Problem Statement: Artificial Intelligence

Title: Sentiment Analysis and Intervention Chatbot

Proof of Concept

Group Name:

LexicalBot

Group Members:

jayasurya sakthivel (Group Leader)

Group ID:

OcfyyZv71TlcOuL

GitHub Link : https://github.com/jayasurya3012/ChatBot-Intervention-System.git

Technical Implementation

Here is an overview of the core technical implementation of my project:

1. Programming Languages:
    - Python: The main programming language used for developing the proof of concept.
2. Libraries and Frameworks:
    - gtts (Google Text-to-Speech): A library used to convert text into speech.
    - transformers: A library that provides pre-trained models for natural language processing tasks, such as conversational AI and question answering.
    - vaderSentiment: A library for sentiment analysis, specifically designed for analyzing social media text.
    - PIL (Python Imaging Library): A library used for image processing tasks, such as opening, manipulating, and saving images.
    - OpenCV (Open Source Computer Vision): A computer vision library used for image pre-processing and filtering tasks.
    - pytesseract: A library that enables optical character recognition (OCR) to extract text from images.
    - matplotlib: A plotting library used for visualizing images.
    - numpy: A library for numerical computing used for array manipulation and mathematical operations.
    - Streamlit: A library for creating a professional UI for any type of python application like Dashboards, webpages and apps.
3. System Design Overview:
    - The proof of concept consists of a ChatBot class that simulates a conversational AI assistant. The ChatBot class has methods for receiving user input, converting text to speech, and initiating a chat session.
    - The sentiment() function performs sentiment analysis on the extracted text from social media posts using the vaderSentiment library.
    - The preprocess() function applies image preprocessing techniques using OpenCV to enhance the quality of text extraction from images.

- The Sentiment_analysis() function processes a collection of social media post images, applies sentiment analysis to each image's text, and returns the sentiment context.
- If the context has more negative (-1), then the Chatbot is initiated.
- The start_robert() function initializes the ChatBot, sets up the necessary NLP models and libraries, and handles the conversation loop.
- The proof of concept utilizes pre-trained models such as DialoGPT for conversational AI and DistilBERT for question answering and provide emotion support to the user.

Running the Code

To execute and test the proof of concept, you can follow these step-by-step instructions:

Note: The following instructions assume that you have Python installed on your machine. If not, please install Python before proceeding.

1. Set up the Python environment:
    - Create a new directory for your project.
    - Open a command-line interface (CLI) or terminal and navigate to the project directory.
2. Install the required dependencies:
    - Run the requirements.txt in terminal to install the necessary libraries:

      pip install -r requirements.txt
3. Download the pre-trained NLP models:
    - Download the pre-trained DialoGPT model from the Hugging Face model hub. You can find it at: https://huggingface.co/microsoft/DialoGPT-large.
    - Save the downloaded model files in your project directory.
4. Obtain the social media post images:
    - Prepare a collection of social media post images (in PNG or JPEG format) that you would like to analyse for sentiment.
5. Set up the code:
    - Create a new Python file, e.g., **chatbot.py**, in your project directory.
    - Copy the code for the proof of concept into the **chatbot.py** file.

6. Configure the code:
    - In the **chatbot.py** file, ensure that the file paths for the pre-trained NLP models are correctly set to the downloaded model files.
    - Adjust any other configurable parameters or settings according to your preferences.
7. Run the code:
    - Save the **chatbot.py** file.
    - Open a CLI or terminal and navigate to the project directory.
    - Run the following command to execute the code:

      python chatbot.py
8. Interact with the ChatBot:
    - The ChatBot will initiate a conversation and provide prompts for user input.
    - Type your responses and press Enter to continue the conversation.
    - The ChatBot will process your inputs, convert them to speech when necessary, and provide responses accordingly.
    - You can also test the sentiment analysis feature by providing the path to the social media post images in the code.
9. Initiate UI for ChatBot:
    - Open the terminal and install Streamlit:
      pip install streamlit
    - To run the streamlit code:
      streamlit run UII.py
    - UII.py only contains Chatbot code and does not contain the Sentiment analysis code.
10. Explore the results:
    - Observe the ChatBot's responses, including the converted text-to-speech output.
    - Check the sentiment analysis results if you tested the social media post images. The sentiment context will be displayed for each analysed image.

Input and Expected Output

Input samples:

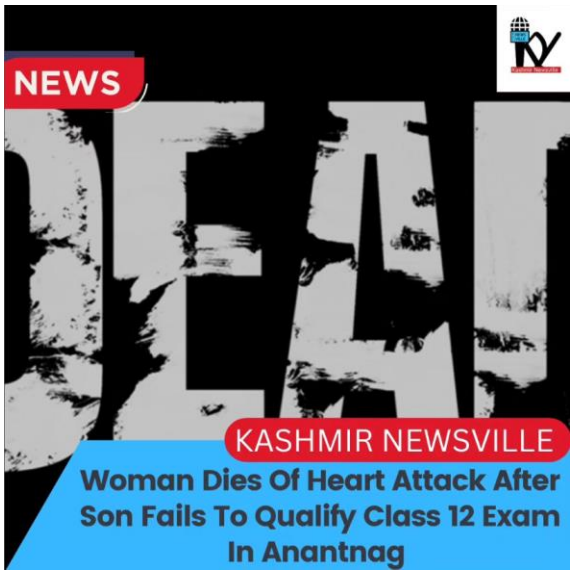
This framework works only if the system detects that the user has viewed/liked more negative social media posts. So let’s assume that these below posts are the posts viewed by the users at particular interval of time:

Pakistan: 34 killed, 145 injured as heavy rain, hail hit Khyber Pakhtunkhwa ahead of cyclone 'Biparjoy'

Pakistan: Heavy rains swept through Pakistan's northwest, causing several houses to collapse and leaving at least 34 people dead and 145 injured, authorities said.

Follow us on Google News

A child and his parents lived in a house away from the city, one day the husband and wife were fighting among themselves and the husband in anger hit the wife with a hard object. Swipe please:-



Expected Output:

As we can see there are many negative posts in the list. When this system detects it the framework activates the chatbot.

This is the processed output of this framework which denotes

- -1 – Negative Post
- 0 – Positive Post
- 1 – Neutral Post

```
# Here is the output of these functions
activity
```

```
[-1, 0, 1, -1, -1, 0, -1, -1, -1, 1]
```

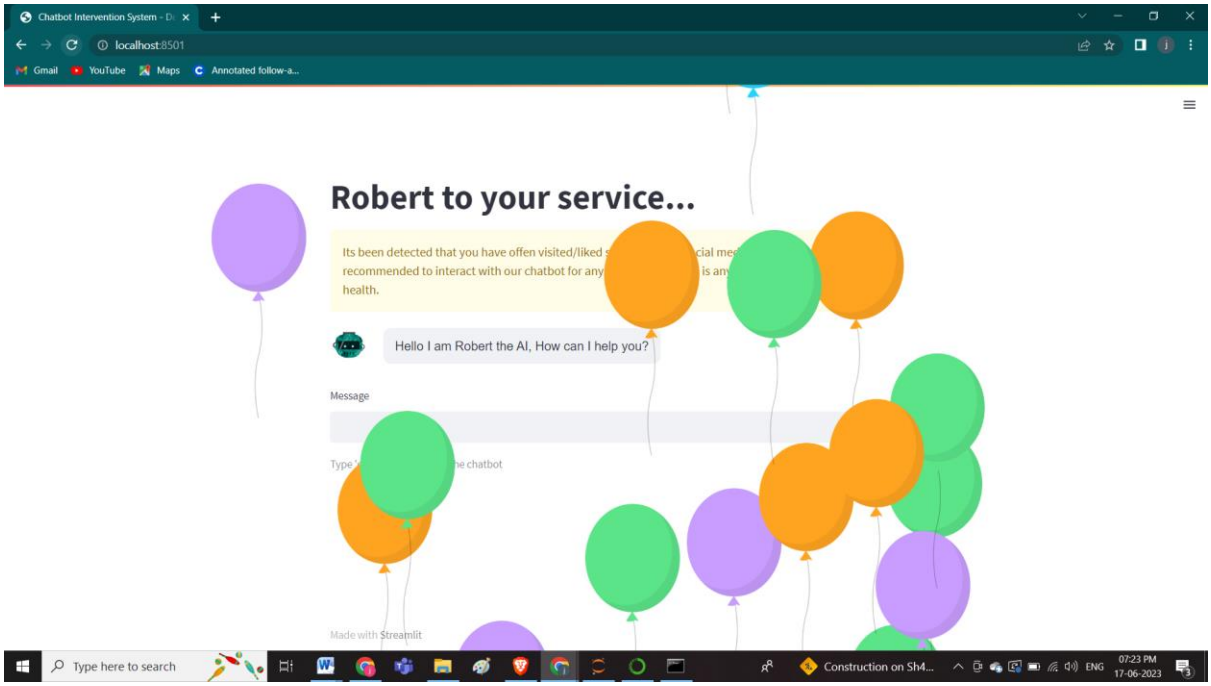
After detection the Chatbot activates, the user can chat with the user. And the Bot will respond with both message and audio output.

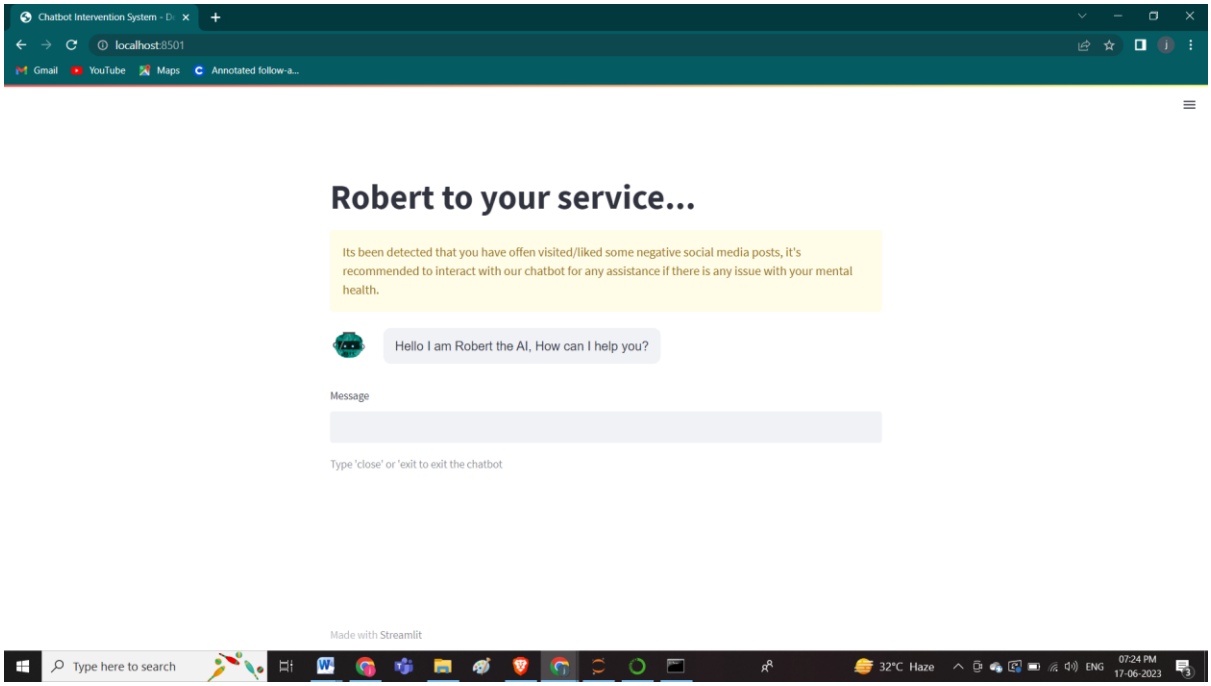
```
# To find the trend, we can sum the list to find if there is a need for our Chatbot Intervention System,
# if required then the system is activated...

if np.array(activity).sum() < -1 :
    print("ChatBot Intervention System Activates...")
    start_robert()
```

```
ChatBot Intervention System Activates...
----- Starting up Robert -----
No model was supplied, defaulted to distilbert-base-cased-distilled-squad and revision 626af31 (https://huggingface.co/distilb
ert-base-cased-distilled-squad).
Using a pipeline without specifying a model name and revision in production is not recommended.
Robert -->  Hello I am Robert the AI, what can I do for you?
can u help me
A decoder-only architecture is being used, but right-padding was detected! For correct generation results, please set `padding
_side='left'` when initializing the tokenizer.
Me  -->  can u help me
Robert -->  yes message me
i feel depressed and i dont know what to do
Me  -->  i feel depressed and i dont know what to do
Robert -->   Take it one step at a time and focus on self-care. What activities help you relax?
when i am depressed, i tend to sleep a lot
Me  -->  when i am depressed, i tend to sleep a lot
Robert -->   Take it one step at a time and focus on self-care. What activities help you relax?
i sleep a lot
```

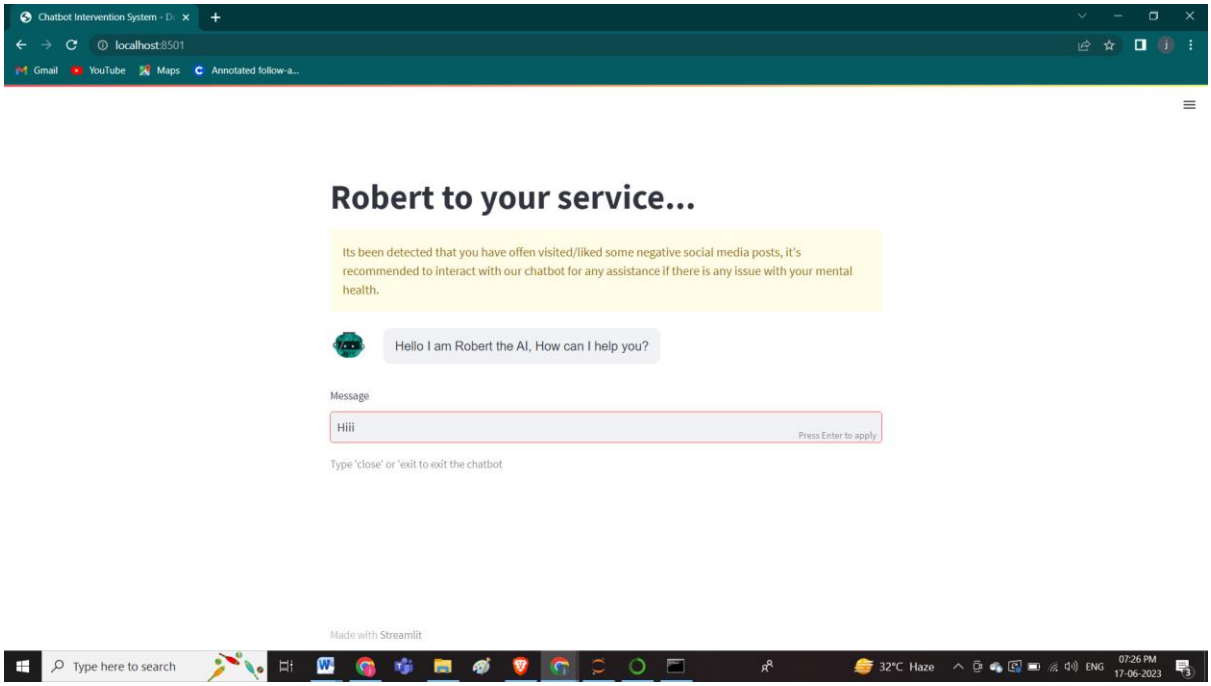
```
A decoder-only architecture is being used, but right-padding was detected! For correct generation results, please set `padding
_side='left'` when initializing the tokenizer.
Me  -->  i sleep a lot
Robert -->  I sleep a lot too.
i want to talk with someone
Me  -->  i want to talk with someone
Robert -->  Have a talk with your friend - 9753086421
thanks
Me  -->  thanks
Robert -->  cool!
exit
Me  -->  exit
Robert -->  Goodbye
----- Closing down Robert -----
```

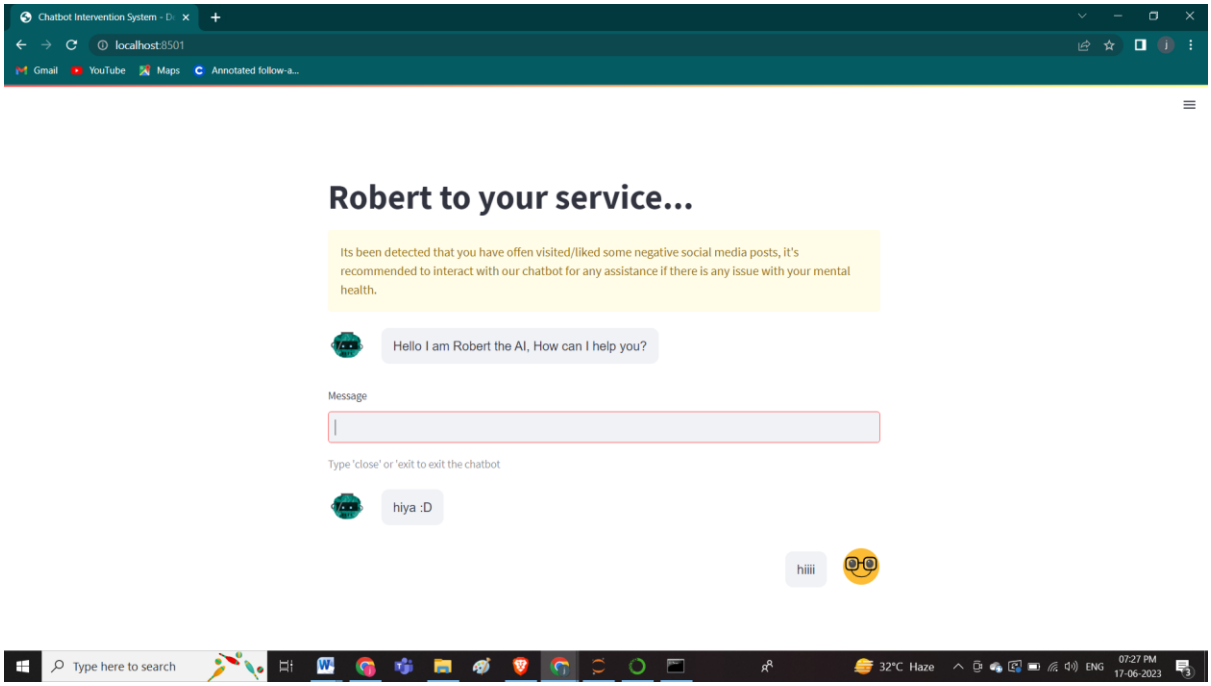
This is the Output of the UII.py which is the Chatbot Only User Interface.

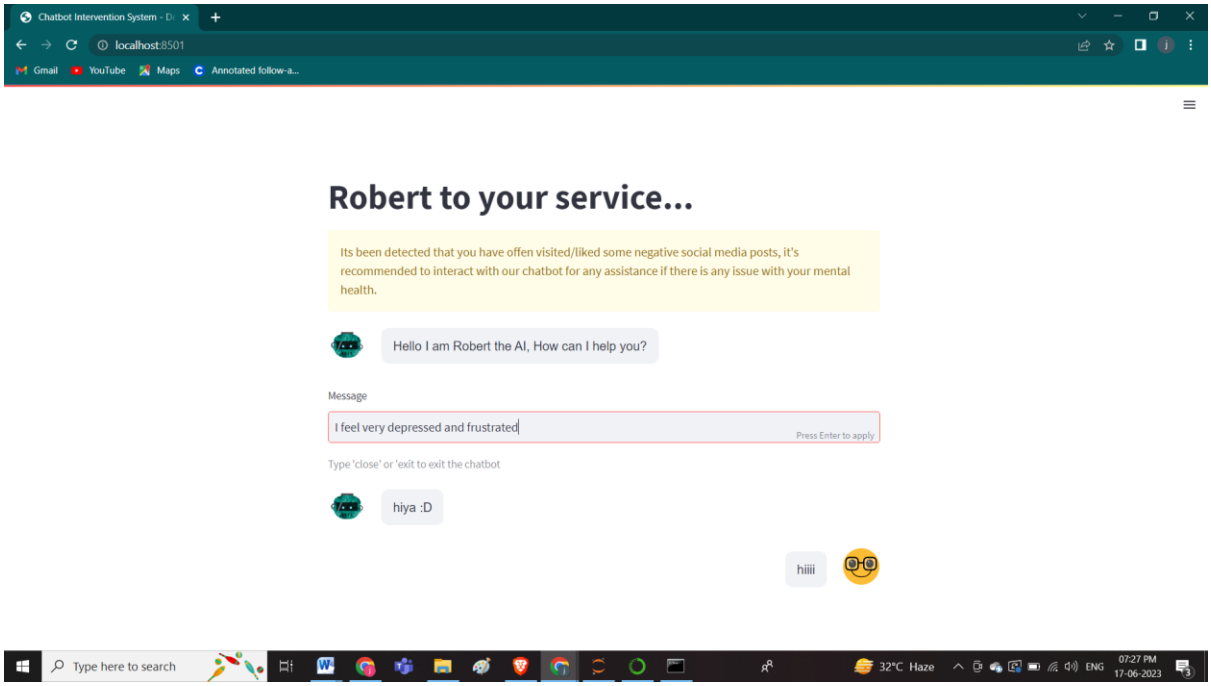


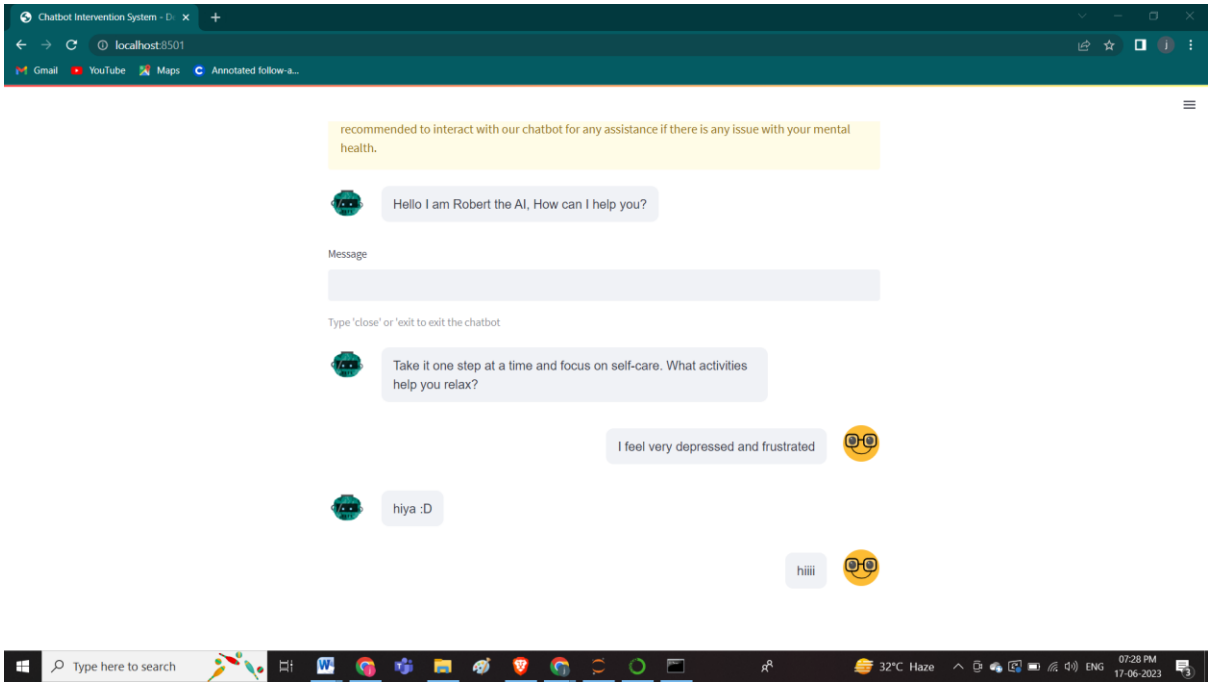


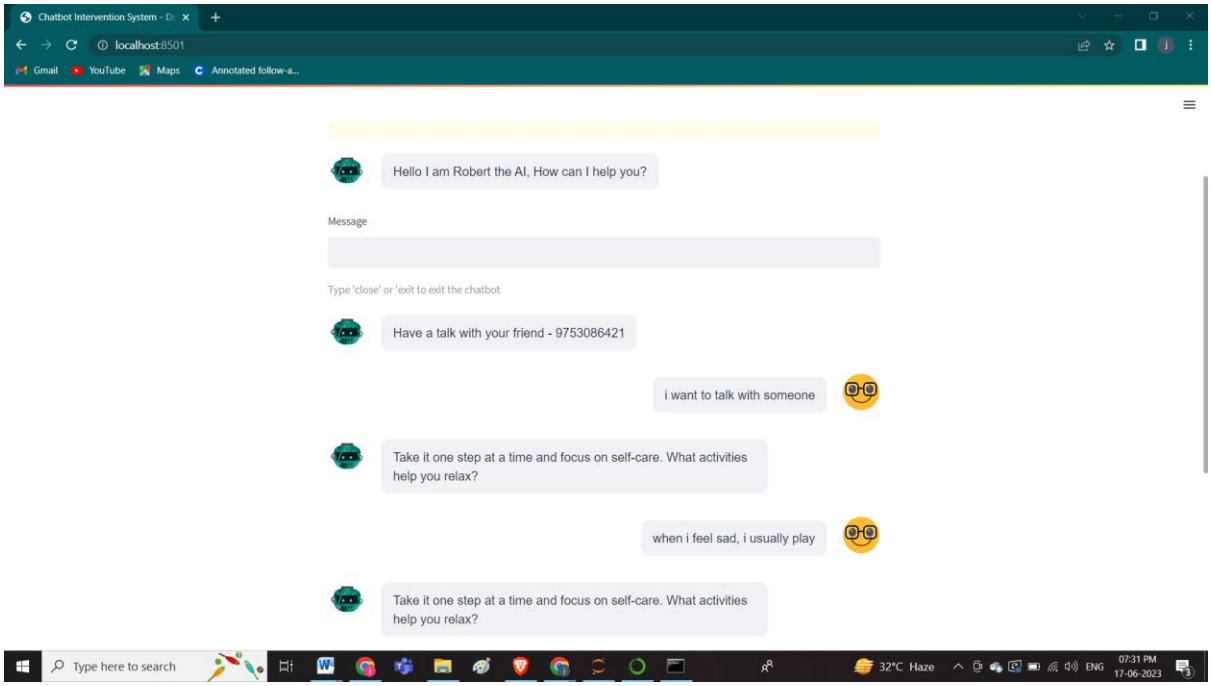
Let’s Type “Hiiii”

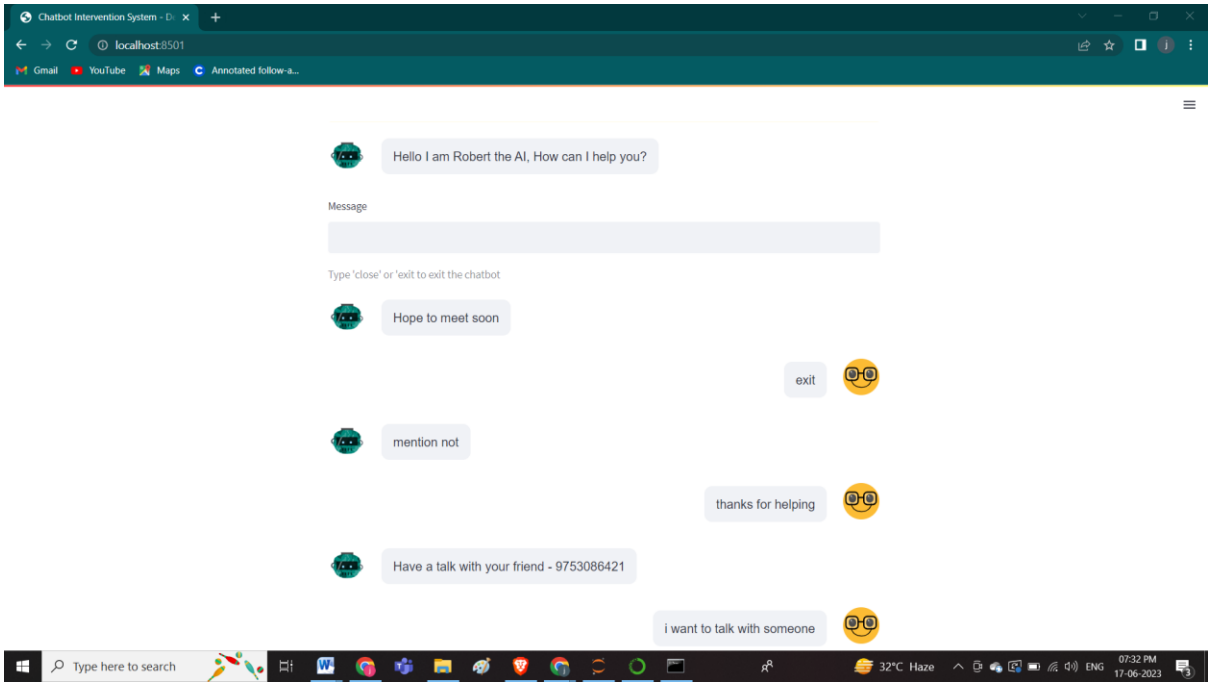



Chatbot Intervention System
localhost:8501
Gmail
YouTube
Maps
Annotated follow-a...
Robert to your service...
Its been detected that you have offen visited/liked some negative social media posts, it's recommended to interact with our chatbot for any assistance if there is any issue with your mental health.
Hello I am Robert the AI, How can I help you?
Message
Type 'close' or 'exit to exit the chatbot
hiya :D
hiiii
Type here to search
32°C Haze
ENG
07:27 PM
17-06-2023

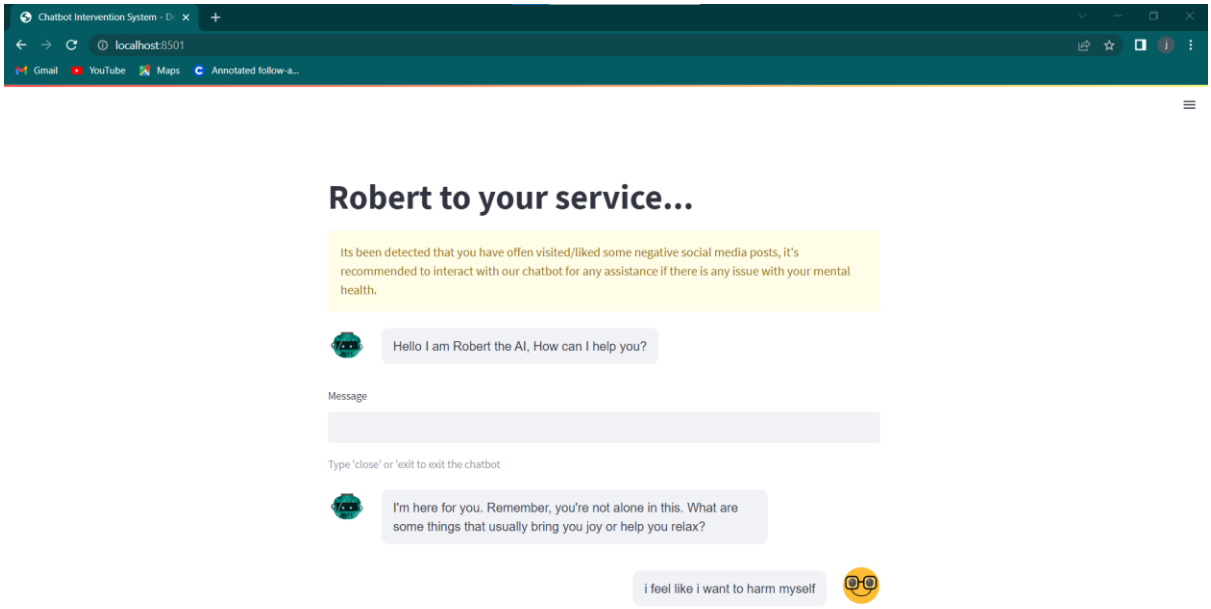

Chatbot Intervention System
localhost:8501
Gmail
YouTube
Maps
Annotated follow-a...
Robert to your service...
Its been detected that you have offen visited/liked some negative social media posts, it's recommended to interact with our chatbot for any assistance if there is any issue with your mental health.
Hello I am Robert the AI, How can I help you?
Message
I feel very depressed and frustrated
Press Enter to apply
Type 'close' or 'exit to exit the chatbot
hiya :D
hiiii
Type here to search
32°C Haze
ENG
07:27 PM
17-06-2023


Chatbot Intervention System
localhost:8501
Gmail
YouTube
Maps
Annotated follow-a...
recommended to interact with our chatbot for any assistance if there is any issue with your mental health.
Hello I am Robert the AI, How can I help you?
Message
Type 'close' or 'exit to exit the chatbot
Take it one step at a time and focus on self-care. What activities help you relax?
I feel very depressed and frustrated
hiya :D
hiiii
Type here to search
32°C Haze
ENG
07:28 PM
17-06-2023


Chatbot Intervention System
localhost:8501
Gmail
YouTube
Maps
Annotated follow-a...
Hello I am Robert the AI, How can I help you?
Message
Type 'close' or 'exit to exit the chatbot
Have a talk with your friend - 9753086421
i want to talk with someone
Take it one step at a time and focus on self-care. What activities help you relax?
when i feel sad, i usually play
Take it one step at a time and focus on self-care. What activities help you relax?
Type here to search
32°C Haze
ENG
07:31 PM
17-06-2023

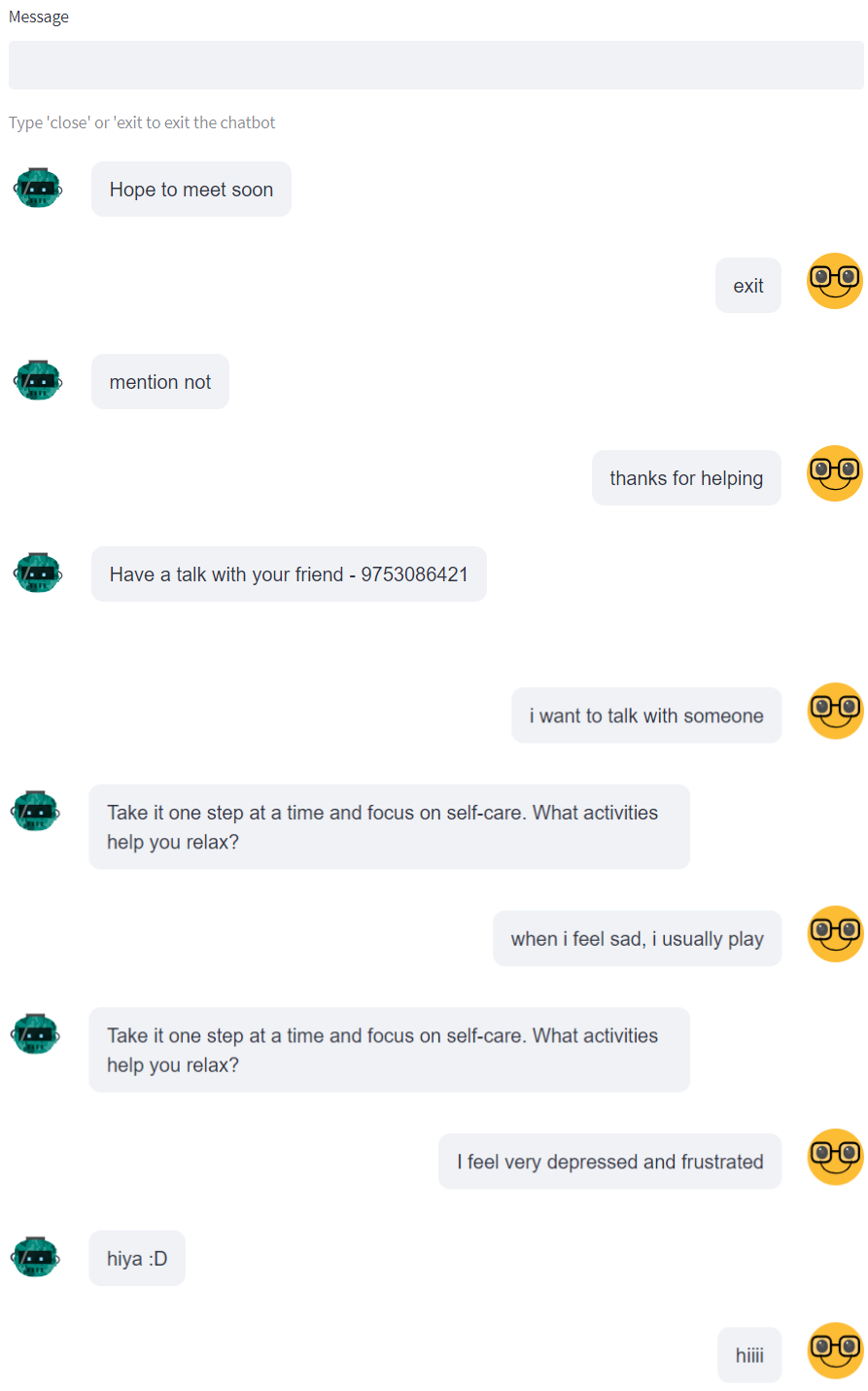


Some other outputs



As mentioned above, this bot can also communicate sarcastically, by using the Microsoft DiabloGPT model and can support the user emotionally with the help of Question-Answering model when the user feels negatively. It also has functions like giving the user the helpline number or a friends number, if the user wishes to talk with someone.

Whole conversation:



Sources and Claims

- https://www.datacamp.com/tutorial/streamlit
- https://github.com/AI-Yash/st-chat/blob/main/examples
- https://huggingface.co/learn/nlp-course/chapter7/7?fw=tf
- https://analyticsindiamag.com/sentiment-analysis-made-easy-using-vader/
- https://towardsdatascience.com/extract-text-from-memes-with-python-opencv-tesseract-ocr-63c2ccd72b69
- https://huggingface.co/microsoft/DialoGPT-medium?text=Hey+my+name+is+Julien%21+How+are+you%3F

Testing Methodologies

1. **Unit Testing:** Unit testing involves testing individual components or units of code in isolation to verify their correctness. In the case of the proof of concept, unit tests can be written for each module or function to ensure that they produce the expected output for a given input. Unit testing frameworks such as pytest or unittest can be used to automate the testing process.

2. **Integration Testing:** Integration testing focuses on testing the interactions and integration between different components of the system. In the proof of concept, integration testing can be performed to ensure that the various modules, such as the sentiment analysis, speech recognition, and image processing, work together correctly and produce the desired results. It can involve testing different combinations of modules and verifying the outputs against expected values.

3. **Functional Testing:** Functional testing evaluates the system's behavior and functionality from an end-user perspective. It tests the system against the specified requirements and ensures that it performs the intended tasks correctly. In the proof of concept, functional testing can involve running the application with sample inputs and verifying that it responds with accurate sentiment analysis results, correctly transcribes speech, and processes images as expected.

4. **Performance Testing:** Performance testing assesses the system's performance and scalability under various load conditions. In the proof of concept, performance testing can involve measuring the response times for different operations, analysing memory and CPU usage, and stress testing the system to evaluate its behaviour under heavy workloads. Tools like JMeter or Locust can be used to simulate concurrent users and measure system performance.

5. **User Acceptance Testing (UAT):** UAT involves testing the system with real end-users to ensure that it meets their requirements and expectations. In the case of the

proof of concept, involving target users to test the application and gather their feedback can provide valuable insights into its usability, effectiveness, and overall user experience.

6. **Metrics and Logging:** Implementing logging mechanisms within the proof of concept can help capture important metrics during testing. Metrics such as response time, error rates, and resource usage can provide quantitative data on the system's performance and behaviour. Logging can also assist in diagnosing issues and debugging the application.

Limitations and Future Enhancements

The proof of concept described above has certain limitations and constraints that should be taken into consideration:

1. **Scalability:** The current implementation is meant for a single user interaction at a time. It doesn't support concurrent conversations or handle multiple users simultaneously. To scale the application for multiple users, it would require additional development, such as implementing a server-client architecture or using a chatbot framework that supports concurrent interactions.

2. **Performance:** The performance of the proof of concept might be affected by the computational resources available. The large DialoGPT model used in the implementation requires significant computing power and can be resource-intensive. Processing long conversations or dealing with complex queries might lead to slower response times. To address this, optimization techniques like model pruning or deploying the application on more powerful hardware can be considered.

3. **Known issues:** As the proof of concept relies on pre-trained models, it inherits any known limitations or biases present in those models. The sentiment analysis module might not be accurate in certain contexts or could be influenced by biases present in the training data. The speech-to-text conversion might not be perfect and could produce inaccuracies or misinterpretations. Additionally, the image processing module might not work optimally for all types of social media post images, depending on factors like image quality or complexity.

Potential areas for improvement and future enhancements include:

1. Enhancing the user interface: The proof of concept currently relies on a command-line interface for interaction. Developing a more intuitive and user-friendly interface, such as a web-based interface or a chatbot widget, would improve the user experience and make it more accessible.

2. Integration with social media platforms: Extending the proof of concept to directly connect with social media platforms would enable real-time analysis of posts and comments, making it more convenient for users to analyse sentiment and engage in conversations within their preferred platforms.

3. Fine-tuning the sentiment analysis module: Fine-tuning the sentiment analysis model using domain-specific data can improve its accuracy for analysing social media content. Additionally, incorporating more advanced natural language processing techniques, such as aspect-based sentiment analysis, can provide more nuanced sentiment analysis results.

4. Multi-modal analysis: Integrating the capabilities of analysing text, speech, and images simultaneously can provide a more comprehensive understanding of social media posts. This can involve combining the sentiment analysis of textual content with the sentiment analysis of images or incorporating audio sentiment analysis.

To overcome the current limitations and further develop the proof of concept, the following ideas and strategies can be considered:

1. Performance optimization: Implementing techniques like model quantization, model parallelism, or utilizing specialized hardware (e.g., GPUs) can enhance the performance and response time of the application, enabling it to handle larger workloads more efficiently.

2. Continuous training and updating: Regularly updating the pre-trained models with new data can improve their accuracy and enable the application to adapt to changing language patterns and sentiments in social media content.

3. User feedback and evaluation: Gathering user feedback and evaluating the performance of the proof of concept in real-world scenarios can help identify areas of improvement, uncover new limitations, and prioritize future enhancements based on user needs and expectations.

4. Collaboration and open-source contribution: Making the codebase open-source and inviting contributions from the developer community can accelerate the development process, allow for diverse perspectives, and foster collaboration to address limitations and introduce new features.

By addressing these limitations and embracing future enhancements, the proof of concept can evolve into a more robust and scalable solution for sentiment analysis and conversational interactions in the context of social media.

Supporting Materials

This is the Only Chatbot UI Code which is run in terminal:

```
import streamlit as st
from streamlit_chat import message
import os
from gtts import gTTS
```

```
import time
from transformers import pipeline,Conversation
import numpy as np
import datetime
from PIL import Image
import cv2
from pytesseract import pytesseract
import matplotlib.pyplot as plt
import numpy as np
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer

pytesseract.tesseract_cmd = r'C:\Program Files\Tesseract-OCR\tesseract.exe'
custom_config = r"--oem 3 --psm 11 -c tessedit_char_whitelist=
'ABCDEFGHIJKLMNOPQRSTUVWXYZ "'
path = [‘post.png' , post3.png' , ' post6.png' , 'post1_1.png' , post1_2.png' ,
        'post1_3.png' , 'post1_4.png' , 'post1_5.png' , 'post_1_6.png' ,  'post_1_7.png']

with open('reply.txt') as f:
   contents = f.read()
contents = contents.replace("\n"," ")
context = contents.replace("A: ","")
contents = contents.split('A:')

st.set_page_config(
   page_title="Chatbot Intervention System - Demo",
   page_icon=":robot:"
)

st.balloons()
st.title ("Robert to your service...")
st.warning("Its been detected that you have offen visited/liked some negative social media
posts, it's recommended to interact with our chatbot for any assistance if there is any issue
with your mental health.")
message("Hello I am Robert the AI, How can I help you?")

if 'generated' not in st.session_state:
   st.session_state['generated'] = []
if 'past' not in st.session_state:
   st.session_state['past'] = []
if 'temp' not in st.session_state:
      st.session_state.temp = ""

def bot(text):
   nlp = pipeline("conversational", model="microsoft/DialoGPT-medium")
   qa_model = pipeline("question-answering", model="distilbert-base-cased-distilled-squad")
```

```
    model = SentimentIntensityAnalyzer()
    os.environ["TOKENIZERS_PARALLELISM"] = "true"
    if any(i in text for i in ["thank","thanks"]):
            res = np.random.choice(["you're welcome!","anytime!","no problem!","cool!","I'm
here if you need me!","mention not"])
        # if user wishes to exit the chatbot
    elif any(i in text for i in ["exit","close"]):
        res = np.random.choice(["Tata","Have a good day","Bye","Goodbye","Hope to meet
soon","peace out!"])
        ex=False
        ## conversation
    else:
            # if user needs some professional help or family intervention
        if any(i in text for i in ["talk with someone","I need help","phone number","mobile
number","helpline"]):
            res = np.random.choice(["Let me help you, call '9876543210' and they will try to help
you as much as possible", "Have a talk with your friend - 9753086421"])
        else:
                # 'tone' is used to find the emotion of the text sent by the user
                # if it is -ve then we go to the question answering model
                # An answer is generated from the query and context which is given below,
                # if there is any sentence in the contents list which contains this answer, we select
that sentence as the reply
            tone = model.polarity_scores(text)
            if tone['compound'] < 0:
                reply = qa_model(question = text , context = context)
                wrd = reply['answer']
                for sentence in contents:
                    if wrd in sentence:
                        res = sentence
                        break
                # else we go to the nlp model for general chating
            else:
                res = nlp(Conversation(text), pad_token_id=50256)
                res = str(res)
                res = res[res.find("bot >> ")+6:].strip()
    return res

def clear_text():
    st.session_state['temp'] = st.session_state["text"]
    st.session_state["text"] = ""

user_input = st.text_input("Message", key="text",on_change=clear_text)
st.caption("Type 'close' or 'exit to exit the chatbot")
user_input = st.session_state['temp']
```

```
if user_input:
    output = bot(user_input)
    st.session_state.past.append(user_input)
    st.session_state.generated.append(output)

if st.session_state['generated']:
    for i in range(len(st.session_state['generated'])-1, -1, -1):
        message(st.session_state["generated"][i], key=str(i))
        message(st.session_state['past'][i], is_user=True, key=str(i) + '_user')
```

Code for sentiment analysis from social media posts:

```
def preprocess(im):
    im = cv2.cvtColor(im, cv2.COLOR_BGR2GRAY)
    im= cv2.bilateralFilter(im,5,30,60)
    _, im = cv2.threshold(im,140, 240, 1)  # try either (im,140, 240, 1) or (im,160, 230, 1) if
not working properly
    plt.figure(figsize=(4,4))
    plt.title('Post')
    plt.imshow(im, cmap='gray'); plt.xticks([]); plt.yticks([])
    return im

def sentiment(im):
    text = pytesseract.image_to_string(im, lang='eng', config=custom_config)
    print(text.replace('\n', ' '))
    s = model.polarity_scores(text)
    if s['compound'] > 0:
        Type = 1
    elif s['compound'] < 0:
        Type = -1
    elif s['compound'] == 0:
        Type = 0
    return Type

def Sentiment_analysis(path):
    context = []
    for i in range(len(path)):
        im = np.array(Image.open(path[i]))
        im = preprocess(im)
        t = sentiment(im)
        context.append(t)
    return context
```