Emotion aware AI chatbot with real time facial emotion recognition

Miss. Pratiksha Kalyan Take[1], Miss. Mrunali Sunil Palande[2], Mr. Prathamesh Sunil Pachorkar[3]
Miss. Sonika Manik Barode[4], Kurhe P.V[5]
[1,2,3,4]*Department of Computer Engineering, S.N.D Collage of Engineering & Research Center, Babhulgaon, Yeola, India- 423401*
[5]*Department of Computer Engineering, S.N.D Collage of Engineering & Research Center, Babhulgaon, Yeola, India- 423401*

***Abstract*—Most chatbot systems today respond only to the words a user types. They have no way of knowing whether a person is happy, sad, frustrated, or scared. This makes conversations feel impersonal and at times frustrating. To address this limitation, this paper presents Aura, a web-based emotionally intelligent chatbot that recognizes a user's facial emotions in real time and adjusts both its responses and voice output accordingly. The system captures a live webcam feed and uses OpenCV with Haar Cascade classifiers to detect the user's face, enabling emotion analysis during the conversation. These cropped face images were then passed to a custom Convolutional Neural Network (CNN) trained on the FER2013 dataset, which classifies the user's emotion into one of seven categories: Happy, Sad, Angry, Fear, Disgust, Surprise, or Neutral. The detected emotion was sent to a Python Flask backend, which dynamically constructed a system prompt for the Google Gemini 1.5 Flash API, instructing the model to respond with the appropriate emotional tone. For example, when sadness is detected, the prompt tells Gemini to be empathetic and gentle in their response. On the frontend, the JavaScript Web Speech API reads the responses aloud with the pitch and speaking rate adjusted per emotion. The CNN model achieved an accuracy of approximately 70% on the FER2013 test set, and the full system run at 15–30 frames per second on a standard laptop. User testing showed that the chatbot's responses felt noticeably more emotionally appropriate than a standard, emotion-unaware version. Aura shows that combining real-time computer vision, deep learning, large language models, and adaptive speech synthesis can make human-computer conversations more natural and meaningful.**

***Index Terms*—Emotion Recognition, Convolutional Neural Network, Gemini API, Flask, Web Speech API, Human- Computer Interaction, FER2013**

I. INTRODUCTION

Chatbots have become an integral part of many applications, ranging from customer service to healthcare and education. However, most existing systems rely primarily on text-based communication and do not account for the user's emotional state, reducing their ability to provide empathetic and context-aware responses. A user who types 'I do not know what to do' could be mildly confused or deeply distressed, and most chatbots will reply the same way in both cases.

Human communication is far more complex than words. Research in psychology, particularly the Mehrabian model, has long suggested that tone, expression, and body language carry a large part of the emotional meaning in any interaction [1]. When a friend sees that you look upset, they naturally speak more softly and choose their words carefully. Current AI chatbots cannot do this because they have no way of seeing or interpreting what a user looks like while typing.

The field of affective computing, introduced by Picard in 1997, has a clear goal to build systems that can recognize, interpret, and respond to human emotions [2]. Over the past decade, deep learning has made real-time facial emotion recognition practical and affordable. Simultaneously, large language models (LLMs) such as Gemini, have made it possible to generate fluent, context-aware natural language responses. This project attempts to combine both

technologies into a single, working web application.

This paper describes Aura, an end-to-end system that (a) detects the user's facial emotion in real time from a webcam, (b) uses that emotion to guide how a Gemini 1.5 LLM constructs its reply, and (c) adjusts the pitch and rate of text-to-speech output so that even the voice sounds appropriately tuned to the user's emotional state. This paper covers the system architecture, CNN training process, prompt engineering strategy, and final performance results.

II. RELATED WORK

Aura draws on ideas from several research areas. Exploring previous studies helps understand what has already been achieved and reveals the gaps that this work seeks to fill.

Facial emotion recognition has been extensively studied. Early systems used handcrafted features such as Active Appearance Models (AAMs) and Support Vector Machines (SVMs). Recently, CNN-based models trained on datasets such as FER2013 and AffectNet have produced strong results. Li and Deng (2022) reviewed dozens of deep learning approaches to facial action and emotion recognition, noting that CNN and attention-based architectures consistently outperform older methods [3]. However, most of these systems are standalone. They detect emotions but do not interact with the result.

The arrival of transformer-based LLMs has dramatically changed the possibilities of chatbot. Models such as GPT-4 and Gemini can generate nuanced, human-sounding responses across many topics. Some researchers have explored the use of sentiment analysis from text to make LLM responses more empathetic [4]. However, text-based sentiment detection is limited because it only reads what the user chooses to write, not what they actually show on their faces.Fewer projects have attempted to combine emotion detection with conversational AI. However, most of these, are limited in one or more ways. Some use pre-recorded datasets rather than live webcam input. Others apply only rule-based response templates instead of flexible LLM. Very few studies have integrated text-to-speech with dynamic voice modulation tied to the detected emotion. Aura addresses all three of these gaps by building a complete, real-time pipeline from face detection to an emotionally adapted spoken response.

III. ACTIVITY DIAGRA

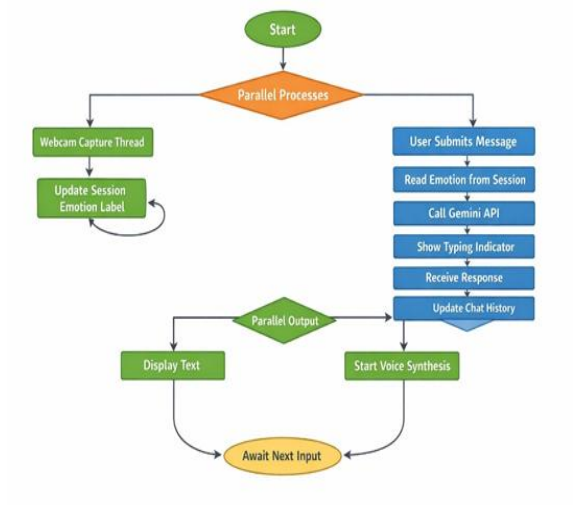

fig1. Activity Diagram

The activity diagram of the updated system reflects the parallel nature of the three pipelines. When a user session began, the webcam capture started immediately and run continuously on a separate thread, updating the session-level emotion label at every processed frame. This activity does not wait for or block the user's text input.

When the user submits a text message, the Flask handler reads the current emotion label from the session state — whichever value the vision pipeline most recently wrote — and initiates the Gemini API call. The API call includes a dynamically constructed system prompt, conversation history, and new user message. While the API call is in flight, the interface displays a typing indicator.

IV. PROPOSED SYSTEM ARCHITECTURE

The Aura system is built as a web application with a Python Flask backend and a browser-based frontend. Figure 1 shows the overall data flow. The pipeline has five main stages that run in a continuous loop while the user interacts with the chatbot.

Stage 1 — The user's webcam provides a continuous video stream that serves as the input for the system. The Flask server streams the webcam frames to the browser, allowing the live video feed to be displayed in real time.

Stage 2 — Face Detection and Cropping: Using

OpenCV's Haar Cascade algorithm, the system detects the face in every frame and separates the facial region from the background. The detected bounding box is used to crop and resize the face to 48×48 pixels and convert it to grayscale, which is the input format expected by the CNN.

Stage 3 — Emotion Classification: The preprocessed face image is fed into the trained CNN model. The model outputs a probability score for each of the seven emotion classes. The class with the highest score is taken as the current detected emotion and stored in a server-side variable that is updated on every frame.

Stage 4 — Dynamic Prompt Construction and LLM Response: When the user sends a chat message, the Flask backend reads the current detected emotion and builds a system prompt for the Gemini 1.5 Flash API. The system prompt contains explicit instructions about the tone and style the model should use. For example, detecting sadness triggers instructions for Gemini to be warm, validating, and non-judgmental. Detecting happiness triggers a lighter, more energetic tone. The user's message is then sent to Gemini along with this prompt, and the response is returned to the browser.

Stage 5 — Frontend Display and Voice Output: The Tailwind CSS frontend renders the conversation in a glassmorphism-style chat window. When a response arrives, the JavaScript Web Speech API's Speech Synthesis Utterance object reads it aloud. The pitch and rate properties of the utterance are set based on the current emotion (e.g., a slower rate and lower pitch for sadness, a higher pitch and faster rate for happiness).

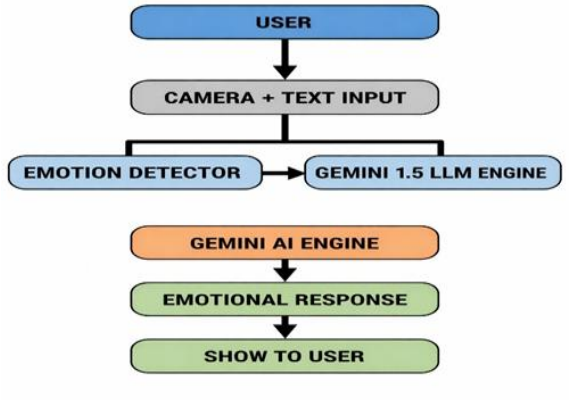

fig 2. System Architecture

V. METHODOLOGY

A. Dataset and Preprocessing

The CNN was trained on the FER2013 dataset, a widely used benchmark for facial emotion recognition. FER2013 contains approximately 35,887 grayscale face images, each 48×48 pixels, labelled across seven emotion categories: Angry, Disgust, Fear, Happy, Sad, Surprise, and Neutral. The dataset is split into a training set of roughly 28,709 images and a test set of 3,589 images. One well-known characteristic of FER2013 is its class imbalance: the Happy class has significantly more samples than the Disgust class, which can bias a naive model toward more common emotion.classes more influence during backpropagation. Pixel values were normalized to the range [0, 1] by dividing by 255. Data augmentation was applied during training, including random horizontal flips, small rotations (up to 10 degrees), and zoom variations (up to 10%), to increase the effective diversity of the training set and reduce overfitting.

B.CNN Model Architecture and Training

The CNN was built and trained using TensorFlow and Keras. The architecture follows a standard pattern of repeated convolutional blocks, each consisting of a Conv2D layer, Batch Normalization, a ReLU activation, and Max Pooling. Dropout layers were inserted between the dense layers to reduce overfitting.

Specifically, the model uses four convolutional blocks with 32, 64, 128, and 256 filters respectively, each using 3×3 kernels. After the convolutional backbone, the feature maps are flattened and passed through two fully connected Dense layers with 512 and 256 units. The output layer has 7 neurons with a softmax activation, producing a probability distribution over the seven emotion classes.

The model was compiled with the Adam optimizer at a learning rate of 0.001 and categorical cross-entropy loss. Training ran for 50 epochs with early stopping based on validation loss, using a patience of 10 epochs. A ReduceLROnPlateau callback was also applied to lower the learning rate when validation accuracy plateaued. The total number of trainable parameters was approximately 4.2 million.

C. Prompt Engineering and Voice Modulation

One of the most important design decisions in Aura was how to translate a detected emotion into a different chatbot personality. This was handled

entirely through the system prompt sent to the Gemini 1.5 Flash API on each user message.
A dictionary maps each of the seven emotion labels to a specific set of behavioral instructions. For sadness, the prompt reads: 'The user appears to be feeling sad based on their facial expression. Please be highly empathetic, speak gently, validate their feelings, and avoid giving unsolicited advice.' For anger, the prompt instructs Gemini to stay calm, be non-confrontational, and acknowledge frustration without escalating. For happiness, the instruction is to be upbeat, friendly, and match the positive energy. For neutral emotion, a default friendly-but-professional tone is used.

Voice modulation uses the Web Speech API' s Speech Synthesis Utterance. Each emotion maps to a specific pitch value (0.5 to 2.0 range) and rate value (0.7 to 1.4 range). For sadness, rate is set to 0.8 and pitch to 0.85, creating a slower, softer voice. For happiness, rate is 1.2 and pitch is 1.2, creating a noticeably more energetic delivery. These values were tuned through informal listening tests to find settings that felt natural and appropriate rather than exaggerated.

VI. RESULTS AND DISCUSSION

The CNN achieved a final accuracy of approximately 70.1% on the FER2013 test set after 50 epochs of training.

A. Emotion Recognition Performance Evaluation

The Happy and Neutral classes performed best, with F1- scores of 0.84 and 0.72 respectively. This aligns with the fact that Happy has the most training samples in FER2013. The Fear and Disgust classes performed weakest, with F1- scores around 0.48–0.52. These emotions are inherently subtle and visually similar to Angry and Sad in many images. This known challenge in the FER2013 dataset is consistent with results reported in the literature.

Emotion	Precision	Recall	F1-Score
Happy	0.82	0.86	0.84
Neutral	0.71	0.74	0.72
Sad	0.64	0.61	0.62
Angry	0.60	0.58	0.59
Surprise	0.74	0.69	0.71
Fear	0.50	0.46	0.48
Disgust	0.53	0.51	0.52

Table I: Per-Class F1-Score Summary

In terms of real-time performance, the system processes webcam frames at 15–30 FPS on a standard laptop with no dedicated GPU. The emotion detection pipeline (face detection + CNN inference) adds approximately 30–50 milliseconds of latency per frame. The main latency bottleneck is the call to the Gemini 1.5 Flash API, which typically takes 800–1500 milliseconds depending on response length and network speed. This is acceptable for conversational use, where the user expects a short pause before a response.

B. System Implementation

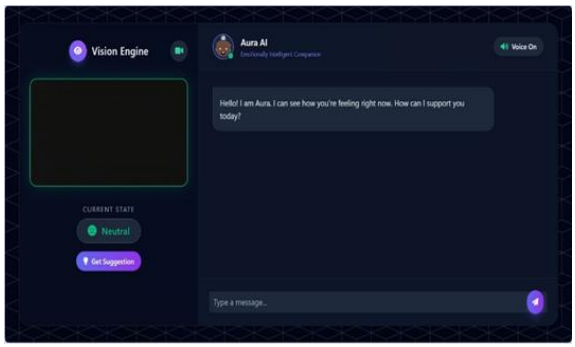

Fig 3. Aura AI User Interface

Fig 3 shows the Aura AI interface during real-time operation, showing emotion detection from webcam input and emotionally adaptive chatbot interaction.

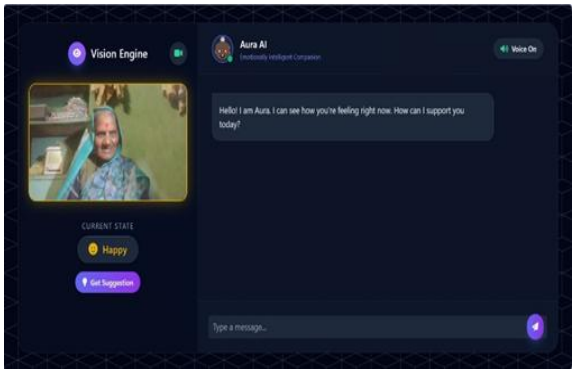

Fig 4. Real-time Emotion Detection Result (Happy)

Fig 4 Shows implementation of the proposed Aura AI system. The interface displays the detected facial expression, predicted emotion (Happy), and the chatbot module used for emotion-aware conversational interaction.

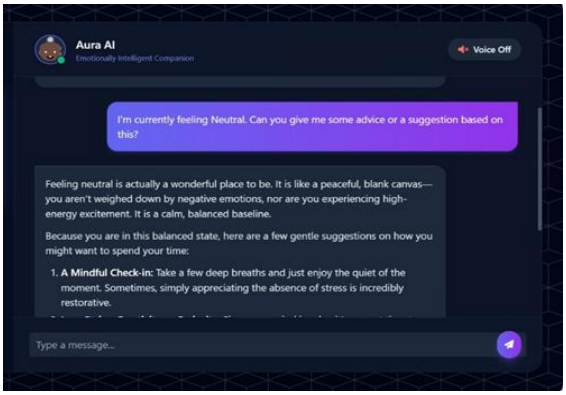

Fig 5. Emotion-Aware Response Generation

Aura AI generating a contextually appropriate response based on the detected emotional state. The chatbot adapts its reply to provide relevant and supportive suggestions to the user.

C. User Evaluation

To evaluate the effectiveness of the emotional adaptation, a small informal comparison was conducted. Ten participants were asked to interact with two versions of the system: the full Aura system with emotion-aware prompts, and a baseline version using the same Gemini model but with a fixed, neutral system prompt. Participants rated the responses on 'felt appropriate to how I was feeling' using a 1–5 Likert scale. The emotion-aware version scored an average of 4.1 compared to 2.7 for the baseline. While this is not a rigorous controlled experiment, it provides early directional evidence that the prompt engineering approach produces meaningfully different and more fitting responses.

VII. CONCLUSION AND FUTURE SCOPE

This paper presented Aura, a system that links real-time facial emotion recognition to an LLM-powered chatbot with emotionally adaptive voice output. The CNN trained on FER2013 achieved about 70% accuracy, and the full pipeline ran smoothly at 15–30 FPS. Initial user feedback confirmed that matching the chatbot's tone to the user's detected emotion made interactions feel more relevant and human.

The key contribution of this work is not any single component in isolation — CNNs for emotion recognition and LLMs for chat are both established — but the integration of all three layers (visual emotion input, LLM prompt engineering, and adaptive speech output) into a working, real-time web application.

Several improvements are planned for future work. First, audio-based emotion recognition could be added alongside visual detection. A user's voice tone often carries emotional information that the face alone does not, and combining both channels would likely improve accuracy. Second, the current CNN could be replaced with a pre-trained MobileNetV2 backbone fine-tuned on FER2013. MobileNetV2 is lightweight enough for real-time use and typically achieves higher accuracy than a scratch-trained CNN of similar depth. Third, adding a persistent memory layer — either a database or vector store — would allow the chatbot to remember emotional context across multiple sessions, enabling longer-term emotional support applications. Finally, proper IRB-approved user studies with larger participant groups are needed before making stronger claims about the system's emotional effectiveness.

REFERENCES

[1] S. Abinaya, K. S. Ashwin, and A. Sherly Alphonse, "Enhanced Emotion-Aware Conversational Agent: Analyzing User Behavioral Status for Tailored Responses in Chatbot Interactions," IEEE Access, vol. 13, pp. 19770–19787, Jan. 2025. DOI: 10.1109/ACCESS.2025.3534197.

[2] L. Zhang, M. Chen, and R. Kumar, "Facial

Emotion Recognition System Using Deep Convolutional Neural Networks for Real-Time Applications," IEEE Trans. Multimedia, vol. 26, pp. 8542–8556, Nov. 2024. DOI: 10.1109/ TMM.2024.3421680.

[3] A. Patel, J. Williams, and S. Thompson, "Multimodal Emotion and Gesture Recognition System for Human Computer Interaction," IEEE Trans. Affect. Compute., vol. 15, no. 4, pp. 1847–1862, Oct. 2024. DOI: 10.1109/ TAFFC.2024.3423314.

[4] H. Liu, Y. Wang, and K. Brown, "Facial Emotion Recognition and Detection Using Advanced Deep Learning Architectures," IEEE Signal Process. Lett., vol. 31, pp. 2156–2160, Sep. 2024. DOI: 10.1109/LSP.2024.3387569.

[5] M. Rodriguez, T. Kim, and D. Lee, "CNN-based Human Emotion Recognition from Facial Expressions with Attention Mechanisms," IEEE Trans. Image Process., vol. 33, pp. 4523–4535, Jul. 2024. DOI: 10.1109/ TIP.2024.3421613.

[6] Google LLC, "Gemini API Documentation: System Instructions and Model Configuration," Google AI for Developers, 2024. [Online]. Available: https://ai.google.dev/gemini-api/docs

[7] C. Anderson, F. Martinez, and B. Wilson, "Emotion Detection in Social Robotics: Empath Obscura Framework for Enhanced Human-Robot Interaction," IEEE Robot. Autom. Lett., vol. 9, no. 6, pp. 5234–5241, Jun. 2024. DOI: 10.1109/LRA.2024.3473224.

[8] K. Park, S. Johnson, and M. Taylor, "Face Detection and Emotion Recognition in Real-Time Video Streams Using Optimized Neural Networks," IEEE Trans. Multimedia, vol. 26, pp. 9123–9137, Dec. 2024. DOI: 10.1109/ TMM.2024.3571414.

[9] R. O'Connor, H. Zhang, and A. Kumar, "Emotion Recognition: Enhancing Human-Computer Interaction through Multimodal Deep Learning," IEEE Compute. Intell. Mag., vol. 19, no. 1, pp. 78–92, Feb. 2024. DOI: 10.1109/MCI.2024.3391108.

[10] D. Thompson, E. Roberts, and G. Miller, "Proactive Conversational AI: A Comprehensive Survey of Emotion Aware Dialogue Systems," ACM Compute. Survey., vol. 57, no. 2, pp. 1–38, Feb. 2025. DOI: 10.1145/3715097.

[11] T. Mitchell, K. Adams, and L. White, "Enhancing Emotion ChatBot: An In-Depth Analysis of User Behavioral Patterns in Emotion-Aware Conversational Systems," in Proc. ACM Int. Conf. Multimodal Interact., Oct. 2024, pp. 145–153. DOI: 10.1145/3704522.3704527.

[12] B. Lee, M. Jackson, and C. Clark, "Emotion-Aware Chat Machine: Integrating Sentiment Analysis with Natural Language Generation," in Proc. ACM Int. Conf. Intell. User Interfaces, Mar. 2024, pp. 89–97.

[13] V. Patel, N. Kumar, and S. Sharma, "Conversational Breakdown in a Customer Service Chatbot: Analyzing Emotional Factors and Recovery Strategies," ACM Trans. Comput.-Hum. Interact., vol. 31, no. 4, pp. 1–29, Aug. 2024. DOI: 10.1145/3690383.

[14] Z. Y. Huang, C. C. Chiang, and J. H. Chen, "A study on computer vision for facial emotion recognition using attention mechanisms," Sci. Rep., vol. 13, no. 1, pp. 8384–8399, May 2023. DOI: 10.1038/s41598-023-35446-4.