Developing a medical LLM Chatbot for Personalised Mental Health Care

ANKITA SARKAR
P24AI0001
IIT JODHPUR
p24ai0001@iitj.ac.in

AVNI SINGH
R24AB0001
IIT JODHPUR
r24ab0001@iitj.ac.in

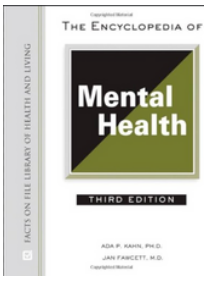
GOURAV SEN
M24CSA011
IIT JODHPUR
m24csa011@iitj.ac.in

ABSTRACT

This project focuses on creating a Medical Large Language Model (LLM) chatbot to help users with personal mental health questions using Retrieval-Augmented Generation (RAG). The chatbot uses advanced natural language processing(NLP) to understand what users are asking and give clear, reliable answers based on a wide range of medical knowledge. It provides accurate information on symptoms, diagnoses, treatments, medications, and general health advice. The goal is to give users quick and easy access to helpful mental health information. The chatbot connects to mental health medical databases to ensure that its answers are up-to-date and based on the latest research. It also keeps patient information private and secure.

1. DATASET

Our LLM model incorporates content from the Encyclopedia of Mental Health (Vol. I & II), 2nd Edition, edited by Howard S. Friedman and published on August 26, 2015, to deliver accurate and comprehensive information on mental health topics.



2. PROPOSED METHODOLOGY

2.1. Data Collection (Backend):

The initial step involves gathering medical data focused on mental health from diverse sources, including PDF documents and structured databases. This raw data is systematically processed, breaking it down into manageable text chunks. These chunks are carefully extracted to ensure they are relevant and ready for subsequent analysis.

2.2. Embedding Generation:

Once the text chunks are prepared, they are transformed into numerical vectors known as embeddings. This process employs Natural Language Processing (NLP) techniques to capture the semantic essence of the text. These embeddings serve as compact, machine-readable representations of the data, enabling more efficient comparison and retrieval in later stages.

2.3. Indexing (Backend):

The generated embeddings are organized into a semantic index. This index acts as a powerful backend structure that allows for rapid and precise retrieval of information from the knowledge base. By leveraging the semantic relationships embedded within the data, this indexing system ensures quick access to contextually relevant information.

2.4. Query Handling (Frontend):

When a user interacts with the chatbot, their question or prompt is received through the interface. The chatbot processes this input by converting it into an embedding using the same NLP model applied during the embedding generation phase for the knowledge base. This ensures consistency in understanding between the input query and the indexed knowledge.

2.5. Knowledge Base Search:

The query embedding is then compared against the semantic index, enabling the chatbot to identify and retrieve the most relevant text chunks from the database. The retrieved results are ranked based on their relevance to the user's query, ensuring that the most pertinent information is prioritized.

2.6. LLM Integration:

The top-ranked information from the knowledge base is used as input for the Large Language Model (LLM) in a process known as Retrieval-Augmented Generation (RAG). The LLM synthesizes a response by combining the contextual data retrieved from the knowledge base with its generative capabilities. This integration ensures that the responses are both contextually accurate and conversationally natural.

2.7. Response Delivery (Frontend):

Finally, the chatbot delivers a clear and concise response to the user. The output combines the retrieved and processed information with the LLM's ability to generate coherent and user-friendly text. The chatbot maintains a conversational tone, enhancing the user experience while ensuring the information provided is relevant and accurate.

The overall workflow is illustrated in Fig. 1, emphasizing the seamless integration of data retrieval, semantic indexing, and LLM capabilities to create a highly effective mental health chatbot.

3. TECHNICAL SPECIFICATIONS

- Embeddings used : openAI Embeddings
- Vector database used : FAISS (Facebook AI Similarity Search)
- LLM Framework used : Langchain

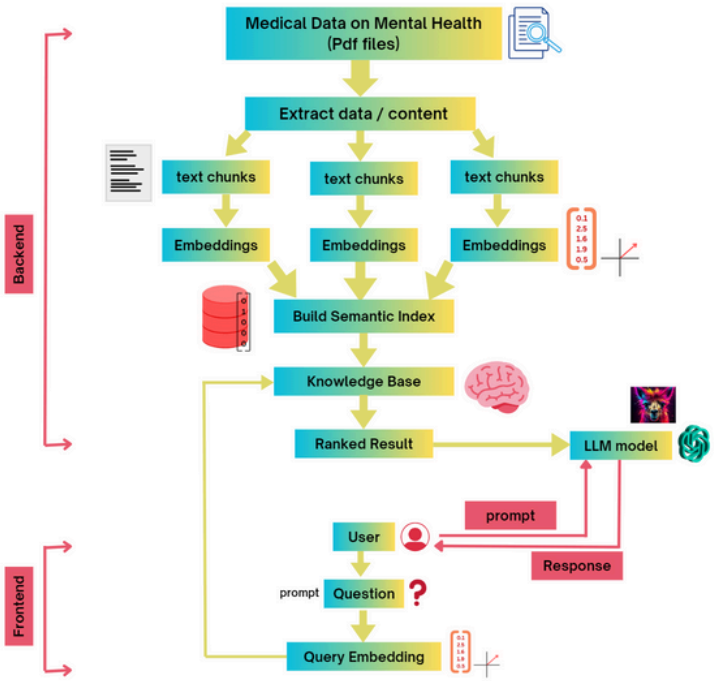


Fig. 1. Flowchart of the model

4. WORK DONE

The current code performs a series of steps to enable efficient retrieval-based question answering from a PDF document. Initially, it processes the given PDF by dividing its content into manageable text chunks. These chunks are then transformed into numerical representations, or embeddings, using OpenAI's embedding model. To facilitate quick and relevant information retrieval, a local vector database is built using FAISS (Facebook AI Similarity Search), which efficiently indexes the embeddings. When a user poses a question, the system retrieves relevant context from the local vector database by performing a similarity search between the question's embedding and the stored embeddings. The model then utilizes this retrieved-context, along with the question and any previous chat history, to generate an informative and coherent response. In simpler words, it reads the PDF and breaks the text into smaller chunks so that it's easier to work with. After that, it changes these text chunks into something called "embeddings," which are special representations of the text using OpenAI's embedding tools. These embeddings are then stored in a local vector database. The database is made using a tool called FAISS, which helps to store and search through these

embeddings quickly and find similar text. When a user asks a question, the code looks for the most relevant parts of the PDF by comparing the question with the stored embeddings in the vector database. It uses a process known as similarity search to find the chunks that are closest in meaning to the question. Once it finds these similar pieces of text, it combines them with the original question and the past conversation to give a more complete response. This helps the model to provide an answer that is more detailed and relevant based on the content from the PDF.

5. OUTPUTS & EVALUATION

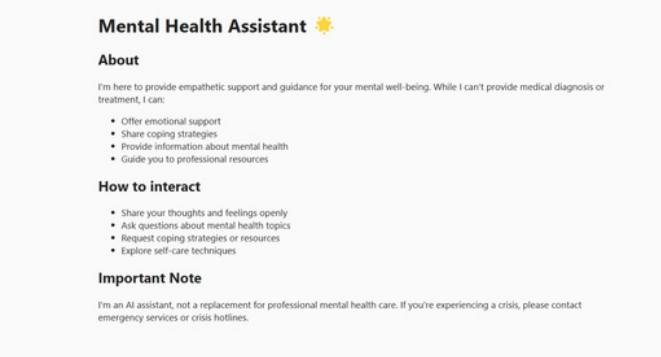


Fig 2 : Output 1

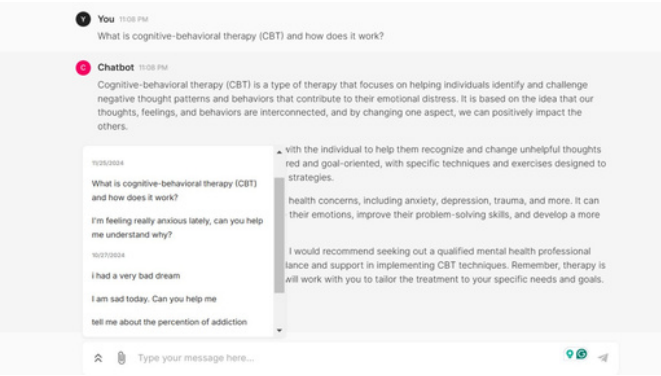


Fig 3 : Output 2

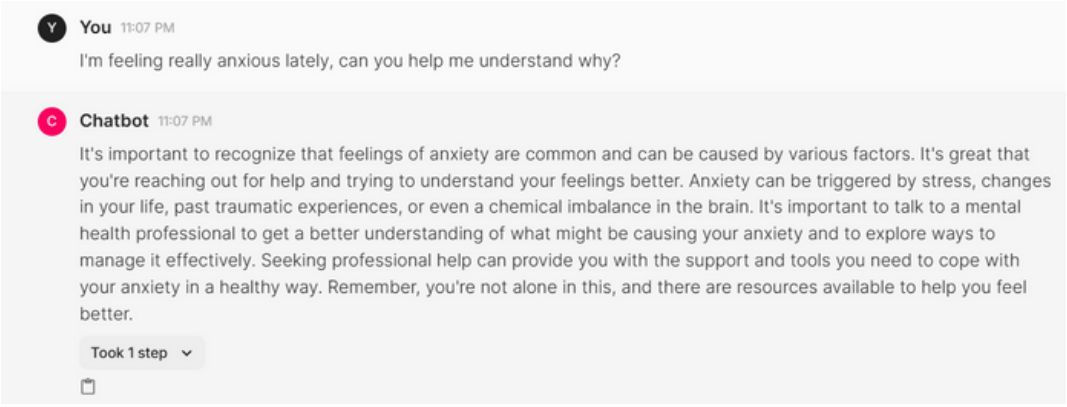


Fig 4 : Output 3

We created a small dataset for evaluation with 15 queries and their expected responses as the ground truth from the pdf. We evaluated our chatbot's responses using BLEU SCORE with a value of around 0 and Semantic Similarity (Cosine Similarity) between the generated responses and expected responses with a score of 75%. Hence, we were able to evaluate our chatbot better using Semantic similarity instead of word-to-word overlap done in calculating the BLEU score.

6. CODE LINK

https://github.com/sengourav/Mental-Health-Chatbot.git

7. REFERENCES

[1] Quidwai, M. A., & Lagana, A. (2024). A RAG Chatbot for Precision Medicine of Multiple Myeloma. medRxiv, 2024-03.

[2] Ghanbari Haez, S., Segala, M., Bellan, P., Magnolini, S.,Sanna, L., Consolandi, M., & Dragoni, M. (2024, July). A Retrieval-Augmented Generation Strategy to Enhance Medical Chatbot Reliability. In International Conference on Artificial Intelligence in Medicine (pp. 213-223). Cham: Springer Nature Switzerland.

[3] Torres, J. J. G., Bîndilă, M. B., Hofstee