CREATE A CHATBOT IN PYTHON

Team Leader : Saravanakkumar D
Team Member 1 : Kumarakabilan P
Team Member 2 : Vijay I
Team Member 3 : Veerendran A
Team Member 4 : Saran R

Problem Definition : The challenge is to create a chatbot in Python that provides
exceptional customer service, answering user queries on a website or application. The
objective is to deliver high-quality support to users, ensuring a positive user experience and
customer satisfaction.

Design Thinking :
1. Functionality: Define the scope of the chatbot's abilities, including answering common questions, providing guidance, and directing users to appropriate resources.
2. User Interface: Determine where the chatbot will be integrated (website, app) and
design a user-friendly interface for interactions.
3. Natural Language Processing (NLP): Implement NLP techniques to understand and
process user input in a conversational manner.
4. Responses: Plan responses that the chatbot will offer, such as accurate answers,
suggestions, and assistance.
5. Integration: Decide how the chatbot will be integrated with the website or app.
6. Testing and Improvement: Continuously test and refine the chatbot's performance
based on user interactions.

Phases of development :

1. Data Collection
- Acquire the dataset necessary for the project.

- Evaluate the data sources and determine the relevance and quality of the data.

2.Data Preprocessing

- Handle missing data: Impute missing values or remove instances.
- Data cleaning: Address outliers, inconsistencies, or errors in the dataset.
- Data transformation: Normalize, scale, or encode features as required.
- Split data: Divide data into training, validation, and test sets.

3. Feature Engineering

- Extract relevant features from the dataset or create new features based on domain knowledge.
- Feature selection: Identify the most informative features for the model.

4. Model Selection and Design

- Choose appropriate machine learning algorithms based on the problem type (e.g., classification, regression).
- Experiment with different models and architectures to select the most suitable one.
- Define the structure of the model and its components.

5. Model Training and Validation

- Train the chosen model using the training dataset.
- Validate the model's performance using the validation dataset and fine-tune hyperparameters.

6. Model Evaluation

- Evaluate the model's performance using appropriate metrics (accuracy, precision, recall, F1-score, etc.).
- Compare and analyze the model's performance against baseline or benchmark models.

7. Model Optimization and Tuning

- Fine-tune the model by adjusting hyperparameters or employing techniques like regularization or feature selection.

8. Testing and Deployment

- Test the model's performance on the test dataset, unseen during training or validation.
- If satisfactory, deploy the model in a production environment for real-world use.

9. Monitoring and Maintenance

- Monitor the model's performance in the production environment.
- Maintain the model by retraining it with new data and making necessary updates.

10. Executing
integrating it into a web app using Flask

Dataset Description and Preprocessing Steps:

Dataset : https://www.kaggle.com/datasets/grafstor/simple-dialogs-for-chatbot consists of dialogs, which presumably include questions and their corresponding answers for a chatbot. In the code, the dataset is loaded from a file named `dataset.xlsx`.

The data preprocessing steps conducted in the code include:

1. Loading the dataset from the Excel file using Pandas.
2. Creating a knowledge base dictionary from the dataset where questions serve as keys, and answers [Answer1 and Answer2] act as the respective values.
3. Handling missing values: If Answer1 exists, it's associated with the question; otherwise, Answer2 is used. This process populates the knowledge base dictionary.

Feature Extraction Techniques:
In this context, feature extraction involves converting the dialogue-style dataset into a question-answer format. The questions are the features used to query the chatbot for responses, and the corresponding answers are the targets generated by the chatbot.

Choice of Machine Learning Algorithm, Model Training, and Evaluation Metrics:

The selected machine learning model for generating responses is GPT-2 [Generative Pre-trained Transformer 2 or eiher we use GPT-3]. GPT-2 or GPT-3 is a state-of-the-art language generation model developed by OpenAI. The code initializes and uses a pre-trained GPT-2 and GPT-3 model and tokenizer provided by the `transformers` library.

The model training is not explicitly performed in this code since GPT-2 is a pre-trained model. The model is fine-tuned or conditioned based on the loaded dataset during inference by generating responses to user-input questions.

The evaluation metric for the chatbot performance isn't explicitly measured in the provided code snippet. In practical scenarios, evaluation metrics such as BLEU score, perplexity, or human evaluation could be employed to assess the quality of generated responses against ground truth answers.

Innovative Techniques or Approaches:

The innovation in this code lies in the use of a pre-trained language model like GPT-2 and GPT-3 for generating conversational responses. The model is capable of understanding the context of the input question and generating relevant and contextually appropriate responses.

This approach streamlines the process of creating a conversational chatbot by utilizing pre-existing language models, potentially reducing the need for extensive hand-crafted feature engineering.

To further enhance the chatbot's performance, iterative fine-tuning on the dataset or employing more advanced models might be considered. Additionally, implementing a more extensive evaluation setup to measure the chatbot's response quality and coherence could be beneficial.