CS467 - Capstone

Project Plan - Enhancing a LLM with Vector Stores for Improved Interpretation of Clinical Guidelines

Chris Mannina
July 9, 2023

Introduction

The ability of AI to comprehend, interpret, and apply complex datasets has substantial implications for clinical decision-making processes. This project aims to develop an application uses a Large Language Model (LLM) integrated with a vector database to interpret and apply clinical guidelines effectively.

Background

Clinical practice guidelines provide recommendations for healthcare practitioners about the care of patients with specific conditions. These medical guidelines are verbose and typically algorithmic in nature. They are used to guide decisions about appropriate and effective patient care. However, it's their complexity and volume that can be a hurdle in learning and quick decision-making processes.

A LLM, like GPT-3.5/4, is an advanced AI model capable of understanding and generating human-like text. It's trained on a vast array of internet text, and hence it can answer questions, write essays, summarize texts, translate languages, and even perform tasks like writing Python code. These models can generate diverse responses and are flexible in handling a wide array of tasks without needing task-specific training data. Instead of training a LLM, what if you could have it read and interpret a document to then use as a source of truth when asking subsequent questions.

Therefore, the thought arose that we can leverage a LLM to interpret and help answer patient care questions by referring to the medical guidelines. This project explores how we can accurately provide medical literature to a LLM to reference, and then proceding to ask questions.

We aim to accomplish this by enchaning GPT-3.5/4 with a vector store. A vector store, also known as a vector database, is a data storage and retrieval system designed to handle high-dimensional vector data. Vector stores make it possible to conduct similarity searches among vectors, which are often used to represent complex concepts, entities, or documents in a machine-readable format. In our case, this allows the model to draw upon the specific knowledge encapsulated in the guidelines.

In summary, by leveraging the combined potential of LLMs and vector stores, we aim to construct an application that can adeptly ingest, comprehend, and interpret medical documents. Once these documents are converted into vectorized form, the LLM can draw upon this information effectively, permitting users to pose queries that relate directly to the stored guidelines. This innovation is about intelligently bridging the gap between extensive clinical guidelines and real-time, relevant queries. Our project aims to facilitate efficient interpretation and utilization of clinical guidelines, offering significant prospects for enhancing healthcare decision-making processes.

Program Description from the User's Perspective

Users of this application will have the ability to upload clinical guidelines into the system. Once the guidelines are in the system, users can then ask the application questions related to those specific guidelines. The application, utilizing the LLM (GPT-3.5/4), will reference the provided guidelines to provide an accurate response. This system allows healthcare professionals to interpret and apply clinical guidelines more effectively and efficiently.

Initial Thoughts on the Software Structure

The general structure of the code will be modularized by:

1. Document/file ingestion: Allows users to upload PDFs, text documents, or URLs of medical guidelines.
2. Embedding Model: Creating embeddings and vectorization of documents.
3. Vector database: Will start with either Chroma or Pinecone for simplicity. Considering Postgres + pgvector if time permits.
4. User intput/query: Takes user input for a question.
5. LLM calls: Use OpenAI's API to connect our user with GPT-3.5/4.
6. User Interface (UI): Allows users to interact with our application by uploading guidelines, asking questions, and receiving responses. This will initially be done through CLI, but a stretch goal will be to create a webapp. The potential webapp will be done using either Flask, Gradio, or Streamlit. Ideally, Flask as it would be the most customizable and scalable, however Gradio/Streamlit are interesting proscpects for this use case.
7. Settings/Config options: Either through a settings file or constants, we can allow for features to be turned on/off or selected. The initial idea is to allow users to pick between GPT-3.5 and 4 models. Perhaps, we can use this for embedding models as well or even database connections.

Required Tools and Technologies

The project will require the following technologies:

1. Large Language Model: ChatGPT-3.5/4 (via OpenAI API key)

2. Embedding Model: text-embedding-ada-002 (OpenAI). Will then branch out to test others as well to see the accuracy difference.
3. Vector Database: Chroma or Pinecone to start with. Potentially utilize pgvector + Postgres.
4. LangChain: Framework for developing applications powered by language models. It enables applications that are data-aware (connect a language model to other sources of data), and agentic (allow a language model to interact with its environment). While LangChain will abstract some of the project away, it should allow me to build a prototype quickly, test, iterate, and improve. Once our prototype is built, we can then begin to customize for our use case. After researching, LangChain seems like the best option for this.
5. Flask/Streamlit/Gradio: Stretch goal will be to build a GUI for the user. Undecided on which technology I will take this. Since the project will be completed in python, it would be easiest to continue with python.
6. Docker: Stretch goal to containerize this program. I've never worked with docker, but am highly interested in learning if I have time.

High-Level Project Tasks

Note that I am approved to work solo on this project. All tasks below will be completed by myself.

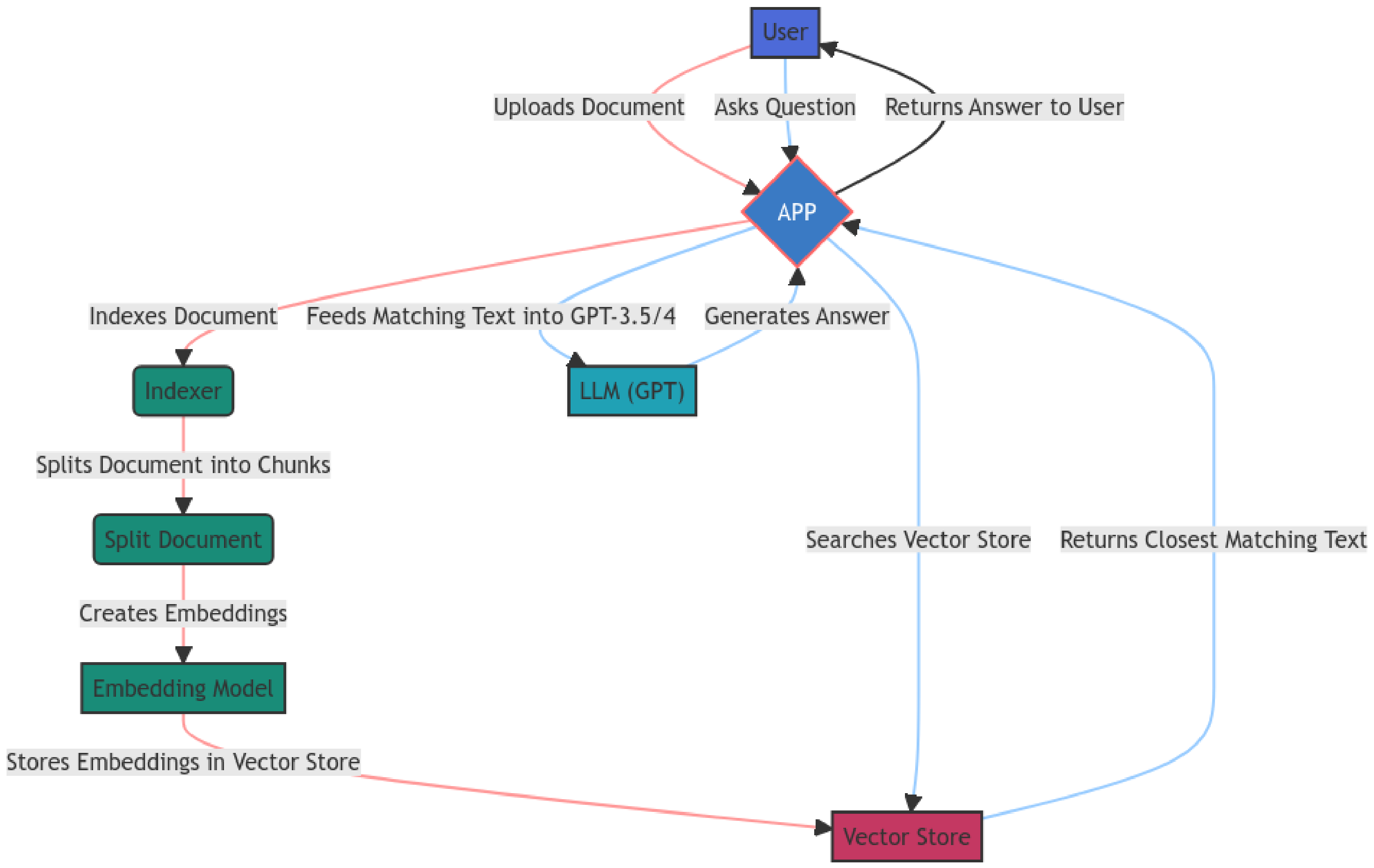
Week	Task	Time Estimate (Hours)
1	Study vector databases, OpenAI API docs, and LangChain docs.	10
2	Setting up the Initial Infrastructure and Experimentation with OpenAI API. Write code to send queries and provide responses using API calls.	10
3	Implment code for ingestion of documents. PDF will be main source of documents, however it should accept text and URLs. Documents should be split into managable chunks.	10
4	Implement embedding models and vector database. Using OpenAI's text-embedding-ada-002 to start, have the model vectorize the ingested document and store in Chroma or Pinecone.	10
5	Test and validate individual components. Continue development. Work on stretch goals if time permits. Additional application features will be added around this time as well.	10
6	Integrate all components and test.	10
7	Conduct end-to-end system tests. Meet with Clinical Pharmacists for testing and accuracy.	10
8	Implement multi-document capabilities if possible and experiment on the trade-off between the number of documents and accuracy.	10

Week	Task	Time Estimate (Hours)
9	Fine-tune the system based on test results (change embedding models, consider training embedded models, etc.)	10
10	Prepare documentation and reports.	10

Stretch Goals:

- GUI (Flask/Streamlit/Gradio)
- Mutli-document capabilities
- Persistent storage
- Options to choose embedding and LLM models
- Testing / comparison of embedding models

Program Flow Chart



This graphical example represents a basic layout of our proposed application, showcasing how the user will interact with the system to upload clinical guidelines and ask questions.

Other Requirements

- Clinical Expertise: Confer with Clinical Pharmacists to help test the application. They will be tasked with coming up with a question and answer set for testing.
- Mentor Input: Regular interaction with Karl Renius (boss and project mentor) to ensure that the project is progressing as planned. Discussion about direction, research, and potential writeup.

Conclusion

By combining the power of LLMs and vector databases, we aim to create a tool that could greatly simplify the interpretation of clinical guidelines for healthcare professionals.