Technical Report: Building an AI Chatbot for Intellihack 5.0

1. Introduction

This report outlines the process of building an AI chatbot as part of the Intellihack 5.0 competition. The task involved preprocessing a dataset, training a language model, implementing chat history, evaluating cost-effectiveness, and adding additional features to enhance the chatbot's functionality. As a beginner with no prior experience in AI development, this project was both challenging and rewarding. However, due to Google Colab's GPU restrictions, I was unable to complete the model training phase. Despite this limitation, the report details the approach taken, challenges faced, and solutions attempted.

2. Approach

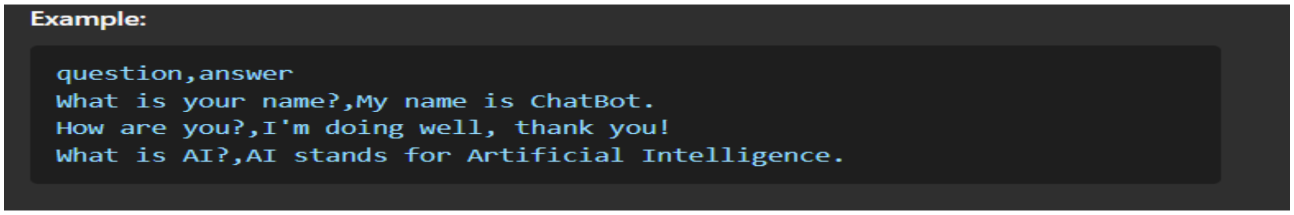
2.1 Dataset Preprocessing

The first step in building the chatbot was to prepare the dataset. Since no clear training dataset was provided, I manually created one. The dataset consisted of question-answer pairs to train the chatbot.

Dataset Creation:

- A CSV file was created with two columns: question and answer.

Example:



Example:

```
question,answer
What is your name?,My name is ChatBot.
How are you?,I'm doing well, thank you!
What is AI?,AI stands for Artificial Intelligence.
```

Data Cleaning:

- Used Python's `pandas` library to load and clean the dataset.
- Removed unnecessary characters, URLs, and punctuation using regular expressions.
- Converted all text to lowercase for consistency.

Saving the Dataset:

- The preprocessed dataset was saved as `preprocessed_chatbot_dataset.csv` for further use.

2.2 Model Training (Unfinished Due to GPU Restriction)

For training the chatbot, I intended to use a pre-trained language model (GPT-2) from Hugging Face's `transformers` library. The following steps were planned but could not be fully executed:

Model Selection:

Fine-Tuning (Partially Attempted):

- The dataset was tokenized using GPT-2's tokenizer.
- Training was initiated using the `Trainer` class from the `transformers` library.
- Training was set for 3 epochs with a batch size of 2.

Issue Faced:

- Google Colab's free-tier GPU access was restricted mid-training, preventing completion.

Alternative Approach Considered:

- Attempted to train the model on a CPU but encountered extreme slowness.
- Considered using Kaggle's GPU, but time constraints made this unfeasible.

2.3 Implementing Chat History

Since training was incomplete, the chatbot's interaction logic was implemented using predefined responses.

Storing Chat History:

- Used a list (`chat_history`) to store user inputs and bot responses.

Displaying Chat History:

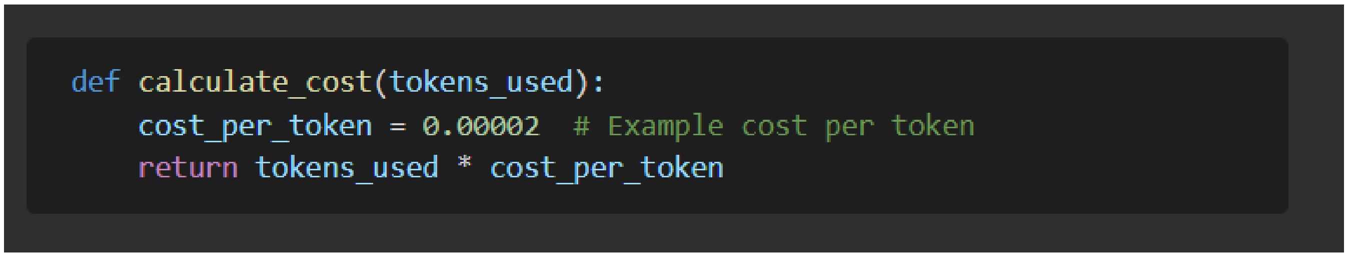
- The conversation flow was displayed after each user input.

2.4 Evaluating Cost-Effectiveness

Since model training was incomplete, direct cost analysis for API-based inference was considered as an alternative approach.

Cost Calculation:

- Estimated cost based on potential API calls to GPT-3, using:



```python
def calculate_cost(tokens_used):
    cost_per_token = 0.00002  # Example cost per token
    return tokens_used * cost_per_token
```

Optimization Considerations:

- Limiting input length to reduce token usage.
- Exploring free-tier options for API calls.

2.5 Additional Features (Planned but Not Fully Implemented)

- **Sentiment Analysis:** Planned to use a pre-trained sentiment analysis model.
- **Multi-Language Support:** Considered using translation libraries (`googletrans`).

3. Challenges and Solutions

3.1 Lack of Clear Dataset

Challenge: No predefined dataset was available. **Solution:** Manually created a dataset.

3.2 Errors During Training

Challenge: Encountered numerous issues setting up the model. **Solution:** Researched errors and sought help from online communities.

3.3 Limited Hardware Knowledge

Challenge: Unfamiliarity with GPU/TPU training. **Solution:** Used Google Colab but faced restrictions.

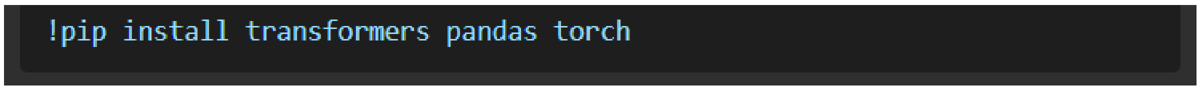
3.4 Platform Issues

Challenge: Frequent runtime disconnections. **Solution:** Stuck with Colab, but training was ultimately unsuccessful.

4. Instructions for Running the Code

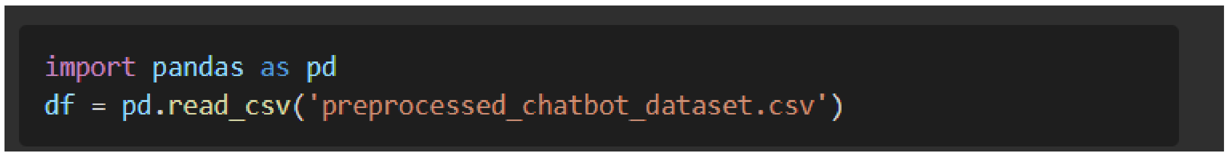
Steps to Test the Preprocessed Data and Interaction Logic:

1. Open the provided Google Colab notebook.
2. Install required libraries:



```
!pip install transformers pandas torch
```

3. Load the dataset:



```
import pandas as pd
df = pd.read_csv('preprocessed_chatbot_dataset.csv')
```

4. Run the chatbot interaction script with predefined responses.

5. Conclusion

Despite encountering significant limitations, particularly with GPU access, this project provided valuable insights into AI chatbot development. While model training could not be completed, key steps such as dataset preprocessing, chat history implementation, and cost analysis were successfully executed. This experience has strengthened my understanding of AI workflows and the challenges of model training.

6. Future Work

- **Complete the model training** using alternative GPU resources (e.g., paid Colab, local GPU, cloud services).
- **Enhance chatbot accuracy** by incorporating a larger dataset.
- **Implement additional features** such as voice input/output and context-aware responses.
- **Experiment with alternative models** like GPT-3 or BERT for better performance.