ChatGPT 3.5 Chatbot

Documentation:

Overview:

The NutriHelp Chatbot is a web-based application, designed to provide information and assistance related to nutrition leveraging Gpt3.5 extensive knowledge, as well as answer questions about NutriHelp at Deakin. This document serves as an extensive documentation report for the chatbot codebase.

Prerequisites:

Before deploying and using the NutriHelp Chatbot, ensure the following prerequisites are met:

- Python 3.x installed on the system.
- Flask: A micro web framework for Python.
- Flask-Session: An extension for managing user sessions in Flask applications.
- OpenAI Python library: Used for accessing OpenAI's GPT models.

Code Overview:

The NutriHelp Chatbot codebase is organized into distinct components, each serving a specific purpose within the application's architecture. After importing necessary libraries, we have the **Secret Key Generation**, where functions for generating secret keys for Flask application and session management are defined, These keys play a crucial role in ensuring the security and integrity of the application’s communication and data storage mechanisms.

Then we have the **Flask application configuration** where the configuration settings for the Flask application and session management are specified. These configurations dictate various aspects of the application’s behaviour. After this we have the **Relevance Check Function** which is responsible for determining the relevance of a user’s question to NutriHelp or nutrition. Here we make out first API call, and leverage OpenAI’s GPT model, it evaluates the semantic context of the question and provides a binary classification indicating relevance.

We then have the main **GPT Response Generation Function** which facilitates interaction with the GPT model to generate responses to user inquiries. It manages the conversation history using Flask session and utilizes the relevance check function to ensure that only relevant questions are processed.

Towards the end of the script we have the **Flask Routes** which include various HTTP requests received by the application These routes serve specific endpoints such as rendering the chat interface, processing user queries, and resetting session data. Finally the **Main Function** serves as the entry point of the application, it initializes and runs the Flask server, allowing the application to handle incoming requests and responds accordingly.

Usage:

- Clone the repository containing the NutriHelp Chatbot code.
- Run the Flask application using python app.py.
- Access the chat interface via a web browser at http://localhost:5000/.
- Enter questions related to NutriHelp or nutrition, and interact with the chatbot.

Future Improvements:

While the NutriHelp Chatbot leverages OpenAI's GPT model to provide insightful responses to user inquiries regarding nutrition, there are several areas for enhancement and refinement to further elevate its functionality and user experience.

- Prompt Engineering for Domain Constraints:

Although the GPT model possesses extensive knowledge about nutrition, it's essential to implement prompt engineering techniques to constrain its responses within the domain of nutrition and NutriHelp-related topics. By fine-tuning the prompts provided to the model and incorporating domain-specific keywords and constraints, we can ensure that the chatbot's responses remain relevant and accurate. This approach helps mitigate the risk of the model generating irrelevant or misleading information outside the intended scope.

- Integration with External Databases:

To augment the chatbot's knowledge base and provide more comprehensive responses, integrating it with external databases containing curated nutritional information could be beneficial. By accessing verified and up-to-date nutritional data from reputable sources, the chatbot can offer more precise and detailed responses to user queries. Additionally, linking the chatbot to databases that store information about NutriHelp itself, including its projects, initiatives, and research findings, enables the chatbot to answer questions about the company with greater depth and accuracy.

- Enhanced Personalization and User Authentication:

Implementing features such as user authentication allows the chatbot to personalize interactions based on individual user profiles and preferences. By capturing user data such as dietary preferences, health goals, and medical history (with appropriate consent and privacy measures), the chatbot can tailor its recommendations and advice to better suit the needs of each user. Personalization not only enhances user engagement but also improves the relevance and effectiveness of the chatbot's responses.

- Continuous Training with Relevant Datasets:

To further enhance the chatbot's understanding and response capabilities, continued training with relevant datasets is crucial. By periodically updating the model with new data and insights from the field of nutrition and MedTech, we can ensure that the chatbot remains current and informed about the latest developments, trends, and research findings. This iterative learning process enables the chatbot to evolve over time and adapt to changing user needs and preferences.

- User Interface Enhancements:

Improving the user interface (UI) of the chatbot can contribute to a more engaging and intuitive user experience. Incorporating features such as interactive elements, visual aids, and natural language understanding (NLU) capabilities enhances usability and accessibility for users of varying backgrounds and proficiency levels. A visually appealing and user-friendly interface encourages users to interact more frequently with the chatbot, fostering deeper engagement and satisfaction.