HIPPO AI
by Pendium Health

Coding Assignment: Building a Retrieval-Augmented Generation System

Task Description:

Your task is to develop a retrieval-augmented generation system that can answer questions based on information extracted from three PDF documents. You will use OpenAI's GPT-3.5 API (or any other publicly available Language Model) as the foundational model for this task. The chosen tech stack includes Langchain for the Language Model framework, Conversational Retrieval Chain for context handling, Pinecone for the PDF vector store, and Jupyter Notebook for testing and demonstration. The focus is on creating a functional system within Jupyter Notebook, without requiring a UI or frontend development. Additionally, the system should handle follow-up questions, effectively implementing a chat history.

Requirements:

Language Model Integration: Utilize OpenAI's GPT-3.5 API (or any other suitable LLM) to handle the language generation part of the system. This includes generating answers to questions and follow-up queries.

Retrieval Component: Implement a retrieval mechanism using the Conversational Retrieval Chain provided by Langchain. This mechanism should take the user's question, retrieve relevant information from the three PDFs stored in Pinecone's PDF vector store, and pass this context to the Language Model for generating responses.

PDF Vector Store: Utilize Pinecone to set up the PDF vector store for efficient retrieval. You should be able to query the store using user questions and obtain relevant PDF excerpts.

Chat History: Develop a method to manage and maintain chat history. The system should allow for the inclusion of previous interactions, enabling context-aware conversations and accurate follow-up question handling.

Jupyter Notebook Implementation: Build the entire system within a Jupyter Notebook environment. Showcase how the retrieval-augmented generation works in a step-by-step manner, including interactions, PDF retrieval, context handling, and responses.

Documentation and Demonstration: Provide clear documentation within the Jupyter Notebook to explain the code logic, system architecture, and how the different components interact. Additionally, demonstrate the functionality of the system using example questions and follow-up interactions.