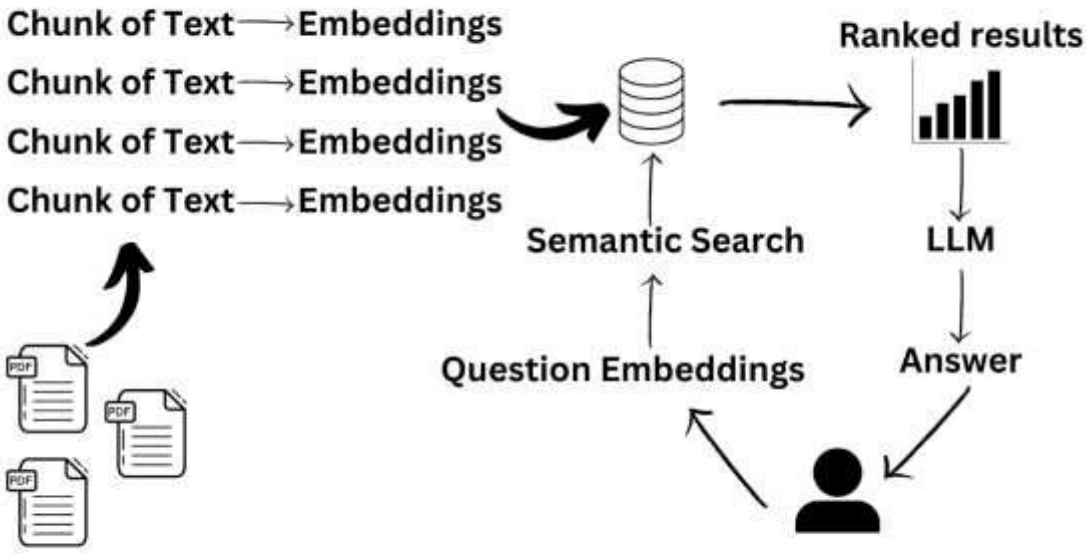
Chatbot Implementation using Langchain and Cohere Embeddings

Introduction

This document outlines the implementation of a chatbot that utilizes Langchain and CohereEmbeddings to provide accurate and relevant responses to user queries. The chatbot is designed to work with a PDF document, breaking it down into chunks of text, creating vector embeddings, and storing them in a Chroma vector store. When a user asks a question, the chatbot converts the query into vector embeddings and performs a similarity search to find the most relevant context. The final output is generated by a Large Language Model (LLM).

Technical Implementation



The chatbot uses the following technologies:

- Langchain for the backend
- Flask for the frontend
- Cohere Embeddings for vector embeddings
- Chroma for vector storage
- Groq for speed and accuracy (LLM)
- PyPDFLoader for loading PDF documents
- RecursiveCharacterTextSplitter for splitting documents into chunks
- ChatGroq for interacting with the LLM

Code Overview

The code consists of a DocumentResponder class that initializes the chatbot by loading the PDF document, splitting it into chunks, creating vector embeddings, and storing them in the Chroma vector store. The response method takes a user query and history as input, formats a prompt using a template, and invokes the retrieval chain to generate a response.

Key Components

- DocumentResponder class: Initializes the chatbot and provides a response method for user queries.
- PyPDFLoader: Loads the PDF document and splits it into chunks.
- RecursiveCharacterTextSplitter: Splits documents into chunks of text.
- Cohere Embeddings: Creates vector embeddings from the chunks of text.
- Chroma: Stores the vector embeddings in a vector store.
- ChatGroq: Interacts with the LLM to generate responses.
- PromptTemplate: Formats a prompt using a template and user input.

Future Development

The chatbot can be further improved by:

- Integrating with other social media platforms to increase its reach.
- Enhancing the user interface to provide a better user experience.
- Deploying machine learning algorithms to better understand user needs.

Conclusion

The chatbot implementation using Langchain and Cohere Embeddings provides a robust and accurate solution for responding to user queries. The code provides a solid foundation for further development and improvement.