summarize* method was selected, as it provided the most consistent answers. This method summarizes the different retrieved chunks in order to produce a smaller context.

*Prompting.* The choice of the prompt was crucial to obtain the most precise answer for our task. This was identified as an interesting prompt engineering task in itself, but the aim was to develop the simplest possible prompt template. The goal of this pipeline was to extract the salient features of the given KG to build conceptual spaces. Efforts were made to provide the LLM with the complete definition of a conceptual space, which helped achieve more controlled responses. Additionally, specifications were made regarding the presentation of the answers to facilitate any subsequent automated processing.

The final prompt we used is presented in Figure 1.

**Prompt.**

Context: {retrieved context}

Question: You are a useful information retrieval agent, A conceptual space is a geometric structure that represents a number of quality dimensions, which denote basic features by which concepts and objects can be compared, From the context, extract 20 quality dimensions, present the result as a list.

Fig. 1. Prompt example used for the RAG pipeline

For simplicity, we only present the question in the next parts.

*Limitations.* This approach appeared promising due to the potential abilities of LLMs in reasoning and summarizing tasks, which we aimed to incorporate into our overall strategy. One limitation encountered was the lack of transparency of the answers generated. Despite RAG helping to mitigate potential hallucinations of the LLMs, uncertainty remained regarding whether the answers were derived from actual knowledge. It was also unclear whether the responses were formulated solely from the LLM’s training or from the constructed database.

The quality of the produced answers was crucial, as it laid the foundation for our future conceptual space. Unfortunately, no metrics were available to measure answers quality, making it challenging to evaluate the various responses beyond a few exploratory experiments. The only viable method was human feedback on each answer, which was inherently biased.

Finally, by taking a step backwards and looking at the whole RAG pipeline, only some chunks are retrieved from the transformed KG to produce the answer. Hence, the LLM has no global view of the KG. Even though LLMs are good at summarizing the information, here, the retriever hides most of the information. The produced answer is probably a good summarize but only regarding the retrieved context. This can explain the quality observed in the generated answer: if the retrieved context represents well a majority of the KG entities, the summarize (through quality dimensions) produced by the LLM is a rough answer but not so bad.

*Illustrative Results.* The quality of the results is not bad overall. In Figure 2, we provided two examples of answers we got from the pipeline. We can observe that some suggested quality dimensions belongs to the most represented relations present in Yago3-10. Some presented quality dimensions were also used in our study. But sometimes, like in the right example, the suggested dimensions were too specific. The difference between both examples is the prompt used which slightly differs. This also raised another phenomenon which is the stability of the LLM regarding

the prompt. The produced results were sometimes unstable leading to a prompt engineering task to find the potential best prompt. As this part is not fundamental for our study since the statistical approach is efficient, we decided not to go deeper in the prompt engineering task although we were aware that other techniques could have produced potentially better results. We also did not consider the polysemanticity of the suggested salient features, we were strictly expecting existing relations and not similar words.

In light of the unsatisfactory results, we opted not to pursue this approach and instead adopted a more manageable, interpretable, and systematic method of performing data-driven analysis in the *InterpretE* method, which was discussed in detail in the *Data Analysis and Selecting Entity Aspect* section.

## 1.2. Semantic Similarity Evaluation with LLMs

We also explored an alternative way to the *simtopk* metric using a large language model (LLM) in a limited experiment. We attempted this approach with few-shot prompting using Llama3-70B [3]. Additionally, we experimented with a RAG pipeline using the entire initial knowledge graph with Mistral7B [1] and LlamaIndex. However, the results were not consistently convincing, and the model sometimes contradicted itself.

In our prompt to the LLM, we provided two examples: one positive and one negative, randomly chosen from all possible entities. We also specified the type of similarity we were evaluating, as it depended on the selected feature for a given experiment. This method allows us to assess our approach by examining how similar the neighborhood

**Question.** You are a useful information retrieval agent, A conceptual space is a geometric structure that represents a number of quality dimensions, which denote basic features by which concepts and objects can be compared, From the context, extract 20 quality dimensions, present the result as a list.

1. Affiliation to a football club or basketball team
2. Gender (male or female)
3. Is influenced by another person or concept
4. Influences another person or concept
5. Acts in a film or play
6. Marital status
7. Citizenship of a country or region
8. Plays for a football club or basketball team
9. Has won a prize or award
10. Is connected to an airport or location
11. Graduated from a specific university
12. Is affiliated to a specific organization, be it sports-related or not
13. Has musical role (in this case, plays the guitar)
14. Plays for a national football team
15. Is located in a country or city
16. Died in a specific location
17. Belongs to a specific political party
18. Created an album or song
19. Has a specific role in a sports team (goal-keeper, midfielder, forward)
20. Is affiliated to a football club from a different country

**Question** You are a useful information retrieval agent, answer the question: From the context, can you make a list of 20 concepts that represent the knowledge graph?

1. Country (mentioned multiple times)
2. Western Cape (a specific region)
3. South Africa (a country)
4. Algeria (a country)
5. African Union
6. United Nations
7. World Trade Organization
8. Arab League
9. Organisation of Islamic Cooperation
10. International Bank for Reconstruction and Development
11. International Development Association
12. International Finance Corporation
13. Multilateral Investment Guarantee Agency
14. International Centre for Settlement of Investment Disputes
15. African Development Bank
16. Arab Bank for Economic Development in Africa
17. Asian Development Bank
18. Organisation internationale de la Francophonie
19. UNESCO
20. Universal Postal Union

Fig. 2. Example results using LlamaIndex with Mistral-7B for Yago3-10

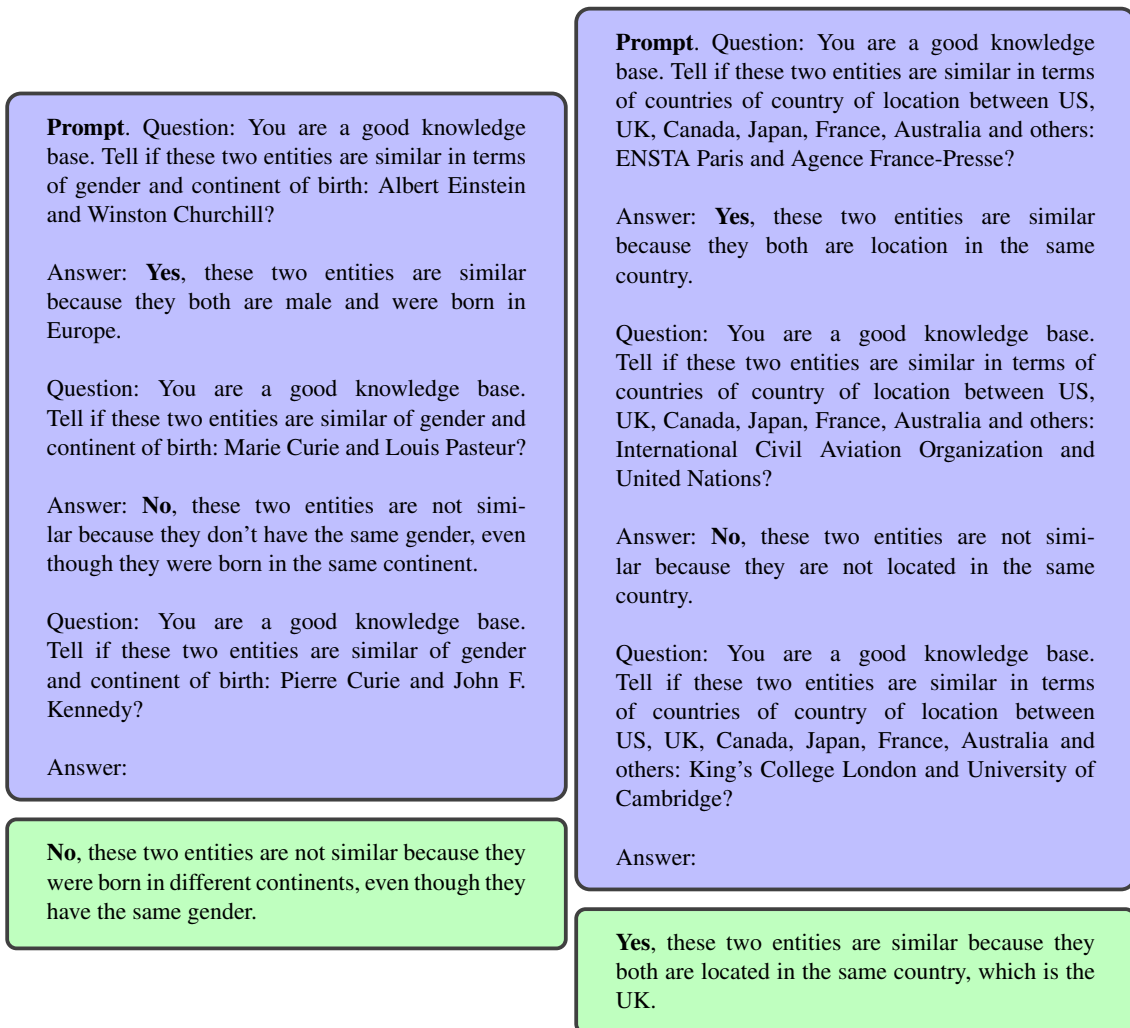


Fig. 3. Partial example of few-shot prompts with Llama 3 70B using HuggingChat

of a given entity is to the entity itself. An example is shown in Figure 3. This approach seemed promising and needs to be applied to all entities to obtain a global evaluation metric, which we plan to explore in future work.

## References

- [1] A.Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D.S. Chaplot, D. de las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L.R. Lavaud, M.-A. Lachaux, P. Stock, T.L. Scao, T. Lavril, T. Wang, T. Lacroix and W.E. Sayed, Mistral 7B, 2023.
- [2] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, S. Riedel and D. Kiela, Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks, in: *Advances in Neural Information Processing Systems*, Vol. 33, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan and H. Lin, eds, Curran Associates, Inc., 2020, pp. 9459–9474.
- [3] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar et al., Llama: Open and efficient foundation language models, *arXiv preprint arXiv:2302.13971* (2023).