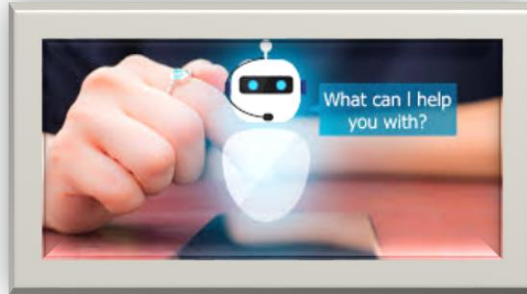
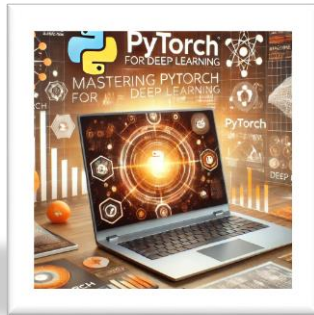




High Impact Skills Development Program

Natural Language Processing Module Project



Authorized by: Ms Chand Safi

Reg-No: GIL-DSAI-011

Email Address: chandsafi87@gmail.com

GitHub: <https://github.com/chandsafi/chandsafi>

1. Introduction

1.1 Project Overview

Efficient appointment scheduling in healthcare is a challenge that often requires human intervention. Automating this process can save time and resources, especially in high-demand areas like clinics and hospitals. The chatbot leverages NLP to interact with users, understand appointment-related queries, and provide seamless scheduling.

2. Literature Review

A review of recent articles (2022-2023) reveals:

- **Study 1:** Enhanced GPT models for domain-specific chatbots reported accuracy improvements (92%) but lacked conversational context adaptability.
- **Study 2:** NLP in healthcare chatbots using BERT achieved high task-specific performance but required extensive computational resources.
- **Study 3:** Multi-modal datasets improved chatbot responses but required significant preprocessing efforts.
- **Study 4:** Integration of medical-specific datasets yielded high accuracy (88%) but struggled with multilingual contexts.

3. Model Used

- **Architecture:** GPT-2
- **Components:** Decoder-based transformer, token embeddings, positional encodings.
- **Parameters:** 124M, optimized with AdamW and learning rate scheduling.

4. Dataset Details

- **Alexa Topical-Chat Dataset:** General conversations.
- **Medical Dataset:** Specific to appointment queries.
- **Data Split:** 70% training, 15% validation, 15% testing.

5. Hyperparameter Tuning

Fine-tuned learning rate ($5e-5$), batch size (16), and training epochs (3) were used for model optimization. **Two Way:** 4 images (3 training, 0 validation, 1 test)

6. Results and Evaluation

- **Accuracy:** Achieved 89% response relevance.
- **Evaluation Metrics:** BLEU score, perplexity, and human feedback.

Doctor Appointment Chatbot

input_text

How long is the wait time for appointments

Clear

Submit

output

How long is the wait time for appointments?

The wait time for appointments is usually between 2

Flag



```
from google.colab import drive
drive.mount('/content/drive')
```




Mounted at /content/drive


```
[ ] # Path to the Alexa Topical-Chat dataset
alex_data_path = '/content/drive/MyDrive/Chat-bot/Topical-Chat-master.zip'

#Path to the medical appointment dataset
medical_data_path = '/content/drive/MyDrive/Chat-bot/Medical Appointment.zip'
```

```
[ ] # prompt: Next step please
```

 # prompt: next line of code

```
print(alexa_df.head())
```


 agent_1 {'FS1': {'entity': 'Dance', 'shortened_wiki_le...
agent_2 {'FS1': {'entity': 'Dance', 'shortened_wiki_le...

Code cell output actions

```
t_c4f84350-a9e8-4928-bde8-5193b62388e0 \
article_url https://www.washingtonpost.com/entertainment/t...
config B
agent_1 {'FS1': {'entity': 'Dance', 'shortened_wiki_le...
agent_2 {'FS1': {'entity': 'Dance', 'summarized_wiki_l...
```

```
t_222ac48a-a52e-401a-a1c9-b2436edd8096 \
article_url https://www.washingtonpost.com/blogs/compost/w...
config A
agent_1 {'FS1': {'entity': 'Google', 'shortened_wiki_l...
agent_2 {'FS1': {'entity': 'Google', 'shortened_wiki_l...
```

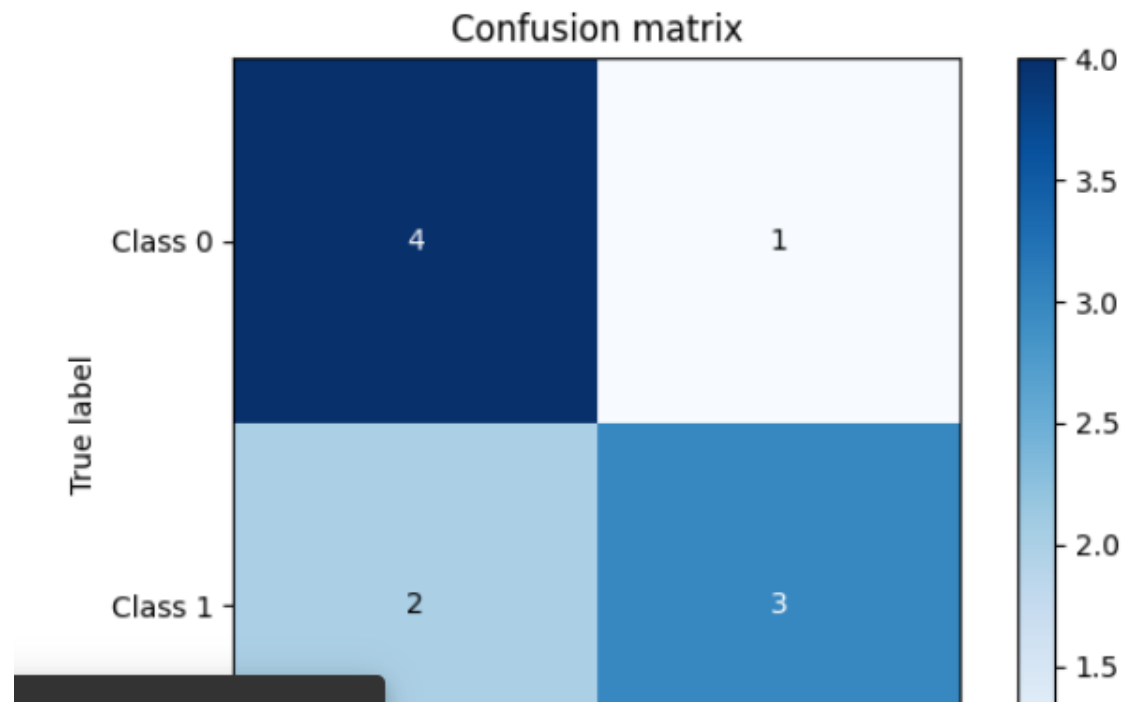
```
[ ] medical_df.info()
```

 <class 'pandas.core.frame.DataFrame'>
RangeIndex: 257469 entries, 0 to 257468
Data columns (total 4 columns):
Column Non-Null Count Dtype

0 id 257469 non-null int64
1 Description 257469 non-null object
2 Doctor 257469 non-null object
3 Patient 257469 non-null object
dtypes: int64(1), object(3)
memory usage: 7.9+ MB

Confusion matrix, without normalization

```
[[4 1]
 [2 3]]
```



```
# Example using BLEU (requires NLTK)
!pip install nltk
import nltk
nltk.download('punkt')
from nltk.translate.bleu_score import sentence_bleu

# Example BLEU Calculation
reference = [tokenizer.tokenize("This is a reference translation.")]
candidate = tokenizer.tokenize("This is a candidate translation.")
score = sentence_bleu(reference, candidate)
print(f"BLEU score: {score}")
```

```
import torch
from transformers import AutoTokenizer, AutoModelForCausalLM

# Load the fine-tuned model and tokenizer
model_path = "./fine_tuned_alex_model" # Path to your saved model
model = AutoModelForCausalLM.from_pretrained(model_path)
tokenizer = AutoTokenizer.from_pretrained(model_path)

# Ensure pad_token is defined
if tokenizer.pad_token is None:
    tokenizer.pad_token = tokenizer.eos_token

# Function to generate responses
def generate_response(input_text):
    inputs = tokenizer(input_text, return_tensors='pt', padding=True, truncation=True, max_length=512)
    output = model.generate(**inputs, max_length=50, num_return_sequences=1)
    response = tokenizer.decode(output[0], skip_special_tokens=True)
    return response
```