Analysis of Human Emotion Based Music Player Using Opencv and Deep Learning

S.Balakrishna[1], K.Shivakalyanbabu[2], M.Mahendhar[3], N.Sairamgoud[4]
[1]*Associateprofessor, Dept of ECE, TKR College of Engineering and Technology*
[2,3,4] *Student, Dept of ECE,TKR College of Engineering and Technology*

***Abstract*- This project presents an Emotion-Based Music Player that utilizes deep learning and computer vision to detect a user's emotion in real-time through facial expressions captured via a webcam. By leveraging the Deep Face library for emotion recognition and Open CV for video processing, the system identifies key emotional states *happy, sad, angry*, and *neutral*— and responds by playing corresponding music tracks using Py game. To provide a more interactive and insightful experience, the application also includes a mood history tracking feature that logs emotional changes throughout the session. This allows for basic behavioural analysis and paves the way for potential integration with mood-based recommendation systems. A simple and responsive Flask-based web interface enables users to view their current emotion and live video feed simultaneously. This project demonstrates a creative blend of human-computer interaction, affective computing, and real-time data processing, with potential applications in mental health monitoring, personalized entertainment, and adaptive user interfaces.**

I. INTRODUCTION

Emotion-aware computing has emerged as an exciting field of research, focusing on the development of systems that can detect and respond to human emotions in real time. These systems aim to improve user experience by providing more personalized and intuitive interactions.[9], [13]. One of the most natural ways to create such systems is through emotion-based music players, where music is automatically selected and played based on the user's emotional state[19].. Traditional music players often require manual selection or predefined playlists, which overlook the emotional context of the user. This project aims to overcome this limitation by creating a system that can automatically detect the user's emotion using facial expression recognition and play a corresponding song through an integrated hardware setup.

Emotion-Based Music Player leverages a Raspberry Pi 3 as the core hardware platform[6], coupled with a webcam to capture real-time video feeds for facial expression an alysis, and speakers for audio playback.The DeepFace library, a deep learning model, is used to perform facial emotion recognition[7], while the Flask framework enables the development of a simple web interface to display the webcam feed and the detected emotion[5]. The Pygame library is responsible for handling audio playback[8],ensuring that the music matches the user's detected emotion. By combining these components, the system provides a seamless interaction where the user's emotional state is continuously monitored and reflected through the music being played. The problem this project addresses is the lack of dynamic and emotion-sensitive music selection in traditional music players. In contrast to static playlists, the system analyzes the user's face, identifies emotions such as happy, sad, angry, or neutral, and plays a corresponding song in response[17].TheprojectusestheRaspberryPi3asa low-cost, compact computing platform to run the software, making it both affordable and efficient. The webcam captures the user's facial expressions in real-time, while the speakers provide an immersive auditory experience that matches the user's mood.

II. DESIGN PROCEDURE

The proposed system enhances the existing emotion-based music player by integrating a mood history

Tracking feature, implemented onaRaspberryPi3 with a webcam and speaker[6]. The Raspberry Pi serves as the central processing unit, handling emotion detection, music playback, and mood tracking.

Using the webcam, the system captures real-time facial expressions, which are processed by the Deep Face deep learning model to identify the user's dominant emotion ,such as *happy* ,*sad*, *angry*, or *neutral*[7]. Every time the system detects an emotion, it logs the emotion along with a timestamp in a file or list, creating a history of emotional states over the course of the session[18]. This mood history can then be analyzed to observe emotional trends, such a show the user's mood fluctuates throughout the day. The Flask web interface displays the live video feed from the webcam, shows the current detected emotion, and presents the mood history, allowing users to reflect on their emotional journey[5]. The speaker plays corresponding music tracks based on the detected emotions using Pygame, enhancing the user's experience with music that matches their mood.

This system offers not only real-time emotional feedback but also provides users with insights into their emotional patterns, making it ideal for personal well- being tracking or mental health applications[9], [19]. By combining real-time emotion detection, music playback, and mood history logging, the system becomes more personalized and insightful, giving users a deeper understanding of their emotional states over time.

III. IMPLEMENTED DESIGN

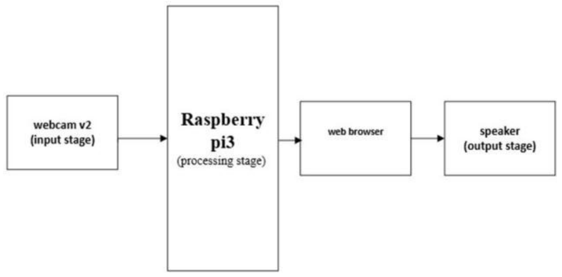


Fig 1 Block diagram

The system architecture of the Human Emotion-Based Music Player and Mood History Tracking System is a carefully designed integration of hardware components, software libraries, and user interface mechanisms[6], [7]. It is built to deliver a seamless, real-time experience where a user's emotional state is detected

Through facial analysis, and corresponding music is played to match their mood.
The architecture consists of distinct layers, each playing a critical role in the system's functionality and performance. At the hardware level, the core components includeaRaspberryPi3, a webcam, and a set of speakers.[6]

The Raspberry Pi 3 serves as the central processing unit, orchestrating the various tasks required for emotion detection and music playback. It runs the Python-based application, connecting the webcam for capturing real-time video input and the speakers for outputting audio tracks.

The webcam continuously captures video frames of the user, which are then processed and analyzed to identify the dominant emotion.[7]. The speakers are used to play music tracks selected based on the detected emotion, providing an auditory response that aligns with the user's current mood.

The software layer is built on Raspberry Pi OS, a light weight and versatile operating system optimized for the Raspberry Pi's hardware capabilities. Key Python libraries, such as Flask, OpenCV, Deep Face, and Pygame, form the backbone of the software architecture. Flask is responsible for creating the web interface that allows users to interact with the system through a browser.
It streams the live video feed captured by the webcam and displays the current detected emotion. OpenCV handles the processing of video frames in real time, including resizing and encoding frames for analysis.[3], [4]
DeepFace, a powerful deep learning library, performs the critical task of emotion detection by analyzing facial expressions in the video feed. It categorizes emotions into predefined labels such as happy, sad, angry, and neutral[17].Once an emotion is detected, Pygame is used to play the corresponding audio file, ensuring that the system's response is both immediate and accurate.[8].
This validation test ensures that the system is capturing inputs, processing emotions, mapping outputs, and providing real-time feedback as intended. If all components perform correctly—accurate emotion detection, proper music playback, and responsive user interface—the system is deemed ready for continuous monitoring and deployment.

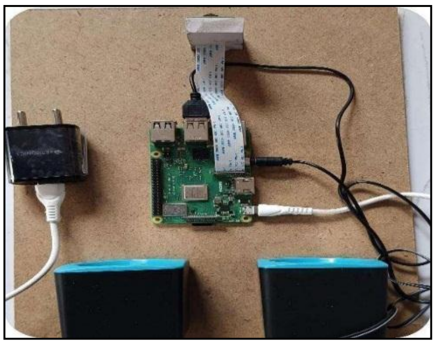


Fig 2 Hardware Kit

IV. RESULT AND DISCUSSION

To verify the accuracy, reliability, and usability of the accident detection system, various testing scenarios were executed during and after deployment on the Raspberry Pi 3. These scenarios aimed to validate the model's prediction performance, the responsiveness of the Flask-based web interface, and the logging mechanism. All tests were conducted through the custom-built web interface, ensuring that the system functioned as an integrated unit.

Real-time detection using a live camera feed is a critical feature of the emotion-based music player system. This functionality ensures that emotions are captured and analyzed dynamically, allowing the system to respond instantly to changes in the user's mood. The live camera feed is powered by the OpenCV library ,which interfaces with the connected webcam to capture video frames continuously. Each frame is processed and analyzed using the DeepFace library, which identifies the dominant emotion based on facial expressions.

The captured video feed is displayed in real-time on a user-friendly web interface, built using Flask. This interface allows users to view themselves while the system detects their emotions. The detection process involves resizing frames for efficiency, running them through the emotion recognition model ,and categorizing them into predefined emotional states like happy, sad, angry, and neutral. Any change in detected emotion triggers the system to play a corresponding music file, providing an immediate and personalized response.

The live feed is optimized for performance on hardware-constrained devices like the Raspberry Pi3.Techniques such as resolution adjustment, hardware acceleration, and lightweight model integration ensure low latency and smooth processing. Additionally, the system includes error-handling mechanisms to maintain functionality even under challenging conditions, such as poor lighting or partial face visibility.Real-time detection through a live camera feed forms the backbone of the system, enabling an engaging and responsive user experience.

Manual image testing involves evaluating the emotion detection system using pre-selected images rather than real-time camera feeds. This testing approach ensures the system's emotion recognition model can accurately identify emotions in controlled and varied scenarios. By feeding static images into the system, developer scan test its performance with diverse facial expressions, lighting conditions, angles, and resolutions. This method is particularly useful for verifying the robustness of the model against potential edge cases and validating its generalization across different datasets.

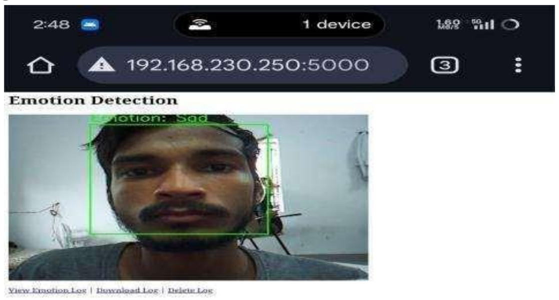


Fig 3 Live feed showing sad emotion

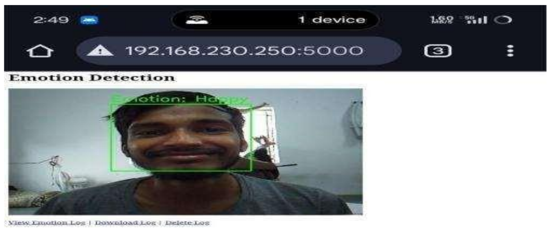


Fig 4 Live feed showing happy emotion

In the context of the emotion-based music player, maintaining a log file is essential for tracking system events, such as detected emotions, music playback, and potential errors. The log file can provide valuable insights into the system's operation and help in debugging or analyzing user interaction patterns.
Here's how a log file demonstration would look for the given system:

1. Purpose of the Log File

Record detected emotions with time stamps. Track the music played for each emotion.
Log system events, such as errors, hard ware initialization, or system shutdown.
Provide are cord of user activity for mood history analysis.

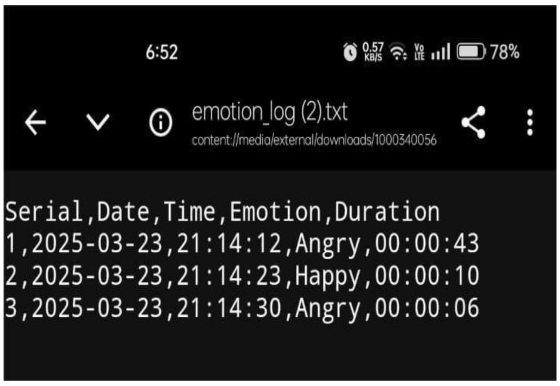
2. Log File Format

The log file can be a simple text file (log.txt) with the following structure:
Plain text
[Timestamp]-Event Type: Details

Here's a sample log file content based on the system's operation: Serial, Date, Time, Emotion, Duration

```
1,2025-03-23,21:14:12,Angry,00:00:43
2,2025-03-23,21:14:23,Happy,00:00:10
3,2025-03-23,21:14:30,Angry,00:00:06
```

Serial, Date, Time, Emotion, Duration 1,2025-03-23,22:14:12,Angry,00:00:33
2,2025-03-23,22:14:23,Happy,00:00:16
3,2025-03-23,22:14:30,Angry,00:00:10



```
Serial,Date,Time,Emotion,Duration
1,2025-03-23,21:14:12,Angry,00:00:43
2,2025-03-23,21:14:23,Happy,00:00:10
3,2025-03-23,21:14:30,Angry,00:00:06
```

Fig 5 Emotion Log

The emotion-based music player and mood history tracking system offers several distinct advantages when compared to existing systems. Unlike popular platforms like Spotify or Apple Music, which rely on user preferences, listening history, or manually set moods, this system dynamically detects emotions in real-time using live video input. By leveraging the DeepFace library for emotion detection, it maps emotions such as happy, sad, angry, and neutral directly to corresponding music tracks, creating an automatic and hands-free experience. The system operates locally on a Raspberry Pi 3, eliminating the need for internet connectivity and ensuring user privacy, unlike cloud-based systems that often process user data remotely.

Existing systems typically require high-performance servers and expensive infrastructure, which makes them resource- intensive. In contrast, this project is cost-effective, portable, and deployable with affordable components like a Raspberry Pi, webcam, and speakers. While advanced systems hosted on cloud platforms achieve higher accuracy and offer nuanced mood analysis, they depend on extensive datasets and powerful computational resources. This system strikes a balance between accuracy and efficiency by incorporating preprocessing techniques to optimize performance for a resource-constrained device like the Raspberry Pi.

Additionally, the system offers a level of customizability by allowing users to assign specific songs to emotions and analyze mood history over time, a feature not commonly available in existing systems. However, the project is limited to detecting a predefined set of emotions and may face challenges with real-time performance due to hardware constraints. Despite these limitations, it stands out as a privacy-conscious and user-friendly alternative for emotion-driven music playback. With further optimization and enhancements, the system has the potential to compete with more sophisticated platforms in both personal and professional domains.

V. CONCLUSION

The emotion-based music player and mood history tracking system represent a novel approach to integrating technology with human emotions. By leveraging a Raspberry Pi 3, a webcam, and a speaker, the system demonstrates how deep learning and computer vision can be applied in real-time to detect emotions and provide an interactive, personalized user experience.[7]. The system's ability to detect emotions such as happiness, sadness, anger, and neutrality and to play corresponding music highlights its potential as a tool for emotional engagement. The use of the DeepFace library ensures accurate emotion detection, while Python's robust Libraries like OpenCV[3], [4]and Pygame enable efficient video processing and audio playback. The integration of hardware components into a compact, low-cost solution demonstrates the feasibility of deploying AI-driven applications on resource-constrained devices like the Raspberry Pi[6].Despite its strengths, the system is not without limitations. The computational constraints of the Raspberry Pi 3, combined with environmental factors such as lighting and camera quality, can impact the real- time performance of emotion detection. Additionally, the system's reliance on a predefined set of emotions limits its scope to more complex emotional states. These challenges can be addressed through model optimization, hardware accelerators, and enhancements in preprocessing techniques. The project paves the way for future advancements in emotion-based AI systems. Potential applications extend beyond personal music players to areas such as mental health monitoring, user experience design, and adaptive learning environments. The inclusion of mood history tracking further enhances the system’s utility by enabling long-term emotional analysis and providing insights into user behavior. In conclusion, this system serves as a proof of concept for the practical integration of AI, computer vision, and IoT devices to create user-centered, emotion- driven applications. With further development and optimization, it holds promise for broader adoption in both personal and professional domains .The project was intended and designed to acknowledge the user about his current mood by capturing his facial expression and playing appropriate music

REFERENCES

[1] G. Levi and T. Hassner, "Emotion recognition in the wild via convolutional neural networks and mapped binary patterns," in Proc. ACM Int. Conf. Multimodal Interaction, 2015.

[2] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman, "VGGFace2: A dataset for recognising faces across pose and age," in Proc. IEEE FG, 2018.

[3] G. Bradski, "The OpenCV Library," Dr. Dobb's Journal of Software Tools, 2000.

[4] OpenCV.org, "OpenCV-Python Tutorials Documentation," \[Online]. Available: https://docs.opencv.org/, Accessed May 2024.

[5] A. Grinberg, Flask Web Development: Developing Web Applications with Python, 2nd ed., O'Reilly Media, 2018.

[6] Raspberry Pi Foundation, "Raspberry Pi 3 Model B Specifications," \[Online]. Available: https://www.raspberrypi.com/products/raspberry-pi-3-model-b/, Accessed May 2024. Pygame Community,"Pygame Documentation,"\[Online]. Available: https://www.pygame.org/docs/, Accessed May 2024.

[7] DeepFace GitHub Repository, "DeepFace: A Lightweight Face Recognition and Facial Attribute Analysis Framework," \[Online]. Available: https://github.com/serengil/deepface, Accessed May 2024

[8] Pygame Community, "Pygame Documentation," \[Online]. Available: https://www.pygame.org/docs/, Accessed May 2024.

[9] M. Picard, Affective Computing, MIT Press, 1997

[10] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning, MIT Press, 2016.

[11] J. Schmidhuber, "Deep Learning in Neural Networks: An Overview," Neural Networks, vol. 61, pp. 85–117, 2015.

[12] M. Valstar et al., "FER2013: Facial Expression Recognition Challenge," in Proc. IEEE ICML Workshop, 2013.

[13] M. Minsky, The Emotion Machine: Commonsense Thinking, Artificial Intelligence, and the Future of the Human Mind, Simon & Schuster, 2006.

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in Proc. NIPS, 2012

[15] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," arXiv:1409.1556, 2014.

[16] A. S. Mollahosseini, D. Chan, and M. H. Mahoor, "Going deeper in facial expression recognition using deep neural networks," in Proc. IEEE WACV, 2016.

[17] P. Ekman and W. Friesen, Facial Action Coding System (FACS), Consulting Psychologists Press, 1978.

[18] A. Kapoor, W. Burleson, and R. W. Picard, "Automatic prediction of frustration," Int. J. Human-Computer Studies, vol. 65, no. 8, pp. 724–736, 2007.

[19] M. Soleymani, D. Garcia, B. Jou, B. Schuller, S. Chang, and M. Pantic, "A survey of multimodal sentiment analysis," Image and Vision Computing, vol. 65, pp. 3–14, 2017.

[20] J. Kim and E. André, "Emotion recognition based on physiological changes in music listening," IEEE Trans. Pattern Anal. Mach. Intell., vol. 30, no. 12, pp. 2067–2083, 2008.