HelpMate AI Project

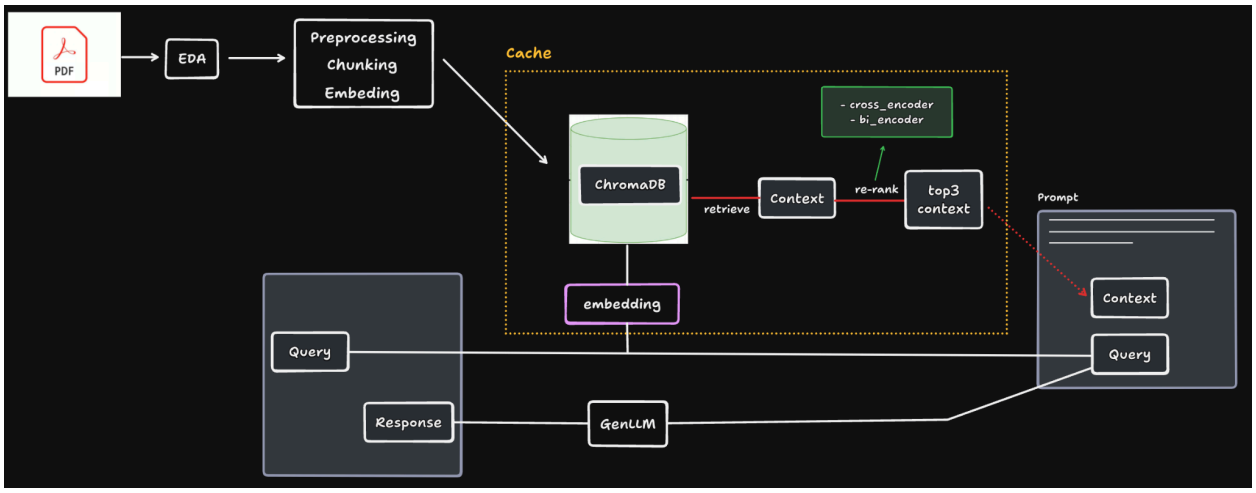
Project goal

The project will aim to build a robust generative search system capable of effectively and accurately answering questions from a policy document.
For more details, you can access my notebook

Data source

I use a single long-life insurance policy document for this project. The PDF for the document can be downloaded at link_to_download

Design choices



I implement step by step:

- Embedding Layer
- Search Layer
- Generative Layer
- Query Search

1. **Embedding layer:**
    - *Document Processing*: I use pdfplumber library to extract text from pdf file to dataframe

- *Chunking Strategy*: Experiment with different chunk sizes and overlap strategies to ensure that important information isn't missed during retrieval. I try to choose one of 2 methods: RecursiveCharacterTextSplitter And CharacterTextSplitter from langchain
- *Embedding Model Choice:* Use OpenAI's embedding models or models from the SentenceTransformers library.

2. **Search Layer**
    - *Design Queries*: Create at least three queries based on a thorough understanding of the insurance policy document. These should reflect typical questions a user might ask about the policy.
    - *Caching*: Implementing a caching mechanism was essential for improving the system's efficiency
    - *Embedding Queries and Searching*: Embed these queries and use the ChromaDB vector database for efficient searching. Implement caching to improve performance and avoid redundant computations.
    - *Re-ranking*: Utilize cross-encoding models from HuggingFace for re-ranking the search results. This step ensures that the most relevant chunks are prioritized. Besides, I also use a bi-encoder for comparison.
3. **Generative Layer**
    - *Prompt Design*: The prompt for the LLM should be detailed, guiding the model to use the retrieved information effectively. Consider adding context from the retrieved chunks to make the generated response more relevant.
    - *Few-shot Examples*: Including examples in the prompt might improve the LLM's performance by providing a clear understanding of the expected output format and style.
4. **Query Search**
    - Testing with self-designed queries will help gauge the system's accuracy and relevance.

Challenges faced

Most of the challenges are from the parameter selection. It is tough to find the chunking size, and top k retrieval documents.
When we have a lot of retrieved documents, the LLM can not find the important information to generate a good answer, besides we have a limitation of input and output tokens.
While we had fewer retrieved documents, we did not have enough pieces of information to generate the answer.

Lessons Learned

1. Importance of Preprocessing and Chunking Strategies: The effectiveness of the embedding and retrieval process heavily depends on how well the document is preprocessed and chunked. Different chunking strategies can significantly affect the quality of the embeddings and, consequently, the relevance of the retrieved results.

2. Model Selection Matters: The choice of the embedding model directly impacts the quality of the semantic understanding. Different models have strengths in different contexts, and not all models perform equally across all types of content.

3. Query Design is Crucial for Evaluation: The quality of the system's output is closely tied to the quality of the queries used for testing. Well-designed queries that are representative of real-world questions can

4. Effective Caching Improves Performance: Implementing a caching mechanism was essential for improving the system’s efficiency, especially during repeated query testing and tuning.

5. Re-ranking Enhances Result Relevance: The re-ranking step using cross-encoding models was effective in refining the search results. It added an extra layer of accuracy, ensuring that the most relevant results were prioritized.

6. Prompt Engineering is an Art: The design of the final prompt in the generation layer significantly influenced the quality of the responses generated by the LLM. A well-crafted prompt that includes clear instructions and context improves the quality of the output.