JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Developing an Empathetic AI Companion Using Sentimental Analysis

[1] Nanda M B, [2]Akhansha Chakraborty, [3]Varsha Mathur, [4]Vidyasree.M.G., [5]Fardeen Hussain
[1] Assistant Professor [2,3,4,5] UG Students
Department of Artificial Intelligence and Machine Learning
Sri Krishna Institute of Technology, Bengaluru, Karnataka, India

Abstract: The Emotion Recognition and Voice Command Chatbot System merges advanced AI technologies to create an empathetic and interactive experience. Using facial emotion detection (via DeepFace), speech recognition, and AI-based responses, it analyses user inputs to provide meaningful interactions. The system features a modular design with a webcam-based emotion analyser, a microphone-driven speech recognizer, and a text-to-speech engine for real-time, context-aware communication. It identifies emotions like happiness, sadness, and anger, making it suitable for applications in mental health, education, and customer service. Emphasizing privacy and inclusivity, future developments aim to enhance personalization, multilingual capabilities, and emotional granularity.

***Keywords* – Emotion Recognition, Speech Recognition, Facial Expression Analysis, Speech-to-Text, Text-to-Speech, Natural Language Understanding (NLU), Personalized Interaction, Mental Healthcare.**

I INTRODUCTION

The advent of artificial intelligence (AI) has revolutionized various aspects of human interaction, enabling systems to assist, inform, and even engage in simulated conversations. While chatbots and virtual assistants, such as Siri, Alexa, and Google Assistant, have become commonplace, their lack of emotional intelligence limits their effectiveness in creating meaningful connections. These systems are designed for functional tasks, offering information and performing simple actions, but they fail to understand or respond to the emotional context behind user's queries.

In contrast, the empathetic chatbot described in this report aims to bridge this gap by incorporating emotional intelligence into AI-powered interactions. Through advanced sentiment analysis and speech emotion recognition, this chatbot is not just designed to engage in basic conversations, but to understand and respond empathetically to users' emotional states. It aims to serve as a virtual companion capable of providing comfort, emotional support, and practical assistance.

This empathetic chatbot can detect emotions through both text and speech inputs, enabling it to respond in a way that aligns with the emotional tone of the conversation. By doing so, it enhances user experiences across a range of fields, such as mental health, education, customer service, and beyond. The system's goal is to create meaningful interactions that improve user well-being and satisfaction, whether by providing emotional support, helping users manage stress, or simply offering a friendly and understanding presence.

II LITERATURE SURVEY

The reviewed studies highlight significant advancements and challenges in using AI and technology for health and wellness interventions:

Deep Learning-Based Chatbots for Health Education

John Batani et al. developed a deep learning-powered chatbot to enhance maternal health education. The chatbot improved users' accessibility to crucial maternal health knowledge, though its application was limited to this specific domain.

AI Chatbot for Mental Health in Kenyan Universities

Maria Oganda et al. created *Wellness Buddy*, an AI chatbot tailored to address mental health issues among Kenyan university students. The study demonstrated the chatbot's effectiveness in increasing student engagement and mental health awareness, though scalability and long-term impact remain areas for further exploration.

Trust in Mobile Health and Wellness Chatbots

De Mugdha Kulkami investigated user trust in mobile health and wellness chatbots using the Technology Acceptance Model (TAM) and Health Belief Model (HBM). Factors like perceived usefulness, ease of use, and security were identified as key to trust. However, cultural and demographic differences may affect outcomes.

Challenges in Using Actigraphy Data for Depressive Symptoms

This study addressed the difficulties in classifying depressive symptoms based on actigraphy data. The findings highlighted the limitations of wearable devices in distinguishing depressive symptoms due to data quality issues and symptom overlaps.

Smart Home Data Visualization for Mental Health Monitoring

Youngji Koh et al. explored the use of smart home environments for mental health monitoring through data visualization. While the approach offered personalized and real-time mental health insights, concerns regarding scalability and user privacy were raised.

Keyword Matching Algorithm for Early Warning of Mental Health Issues

Lee Zhang and Chas Huang developed an early warning platform for mental health issues among college students using keyword matching algorithms. The platform effectively identified signs of mental distress, enabling timely intervention, but faced challenges with false positives and the lack of contextual analysis.

III EXISTING SYSTEMS

Current chatbot systems are primarily designed for task execution, such as providing weather updates, setting reminders, or controlling smart home devices. While these virtual assistants are highly functional, their interactions are often limited to transactional exchanges. They operate under a set of predefined scripts and are not capable of detecting the emotional context of user inputs.

For instance, when a user expresses frustration or sadness, systems like Siri or Alexa do not acknowledge or respond to these emotions. This creates a barrier to forming meaningful connections between the user and the AI. The lack of emotional awareness in these systems can result in impersonal, robotic responses that fail to resonate with users on a deeper level.

IV PROPOSED SYSTEM

The proposed system overcomes these limitations by integrating multimodal emotion recognition technologies. By combining text analysis with speech emotion recognition, the chatbot can understand emotional nuances beyond the content of the message itself. For example, if a user expresses sadness in their speech or text, the system can detect this emotion and respond with comforting, supportive language.

Additionally, the chatbot will be designed to provide context-aware, empathetic responses tailored to the emotional state of the user. Whether the user is expressing happiness, frustration, or sadness, the chatbot will adapt its responses to align with the emotional context. This dynamic interaction fosters trust and helps create more human-like exchanges, which are essential for the success of virtual companions.

To further enhance the system, the chatbot could incorporate memory features, allowing it to remember key details from previous interactions and offer personalized support over time. Moreover, the chatbot’s adaptive learning algorithms can enable it to improve and adjust its responses based on ongoing user feedback and evolving emotional states.

The system will also be developed with cultural and linguistic adaptability in mind, ensuring that it can respond appropriately to users from diverse backgrounds. This is especially important when considering the global nature of mental health issues and the increasing demand for personalized, inclusive AI systems.

V System Architecture

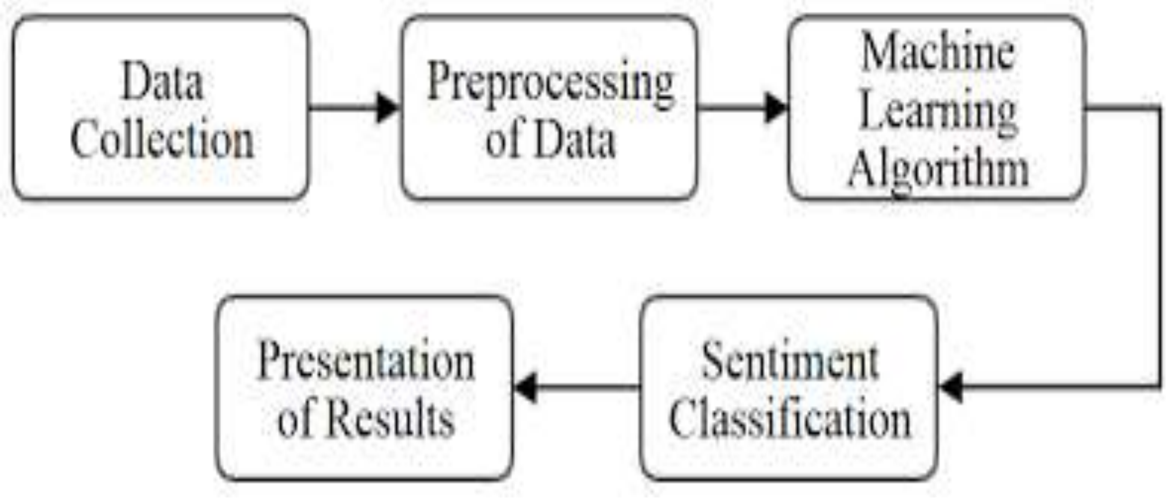


Fig 1. Architecture of virtual traffic police

IMPLEMENTATION

5.1 System Requirements:

1) Operating System: Linux (e.g., Ubuntu) or Windows
2) Programming Language: Python Version 10.
3) OpenCV: For video processing.
4) DeepFace: Supports facial emotion recognition
5) IDE: Any Python IDE or VS Code.

5.2 Hardware Requirements:

- This chatbot needs a powerful computer to function smoothly. It requires a strong processor like an Intel Core i5 or AMD Ryzen 5 to handle the complex calculations involved. Sufficient RAM is essential, with 8GB being the minimum, and 16GB recommended for optimal performance. A fast SSD is crucial for quick loading times.

- Additionally, a dedicated graphics card can significantly speed up the processing of images and videos for facial emotion detection. A high-quality microphone is necessary for accurate voice input, and a webcam (optional) is required for capturing facial expressions. These hardware components ensure the chatbot can efficiently analyse user input, generate responses, and provide a smooth, responsive user experience.

5.3 METHODOLOGY

The methodology for building the interactive emotion recognition and voice command chatbot consists of several phases, from design to implementation and evaluation. Each phase follows an agile, iterative approach where components are designed, integrated, and tested incrementally. Below are the key steps:

1. System Design and Planning

- **Objective:** Define the purpose and features of the system, including video capture, emotion recognition, voice recognition, and response generation.
- **Tools and Technologies:** Python programming language, OpenCV, Deep Face, pyttsx3, Speech Recognition, and other libraries.
- **System Components:** The system will have various interconnected modules like webcam input, speech recognition, emotion detection, and response generation.

2. Video and Audio Capture

- **Objective:** Capture video and audio from the user using the webcam and microphone, respectively.
- **Tools:** OpenCV is used to capture video from the webcam, while the Speech Recognition library processes audio input.
- **Implementation:** Continuously capture video frames for emotion analysis and record audio for speech recognition.

3. Emotion Detection using Deep Face

- **Objective:** Analyse the captured video frames to detect the user's emotions (e.g., happy, sad, angry).
- **Tools:** Deep Face, a deep learning library, is used to detect emotions from facial expressions.
- **Implementation:** Once a frame is captured, it is passed through Deep Face to determine the dominant emotion and return the result.

4. Speech Recognition

- **Objective:** Convert spoken words into text using the microphone input.
- **Tools:** Google Web Speech API via Speech Recognition library.
- **Implementation:** The system listens for user input, recognizes speech, and processes it as commands or requests.

5. Response Generation

- **Objective:** Generate an appropriate response based on the user's emotion or voice command.
- **Tools:** Predefined responses for each emotion, along with additional handling for speech commands.
- **Implementation:** Depending on the emotion detected or the command recognized, the system generates a text-based response.

6. Text-to-Speech Output

- **Objective:** Convert the text-based response into spoken words.
- **Tools:** pyttsx3 library for text-to-speech conversion.
- **Implementation:** The generated response is then converted into speech and output to the user through speakers.

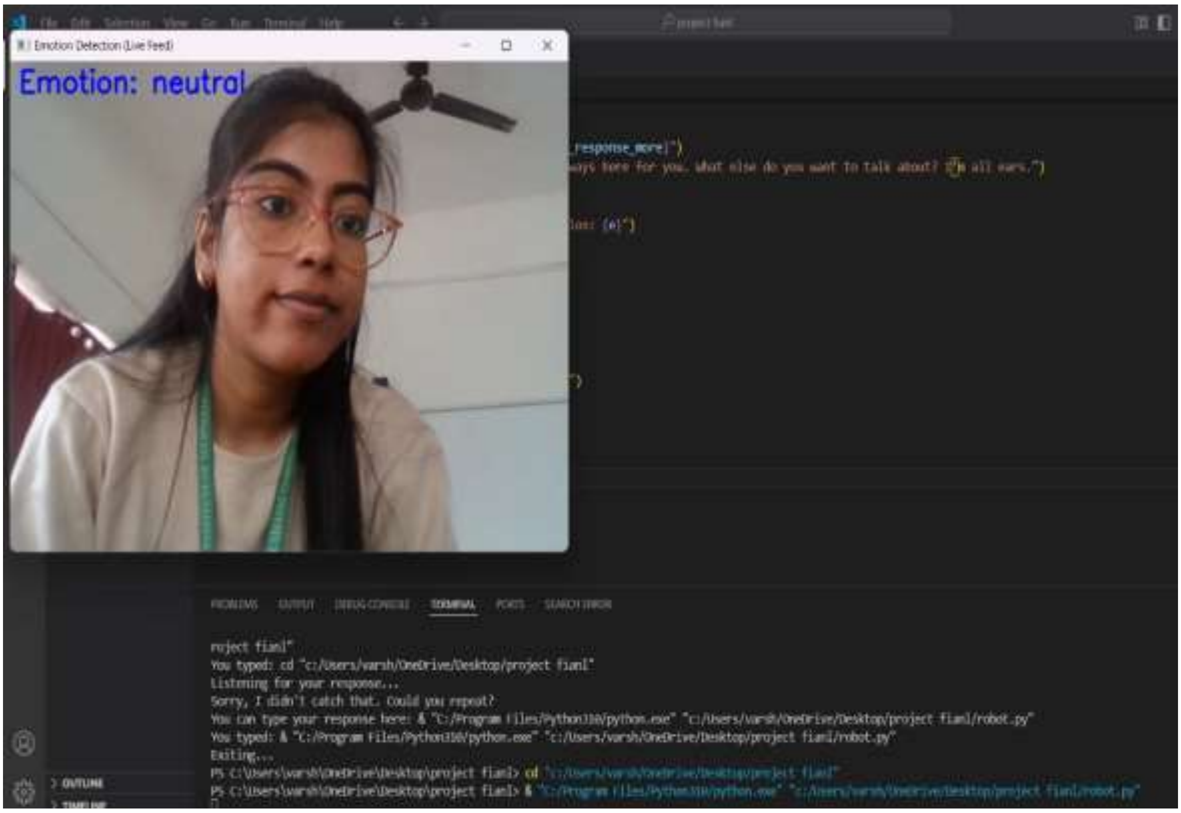
7. Image and Video Saving

- **Objective:** Periodically save images from the webcam and the video feed for future analysis or storage.
- **Tools:** OpenCV's cv2.imwrite and cv2.VideoWriter.
- **Implementation:** Every fixed interval (e.g., every 5 seconds), the system captures an image and saves it to a specified directory. The entire video stream is also saved as a file.

8. Error Handling and Logging

- **Objective:** Ensure the system handles failures gracefully and provides user feedback.
- **Tools:** Python error handling (try-except blocks) and logging.
- **Implementation:** The system is programmed to log errors and provide verbal feedback to the user when something goes wrong (e.g., when the system fails to recognize speech).

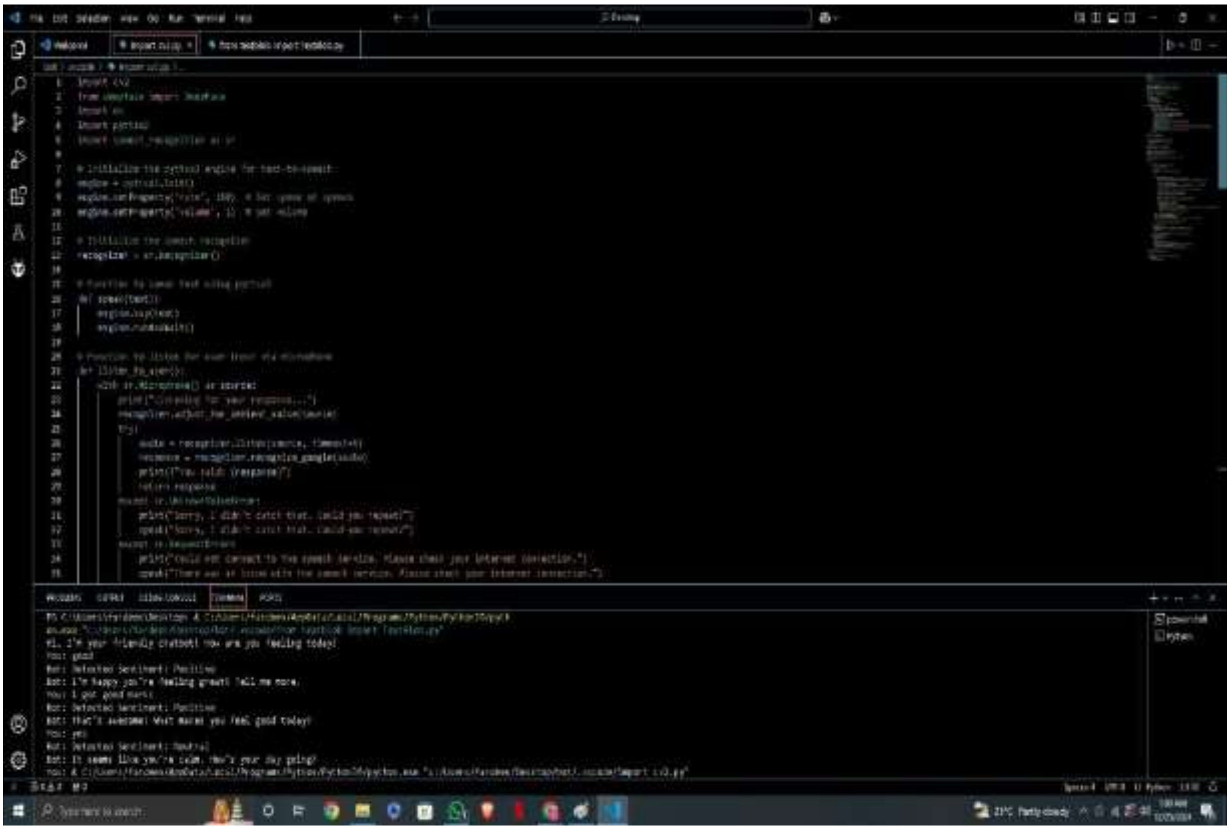
VI SCREENSHOTS



```
_response_more]")
ways here for you. What else do you want to talk about? I'm all ears.")

tion: {e}")
```

```
roject fianl"
You typed: cd "c:/Users/varsh/OneDrive/Desktop/project fianl"
Listening for your response...
Sorry, I didn't catch that. Could you repeat?
You can type your response here: & "C:/Program Files/Python310/python.exe" "c:/Users/varsh/OneDrive/Desktop/project fianl/robot.py"
You typed: & "C:/Program Files/Python310/python.exe" "c:/Users/varsh/OneDrive/Desktop/project fianl/robot.py"
Exiting...
PS C:\Users\varsh\OneDrive\Desktop\project fianl> cd "c:/Users/varsh/OneDrive/Desktop/project fianl"
PS C:\Users\varsh\OneDrive\Desktop\project fianl> & "C:/Program Files/Python310/python.exe" "c:/Users/varsh/OneDrive/Desktop/project fianl/robot.py"
```

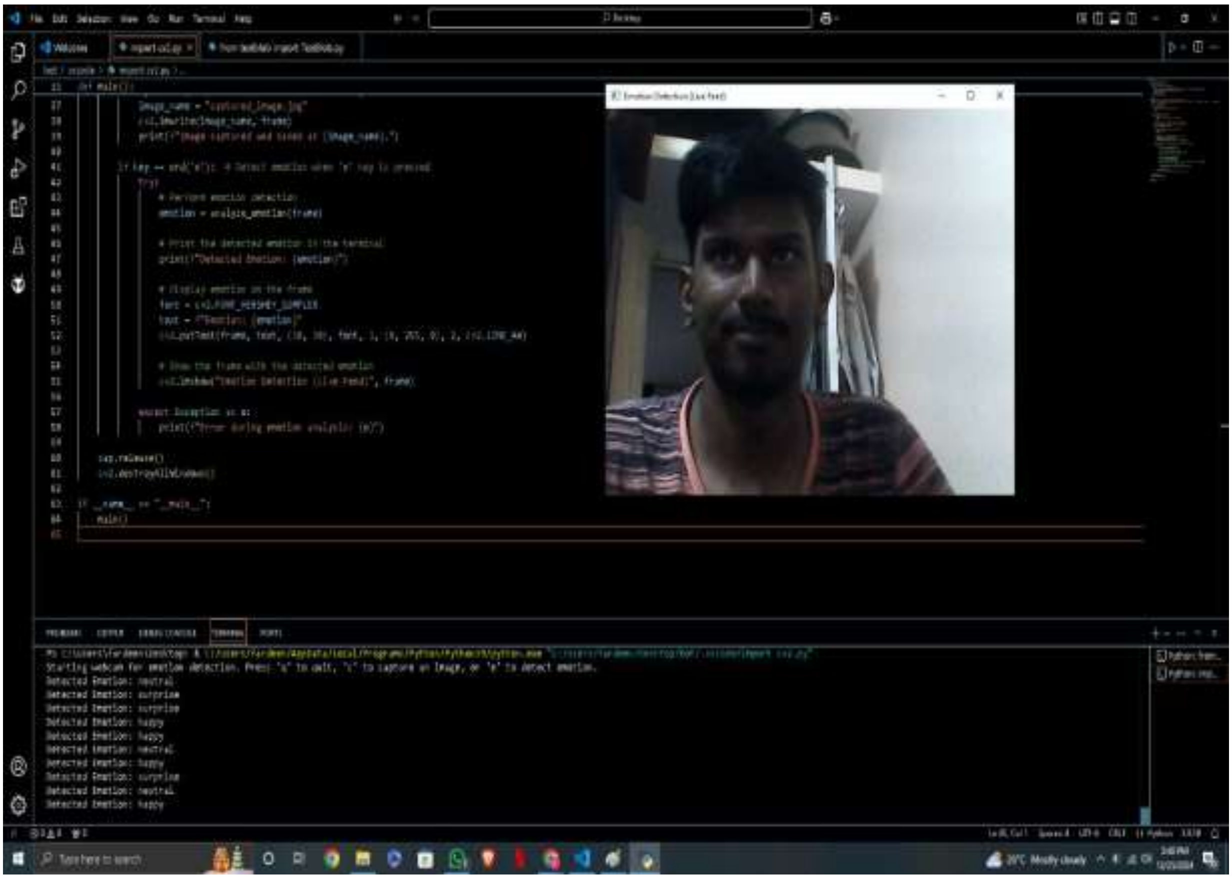
Fig 1 The Terminal



```
import cv2
from deepface import DeepFace
import os
import pyttsx3
import speech_recognition as sr

# Initialize the pyttsx3 engine for text-to-speech
engine = pyttsx3.init()
engine.setProperty('rate', 150)  # Set speed of speech
engine.setProperty('volume', 1)  # Set volume

# Initialize the speech recognizer
recognizer = sr.Recognizer()

# Function to speak text using pyttsx3
def speak(text):
    engine.say(text)
    engine.runAndWait()

# Function to listen for user input via microphone
def listen_to_user():
    with sr.Microphone() as source:
        print("Listening for your response...")
        recognizer.adjust_for_ambient_noise(source)
        try:
            audio = recognizer.listen(source, timeout=5)
            response = recognizer.recognize_google(audio)
            print(f"You said: {response}")
            return response
        except sr.UnknownValueError:
            print("Sorry, I didn't catch that. Could you repeat?")
            speak("Sorry, I didn't catch that. Could you repeat?")
        except sr.RequestError:
            print("Could not connect to the speech service. Please check your internet connection.")
            speak("There was an issue with the speech service. Please check your internet connection.")
```

```
PS C:\Users\Fardeen\Desktop> & C:/Users/Fardeen/AppData/Local/Programs/Python/Python312/python
.exe "c:/Users/Fardeen/Desktop/bot/.vscode/from textblob import TextBlob.py"
Hi, I'm your friendly chatbot! How are you feeling today?
You: good
Bot: Detected Sentiment: Positive
Bot: I'm happy you're feeling great! Tell me more.
You: i got good marks
Bot: Detected Sentiment: Positive
Bot: That's awesome! What makes you feel good today?
You: yes
Bot: Detected Sentiment: Neutral
Bot: It seems like you're calm. How's your day going?
You: & C:/Users/Fardeen/AppData/Local/Programs/Python/Python312/python.exe "c:/Users/Fardeen/Desktop/bot/.vscode/import cv2.py"
```

Fig 2 The Terminal Conversation

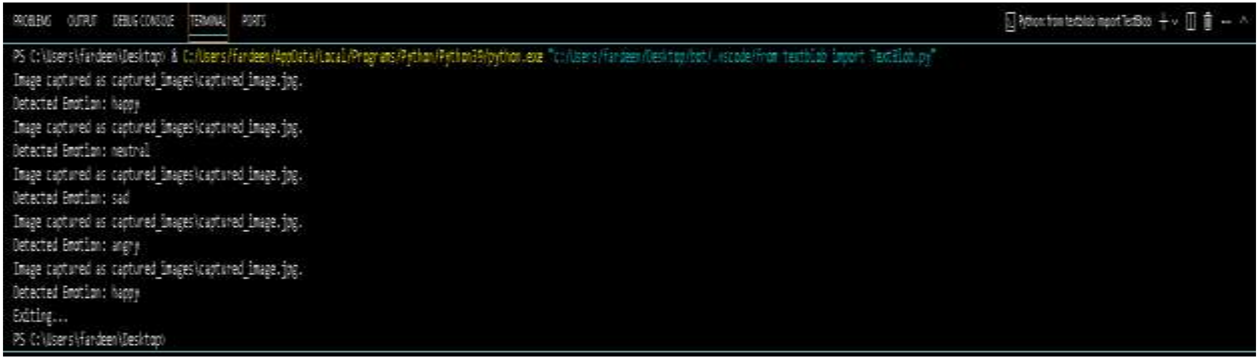


Emotion Detection (Live Feed)

```
def main():
            image_name = "captured_image.jpg"
            cv2.imwrite(image_name, frame)
            print(f"Image captured and saved as {image_name}.")

        if key == ord('e'):  # Detect emotion when 'e' key is pressed
            try:
                # Perform emotion detection
                emotion = analyze_emotion(frame)

                # Print the detected emotion in the terminal
                print(f"Detected Emotion: {emotion}")

                # Display emotion on the frame
                font = cv2.FONT_HERSHEY_SIMPLEX
                text = f"Emotion: {emotion}"
                cv2.putText(frame, text, (10, 30), font, 1, (0, 255, 0), 2, cv2.LINE_AA)

                # Show the frame with the detected emotion
                cv2.imshow("Emotion Detection (Live Feed)", frame)

            except Exception as e:
                print(f"Error during emotion analysis: {e}")

    cap.release()
    cv2.destroyAllWindows()

if __name__ == "__main__":
    main()
```

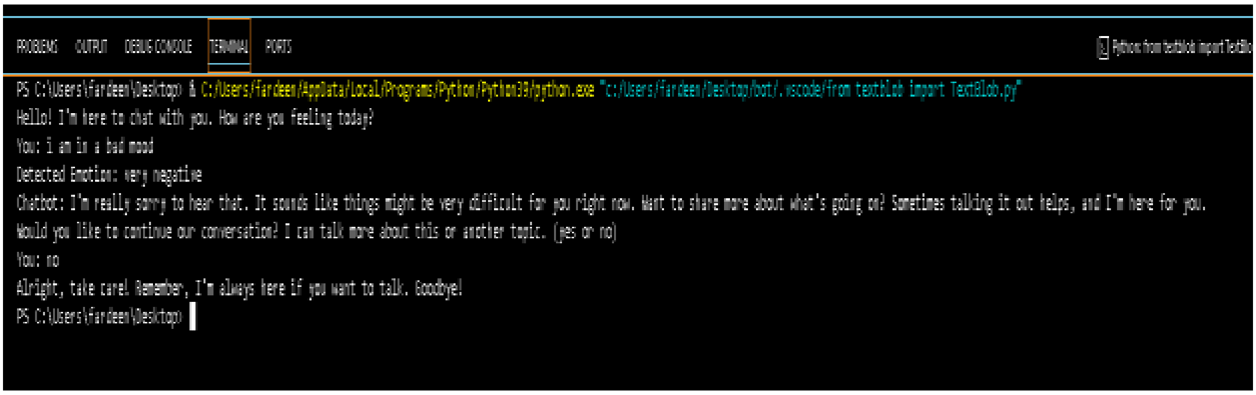
```
PS C:\Users\Fardeen\Desktop> & C:/Users/Fardeen/AppData/Local/Programs/Python/Python312/python.exe "c:/Users/Fardeen/Desktop/bot/.vscode/import cv2.py"
Starting webcam for emotion detection. Press 'q' to quit, 'c' to capture an image, or 'e' to detect emotion.
Detected Emotion: neutral
Detected Emotion: surprise
Detected Emotion: surprise
Detected Emotion: happy
Detected Emotion: happy
Detected Emotion: neutral
Detected Emotion: happy
Detected Emotion: surprise
Detected Emotion: neutral
Detected Emotion: happy
```

Fig 3 Live Webcam



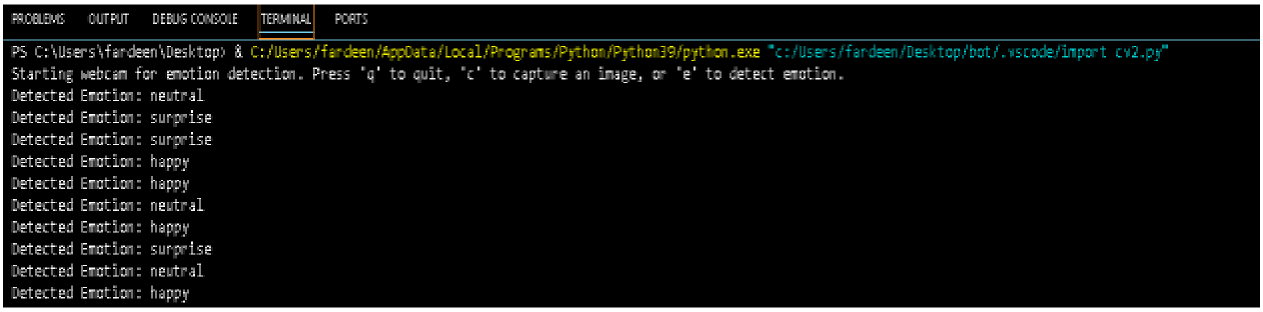
```
PS C:\Users\fardeen\Desktop> & C:/Users/fardeen/AppData/Local/Programs/Python/Python39/python.exe "c:/Users/fardeen/Desktop/bot/.vscode/from textblob import TextBlob.py"
Image captured as captured_images\captured_image.jpg.
Detected Emotion: happy
Image captured as captured_images\captured_image.jpg.
Detected Emotion: neutral
Image captured as captured_images\captured_image.jpg.
Detected Emotion: sad
Image captured as captured_images\captured_image.jpg.
Detected Emotion: angry
Image captured as captured_images\captured_image.jpg.
Detected Emotion: happy
Exiting...
PS C:\Users\fardeen\Desktop>
```

Fig 4 Image Capturing



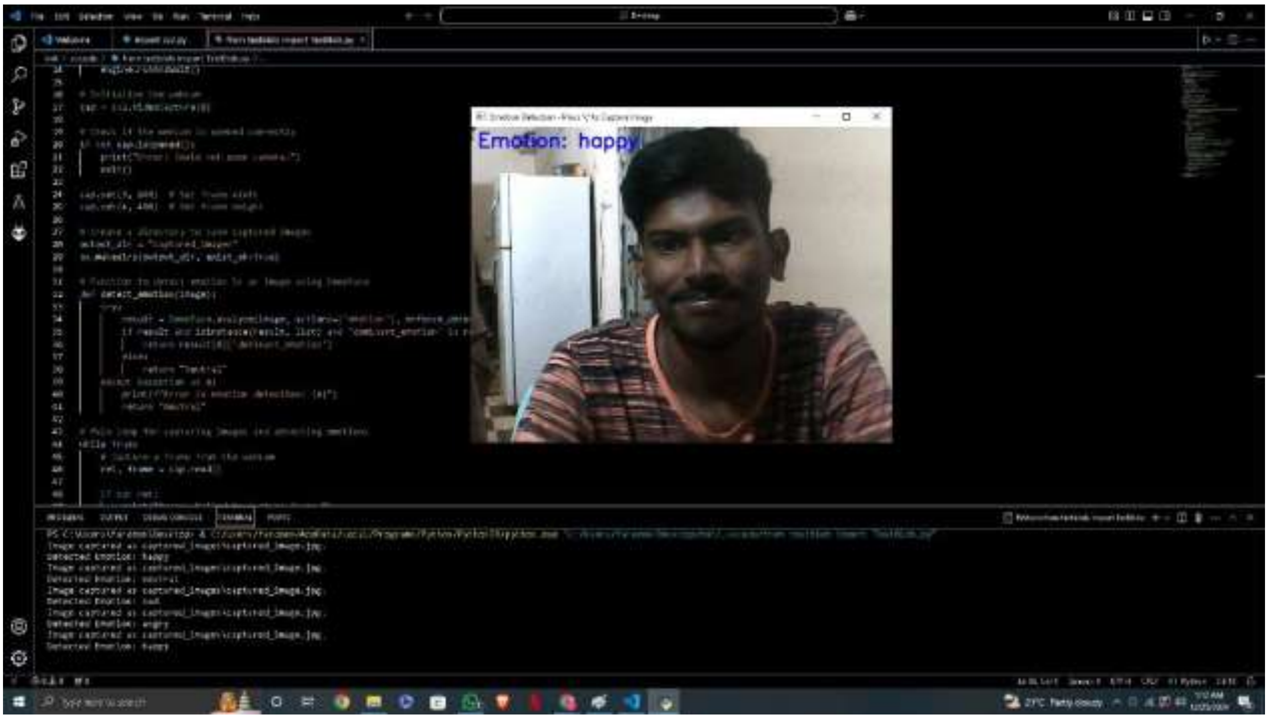
```
PS C:\Users\fardeen\Desktop> & C:/Users/fardeen/AppData/Local/Programs/Python/Python39/python.exe "c:/Users/fardeen/Desktop/bot/.vscode/from textblob import TextBlob.py"
Hello! I'm here to chat with you. How are you feeling today?
You: i am in a bad mood
Detected Emotion: very negative
Chatbot: I'm really sorry to hear that. It sounds like things might be very difficult for you right now. Want to share more about what's going on? Sometimes talking it out helps, and I'm here for you.
Would you like to continue our conversation? I can talk more about this or another topic. (yes or no)
You: no
Alright, take care! Remember, I'm always here if you want to talk. Goodbye!
PS C:\Users\fardeen\Desktop>
```

Fig 5 Text Response



```
PS C:\Users\fardeen\Desktop> & C:/Users/fardeen/AppData/Local/Programs/Python/Python39/python.exe "c:/Users/fardeen/Desktop/bot/.vscode/import cv2.py"
Starting webcam for emotion detection. Press 'q' to quit, 'c' to capture an image, or 'e' to detect emotion.
Detected Emotion: neutral
Detected Emotion: surprise
Detected Emotion: surprise
Detected Emotion: happy
Detected Emotion: happy
Detected Emotion: neutral
Detected Emotion: happy
Detected Emotion: surprise
Detected Emotion: neutral
Detected Emotion: happy
```

Fig 6 Emotion Detected



```
Image captured as captured_images\captured_image.jpg.
Detected Emotion: happy
Image captured as captured_images\captured_image.jpg.
Detected Emotion: neutral
Image captured as captured_images\captured_image.jpg.
Detected Emotion: sad
Image captured as captured_images\captured_image.jpg.
Detected Emotion: angry
Image captured as captured_images\captured_image.jpg.
Detected Emotion: happy
```

Fig 7 Terminal Response with Webcam

CONCLUSION

The development of the Emotion Recognition and Voice Command Chatbot System represents a significant leap in the way humans interact with machines, particularly in the domain of emotional intelligence and empathy. The system combines multiple advanced technologies—computer vision, speech recognition, and natural language processing—to create an interactive interface that can understand both emotional and verbal cues. By using computer vision techniques to capture and analyse emotions from facial expressions and voice recognition to interpret spoken commands, the system aims to create a more natural, responsive, and empathetic interaction.

This multi-modal approach ensures that the system can engage in meaningful conversations with users, tailoring its responses based on real-time emotional input, which is a major step forward compared to traditional chatbots that only focus on text-based or command-driven interactions. By leveraging Deep Face for emotion detection, the system accurately identifies various human emotions such as happiness, sadness, anger, surprise, and more. This emotion detection is essential for providing contextually appropriate responses, which can help improve user experiences in sectors such as mental health support, customer service, and education.

In conclusion, while the system already provides significant value, it is only the beginning. With ongoing improvements and adaptation to user feedback, the Emotion Recognition and Voice Command Chatbot System has the potential to become an indispensable tool in a variety of fields, promoting more human-like, emotionally aware interactions that cater to the needs of users in an increasingly digital world.

REFERENCE

[1] John Batani, Elliot Mhunge, Rambang, Eswarini, Maseru Lesotho A Deep Learning-Based Chatbot to Enhance Maternal Health Education 2034 Conference on Information Communications Technology and Society.

[2] Maria Oganda, John Cluksara, Strathmore Elenersity Namobi, Joseph Gitongs, Joseph Sevilla, Betry Mariti Wellness Buddy: An Al Mental Health Chatbot for Kenyan University Students.

[3] Can we trust Health and Wellness Chatbot going mobile? Empirical research using TAM and HBM De Mugdha Kulkami Symbi Copy of m Technology (SCT).

[4] Challenge in Classification of Depressive Symptoms Using Actigraphy Data

[5] Data Visualization for Mental Health Monitoring In Smart Home Environment: A Case Study Youngji Koh Soheal el Computing KAIST, Chanhee Lee Saneel or Computing KAIST, Yunhee Ku LG Electronics Uichin Lee School of Computing KAIST.

[6] College Student Mental Health Early Warning Platform Based on Keyword Matching Algorithm Lee Zhang, Chas Huang.