Final Report: Therapist Helper Chatbot

1. Introduction

The **Therapist Helper Chatbot** project is designed to assist therapists by enhancing their interaction with patients during therapy sessions and to lighten therapists' administrative load so they can spend more quality time with patients. It integrates into a therapist's workflow and simplifies pulling up need-to-know details from previous sessions through natural questions by storing therapist-patient conversations as vector embeddings in Pinecone for efficient retrieval. The chatbot leverages OpenAI's text-embedding-ada-002 model for embedding generation and **Retrieval-Augmented Generation (RAG)**, this chatbot provides accurate and contextually relevant responses based on pre-embedded transcripts. The project demonstrates the integration of **Generative AI**, **Large Language Models (LLMs)**, and **LangChain** to optimize the chatbot's performance for therapeutic use cases. This report details the approach taken, challenges faced, and the solutions implemented.

Project Objectives:

- **Enhancement of RAG Pipeline:** Optimize the retrieval and generation process to ensure accurate and contextually relevant responses.
- **Creation of Custom Questionnaires**: Generate patient-specific questionnaires based on their current and desired emotional states.
- **Note-Taking Feature:** Enable real-time note-taking during therapy sessions with downloadable options.
- **Model Fine-Tuning:** Fine-tune the chatbot model to better align with therapy-specific contexts and improve overall performance.

2. You-Tube Link: https://youtu.be/SHKgwbI4lN4

3. Git Hub Link: https://github.com/Mansi-Dabriwal/TherapistHelperBot

4. Use Case Explanation

Generative AI, RAG, LLMs, and LangChain Integration:

The chatbot utilizes **Generative AI** to create text-based responses that are both contextually accurate and empathetically appropriate. The **RAG pipelin**e is central to

the project, as it combines retrieval-based methods with generative models. This ensures that responses are grounded in the actual content of therapy session transcripts, leading to more reliable interactions.

Large Language Models (LLMs) such as GPT-4o-mini are employed for both retrieval and generation tasks. The LLMs are responsible for understanding the context of user queries and generating suitable responses. **LangChain** is integrated to manage the chain of events in the chatbot's architecture, allowing for seamless transitions between retrieval and generation phases.

Key Features and Functionalities:

1. Contextual Response Generation: The chatbot retrieves relevant information from pre-embedded therapy session transcripts and generates responses based on that information.

2. Customized Questionnaire Creation: Therapists can create questionnaires tailored to the patient's current and desired emotional states. This functionality is particularly useful for tracking progress and guiding sessions also made available for download.

3. Real-Time Note-Taking: Therapists can take notes during sessions, which are then curated and made available for download. This feature helps in maintaining comprehensive session records.

4. Model Fine-Tuning: The chatbot can be fine-tuned using therapy-specific data to better cater to the needs of therapists and patients.

5. Approach

Data Collection and Preprocessing:

- I started by collecting therapist-patient conversations from a CSV file.
- The text data was cleaned to remove any missing values and irrelevant information.

- Example data:

Client_ID,Age, Gender, Transcript
1, John Doe, 35, Male,"Therapist: How are you feeling today? Patient: I'm feeling anxious." 2, Jaxonina Peter, 25, Female, "Therapist: Can you tell me more about that? Patient: I've been having trouble sleeping."

Generating Embeddings:

- We used the ‘text-embedding-ada-002’ model to convert each conversation into a vector representation.

- This model from OpenAI provides high-quality embeddings suitable for semantic similarity tasks.

Storing Embeddings in Pinecone:

- We initialized Pinecone, a scalable vector database, using an API key.

- The conversation embeddings were stored in Pinecone, allowing for efficient retrieval based on similarity.

- We created a mapping between conversation IDs and their text, saving this mapping to a JSON file for later use.

Querying the Vector Database:

- A function was implemented to convert user queries into vectors and perform a similarity search in Pinecone.

- The top relevant conversations were retrieved based on the similarity score.

- OpenAI's language model was used to generate responses by combining the retrieved conversation snippets with the user query.

- A Streamlit web application was set up to provide an interactive user interface for therapists.

6. Key Features and Functionalities

1. Customized Questionnaire Creation:

- This feature allows therapists to create questionnaires based on the patient's emotional state. The therapist selects the current and desired feelings of the patient, and the chatbot generates relevant questions.
- **Process:** The therapist selects the emotional states → The bot generates questions → The questionnaire is downloadable as a file.

2. Real-Time Note-Taking:
- During sessions, therapists can input notes into a text box. These notes are curated and available for download after the session.
- **Process:** The therapist inputs notes during the session → The bot curates the notes → The notes are downloadable as a file.

3. Model Fine-Tuning:
- The chatbot is fine-tuned to align more closely with therapy-specific contexts. This is achieved by training the model on therapy-related datasets, ensuring that it can handle therapy-related queries more effectively.

7. Retrieval Metrics Calculation

Context Precision:

- Measures the proportion of retrieved contexts that are relevant to the user's query.
- Calculation: Precision = Number of relevant contexts retrieved / Total number of contexts retrieved

Context Recall:

- Assesses the system's ability to retrieve all relevant contexts.
- Calculation: Recall = Number of relevant contexts retrieved / Total number of relevant contexts

Context Relevance:

- Evaluates the relevance of the retrieved contexts using cosine similarity between query and context embeddings.

- Calculation: Average cosine similarity score between query embedding and retrieved context embeddings.

Noise Robustness:

- Tests the system's performance under noisy inputs by adding random noise to the query.
- Calculation: Precision and recall metrics are recalculated after introducing noise to the query.

8. Generation Metrics Calculation

Faithfulness:

- Measures how accurately the generated answers reflect the reference answers.
- Calculation: Faithfulness = Number of correct answers / Total number of reference answers

Answer Relevance:

- Evaluates how relevant the generated answer is to the query.
- Calculation: Cosine similarity between the query and the generated answer embeddings.

Information Integration:

- Assesses the ability of the system to integrate and present information cohesively.

- Calculation: Cosine similarity between the combined embeddings of retrieved contexts and the generated answer.

Counterfactual Robustness:

- Measures the system's response to counterfactual or contradictory queries.
- Calculation: Proportion of answers that appropriately differ from counterfactual queries.

Negative Rejection:

- Tests the system's ability to reject or handle inappropriate queries.
- Calculation: Proportion of answers that appropriately handle or reject negative queries.

Latency:

- Measures the time taken from receiving a query to delivering an answer.
- Calculation: Time taken in seconds for retrieval and generation processes.

9. Retrieval Metrics-

- Context Precision: 1.0
- Context Recall: 0.75
- Context Relevance: 0.687
- Noise Robustness Precision: 1.0

10. Generation Metrics

- Faithfulness: 0.0
- Answer Relevance: 0.696
- Information Integration: 0.54
- Counterfactual Robustness: 0.25 - Negative Rejection: 0.0
- Latency: 1.476

11. Challenges Faced

Data Cleaning and Preprocessing:
- **Challenge**: Ensuring data quality and consistency was crucial for generating accurate embeddings.
- **Solution:** Implemented deep data cleaning processes to handle missing values and irrelevant information.

Choosing the Right Embedding Model:
- **Challenge:** Selecting an embedding model that balances performance and efficiency.
- **Solution:** After evaluating several models, I chose 'text-embedding-ada-002' for its high-quality embeddings.

Scalability of Storage:
- **Challenge:** Efficiently storing and retrieving large amounts of vector data.
- **Solution:** Pinecone was chosen for its scalability and speed, making it suitable for handling high-dimensional vector data.

Generating Contextually Accurate Responses:
- **Challenge:** Ensuring the chatbot provides contextually relevant and accurate responses.

- **Solution:** By combining similarity search results with OpenAI's language model, I achieved high-quality responses tailored to user queries.

User Interface Design:
- **Challenge:** Creating an intuitive and easy-to-use interface for therapists.
- **Solution:** Streamlit was used to build a simple and interactive web application, allowing therapists to input queries and receive responses seamlessly.

Handling Noisy Inputs:
- Ensuring that the chatbot effectively manages irrelevant or noisy inputs without compromising the quality of retrieved contexts was a significant challenge. Various filters and validation checks were implemented to address this issue.

Token Limit Management:
- Managing the token limits in both retrieval and generation stages was crucial to maintaining the efficiency and effectiveness of the chatbot. This required optimizing the prompt design and implementing dynamic token management strategies.

Ensuring Faithfulness:
- Another challenge was ensuring that the generated responses remained faithful to the original therapy session transcripts. Cross-verification of responses was introduced to tackle this issue.

12. Conclusion and Future Scope

The **Therapist Helper Chatbot** has successfully demonstrated the integration of RAG, Generative AI, LLMs, and LangChain to enhance therapy sessions. The chatbot not only provides contextually accurate responses but also offers valuable features such as customized questionnaire creation and real-time note-taking.

Future Scope:
- **Enhanced Emotional Intelligence:** Further fine-tuning to improve the chatbot's ability to recognize and respond to complex emotional states.
- **Voice Interaction Integration:** Adding voice recognition and response generation capabilities to make the chatbot more accessible.
- **Expanded Dataset:** Incorporating a broader range of therapy session transcripts to improve the chatbot's versatility and accuracy.

The **Therapist Helper Chatbot** represents a significant step forward in leveraging AI to support mental health professionals. With continuous improvement and the integration of additional features, it has the potential to become an indispensable tool in therapy settings.

13. Acknowledgments

I would like to thank the developers of OpenAI, Pinecone, and Streamlit for their powerful tools and libraries that made this project possible.

14.References

- OpenAI: https://www.openai.com/
- Pinecone: https://www.pinecone.io/
- Streamlit: https://www.streamlit.io/