Anirudha Joshi (NUID 002991365)

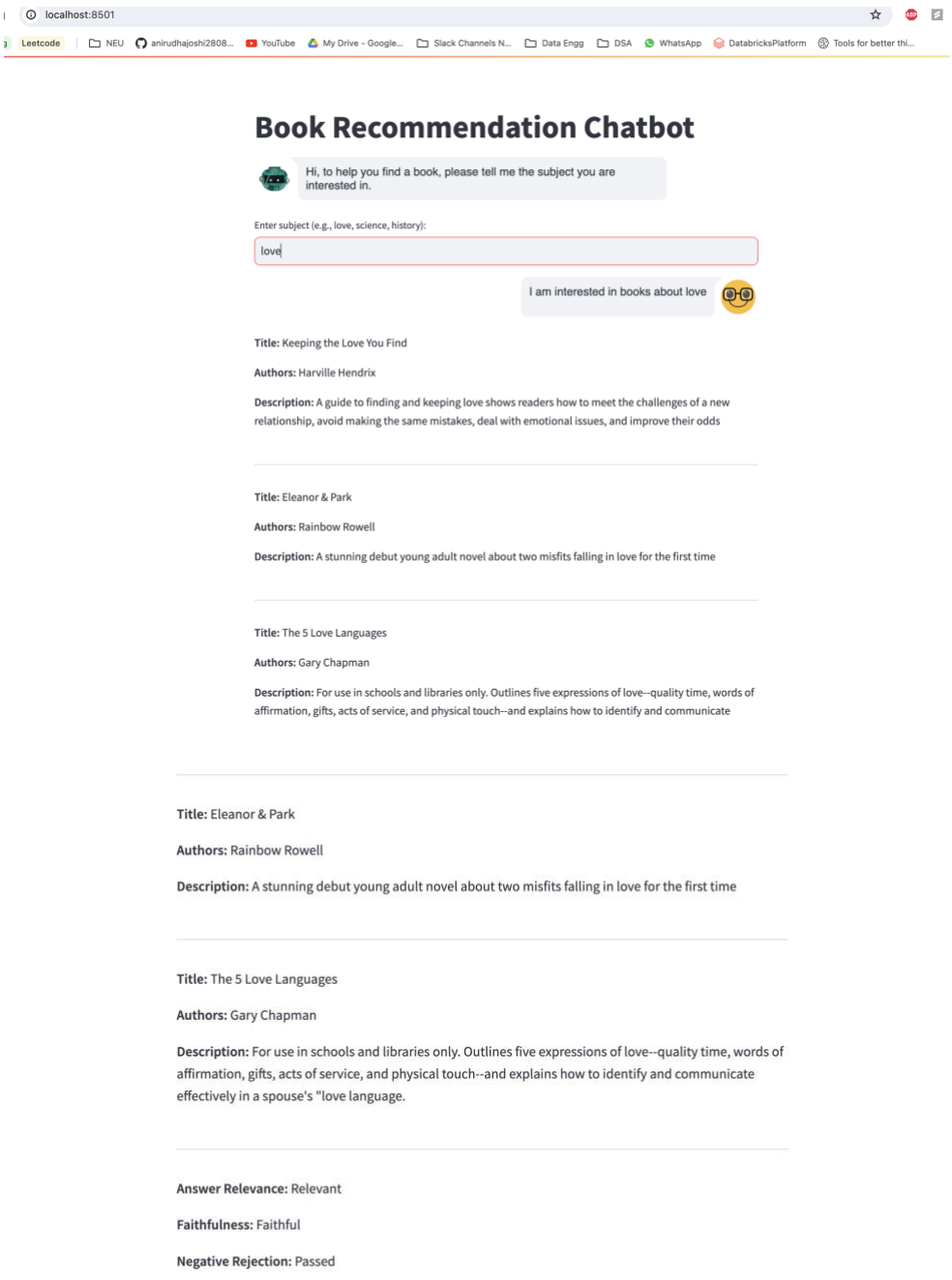
Prompt Engineering & AI Summer 2024

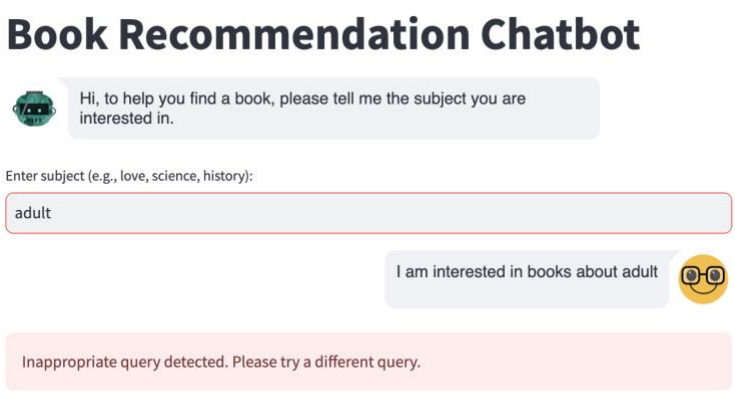
Adaptive Recommendation Chatbot with RAG and Vector Database

Introduction

This report outlines the development process of a domain-specific chatbot designed to recommend books based on user-specified subjects. The project leverages Pinecone for vector storage, OpenAI's GPT-4 for natural language processing, and a retrieval-augmented generation (RAG) approach to enhance the relevance and quality of recommendations. The chatbot interface is built using Streamlit.

Screenshots





Approach Taken

1. **Data Collection and Preprocessing**:
   - **Data Fetching**: We used the Open Library API to fetch book data based on user-specified subjects. Each query fetches up to 25 books to ensure a manageable dataset size for processing and embedding generation.
   - **Preprocessing**: Data preprocessing involved cleaning and structuring the fetched data to ensure consistency. Books without descriptions were populated with their titles as placeholders.
2. **Embedding Generation**:
   - **Model Selection**: SentenceTransformer from HuggingFace (all-MiniLM-L6- v2) was chosen for embedding generation due to its balance between performance and computational efficiency.
   - **Embedding Process**: Each book's description was converted into embeddings, which were then used to represent the books in the vector space.
3. **Storage in Pinecone**:
   - **Index Creation**: A Pinecone index was created to store the embeddings. The index was configured with cosine similarity as the metric.
   - **Vector Upsert**: The embeddings, along with metadata (title, authors, description, and subject), were upserted into the Pinecone index, allowing for efficient similarity search during recommendations.
4. **Chatbot Interface**:
   - **Streamlit Integration**: The chatbot interface was developed using Streamlit, providing a user-friendly way to interact with the system.
   - **User Query Processing**: User inputs were processed using GPT-4 to understand and refine the queries before fetching book recommendations.
   - **Recommendation Generation**: Recommendations were fetched from Pinecone based on the processed user input and displayed on the Streamlit interface.
5. **Evaluation Metrics**:
   - **Faithfulness**: Ensured the generated answers accurately reflect the source data.

- **Answer Relevance**: Assessed the relevance of the generated answers to the user's query.
- **Information Integration**: Evaluated the chatbot's ability to cohesively integrate and present information.
- **Counterfactual Robustness**: Tested the system's robustness against contradictory queries.
- **Negative Rejection**: Verified the system's ability to reject and handle inappropriate queries.

Challenges Faced and Solutions

1. **Handling Large Data Volumes**:
   - **Challenge**: Processing and storing large volumes of book data while maintaining performance.
   - **Solution**: Limited the number of fetched books to 25 per query to ensure manageable data processing and embedding generation. Pinecone's efficient indexing and query capabilities helped maintain performance.
2. **Ensuring Relevant and Faithful Recommendations**:
   - **Challenge**: Ensuring that the recommendations are both relevant to the user's query and faithful to the source data.
   - **Solution**: Implemented relevance and faithfulness checks within the Streamlit interface. Used GPT-4 to process user inputs and refine queries for better recommendation accuracy.
3. **Inappropriate Query Handling**:
   - **Challenge**: Handling and rejecting inappropriate or out-of-context queries.
   - **Solution**: Added a function to detect inappropriate queries based on predefined keywords and reject them with an appropriate message.
4. **Integrating Pinecone with the Latest API Changes**:
   - **Challenge**: Adapting to changes in Pinecone's API, such as the removal of the `init` function.
   - **Solution**: Followed Pinecone's updated documentation to correctly initialize and interact with the Pinecone index, ensuring compatibility and functionality.
5. **Counterfactual Robustness**:
   - **Challenge**: Ensuring the chatbot can handle counterfactual or contradictory queries effectively.
   - **Solution**: Implemented checks within the chatbot to identify and appropriately respond to counterfactual queries, enhancing robustness.

Conclusion

The development of the book recommendation chatbot involved integrating several advanced technologies to create a robust and user-friendly system. By leveraging Pinecone for efficient vector storage and retrieval, HuggingFace's SentenceTransformer for embedding generation, and OpenAI's GPT-4 for natural language processing, we were able to build a system capable of delivering relevant and accurate book recommendations. The addition of evaluation metrics and checks for inappropriate queries ensured the system's reliability and robustness. This project demonstrates the potential of combining multiple AI and ML tools to create sophisticated and practical applications.