Retrieval Augmented Generation (RAG) Model

Model Architecture

The RAG model presented here is a hybrid system that leverages both a **Retrieval** component and a **Generative** component.

1. **Retrieval Component:**
    - **Vector Store:** A FAISS index is used to store embeddings of text chunks extracted from the PDF document.
    - **Embedding Model:** HuggingFaceEmbeddings is used to generate embeddings for the text chunks.
    - **Retrieval:** When a query is provided, the model searches the vector store for similar embeddings and returns the most relevant text chunks.
2. **Generative Component:**
    - **LLM:** Ollama, a language model based on the LLaMA 3.1 architecture, is used as the generative component.
    - **Contextualization:** The retrieved text chunks are provided as context to the LLM, allowing it to generate responses that are relevant to the query and grounded in the provided information.

Approach to Retrieval

The model uses a similarity search approach to retrieve relevant information from the vector store. When a query is provided, its embedding is calculated using the same HuggingFaceEmbeddings model. This embedding is then compared to the embeddings of the stored text chunks using cosine similarity. The text chunks with the highest similarity scores are returned as context to the LLM.

How Generative Responses are Created

1. **Contextualization:** The retrieved text chunks are concatenated into a single context string and provided as input to the LLM.
2. **Generation:** The LLM generates a response based on the provided context and the query. The model's temperature setting (0 in this case) ensures that the generated responses are deterministic and focused on providing informative answers.
3. **Output:** The generated response is returned as the final output of the RAG model.

Overall Workflow

1. **Query Input:** A user provides a query.
2. **Embedding:** The query is embedded using the same HuggingFaceEmbeddings model.

3. **Retrieval:** The embedding is compared to the embeddings of the stored text chunks to retrieve the most relevant ones.
4. **Contextualization:** The retrieved text chunks are concatenated into a single context string.
5. **Generation:** The LLM generates a response based on the context and query.
6. **Output:** The generated response is returned to the user.

This RAG model provides a powerful framework for answering questions and generating informative text based on a given corpus of documents. By combining the strengths of retrieval and generation, it can effectively leverage the information contained in the documents to produce relevant and accurate responses.