Report on FAQ Chatbot Development

Introduction

The goal of this project was to build an FAQ chatbot that efficiently retrieves relevant answers while handling variations in user queries. As the primary retrieval technique, I concentrated on Facebook AI Similarity Search (FAISS) and investigated various large language models (LLMs) for response generation.

Research and Approach

Initial Considerations

1. **Simple Rule-Based Retrieval Methods**:
    - Initially, I considered rule-based retrieval techniques such as keyword matching and TF-IDF (Term Frequency-Inverse Document Frequency).
    - These methods are computationally inexpensive and easy to implement but lack semantic understanding, making them ineffective for varied phrasings of the same question.
    - Due to their limitations in handling complex queries, I opted for FAISS, which provides more robust similarity matching.
2. **Using FAISS Alone for Retrieval**:
    - FAISS enables fast retrieval of similar questions from a pre-indexed FAQ dataset.
    - However, FAISS alone only retrieves similar entries without understanding nuances or generating natural language responses.
    - It struggles with cases where exact matches are unavailable or when a more conversational response is needed.
    - Due to these limitations, I decided to enhance FAISS with an LLM for better response generation.
3. **Retrieval-Augmented Generation (RAG) with FAISS and LLM**:
    - FAISS retrieves the most relevant FAQ entry.
    - An LLM refines and generates the response based on retrieved data, ensuring fluency and coherence.
    - This approach combines fast retrieval with intelligent text generation, leading to more precise and natural interactions. So I moved forward with implementing it.
4. **Explored LLMs for Response Generation**:
    - I tested different transformer-based models for text generation, including:
        - **GPT-2 models** for their fluency and coherence but struggled with generating coherent responses.

        - **Mistral-7B-v0.1** for its good performance but too heavy weight for such a simple task.
        - **h2oai/h2o-danube3-500m-chat** for lightweight performance with high accuracy which I settled on.
    - The final choice was based on balancing accuracy, response coherence, and computational efficiency choosing **h2oai/h2o-danube3-500m-chat**.

Functional Implementation

1. Building the Knowledge Base

- The FAQ dataset was scraped from Danson Solutions official website.
- The FAQ dataset was embedded using **all-MiniLM-L6-v2**, creating vector representations of questions.
- FAISS indexed these embeddings, allowing rapid similarity search.
- When a user submits a query, FAISS retrieves the closest match.

2. Query Processing Pipeline

- **Embedding the User Query**: The input query is converted into an embedding using the same model as the FAQ dataset.
- **Retrieval from FAISS**: FAISS identifies the nearest matching question from the knowledge base.
- **Response Generation Using LLM**:
    - If the confidence level is moderate to high, the chatbot first checks whether the user's query is a greeting. If it is, the chatbot responds with a greeting and encourages the user to begin FAQ queries. Otherwise, it retrieves a relevant question and answer using FAISS and passes them to the LLM to generate a coherent response.
    - If confidence is low (< 0.5), the chatbot notifies the user that it cannot answer the query.

3. API Development with FastAPI

- Implemented a REST API using FastAPI to handle user queries.
- The API processes the input, retrieves responses using FAISS, and refines them via LLM before returning a structured JSON response.

4. Frontend UI with Streamlit

- Built an interactive chat interface using Streamlit.
- Messages were dynamically updated in real-time.
- Ensured a smooth and intuitive user experience.

Challenges and Solutions

1. **Response Quality Optimization**:
    - Adjusted LLM hyperparameters like temperature and max tokens to prevent verbosity or irrelevant responses.
2. **Handling Uncertain Queries**:
    - Introduced a confidence threshold to avoid misleading responses.
3. **Performance Considerations**:
    - Used a compact LLM to reduce latency while maintaining accuracy.
4. **Prompt Engineering** :
    - Enhancing the prompt through various iterations for the exact required performance.

Conclusion

This project effectively created an advanced FAQ chatbot by leveraging FAISS for efficient retrieval and LLMs for generating responses. The integration of FAISS for similarity matching, coupled with LLM-driven refinements, ensures both accuracy and a natural conversational flow. Potential enhancements include support for multi-turn interactions, voice-based inputs, and adaptive learning through real-time user feedback.