PhD Interview Assignment

Cristobal Donoso-Oliva

November 27, 2024

The assignment involves developing a retrieval-augmented generation (RAG) system to retrieve relevant tables. A collection of 1001 tables and 100 queries are provided. The goal is to identify and rank the five most relevant tables for each query. In this written we extend this idea for recommending highlight from the structured data.

In general terms, our end-to-end pipeline should take a user query as the input and produce a referenced text as the output. Intermediate steps are crucial for producing valuable insights. Figure 1 shows our proposed pipeline which consist in specialized modules to control different levels of reasoning.

We start by receiving a collection of tables which constitute our knowledge database. A semantic splitter is necessary to control the number of tokens for the LLM while preserving the semantic meaning of the tuples. Even though we can split the table, the column elements may still be too large. Here, we may use a larger model that can accept more tokens or try to summaries the entry data. Similarly, the context splitter extracts relevant information from the table structure, such as column names, data types, table name, and table ID. Intuitively, when searching for information, we first select tables based on their general structure before delving into the details. Potential problems may arise if tables lack meaningful contextual data, such as table descriptors. Until step 3 corresponds to inner operations of our pipeline. (4) The input consists of a user query in natural language. (5) We encode the semantic and context data, as well as the query, using the same LLM. Following this process, we index the embeddings in a vector database.

Decision module (6) is a critical engine that look for the most similar vectors from the semantic and contextual databases. We calculate the similarity between the query and the vectors in each database. This step allows us to capture both semantic information and contextual information from the relevant tables, in order to answer the question coherently. We expect the tables selected based on the semantic and context vectors to be the same, forming a cohesive response. Then, we use the intersection between the vectors as input for another LLM that generates the answer for the user.

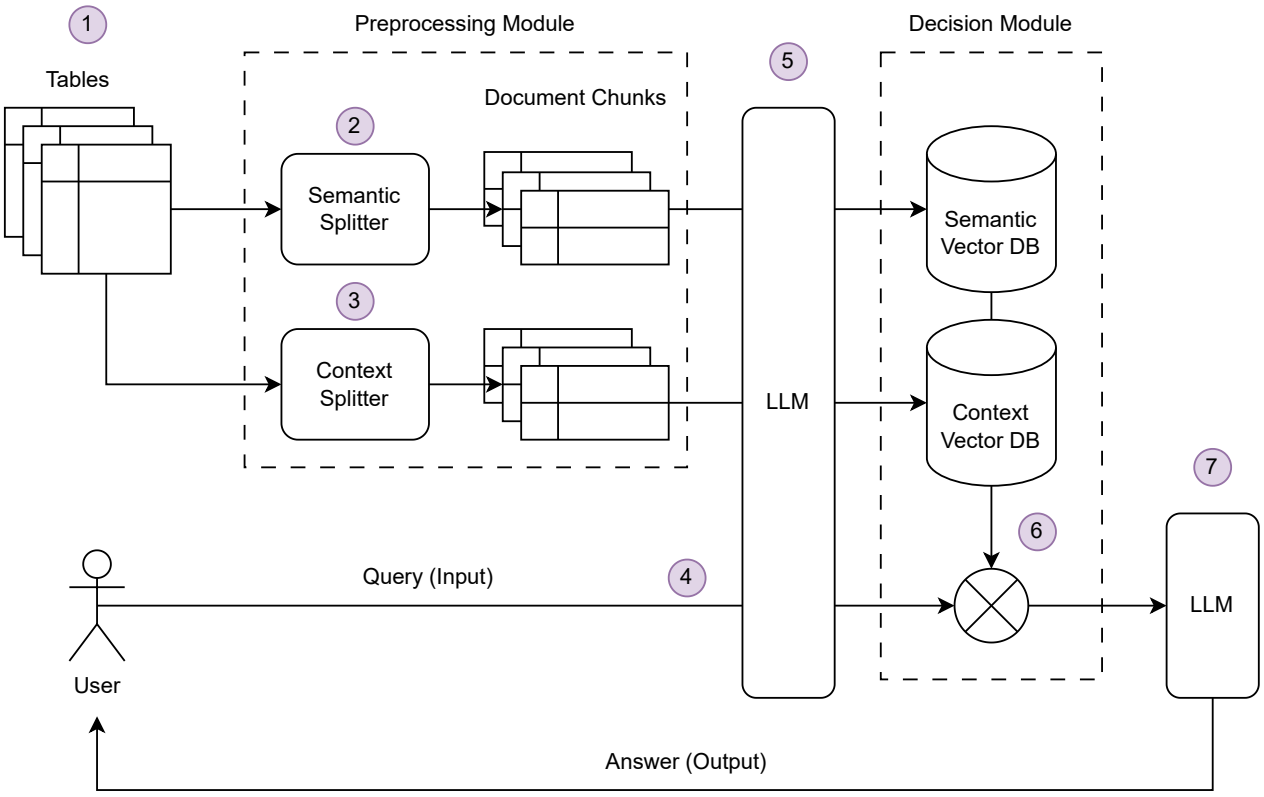

Figure 1: Pipeline Diagram. (1) Collection of tables containing specific domain knowledge. (2) Semantic splitter that reduces the tables into tuples with fewer tokens. (3) Context splitter that extracts general structural features from the tables, such as columns and data types. (4) Query from the user in plain text. (5) LLM that encodes the query, semantic, and context documents. (6) Distance operation performed between (query,semantic) and (query,context). The final decision in (6) is the intersection between the semantic vectors and the context vector. (7) An LLM takes the final embedding to generate a response.

This pipeline includes all the necessary steps for creating an end-to-end system for highlight recommendation. However, it is not free of problems. Here we enumerate some of them:

- Knowledge selection: Reducing tables in semantic documents is a challenging tasks that requires to identify semantic information from the tables. It has be done in a way we do not loose important information while preserving a manageable number of tokens.

- Scaling: If our number of tables increase significantly, our database will increase subsequentially. We should manage mechanism of avoid duplicating information and update tables.

- Getting context features: When we capture context features, we assume the database contains minimal information or levels of standardization. However, in practical cases tables can be heterogeneous, unclear or ambiguous. An option could be transforming tables to image and use another model to extract descriptors.

- LLM representation: In this example we use an open source LLM model trained on general purpose content. However, it would be beneficial if during the pretrainig we include information that can help the inference part. A potential solution can be finetune our model on our database.

- Selecting relevant vectors is a challenging task. Our current approach relies on a naive definition of a coherent response by taking the intersection between vectors. However, this can be sub-optimal in cases where there is no intersection. Contextual information does not necessarily need to be connected with semantic information, and vice versa. Introducing a deep learning model to assess the quality of the response could provide a more robust solution. This model's parameters could be fine-tuned using active learning from user feedback.