

REAL TIME FACIAL RECOGNITION AND EMOTION DETECTION SYSTEM

*A project report submitted to the department of
Information Technology in partial fulfillment of the requirements
for the Degree of Bachelor of Technology*

By

Khandavalli Munni (21A21A1222)

Valavala Devaki Anandi (21A21A1258)

Baskarla Nikhita (21A21A1263)

Under the guidance of

Mr. M N V L NARAYANA M Tech (Ph. D)
Associate Professor, IT Dept.



DEPARTMENT OF INFORMATION TECHNOLOGY
SWARNANDHRA COLLEGE OF ENGINEERING & TECHNOLOGY

(Autonomous)

Seetharamapuram, Narsapur-534280

(Approved by A.I.C.T.E & Affiliated to JNTU Kakinada) (Accredited by

NAAC with A Grade in 2nd Cycle)

(Accredited by National Board of Accreditation (NBA) under Tier-1)

March 2025

CERTIFICATE

This is to certify that the project entitled “**Real Time Facial Recognition and Emotion Detection System**” submitted by **Khandavalli Munni** (Regd No. 21A21A1222), **Valavala Devaki Anandi** (Regd No. 21A21A1258), **Baskarla Nikhita** (Regd No. 21A21A1263) in the department of **Information Technology**, **Swarnandhra College of Engineering & Technology** for the award of the **Bachelors Degree in Information Technology** is bonafide project work carried out under our supervision.

Project Supervisor

Mr. M N V L Narayana M Tech (Ph.D)

Associate Professor, IT Dept.

Head of the Department

Dr. RVSV Prasad Ph.D

Professor & HOD, IT Dept



External Examiner

DECLARATION

I certify that

- a. The project work contained in the report is original and has been done by us under the guidance of my supervisor.
- b. The work has not been submitted to any other University for the award of my degree or diploma.
- c. The guidelines of the college are followed in writing the project report.

Khandavalli Munni (Regd No. 21A21A1222)
Valavala Devaki Anandi (Regd No. 21A21A1258)
Baskarla Nikhita (Regd No. 21A21A1263)

ACKNOWLEDGEMENT

I whole heartedly and sincerely thank my guide **Mr. M N V L Narayana Associate Professor** and **Dr. RVVSV Prasad** Professor & Head of the Department of Information Technology, Swarnandhra College of Engineering & Technology, Seetharampuram for their valuable suggestions and encouragement during the preparation and progress of my project.

I express my heartfelt thanks to **Dr. S. Suresh Kumar**, Principal and **Dr. A. Gopi Chand**, Vice Principal Swarnandhra College of Engineering & Technology for giving me this opportunity for the successful completion of my degree.

I express my honest thanks to **Management** of Swarnandhra College of Engineering & Technology for providing necessary arrangements for completing the project.

I express my earnest thanks to all the **teaching and non-teaching staff** of department of Information Technology for their valuable guidance and support given for the completion of my project.

Finally, it is pleasure to thank all my **family members and friends** for their constant encouragement and enormous support. Without them I could not go up with my work.

Khandavalli Munni (Regd No. 21A21A1222)
Valavala Devaki Anandi (Regd No. 21A21A1258)
Baskarla Nikhita (Regd No. 21A21A1263)

ABSTRACT

Facial recognition and emotion detection are prominent research areas with a wide range of applications. This project aims to detect and recognize human facial emotions—such as happiness, sadness, disgust, anger, fear, neutrality, or surprise—using the Haar Cascade Classifier for face detection and a Convolutional Neural Network (CNN) trained on the FER2013 dataset for emotion recognition. The system captures a user’s face at 30-second intervals, trains the recognizer based on user input, and subsequently detects the person and their emotions in real time.

The CNN model, trained on the FER2013 dataset (containing 35,887 labeled images of seven emotions), is utilized for emotion classification, ensuring high accuracy. The Haar Cascade Classifier extracts facial regions before feeding them into the CNN for feature learning and classification. This deep learning-based approach enables efficient and accurate emotion recognition compared to traditional handcrafted feature-based methods.

By leveraging a pre-trained CNN model, this project provides a practical and effective solution for real-time facial emotion recognition. The combination of Haar-based face detection and CNN-based emotion classification enhances robustness, making it suitable for various real-world applications.

Keywords:

Face Recognition, Emotion Detection, Haar Cascade Classifier, FER2013 Dataset, Feature Extraction, Real-Time Processing, Facial Emotion Recognition.

LIST OF FIGURES

S.No	FIGURE NO.	FIGURE NAME	PAGE NO.
1	3.1	System Architecture	18
2	7.1	CNN Model Diagram	45
3	7.2	USECASE Diagram	50
4	7.3	CLASS Diagram	52
5	7.4	SEQUENCE Diagram	54
6	9.1	Test Case-1	64
7	9.2	Test Case-2	64
8	9.3	Test Case-3	65
9	9.4	Test Case-4	65
10	9.5	Test Case-5	66
11	10.1	Outline-1	69
12	10.2	Outline-2	70
13	10.3	Outline-3	71
14	10.4	Outline-4	72

CONTENTS

S.No	DESCRIPTION	PAGE No.
Chapter-1	INTRODUCTION	1-5
1.1	General Introduction	2
1.2	Problem Definition	4
1.3	Objective of the Project	5
Chapter-2	LITERATURE SURVEY	6-10
2.1	Introduction	7
2.2	Sentiment and Semantics-Based Approaches	7
2.3	Traditional and Deep Learning Approaches for Facial Emotion Recognition	7
2.4	Psychological perspectives on Emotion Recognition	8
2.5	CNN-Based Approaches for FER	8
2.6	Hybrid and Novel CNN Architectures	9
2.7	Autoencoder and Optimization- Based Approaches	9
2.8	CNN Architectural Review	10
2.9	Conclusion	10
Chapter-3	SYSTEM ANALYSIS	11-24
3.1	Existing System	12
3.1.1	Disadvantages of Existing System	14
3.2	Proposed System	15
3.2.1	Advantages of Proposed System	16
3.3	System Architecture	18
3.3.1	Components of System Architecture	19
3.4	Haar Cascade Classifier (Machine Learning Approach)	22
3.5	Convolutional Neural Networks (Deep Learning)	23
Chapter-4	SYSTEM REQUIREMENTS	25-29
4.1	Software Requirements Specifications	26

4.2	Interface Requirements	27
4.3	Hardware Requirements	27
4.4	Software Requirements	28
4.5	System Workflow	29
Chapter-5	SOFTWARE DESCRIPTION	30-37
5.1	About Python Platform	31
5.1.1	Key Features of Python for this Project	33
5.1.2	Libraries and Frameworks	34
5.2	About Streamlit	35
Chapter-6	SYSTEM STUDY	38-41
6.1	System Study	39
6.1.1	Feasibility Study	39
6.1.2	Economic Feasibility	40
6.1.3	Technical Feasibility	40
6.1.4	Social Feasibility	41
Chapter-7	SYSTEM DESIGN	42-55
7.1	Module Description	43
7.1.1	Input Acquisition Module	43
7.1.2	Face Detection Module	43
7.1.3	Pre processing Module	44
7.1.4	Feature Extraction Module	44
7.1.5	Emotion Recognition Module	47
7.1.6	Real time Visualization Module	47
7.2	Input Design	47
7.3	Output Design	49
7.4	UML Diagrams for Design	50
7.4.1	USECASE Diagram	50
7.4.2	CLASS Diagram	51
7.4.3	SEQUENCE Diagram	53

Chapter-8	IMPLEMENTATION	56-59
8.1	Sample Code	57
Chapter-9	SYSTEM TESTING	60-67
9.1	Testing Techniques	61
9.2	Types of Testing involved in the Code	62
9.3	Test Cases	64
9.4	Test Results	66
Chapter-10	EXPERIMENTAL RESULTS	68-72
10.1	Output Screenshots	69
Chapter-11	Conclusion and Future Scope	73-75
11.1	Conclusion	74
11.2	Future work	75
	References	76-77

CHAPTER-1

INTRODUCTION

INTRODUCTION

1.1 General Introduction

Real-time facial recognition and emotion detection are cutting-edge AI technologies that analyze human faces for identification and emotional understanding. Facial recognition works by detecting a face in an image or video, extracting key facial features such as eyes, nose, and mouth, and then matching these features against a database to identify or verify individuals. This technology is widely used in security systems, smartphone authentication, personalized marketing, and law enforcement. Emotion detection, on the other hand, focuses on analyzing facial expressions to determine emotional states, such as happiness, sadness, anger, surprise, fear, and disgust. These systems rely on deep learning algorithms trained on large datasets of labelled facial expressions to accurately classify emotions in real time.

For real-time processing, facial recognition and emotion detection require efficient AI models, hardware acceleration using GPUs or TPUs, and optimized image processing algorithms. These technologies enable quick and seamless analysis of live video feeds, making them valuable for a variety of applications. In security and surveillance, facial recognition helps identify persons of interest, enhancing public safety and access control.

Despite the numerous advantages, challenges exist in implementing an efficient facial recognition and emotion detection system. Issues such as varying lighting conditions, occlusions, head poses, and diverse facial expressions pose challenges to achieving high accuracy. Addressing these challenges requires a robust feature extraction method capable of handling real-world scenarios. This project aims to mitigate these challenges by employing a combination of well-established techniques, including Haar-like features and pre-trained emotion recognition models.

Haar cascade Classifier :

The Haar cascade classifier serves as an indispensable preprocessing stage in your real-time facial recognition and emotion detection system, primarily functioning to efficiently localize faces within incoming video frames. Its role is to rapidly identify Regions of Interest (ROIs) containing faces, thereby enabling the subsequent deep learning models, specifically

Convolutional Neural Networks (CNNs), to focus their computational resources on relevant facial areas. This initial filtering process is crucial for achieving real-time performance, as Haar cascades are renowned for their speed and computational efficiency, effectively minimizing the processing load on the more computationally intensive CNNs. By acting as a first-pass detector, the Haar cascade simplifies the task for the deep learning component, allowing it to concentrate on extracting discriminative facial features for recognition and analyzing subtle expressions for emotion detection. This hybrid approach, combining the strengths of traditional machine learning for fast detection with deep learning for accurate analysis, ensures both speed and precision in your system, particularly important in resource-constrained environments. Essentially, the Haar cascade provides the essential foundation upon which the more complex deep learning processes can operate effectively.

Facial Emotion Recognition (FER)

Facial Emotion Recognition refers to the process of analyzing facial expressions to classify human emotions. It involves the detection of facial landmarks, extraction of relevant features, and classification of emotions into predefined categories such as happiness, sadness, anger, surprise, fear, and disgust.

Applications of Facial Recognition and Emotion Detection

Facial recognition and emotion detection systems have significant applications in security, education, entertainment, healthcare, and marketing:

1. **Security and Surveillance:** Facial recognition enhances security systems by enabling biometric authentication and preventing unauthorized access. Emotion detection can help identify suspicious behaviours in real-time surveillance systems.
2. **Education:** Teachers can use emotion recognition to assess student engagement and adjust their teaching methods accordingly. Online learning platforms can utilize this technology to provide personalized learning experiences.
3. **Healthcare:** Emotion detection assists in diagnosing mental health conditions such as depression, anxiety, and post-traumatic stress disorder (PTSD). It also helps monitor patients' emotional states during therapy sessions.

4. Marketing and Customer Experience: Companies use emotion detection to analyze customer reactions to advertisements, products, and services, enabling them to tailor their offerings for better engagement.
5. Entertainment and Gaming: Video games and interactive media can adapt to users' emotions, creating more immersive experiences. Personalized content recommendations based on emotional responses enhance user satisfaction.

1.2 Problem Definition

Facial recognition and emotion detection have become crucial areas in artificial intelligence, impacting domains such as human-computer interaction, mental health monitoring, and security. However, traditional methods often struggle with real-time processing, accuracy, and adaptability in dynamic environments.

This project addresses the challenge of real-time facial emotion detection by integrating Haar Cascade Classifier for face detection, MediaPipe for facial landmark tracking, and a Convolutional Neural Network (CNN) trained on the FER2013 dataset for emotion recognition. The system aims to identify seven human emotions—happiness, sadness, disgust, anger, fear, neutrality, and surprise—in real time. A key issue in facial emotion recognition is ensuring both accuracy and computational efficiency. Many models either require high-end GPUs for inference or fail to generalize well across different facial expressions due to variations in lighting, pose, and occlusions. To overcome these limitations, this system combines Haar-based face detection (a lightweight approach for real-time applications) with CNN-based emotion classification (ensuring deep feature extraction and accurate emotion recognition).

Additionally, a Streamlit-based interface provides a user-friendly and interactive platform, allowing real-time emotion detection through a webcam and enabling dataset exploration based on textual input for specific emotions. Thus, this project aims to develop an efficient, accurate, and real-time emotion detection system by leveraging deep learning, traditional computer vision techniques, and modern web-based deployment solutions.

1.3 Objective of the Project

The objective of this project is to develop an efficient and real-time facial emotion recognition system by integrating Haar Cascade Classifier, MediaPipe, and a Convolutional Neural Network trained on the FER2013 dataset. The system is designed to accurately detect and classify human emotions such as happiness, sadness, anger, fear, disgust, surprise, and neutrality in real-time video streams.

One of the main goals is to enhance the accuracy and efficiency of emotion detection by leveraging Haar Cascade Classifier for lightweight face detection, MediaPipe for precise facial landmark tracking, and CNN for deep feature extraction and classification. By combining these techniques, the system ensures fast and reliable emotion recognition while maintaining computational efficiency.

Another key objective is to enable real-time processing and user interaction through a Streamlit-based interface. This interface provides two primary functionalities. The first is live emotion detection, where the system captures real-time video input and continuously detects facial emotions. The second is dataset exploration, allowing users to retrieve and visualize images corresponding to specific emotions from the FER2013 dataset using text-based queries.

CHAPTER-2

LITERATURE SURVEY

2. LITERATURE SURVEY

2.1 Introduction

Facial Emotion Recognition (FER) is an evolving field of artificial intelligence that leverages deep learning techniques for human emotion analysis. Various approaches have been developed, utilizing convolutional neural networks (CNNs), hybrid models, and traditional machine learning techniques. The primary goal of FER is to enable machines to interpret human emotions accurately based on facial expressions. This literature review explores significant studies on FER, focusing on methodologies, datasets, and CNN architectures while addressing challenges and future directions in the field.

2.2 Sentiment and Semantics-Based Approaches

Gupta et al. (2017) [1] introduced a sentiment-and-semantics-based method for detecting emotions in textual conversations, highlighting the importance of integrating linguistic cues with facial recognition techniques. Their study emphasized the role of natural language processing (NLP) in enhancing emotion detection by analyzing textual data alongside visual inputs. This approach has potential applications in chatbots, virtual assistants, and customer sentiment analysis.

2.3 Traditional and Deep Learning Approaches for Facial Emotion Recognition

Ravi and Yadhukrishna (2020) [2] combined CNN and Local Binary Patterns (LBP) for facial expression recognition, demonstrating the effectiveness of handcrafted features with deep learning models. Traditional methods such as Principal Component Analysis (PCA) and Support Vector Machines (SVM) have been used historically for FER, but CNN-based approaches provide superior accuracy. Similarly, Jaiswal and Nandi (2020) [4] proposed a robust real-time FER system using CNN architectures, achieving high accuracy in real-world scenarios. The comparison between traditional and deep learning methods underscores the advantages of automated feature extraction in modern FER systems.

2.4 Psychological Perspectives on Emotion Recognition

Ekman and Friesen (1971) [3] studied the universality of facial expressions across different cultures, establishing the foundation for automated FER systems by categorizing emotions into six primary expressions: anger, happiness, sadness, fear, disgust, and surprise. Their research forms the basis for many FER datasets and models, as these universal emotions are commonly used in training deep learning models. Understanding the psychological aspects of emotion recognition aids in developing models that can generalize across different demographics and cultural backgrounds.

2.5 CNN-Based Approaches for FER

Minaee et al. (2021) [5] presented Deep-Emotion, an attentional convolutional network for FER, improving recognition accuracy using attention mechanisms. Attention mechanisms allow CNN models to focus on the most relevant facial regions, enhancing emotion classification performance. Hussain and Al Balushi (2020) [6] proposed a real-time deep learning-based FER model, demonstrating its application in real-world scenarios, including human-computer interaction and surveillance.

Zahara et al. (2020) [7] explored the use of the FER-2013 dataset in CNN-based emotion detection on Raspberry Pi, highlighting the feasibility of low-power devices for real-time FER. This study is significant for deploying FER models on edge devices. Wang et al. (2019) [8] combined CNN and Random Forest for enhanced classification performance, showing how hybrid models can improve accuracy and robustness.

Bhatti et al. (2021) [9] investigated the use of deep features and Extreme Learning Machines (ELM) for FER in educational environments, helping in analyzing student engagement and emotions. Alzubaidi et al. (2021) [10] reviewed deep learning architectures, challenges, and future directions in FER, emphasizing CNN advancements, including transfer learning and generative adversarial networks (GANs).

Yamashita et al. (2018) [11] and Khan et al. (2020) [12] provided comprehensive overviews of CNN architectures and their applications in image processing, including FER, discussing layer configurations and optimization techniques.

2.6 Hybrid and Novel CNN Architectures

Shi et al. (2021) [13] proposed a multi-branch cross-connection CNN for facial expression recognition, enhancing feature extraction through multiple processing paths. Oloyede et al. (2018) [14] integrated image enhancement techniques with CNN for improved recognition accuracy, demonstrating the role of preprocessing in deep learning models.

Kamencay et al. (2017) [15] introduced a new face recognition method using CNNs, demonstrating the adaptability of CNN models in different facial recognition applications. Sepas-Moghaddam et al. (2020) [16] employed deep attention-based bidirectional LSTMs for facial expression recognition using light field images, introducing temporal analysis in FER.

Lopes et al. (2015) [17] and Shin et al. (2016) [18] analyzed baseline CNN structures for FER, providing insights into the effectiveness of convolutional layers in feature extraction. Zhang et al. (2020) [19] developed RefineFace, a refinement neural network for high-performance face detection, addressing challenges in detecting faces in varying lighting conditions and angles.

Poulose et al. (2021) [20] explored foreground extraction-based FER using the Xception model, improving classification accuracy by reducing background noise. Wegrzyn et al. (2017) [21] mapped emotional facial features, identifying critical regions such as the eyes and mouth for effective emotion classification.

2.7 Autoencoder and Optimization-Based Approaches

Dachapally (2017) [22] applied representational autoencoder units to CNNs for FER, enhancing feature learning by reducing dimensionality and capturing essential emotion-related features. Singh et al. (2020) [23] utilized Action Units (AUs) for facial emotion detection, improving robustness to variations in expressions by focusing on localized facial muscle movements.

Alenazy et al. (2021) [24] introduced an optimized deep learning model using the Gravitational Search Algorithm, demonstrating improvements in facial expression recognition performance by fine-tuning model parameters. Musa (2020) [25] discussed the application of FER in educational environments, including virtual classrooms and student engagement analysis.

2.8 CNN Architectural Review

Stewart (2019) [26] provided an introductory overview of CNNs, explaining their fundamental components such as convolutional layers, pooling layers, and fully connected layers. The study highlighted how CNNs are structured to learn hierarchical features, making them well-suited for FER tasks. This review also explored different activation functions, loss functions, and optimization techniques used in CNNs for facial recognition.

2.9 Conclusion

The reviewed studies highlight the rapid advancements in deep learning for FER, with CNNs playing a crucial role in feature extraction and classification. The evolution of FER models has led to real-time applications in areas such as healthcare, security, and human-computer interaction. Future research should focus on improving real-time recognition, integrating multimodal approaches such as voice and text-based sentiment analysis, and optimizing computational efficiency for deployment in real-world applications. Addressing challenges such as occlusions, variations in lighting, and cross-cultural differences will further enhance the accuracy and robustness of FER systems.

CHAPTER-3

SYSTEM ANALYSIS

3. SYSTEM ANALYSIS

3.1 Existing System

The existing system is designed for facial recognition and emotion detection using a combination of the haar cascade classifier for face detection and a convolutional neural network for emotion classification. The system's primary objective is to classify human emotions into seven categories: happy, sad, disgust, anger, fear, neutral, and surprise. To achieve this, the system relies on the FER2013 dataset, which consists of 35,887 grayscale images labeled according to these seven emotional states. The dataset is divided into training, validation, and testing sets to ensure the model learns effectively and is evaluated for accuracy and generalization. The system is built to process static images and does not perform real-time live emotion detection, meaning it does not continuously analyze video streams but instead processes individual images to recognize facial emotions.

The system begins with face detection, which is handled using the haar cascade classifier. The haar cascade classifier is a feature-based machine learning technique that scans an image to identify facial structures by detecting variations in pixel intensity. It works by applying multiple stages of classifiers that are trained on both positive and negative samples of faces. This method is computationally efficient and allows for quick detection of faces in images. However, while haar cascade is lightweight and fast, it has limitations in detecting faces under extreme lighting conditions, occlusions, and different angles. Once a face is detected, it is extracted as a region of interest (ROI) and undergoes preprocessing steps to prepare it for emotion classification. These preprocessing steps include converting the image to grayscale, resizing it to 48×48 pixels to match the CNN input requirements, and normalizing pixel values to improve model performance. After preprocessing, the extracted facial region is passed through a convolutional neural network, which is responsible for emotion classification. The CNN used in this system consists of multiple layers, including convolutional layers, max-pooling layers, and fully connected layers. The convolutional layers apply filters to extract essential facial features such as edges, contours, and textures. The max-pooling layers reduce the spatial dimensions of the feature maps, helping the model focus on the most important facial details while reducing computational complexity. The fully connected layers then process the extracted features and

classify them into one of the seven emotions using the softmax activation function. The softmax function outputs probability values for each emotion category, and the one with the highest probability is chosen as the final classification. To train the CNN, the system utilizes the Adam optimizer, a popular optimization algorithm known for its adaptive learning rate capabilities. The learning rate is set at 0.001 to ensure stable and efficient weight updates during training. The training process involves feeding labeled images from the FER2013 dataset into the model, allowing it to learn facial patterns associated with each emotion. The model is evaluated using classification metrics such as precision, recall, F1-score, and confusion matrices to assess its accuracy and performance. One of the key limitations of the system is its inability to perform real-time live detection. Since it only processes static images, it cannot analyze emotions dynamically in a continuous video feed. This limits its practical application in areas that require real-time feedback, such as human-computer interaction, surveillance, and behavioral analysis. Another challenge is the difficulty in distinguishing similar emotions, particularly between fear and surprise, which often share overlapping facial expressions.

To address these limitations, the paper suggests several improvements. One proposed enhancement is to incorporate more advanced deep learning techniques, such as attention-based models or transfer learning, to refine emotion detection accuracy. Optimizing the CNN architecture by experimenting with different numbers of layers, activation functions, and hyperparameter tuning could also lead to better performance. Additionally, increasing the diversity of the training dataset by including more varied facial expressions from different ethnicities, age groups, and real-world conditions could help the model generalize better across different populations. Implementing real-time live detection by integrating the model with webcam feeds and processing video frames in real-time would significantly improve the system's applicability in practical scenarios.

In summary, the existing system effectively detects faces using the haar cascade classifier and classifies emotions using a CNN model trained on the FER2013 dataset. However, its reliance on static images rather than real-time video processing limits its usability in real-world applications that require continuous monitoring. While it achieves moderate accuracy in emotion recognition, challenges such as distinguishing similar emotions, handling occlusions, and adapting to different lighting conditions still need to be addressed.

3.1.1 DISADVANTAGES OF EXISTING SOLUTION

Lack of Real-Time Live Detection

The existing system processes only static images and does not support real-time live emotion detection. This means it cannot continuously analyze video streams to detect emotions dynamically. As a result, its usability is limited in applications that require instant feedback, such as human-computer interaction, behavioural analysis, and surveillance. Without real-time capabilities, the system cannot track changes in emotions over time, making it less effective for real-world scenarios that demand continuous monitoring.

Limitations of Haar Cascade for Face Detection

The system relies on the haar cascade classifier for face detection, which, while computationally efficient, has limitations in terms of accuracy. Haar cascade struggles to detect faces under poor lighting conditions, extreme facial angles, and partial occlusions caused by accessories like glasses or masks. These limitations can result in missed detections or incorrect facial region extraction, leading to lower accuracy in emotion recognition. A more robust face detection method, such as deep learning-based approaches, would improve the reliability of the system.

Difficulty in Distinguishing Similar Emotions

The CNN model used for emotion classification faces challenges in differentiating similar emotions, particularly between fear and surprise. These emotions share overlapping facial features, making them difficult to classify accurately. This issue reduces the overall effectiveness of the system, especially in applications where precise emotion recognition is crucial, such as mental health analysis or sentiment detection in customer interactions.

Lack of Generalization Across Different Facial Variations

The accuracy of the system is affected by variations in facial structures, age, gender, and ethnicity. The FER2013 dataset, on which the CNN model is trained, may not include a sufficiently diverse range of expressions, leading to biases and inconsistent predictions when

applied to real-world scenarios. This lack of generalization makes the model less reliable when used on individuals with facial expressions that differ significantly from those in the training dataset.

Moderate Accuracy in Emotion Recognition

The system achieves a validation accuracy of 65.59 percent, which indicates that misclassifications occur frequently. While the model performs reasonably well, it struggles with subtle emotional variations and images that do not closely match the training data. This reduces its reliability, particularly in critical applications such as security monitoring or psychological

3.2 PROPOSED SYSTEM

The proposed system is designed to perform real-time facial emotion recognition by integrating a combination of traditional machine learning and deep learning techniques. The system aims to accurately detect and classify human emotions, including happiness, sadness, anger, fear, disgust, surprise, and neutrality, using a hybrid approach. The primary objective is to improve efficiency and accuracy in emotion detection while ensuring real-time performance through optimized processing techniques.

Face detection is handled using the haar cascade classifier, which is a widely used and computationally efficient technique for detecting faces in images. However, to enhance accuracy and handle variations in lighting conditions, facial angles, and occlusions, the system also incorporates MediaPipe for facial landmark tracking. This hybrid approach ensures that the system can accurately detect facial regions even in challenging scenarios. Once a face is detected, the region of interest is extracted, preprocessed, and fed into a convolutional neural network trained on the FER2013 dataset. The preprocessing steps include grayscale conversion, resizing, and normalization, ensuring consistency in input data before classification.

The convolutional neural network is responsible for extracting meaningful facial features and classifying the detected face into one of the seven emotions. The model is trained on the FER2013 dataset, which contains thousands of labeled images representing different emotional expressions. By leveraging deep learning, the system ensures high accuracy in recognizing subtle

variations in facial expressions. The classification output is processed and displayed in real time, allowing users to see their detected emotions dynamically.

The proposed system also includes a Streamlit-based graphical user interface that provides an interactive platform for users to engage with the application. The interface enables real-time webcam-based emotion detection, where facial expressions are continuously analyzed and displayed. Additionally, the system allows users to explore the dataset by entering text-based queries to retrieve images corresponding to specific emotions. This feature helps in understanding the dataset and analyzing different emotional expressions more effectively.

To optimize performance, the system processes frames at predefined intervals, reducing computational overhead while maintaining real-time responsiveness. GPU acceleration is utilized to speed up model inference, ensuring that emotion classification is performed efficiently. Techniques such as data augmentation and model tuning are applied to improve generalization and reduce misclassification errors. The integration of haar cascade for face detection, MediaPipe for landmark tracking, and CNN for deep learning-based classification ensures a robust and scalable solution for facial emotion recognition.

Overall, the proposed system aims to bridge the gap between deep learning-based emotion recognition and practical real-time applications. By addressing the limitations of existing systems and incorporating advanced techniques, the system provides a more accurate, efficient, and user-friendly approach to facial emotion detection, making it suitable for applications in mental health monitoring, human-computer interaction, and behavioral analysis.

3.2.1 ADVANTAGES OF PROPOSED SYSTEM

Real-Time Live Emotion Detection

The proposed system supports real-time live detection, allowing continuous analysis of facial expressions from a webcam feed. Unlike existing systems that process only static images, this system can track changes in emotions dynamically, making it suitable for applications such as human-computer interaction, behavioral analysis, and mental health monitoring. Real-time processing ensures instant feedback, which is essential for practical deployments in various fields.

Hybrid Face Detection Approach for Improved Accuracy

The integration of the haar cascade classifier and MediaPipe for face detection enhances accuracy and robustness. While the haar cascade classifier provides a computationally efficient method for detecting faces, MediaPipe improves precision by detecting facial landmarks and adjusting for variations in lighting, head angles, and occlusions. This combination allows the system to work effectively in real-world conditions, ensuring that faces are correctly detected before emotion classification.

Deep Learning-Based Emotion Recognition

The convolutional neural network trained on the FER2013 dataset enables highly accurate emotion classification. Unlike traditional machine learning models that rely on handcrafted features, the CNN automatically learns relevant facial features, improving recognition accuracy. The model is capable of detecting subtle variations in facial expressions, leading to more reliable emotion classification across different individuals and environments.

Interactive and User-Friendly Interface

The system includes a Streamlit-based graphical user interface that provides an intuitive and interactive experience. Users can view real-time emotion detection results and explore the dataset using text-based queries. The interface ensures accessibility and ease of use, allowing individuals without technical expertise to interact with the system effortlessly. This makes it practical for deployment in various applications, including education, healthcare, and customer service.

Optimized Performance and Computational Efficiency

To ensure real-time processing, the system optimizes computational efficiency by processing frames at predefined intervals. This reduces unnecessary computations while maintaining responsiveness. Additionally, GPU acceleration is used to speed up model inference, allowing for smooth execution on modern computing devices. The use of optimized deep

learning techniques further enhances the balance between accuracy and efficiency, making the system scalable for different hardware configurations.

Robustness Against Environmental Variations

The system is designed to handle variations in lighting, facial angles, and occlusions more effectively than existing solutions. By incorporating MediaPipe for advanced facial landmark detection, the system can adjust to different environmental conditions and improve detection accuracy. This ensures that emotions are recognized correctly even when facial expressions are partially obscured or captured under challenging lighting conditions.

Scalability and Practical Applications

The system is highly scalable and can be deployed in multiple domains, including mental health monitoring, security surveillance, and sentiment analysis. Its ability to analyze human emotions in real time makes it useful for applications such as stress detection in workplaces, customer feedback analysis in businesses, and emotional support in therapy sessions. The flexibility of the system allows it to be integrated into different platforms, enhancing its real-world applicability.

3.3 SYSTEM ARCHITECTURE

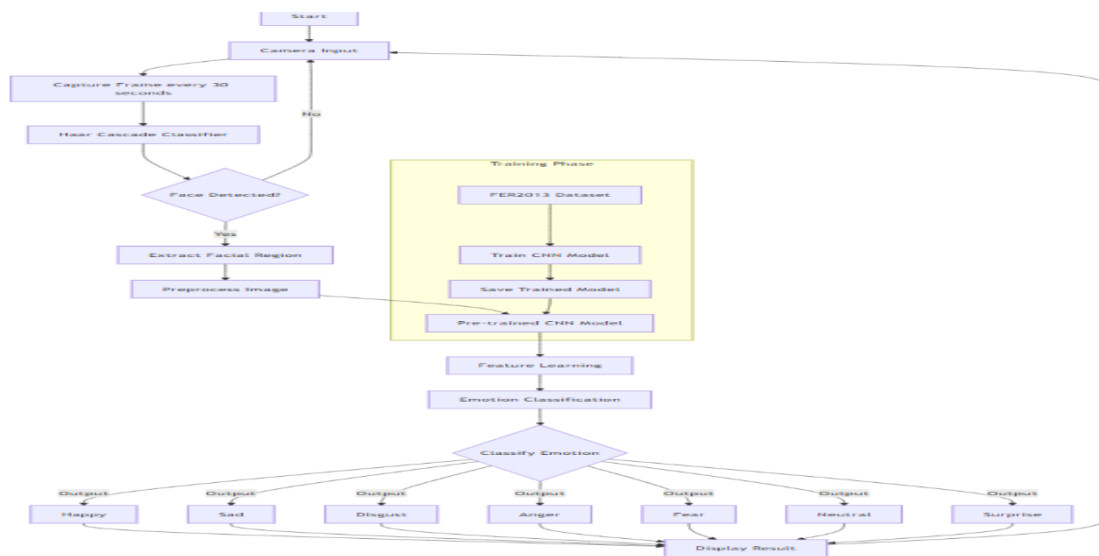


Fig-3.1 System Architecture

The proposed system follows a structured architecture that enables real-time facial emotion recognition by integrating Haar Cascade for face detection, MediaPipe for facial landmark tracking, and a CNN model trained on the FER2013 dataset for emotion classification. The system also includes a Streamlit-based graphical user interface (GUI) for real-time interaction and dataset exploration.

3.3.1 Components of the System Architecture

1. Input Acquisition

The first step in the system architecture involves capturing live video input using a webcam. The webcam continuously records video frames, providing a real-time feed for further processing. Since processing every frame in real time can be computationally expensive, the system processes frames at predefined intervals to maintain a balance between accuracy and efficiency. By selecting key frames at regular intervals, the system minimizes redundant computations while still ensuring smooth detection and recognition of facial expressions. The captured frames act as the primary input to the face detection and emotion recognition pipeline, enabling the system to analyze facial expressions dynamically. The real-time nature of this input acquisition process is essential for applications requiring immediate feedback, such as human-computer interaction, psychological analysis, and real-time emotion-based applications.

2. Face Detection

Once the video frames are captured, the system processes them to detect human faces using a combination of Haar Cascade Classifier and MediaPipe Face Detection. The Haar Cascade Classifier is a machine learning-based object detection algorithm that uses pre-trained Haar-like features to identify facial regions quickly. It operates by scanning the input frame in a hierarchical manner, identifying areas that match the predefined patterns associated with human faces. Although Haar Cascade is lightweight and efficient, it has limitations in handling variations in lighting, occlusions, and different facial angles. To overcome these challenges, MediaPipe Face Detection is integrated into the system to enhance accuracy. MediaPipe utilizes deep learning-based facial landmark tracking, which refines face detection by accounting for slight variations in pose, lighting, and partial occlusions. By combining Haar Cascade with

MediaPipe, the system achieves a higher degree of robustness and precision, ensuring that faces are reliably detected before proceeding to the next stage of the pipeline.

3. Preprocessing

After detecting a face in the captured frame, the system performs multiple preprocessing steps to prepare the image for emotion recognition. The first step in preprocessing is cropping the detected facial region from the image to eliminate unnecessary background details that might interfere with emotion classification. Once the face is extracted, it is converted into grayscale to reduce the computational complexity of the input data. Grayscale conversion removes color information, allowing the model to focus solely on structural and textural features associated with different emotions. The image is then resized to a fixed dimension of 48×48 pixels, which matches the input size required by the CNN model. Standardizing the input dimensions ensures consistency and allows the model to process images efficiently. Furthermore, pixel normalization is applied to scale the image's pixel values to a specific range (typically between 0 and 1) to improve the model's convergence during inference. These preprocessing techniques enhance the overall performance and accuracy of the emotion classification model.

4. Emotion Recognition Using CNN

Once the preprocessed image is ready, it is fed into the Convolutional Neural Network (CNN) model for emotion recognition. The CNN model, which has been trained on the FER2013 dataset, is designed to automatically extract meaningful features from the image and classify the detected face into one of seven emotions: happy, sad, angry, fearful, disgusted, surprised, or neutral. The CNN consists of multiple layers, each playing a distinct role in feature extraction and classification. The convolutional layers detect key patterns such as edges, textures, and facial contours, which help differentiate emotions. The pooling layers reduce the dimensionality of extracted features, preserving essential information while reducing computational complexity. Fully connected layers further process the features and assign a probability score to each emotion. The final softmax classification layer determines which emotion has the highest probability and outputs it as the detected emotion. The use of CNN ensures that the system can

recognize complex facial expressions with high accuracy, even in varying environmental conditions.

5. Real-Time Visualization with Streamlit

To provide an interactive and user-friendly interface, the system incorporates Streamlit, a Python-based framework for building web applications. The Streamlit-based GUI displays the detected face along with the predicted emotion in real time, allowing users to observe their emotional states dynamically. The interface also includes additional functionalities, such as the ability to explore dataset images based on text input. Users can type an emotion name, and the system retrieves and displays corresponding images from the FER2013 dataset, helping them understand how different emotions are visually represented. Additionally, Streamlit allows users to interact with the system by toggling real-time detection on or off, adjusting frame capture intervals, and visualizing the confidence scores of detected emotions. The lightweight and responsive design of the Streamlit interface ensures that users receive an engaging and seamless experience while using the system.

6. Performance Optimization

Since real-time facial emotion recognition requires significant computational resources, multiple performance optimization techniques are implemented to enhance efficiency. The system utilizes GPU acceleration to speed up CNN model inference, reducing processing time and allowing for real-time predictions. Frame-skipping techniques are applied to avoid unnecessary computations; instead of analyzing every frame, the system selectively processes frames at optimal intervals, maintaining accurate detection while reducing workload. Additionally, the system is designed to handle real-world challenges such as lighting variations, facial occlusions, and different facial orientations. Techniques such as histogram equalization can be applied to improve contrast in low-light conditions, while facial alignment strategies help normalize facial positions for better classification accuracy. These optimizations ensure that the system performs efficiently across different environments, making it suitable for practical real-time applications.

3.4 HAAR CASCADE CLASSIFIER (MACHINE LEARNING APPROACH)

Haar Cascade is a machine learning-based object detection algorithm widely used for identifying objects such as human faces in digital images and videos. It is based on the Viola-Jones framework, which introduced an efficient way to detect objects using Haar-like features. These features represent edge and texture patterns by calculating the difference in pixel intensities between adjacent rectangular regions of an image. By applying these features in combination with a cascade of classifiers, the algorithm is able to quickly determine whether a particular region of an image contains a face. The Haar Cascade classifier is trained using a large dataset of positive (faces) and negative (non-face) samples, and the AdaBoost algorithm is used to select the most important features, combining weak classifiers into a strong classifier for robust detection. This process significantly enhances the speed and efficiency of face detection, making it highly suitable for real-time applications.

In this project, Haar Cascade plays a crucial role in the face detection process before emotion recognition is performed. When a live video frame is captured, Haar Cascade is applied to scan the image and detect facial regions. The algorithm works by sliding a detection window over the image at multiple scales, checking for patterns that match those learned during training. If a face is detected, the region containing the face is extracted and passed to the preprocessing stage. The preprocessing step ensures that the extracted face is converted to grayscale, resized to 48×48 pixels, and normalized to standardize pixel values. This prepares the image for input into the convolutional neural network (CNN) used for emotion classification.

The effectiveness of Haar Cascade in real-time face detection is critical for this project, as it allows the system to quickly and accurately locate faces in dynamic video streams. Unlike deep learning-based face detection models, Haar Cascade operates with lower computational requirements, making it highly efficient for real-time applications on standard hardware. However, to address certain limitations, such as sensitivity to variations in lighting, occlusions, and head poses, MediaPipe Face Detection is also integrated to enhance detection accuracy. By combining Haar Cascade with additional techniques, the system ensures robust and efficient face detection, leading to improved performance in the emotion recognition process.

In the overall workflow of this project, Haar Cascade acts as the initial stage, ensuring that only the relevant facial regions are processed by the CNN model. This not only reduces unnecessary computations but also improves the accuracy of emotion detection by eliminating background noise and irrelevant objects. Its ability to quickly detect faces in real-time video streams makes it an essential component of the system, enabling seamless live emotion recognition. By leveraging the advantages of Haar Cascade, the project achieves a balance between computational efficiency and detection accuracy, making it a practical solution for real-world applications.

3.5 CONVOLUTIONAL NEURAL NETWORKS – (DEEP LEARNING)

A Convolutional Neural Network is a deep learning model specifically designed for image processing and pattern recognition tasks. It mimics the way the human brain processes visual information by automatically learning spatial hierarchies of features through convolutional operations. CNNs consist of multiple layers, including convolutional layers for feature extraction, pooling layers for dimensionality reduction, and fully connected layers for classification. The convolutional layers apply filters to detect patterns such as edges, textures, and facial structures, gradually learning more complex representations as the network goes deeper. Pooling layers help reduce computational complexity while retaining essential information, and the final fully connected layers determine the output category, in this case, one of the seven facial emotions.

In this project, the CNN plays a fundamental role in emotion recognition by classifying facial expressions into predefined categories such as happiness, sadness, anger, fear, disgust, surprise, and neutrality. The model is trained on the FER2013 dataset, which contains a large collection of labeled images covering various emotional expressions. Once a face is detected using the Haar Cascade classifier, it undergoes preprocessing steps such as grayscale conversion, resizing, and normalization before being fed into the CNN. The CNN then extracts important facial features and processes them through multiple layers to classify the detected emotion with high accuracy. The final output provides a probability distribution over all possible emotions, selecting the one with the highest confidence score.

CNN is crucial for this project as it eliminates the need for manually extracting facial features, which is often required in traditional machine learning approaches. By leveraging deep learning capabilities, the CNN automatically learns patterns from data, improving recognition accuracy even under varying conditions such as different lighting, head positions, and facial occlusions. The use of CNN ensures that the system is robust and can generalize well to unseen faces. Moreover, GPU acceleration enhances the performance of CNN-based emotion detection, allowing real-time processing without significant delays.

The CNN-based emotion recognition model is integrated with a Streamlit-based graphical user interface, where the detected face and the classified emotion are displayed in real time. Users can interact with the system, and the results update dynamically as new frames are processed. The ability of CNN to accurately classify emotions from facial expressions makes it the core component of this project, enabling practical applications such as sentiment analysis, mental health monitoring, and interactive human-computer interfaces. Its deep learning capabilities, combined with Haar Cascade face detection, result in a highly efficient and effective real-time emotion recognition system.

CHAPTER- 4

SYSTEM REQUIREMENTS

4. SYSTEM REQUIREMENTS

The Software Requirements Specification for the real-time facial recognition and emotion detection system outlines the necessary functionalities, performance expectations, and technical requirements to ensure the smooth development and deployment of the project.

4.1 SOFTWARE REQUIREMENT SPECIFICATIONS (SRS)

The system is designed to detect human faces in real time, classify facial expressions into different emotions using a convolutional neural network, and display the results through an interactive graphical user interface. The requirements are categorized into functional software dependencies that support the system's real-time processing capabilities.

- The system must capture video frames from a webcam at regular intervals.
- It should perform real-time face detection using the Haar Cascade classifier.
- MediaPipe must be integrated to improve facial landmark tracking and handle variations in facial angles and occlusions.
- The detected face should be preprocessed through grayscale conversion, resizing, and normalization before classification.
- A trained convolutional neural network must classify emotions into seven categories: happiness, sadness, anger, fear, disgust, surprise, and neutrality.
- The system should provide a graphical interface using Streamlit to display real-time detection results.
- Users should be able to search for specific emotions within the dataset and retrieve corresponding images.
- The system should handle multiple faces in a frame and provide accurate predictions for each detected face.

The defined software requirements ensure that the system is built with high efficiency, accuracy, and real-time processing capabilities. By integrating deep learning models with optimized preprocessing and user-friendly interfaces, the system is designed to provide accurate facial emotion recognition while maintaining scalability and performance across different hardware configurations. These specifications guide the project's implementation, ensuring that all functional and non-functional goals are met successfully.

4.2 INTERFACE REQUIREMENTS

The interface requirements define how users interact with the real-time facial recognition and emotion detection system. Since the system relies on real-time processing and deep learning models, it is essential to have a well-structured, user-friendly interface that allows smooth interaction with its features. The graphical user interface should provide real-time feedback, display detected emotions clearly, and allow users to explore dataset images. The interface should be designed to be intuitive, requiring minimal technical knowledge to operate while maintaining responsiveness and efficiency.

- The system should provide a real-time video feed displaying the user's face along with the detected emotion, ensuring smooth and responsive updates.
- The detected emotion label and confidence score should be clearly visible, helping users validate the accuracy of predictions based on their facial expressions.
- The interface should include interactive controls to start, stop, and reset the real-time detection process, allowing users to manage the system's functionality easily.
- Users should be able to search and retrieve dataset images based on emotion labels by entering an emotion name, such as "happy" or "sad," and viewing corresponding images from the FER2013 dataset.
- The GUI should display detection logs and history, including timestamps and confidence scores, allowing users to track their detected emotions over time.
- The system should show performance metrics such as FPS and frame processing time, giving users feedback on the efficiency and responsiveness of the model.
- The application should provide visual cues, such as highlighting the detected face and overlaying the predicted emotion on the video feed, making it easier for users to interpret results.
- The system should support error handling mechanisms by displaying messages in case of camera failures, undetected faces, or missing model files, ensuring smooth user experience.

4.3 HARDWARE REQUIREMENTS

The hardware requirements for the real-time facial recognition and emotion detection system are crucial for ensuring smooth processing, real-time performance, and accurate classification.

Since the system relies on deep learning models for emotion recognition, it requires sufficient computational power to handle image processing and model inference efficiently. The primary hardware components include a capable processor, sufficient RAM, a dedicated GPU for accelerated computations, and an HD webcam for real-time video capture.

- processor: intel core i5 (10th gen or later) / amdryzen 5 or higher, recommended intel core i7 or amdryzen 7 for better performance
- ram: minimum 8gb, recommended 16gb for smooth execution of deep learning models
- gpu: nvidiagtx 1050 ti or higher, recommended nvidiartx 2060 or better for fast model inference and processing
- storage: minimum 50gb free disk space, recommended ssd for faster data loading and model execution
- webcam: hd webcam (built-in or external) for real-time facial detection and analysis
- operating system: windows 10/11, ubuntu 20.04+ or macos, linux preferred for deep learning optimizations

4.4 SOFTWARE REQUIREMENTS

The software requirements define the necessary frameworks, libraries, and tools required to develop and execute the system. The software stack includes programming languages, deep learning frameworks, image processing libraries, and user interface development tools. The system is primarily developed in python, which provides extensive support for machine learning and deep learning. OpenCV is used for real-time image processing and face detection, while tensor flow and keras handle deep learning-based emotion recognition.

- programming language: python 3.8 or later
- deep learning frameworks: tensorflow and keras for building and training the cnn model, pytorch (optional) for experimenting with model improvements
- computer vision libraries: opencv for real-time video capture, image processing, and face detection using haar cascade, mediapipe for facial landmark detection and tracking
- frontend and deployment framework: streamlit for developing a lightweight, web-based graphical user interface

- supporting libraries: numpy for numerical computations and matrix operations, pandas for handling datasets and data analysis, matplotlib and seaborn for visualizing model training performance and dataset distributions, scikit-learn for evaluating classification metrics such as accuracy, precision, recall, and confusion matrices
- development environment: jupyter notebook, vs code, or pycharm for writing, debugging, and testing the code
- dataset: fer2013 (facial expression recognition 2013) dataset for model training and validation

4.5 SYSTEM WORKFLOW

- The system starts by capturing live video input from a webcam at predefined intervals.
- Haar Cascade detects the face, and MediaPipe refines face detection using facial landmarks.
- The detected face is cropped, converted to grayscale, resized to 48×48 pixels, and normalized.
- The preprocessed face is fed into a CNN model trained on the FER2013 dataset for emotion classification.
- The CNN extracts facial features, processes them through multiple layers, and classifies the detected face into one of seven emotions.
- The classified emotion is displayed on the Streamlit-based user interface along with the confidence score.
- The system continuously processes new frames, updating the detected emotions in real time.
- Users can interact with the system by exploring dataset images and viewing real-time analysis results.

CHAPTER-5

SOFTWARE DESCRIPTION

5. SOFTWARE DESCRIPTION

5.1 ABOUT PYTHON PLATFORM

Python is a widely adopted programming language known for its simplicity, versatility, and extensive support for machine learning, deep learning, and computer vision applications. It provides a powerful ecosystem of libraries and frameworks, making it an ideal choice for developing artificial intelligence-based projects, including real-time facial recognition and emotion detection systems. Python's high-level syntax enables developers to write clean, readable, and concise code, reducing development time while ensuring efficiency. Its ability to handle large datasets, perform numerical computations, and integrate with deep learning frameworks makes it an essential tool for advanced artificial intelligence applications. Additionally, Python is an open-source language with strong community support, ensuring continuous improvements, updates, and the availability of various resources for developers.

For this project, Python is used as the primary development platform because of its seamless integration with deep learning and computer vision libraries. It supports multiple frameworks such as TensorFlow and Keras, which are essential for building, training, and deploying convolutional neural networks (CNNs) for emotion classification. The CNN model used in this system is trained on the FER2013 dataset and implemented using TensorFlow, which provides highly efficient deep learning operations. Python also integrates well with OpenCV, a leading library for image processing and real-time computer vision tasks. OpenCV is utilized for face detection using the Haar Cascade classifier, a lightweight and effective machine learning-based detection method. To further enhance accuracy, MediaPipe is incorporated for facial landmark detection, helping to refine face localization and improve emotion recognition.

Python's extensive support for scientific computing plays a crucial role in this project. Libraries such as NumPy and Pandas facilitate numerical computations and efficient dataset management, ensuring that facial images are processed correctly before being fed into the CNN model. Matplotlib and Seaborn are used for data visualization, allowing developers to analyze training accuracy, loss curves, confusion matrices, and other performance metrics of the emotion recognition model. Additionally, Python provides multi-threading and parallel processing capabilities, allowing real-time video frame analysis without significant delays. The integration

of GPU acceleration through CUDA further optimizes performance, enabling the deep learning model to run efficiently on compatible hardware.

One of the key advantages of using Python for this project is its ability to support real-time applications through frameworks like Streamlit. Streamlit enables the development of an interactive, web-based graphical user interface (GUI) without requiring extensive front-end programming. With Python's built-in support for multi-threading and asynchronous processing, Streamlit ensures that real-time face detection and emotion classification results are dynamically updated without manual page refreshes. The GUI provides users with a seamless experience, allowing them to view detected emotions instantly, interact with the system, and explore dataset images based on specific emotion labels.

Another important aspect of Python in this project is its cross-platform compatibility. Python applications can run on Windows, Linux, and macOS without requiring significant modifications. This flexibility ensures that the facial recognition and emotion detection system can be deployed across different operating systems, making it accessible for various research and commercial applications. Additionally, Python allows for cloud-based deployment on platforms like AWS, Google Cloud, and Microsoft Azure, enabling scalability for large-scale implementations.

Python also simplifies the implementation of error handling, logging, and debugging, which are essential for maintaining the stability and reliability of the system. The logging module in Python ensures that system performance and errors are tracked efficiently, allowing developers to identify and fix issues promptly. The availability of frameworks like pytest enables unit testing, ensuring that individual components such as face detection, preprocessing, and emotion classification work as expected. These features contribute to the robustness of the system, making Python an excellent choice for developing real-time artificial intelligence applications.

Overall, Python's vast ecosystem, ease of integration with deep learning and computer vision frameworks, and ability to support real-time processing make it the perfect platform for implementing this real-time facial emotion recognition system. Its ability to handle numerical

computations, interface with hardware accelerators, and provide interactive user interfaces ensures that the system is both efficient and user-friendly. Python's open-source nature and extensive community support also provide opportunities for continuous improvement, making it a scalable and adaptable choice for future advancements in facial recognition and emotion detection technologies.

5.1.1 KEY FEATURES OF PYTHON FOR THIS PROJECT

- **Easy Integration with Deep Learning Frameworks**

Python seamlessly integrates with deep learning frameworks like TensorFlow and Keras, enabling efficient training and deployment of the convolutional neural network used for emotion classification. The CNN model, trained on the FER2013 dataset, leverages Python's deep learning libraries to extract and classify facial features accurately.

- **Robust Computer Vision Capabilities**

Python provides extensive support for computer vision tasks through OpenCV, which is used for real-time face detection using the Haar Cascade classifier. Additionally, MediaPipe is integrated for facial landmark tracking, improving accuracy in detecting facial regions and refining emotion recognition.

- **Real-Time Processing and Optimization**

Python supports multi-threading and GPU acceleration, ensuring that the system processes video frames efficiently in real-time. CUDA support for NVIDIA GPUs enables faster inference of deep learning models, reducing delays in emotion detection and making the system responsive.

- **User-Friendly Web Interface with Streamlit**

The system uses Streamlit to create an interactive and lightweight web-based graphical user interface. Python's simplicity allows for dynamic real-time updates, enabling users to view their detected emotions instantly, interact with the system, and explore dataset images based on specific emotions.

- **Extensive Scientific Computing Libraries**

Python includes powerful libraries such as NumPy and Pandas for numerical computations, dataset handling, and preprocessing. These libraries ensure efficient image normalization,

pixel scaling, and structured data management, improving the model's accuracy and performance.

- **Cross-Platform Compatibility**

Python is compatible with multiple operating systems, including Windows, Linux, and macOS, allowing the system to be deployed on various platforms without major modifications. This flexibility ensures wider accessibility and usability.

These key features make Python an essential programming language for implementing this real-time facial recognition and emotion detection system, ensuring efficiency, accuracy, and user-friendly interaction.

5.1.2 LIBRARIES AND FRAMEWORKS

The system utilizes various libraries and frameworks to enable real-time face detection, emotion classification, image processing, and interactive user interface development. These tools ensure efficient execution, high accuracy, and a smooth user experience.

Deep Learning and Machine Learning Frameworks

- **TensorFlow and Keras** – Used for building, training, and deploying the convolutional neural network (CNN) model for emotion classification. Keras provides a high-level API for easier model development.
- **PyTorch (optional)** – Can be used as an alternative deep learning framework for training and evaluating CNN models.

Computer Vision and Image Processing Libraries

- **OpenCV** – Handles real-time video capture, face detection using the Haar Cascade classifier, and image preprocessing.
- **MediaPipe** – Provides advanced facial landmark detection, improving the accuracy of face localization before emotion classification.

Graphical User Interface and Deployment

- **Streamlit** – Enables the creation of a user-friendly web-based interface for real-time emotion detection and dataset exploration. It allows dynamic updates and interaction without requiring extensive front-end development.

Scientific Computing and Data Handling Libraries

- **NumPy** – Used for numerical computations, matrix operations, and handling image data efficiently.
- **Pandas** – Helps in dataset manipulation, preprocessing, and analysis of structured data.

Development and Debugging Tools

- **Jupyter Notebook, VS Code, PyCharm** – Used as integrated development environments (IDEs) for writing, debugging, and testing code.
- **Logging Module** – Helps track system performance, error handling, and debugging logs during real-time execution.

These libraries and frameworks ensure that the system performs real-time face detection, preprocesses images effectively, classifies emotions accurately, and provides an interactive user experience while maintaining high computational efficiency.

5.2 ABOUT STREAMLIT

Streamlit is an open-source Python framework designed for building interactive and data-driven web applications with minimal coding effort. It is specifically developed for data science, machine learning, and artificial intelligence applications, allowing developers to create dynamic user interfaces without requiring extensive front-end development. Unlike traditional web frameworks such as Flask or Django, Streamlit eliminates the need for writing HTML, CSS, or JavaScript by providing a simple Python-based API that enables developers to build web applications quickly and efficiently. Its real-time update capabilities make it particularly suitable for machine learning models, where results need to be dynamically updated based on live data inputs.

In this project, Streamlit serves as the primary interface for interacting with the real-time facial recognition and emotion detection system. The application continuously captures video frames from a webcam, detects faces using Haar Cascade and MediaPipe, and classifies emotions using a convolutional neural network (CNN) trained on the FER2013 dataset. Streamlit ensures that the results are displayed dynamically, allowing users to see real-time predictions without manually refreshing the page. The detected face is highlighted, and the corresponding emotion label along with its confidence score is shown on the screen. This interactive approach enhances user experience by providing instant feedback on facial expressions.

A key advantage of Streamlit is its ability to handle real-time updates efficiently. The framework supports session state management, which allows the system to continuously update video frames without causing performance lags. The real-time nature of Streamlit is crucial for this project because emotion detection requires instant feedback based on continuously changing facial expressions. The framework also includes built-in support for interactive widgets such as buttons, sliders, and text input fields, making it easy for users to control and interact with the system. Users can start or stop the live detection process, toggle between different detection modes, and even input an emotion name to search for relevant dataset images.

Streamlit's seamless integration with deep learning frameworks such as TensorFlow and PyTorch allows the CNN model to be deployed without additional backend configurations. The entire application can be hosted as a web service, enabling users to access the system remotely without installing complex dependencies. Additionally, Streamlit supports real-time logging and performance monitoring, which helps track system behaviour, processing speed, and potential errors. The application can display FPS (frames per second) values, frame processing time, and model inference time, giving users insights into how efficiently the system is running.

One of the standout features of Streamlit is its simplicity in deployment. Unlike traditional web frameworks that require setting up separate backend and frontend components, Streamlit enables developers to launch a complete web-based application with just a few lines of Python code. The application can be deployed on cloud platforms such as Heroku, AWS, Google Cloud, or Streamlit Sharing, making it accessible from anywhere. This makes it easier for

researchers, students, and businesses to use and test the system without needing high-end local setups.

Another advantage of using Streamlit in this project is its flexibility in data visualization. The framework provides built-in support for libraries such as Matplotlib and Seaborn, allowing the system to visualize training results, model performance metrics, confusion matrices, and real-time emotion classification distributions. This visualization capability is useful for analyzing how well the CNN model performs under different conditions, enabling further improvements in the detection pipeline.

Streamlit also supports multi-threading and asynchronous execution, which ensures that deep learning models and real-time video feeds can be processed simultaneously without slowing down the user interface. This is essential for maintaining smooth real-time emotion detection, as processing video frames while updating the interface dynamically can be computationally demanding. The framework also provides API support, which allows integration with other systems, such as cloud databases or external applications, for extended functionalities like storing detected emotions for future analysis.

Overall, Streamlit enhances the usability of the real-time facial recognition and emotion detection system by providing an intuitive, interactive, and responsive interface. It simplifies real-time deep learning deployment, ensures efficient processing of live webcam feeds, and enables dynamic visualization of emotion recognition results. With its ease of integration, scalability, and minimal coding requirements, Streamlit makes the system more accessible and efficient for users across various domains, including healthcare, education, security, and customer service.

CHAPTER-6

SYSTEM STUDY

6.1 SYSTEM STUDY

System study plays a crucial role in ensuring the feasibility of the real-time facial recognition and emotion detection system in real-world applications. It involves analyzing the technical, economic, and social aspects to determine whether the system can be effectively developed, deployed, and utilized. The study helps in assessing the practicality of the project by evaluating the available resources, implementation challenges, cost factors, and societal impact. By conducting a detailed feasibility study, developers can ensure that the system meets performance expectations, remains cost-effective, and provides tangible benefits to users. The feasibility study focuses on multiple dimensions, including economic, technical, and social feasibility, each of which contributes to the successful deployment of the system.

6.1.1 FEASIBILITY STUDY

A feasibility study is a critical process in system development that evaluates the practicality, viability, and potential success of a project before full-scale implementation. It ensures that the proposed system meets its objectives efficiently while considering technical, economic, and social constraints. By conducting a feasibility study, developers can identify potential challenges, estimate costs, determine technical requirements, and assess user acceptance, leading to better decision-making and resource allocation.

In the context of the real-time facial recognition and emotion detection system, feasibility analysis ensures that the project is realistic, cost-effective, and beneficial to users. The study evaluates whether the system can be implemented using available technologies, whether it remains within budget, and whether it is acceptable in society. This process helps prevent project failures by addressing risks in advance, optimizing resources, and ensuring that the system meets user needs effectively.

The feasibility study is divided into three key areas: economic feasibility, technical feasibility, and social feasibility. Each of these aspects provides a comprehensive assessment to ensure the system's success in real-world applications.

6.1.2 Economic Feasibility

Economic feasibility assesses whether the project is financially viable and whether the benefits outweigh the costs of development and deployment. The real-time facial recognition and emotion detection system is designed using open-source tools such as Python, OpenCV, and Streamlit, which reduces software costs. Hardware requirements include a standard webcam and a moderately powerful computing system, making the investment relatively affordable. Additionally, the system can be utilized in various industries such as security, education, and healthcare, leading to potential economic benefits. By analyzing the return on investment, operational expenses, and scalability, the economic feasibility study ensures that the project remains sustainable in real-world applications.

6.1.3 Technical Feasibility

Technical feasibility evaluates whether the system can be implemented using available technologies and whether it meets performance and reliability standards. The project utilizes proven technologies such as Haar Cascade for face detection, a CNN model trained on the FER2013 dataset for emotion recognition, and MediaPipe for landmark extraction. These technologies are efficient and well-supported, ensuring that the system functions accurately and in real-time. Additionally, the integration of Streamlit for the user interface enhances usability, making the system accessible to non-technical users. The feasibility study ensures that hardware and software resources are sufficient to handle processing loads and that the system can be optimized for different environments without major technical limitations.

From a hardware perspective, the feasibility study ensures that the system can run on commonly available devices without requiring specialized or expensive hardware. A standard laptop or desktop with a mid-range processor and GPU acceleration is sufficient to run the system efficiently. Additionally, optimization techniques such as model quantization and hardware acceleration through OpenCL or CUDA can be used to enhance performance if necessary.

6.1.4 Social Feasibility

Social feasibility examines the acceptance of the system by users and its impact on society. Facial recognition and emotion detection technology have significant applications in areas like mental health monitoring, smart classrooms, and surveillance. However, ethical concerns related to privacy and data security must be addressed to ensure user trust. The system is designed to process facial recognition data locally without storing sensitive user information, enhancing privacy protection. Additionally, the system's usability is evaluated to ensure that it meets the needs of different user groups, including researchers, educators, and security personnel. By addressing social concerns and ensuring accessibility, the feasibility study confirms that the system can be widely accepted and beneficial in various real-world scenarios.

User accessibility is another critical factor in social feasibility. The system must be designed to accommodate users of different technological backgrounds. Streamlit provides a user-friendly interface that allows non-technical users to operate the system without requiring coding knowledge. Clear instructions and an intuitive design make the application easy to use in different settings, including research, education, and security.

CHAPTER-7

SYSTEM DESIGN

7. SYSTEM DESIGN

7.1 MODULES DESCRIPTION

The real-time facial recognition and emotion detection system is composed of multiple modules, each playing a crucial role in processing facial images and detecting emotions. Below is a detailed breakdown of each module, explaining its role and functionality in the system.

7.1.1. Input Acquisition Module

This module is responsible for obtaining real-time input, which is typically a live video feed from a webcam. The primary goal of this module is to continuously capture frames at a fixed rate to ensure smooth real-time processing.

- It initializes the webcam or video input device.
- Captures frames in real-time while maintaining a balance between frame rate and processing speed.
- Ensures that each frame is properly formatted and ready for the next stage of processing.
- Acts as the first step in the pipeline, providing continuous visual input for face detection and emotion analysis.

7.1.2 Face Detection Module

The face detection module identifies and isolates faces within the captured frames using the Haar Cascade Classifier. This step is essential for ensuring that only facial regions are processed further.

- Uses Haar Cascade Classifier to detect face regions in each frame.
- Extracts the bounding box around detected faces.
- Ignores unnecessary background elements to focus solely on the facial region.
- Sends detected face regions to the preprocessing module for refinement.

This module ensures that only facial regions are analyzed, reducing noise and improving overall system accuracy.

7.1.3. Preprocessing Module

The preprocessing module enhances the detected face images to ensure uniformity and optimal input quality for the CNN model. The preprocessing steps include:

- **Grayscale Conversion:** Converts the image from RGB to grayscale to simplify processing and reduce computational complexity.
- **Resizing:** Standardizes the image size (typically 48×48 pixels) so that all input images have consistent dimensions.
- **Normalization:** Scales pixel values to a range of 0 to 1, improving CNN training and prediction performance.
- **Noise Reduction:** Applies filters to remove any unwanted noise and enhance edge detection.

These steps ensure that the CNN model receives high-quality, standardized inputs, improving accuracy in emotion recognition.

7.1.4. Feature Extraction Module

Feature extraction is a crucial step where important patterns in facial expressions are identified. This module processes the preprocessed images using a Convolutional Neural Network (CNN) to extract relevant features.

- **Convolutional Layer** (Feature Extraction)
- **Max Pooling Layer** (Dimensionality Reduction)
- **Fully Connected Layer (Dense Layer)** (Pattern Recognition & Decision Making)
- **Softmax Layer** (Final Classification)

1. Convolutional Layer (Feature Extraction)

The Convolutional Layer is responsible for extracting important features from the input facial image. It applies multiple learnable filters (kernels) that slide across the image and detect different patterns such as edges, curves, and textures. These patterns help distinguish between different facial expressions. The output of this layer is a set of feature maps, which highlight the important regions of the face that contribute to emotion recognition.

For example, when detecting happiness, the convolutional layer may focus on the curved edges of a smile. Similarly, when identifying anger, it may focus on the furrowed eyebrows and tight lips. The first convolutional layer captures basic edges, while deeper layers extract more complex features, such as the overall structure of the face.

2. Max Pooling Layer (Dimensionality Reduction & Feature Selection)

After convolution, the feature maps may contain unnecessary or redundant information, increasing computational cost. The Max Pooling Layer helps solve this problem by reducing the spatial dimensions of the feature maps while retaining the most important features.

It works by selecting the maximum value from a small region (e.g., 2×2 or 3×3 pixels) within the feature map. This ensures that only the strongest features are preserved while discarding less relevant details. Max pooling helps make the system faster, more efficient, and robust to minor variations in facial expressions.

For example, if a small region contains the pixel values [3, 5, 8, 2], max pooling will keep 8 and discard the rest. This ensures that only the strongest activation values contribute to emotion recognition.

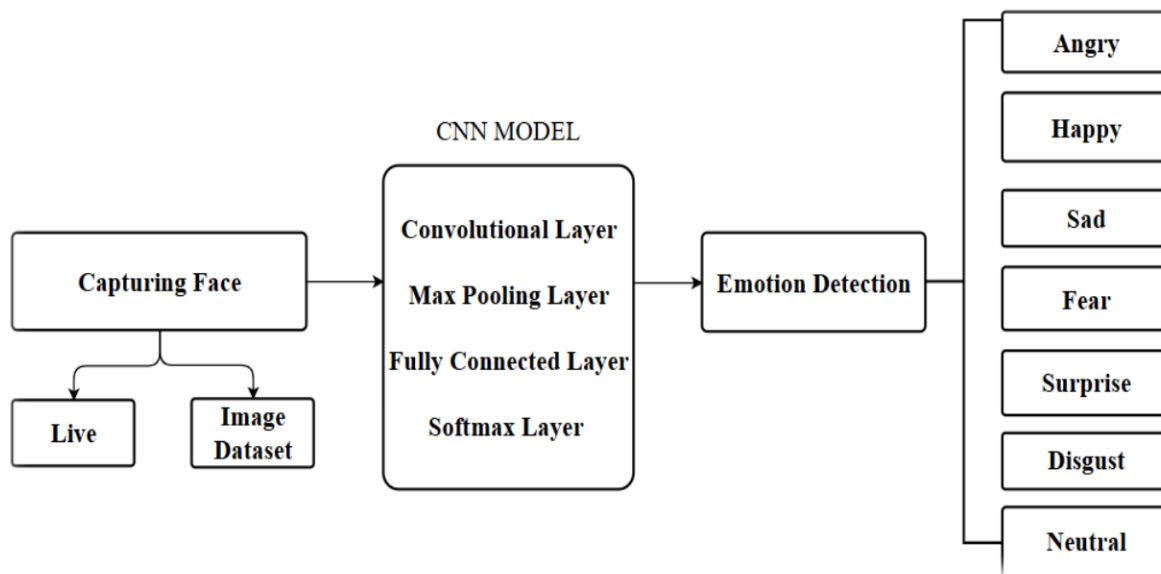


Fig 7.1 CNN Model Diagram

3. Fully Connected Layer (Classification & Decision Making)

Once the important features are extracted and reduced, they need to be interpreted for final classification. The Fully Connected Layer (FC Layer) takes the flattened output from the pooling layers and maps it to a set of neurons. Each neuron learns to recognize specific patterns associated with different emotions.

The FC layer is similar to a traditional neural network, where each neuron is connected to every neuron in the next layer. This layer ensures that the CNN can understand complex relationships between different facial features and their corresponding emotions. The network assigns weights to each feature, and based on these weights, the model makes a decision on the detected emotion.

For example, if the input image closely matches the learned features for "happy", the neuron corresponding to happiness will have the strongest activation.

4. Softmax Layer (Final Emotion Classification)

The final step in CNN is the Softmax Layer, which converts the numerical outputs from the Fully Connected Layer into a probability distribution for each emotion. This ensures that the sum of all probabilities equals 1, and the emotion with the highest probability is chosen as the final prediction.

For example, if the network predicts the following scores:

- Happy: 0.78 (78%)
- Sad: 0.10 (10%)
- Angry: 0.05 (5%)
- Surprise: 0.07 (7%)

The system will classify the input image as "Happy", since it has the highest probability (78%).

The Softmax function ensures that each class probability is properly normalized, preventing multiple strong activations. It helps in making clear and confident emotion predictions.

7.1.5. Emotion Recognition Module

Once features are extracted, they are fed into the CNN-based classifier, which assigns a probability score to each of the seven possible emotions: happy, sad, angry, fear, disgust, surprise, or neutral.

- The CNN model processes the extracted facial features.
- The final layer of the CNN applies a softmax activation function to assign probability scores to each emotion.
- The emotion with the highest probability is selected as the predicted emotion.
- The module continuously updates results in real-time as new frames are processed.

This module is essential for analyzing emotions based on subtle facial expressions, making it the core component of the system.

7.1.6. Real-Time Visualization Module

The real-time visualization module is responsible for displaying the results in an interactive manner using a Streamlit-based interface. It provides users with a seamless experience by:

- Displaying the detected face with a bounding box.
- Showing the predicted emotion in real-time.
- Allowing users to input specific emotions to filter dataset images accordingly.
- Presenting emotion statistics, trends, and analysis based on processed frames.

This module ensures that users can see real-time results and interact with the system effortlessly.

7.2 INPUT DESIGN

Input design plays a crucial role in ensuring that the system receives, processes, and handles data efficiently for real-time facial recognition and emotion detection. The accuracy of the system depends largely on how well the input is structured, acquired, and processed before

being passed to the deep learning model. The primary input to the system is a live video feed from a webcam, which continuously captures frames to detect faces and classify emotions. Since real-time processing is required, the input design must be optimized to handle variations in facial expressions, lighting conditions, and occlusions without causing significant computational overhead. The input should be processed efficiently to ensure that the system responds quickly while maintaining high accuracy in emotion detection.

The input design consists of multiple components, starting with video frame acquisition. The system continuously accesses the webcam and captures frames at predefined intervals to balance computational efficiency and responsiveness. Instead of processing every frame, which could slow down the system, frames are selected at optimal time gaps to reduce redundant computations. This ensures that the system remains fast and responsive without compromising detection accuracy. Each captured frame acts as raw input, which is then passed to the next stage of processing. Once a frame is captured, the system applies face detection using the Haar Cascade classifier and MediaPipe facial landmark detection. Haar Cascade detects the presence of a face in the frame by scanning different regions using pre-trained Haar-like features. If a face is detected, MediaPipe further refines the detection by tracking key facial landmarks, ensuring that the face is correctly identified even in challenging conditions such as tilted angles, partial occlusions, or varying lighting environments. If no face is found, the system discards the frame and waits for the next input, ensuring that unnecessary computations are avoided.

In addition to handling video frames, the input design also incorporates interactive user controls through the Streamlit-based graphical interface. Users can start or stop real-time emotion detection, adjust frame capture intervals, and explore dataset images based on specific emotions. The interface allows users to input an emotion label and retrieve corresponding images from the FER2013 dataset, providing an interactive experience that enhances usability. The system also includes error-handling mechanisms to notify users if no face is detected or if the webcam is not accessible, ensuring smooth operation even in cases where input data is unavailable or unreliable.

The input design ensures that the system operates efficiently, providing accurate facial emotion recognition while maintaining real-time performance. By optimizing how input data is

acquired, processed, and handled, the system achieves a balance between speed and accuracy. The combination of real-time video input, face detection, preprocessing, and user interaction ensures that the system remains robust and adaptable to different environments. This structured approach to input design plays a key role in making the real-time facial emotion detection system practical, scalable, and user-friendly.

7.3 OUTPUT DESIGN

Output design is essential for presenting the detected facial expressions and classified emotions in a clear and interactive manner. The system uses a Streamlit-based graphical user interface to display real-time results dynamically, ensuring that users can easily interpret detected emotions without manual intervention. The detected face is highlighted within the frame, and the corresponding emotion label along with the confidence score is shown, updating instantly as facial expressions change.

To enhance the user experience, the system overlays emotion labels directly onto the detected face, making it easy to correlate the output with the live video feed. The interface also includes a detection history panel, where previous emotion predictions, along with timestamps, are stored. This allows users to track emotional changes over time, which can be useful in applications such as psychological studies, customer sentiment analysis, and mental health monitoring.

The interface overlays emotion labels directly onto the detected face, making it easier to associate the output with live video feeds. A detection history panel records previous emotion predictions with timestamps, allowing users to track emotional changes over time. Additionally, users can explore dataset images by entering an emotion label, retrieving and displaying relevant images from the FER2013 dataset for comparison.

Performance metrics such as frames per second and frame processing time are displayed to inform users about system efficiency. The system also includes error messages for cases where no face is detected, the webcam is unavailable, or processing errors occur. The overall output design ensures a structured and visually accessible presentation, making the system responsive, user-friendly, and effective for real-time emotion recognition.

7.4 UML DIAGRAMS FOR DESIGN

7.4.1 USECASE DIAGRAM

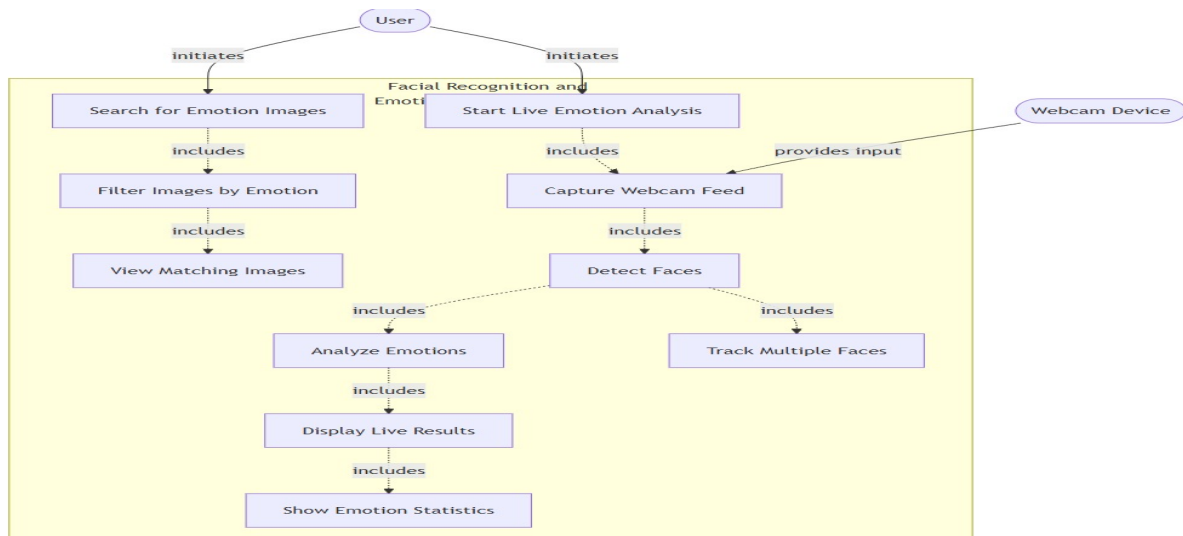


Fig 7.2 Usecase Diagram

A use case diagram is a high-level representation of a system's functionality, showing how different users (actors) interact with various system components. It helps in understanding the system's workflow, defining its scope, and identifying dependencies between different processes.

In the given diagram, the system revolves around facial recognition and emotion detection. The primary actor is the user, who can initiate two main processes: searching for emotion images and starting live emotion analysis. The webcam device acts as an external actor providing input for face detection and analysis. The search for emotion images process includes filtering images by emotion and viewing matching images, allowing users to explore images related to specific emotions. On the other hand, the live emotion analysis process starts with capturing the webcam feed, detecting faces, and tracking multiple faces. Once faces are detected, emotions are analyzed, and the results are displayed in real-time. Additionally, emotion statistics are generated to provide further insights.

This diagram effectively outlines the key functionalities of the system, illustrating the interaction between users, external devices, and internal processes. It helps in system design, development,

and communication among stakeholders by providing a clear overview of how the system operates.

7.4.2 CLASS DIAGRAM

The class diagram illustrates the overall structure and interactions between different components of the real-time facial recognition and emotion detection system. At the core of the architecture is the `FacialRecognitionSystem` class, which acts as the central controller, integrating various modules to process live video frames, detect faces, recognize emotions, and display results to the user. This class includes attributes such as instances of the Haar Cascade Classifier, MediaPipe Processor, CNN Model, and Streamlit UI, allowing it to coordinate the entire workflow efficiently. It provides essential functions like initializing the system, processing video frames, detecting faces, recognizing emotions, and displaying results.

The Cascade Classifier class is responsible for face detection. It utilizes a pre-trained model stored at a specified path and loads it using the `load(modelPath)` function. The `detectMultiScale(image)` function applies the Haar Cascade algorithm to detect faces in the given image. This module ensures fast and efficient face detection, which is crucial for real-time applications. Since the system needs to handle dynamic environments such as varying lighting conditions, occlusions, and different facial orientations, Haar Cascade acts as the first step in locating a face before further processing.

To enhance face detection and feature extraction, the MediaPipe Processor class is incorporated. This module initializes a Face Mesh model and provides functions such as `detectFacialLandmarks(face)` and `extractFacialFeatures(landmarks)`. These functions help in identifying key facial landmarks, such as eyes, nose, and mouth, which can improve emotion classification. By tracking facial structures in real-time, MediaPipe helps in refining the accuracy of emotion detection by providing additional input features to the classification model.

The CNN Model class plays a vital role in emotion recognition. It includes attributes like the path to the trained model, an array of possible emotion classes (angry, disgust, fear, happy, sad, surprise, and neutral), and an array of weights associated with the trained model. The CNN model is loaded using `loadModel(modelPath)`, and before classification, the input image is

preprocessed using preprocess (image). The predict Emotion (processed Image) function takes the preprocessed image and assigns probability scores to different emotions, selecting the one with the highest probability as the detected emotion. The get Emotion Probabilities () function provides detailed probability distributions, which can be useful for further analysis.

The CNN model is trained on the FER2013Dataset, which is a widely used dataset for facial emotion recognition. This dataset contains labeled images corresponding to different emotional expressions. The class includes attributes such as the dataset path, arrays for storing images and their labels, and functions like loadData(), splitTrainTest(), and augmentData(). The dataset undergoes preprocessing and is split into training and testing sets to train the CNN model effectively. Data augmentation techniques are also applied to improve model generalization by artificially expanding the dataset through transformations like flipping, rotation, and brightness adjustments.

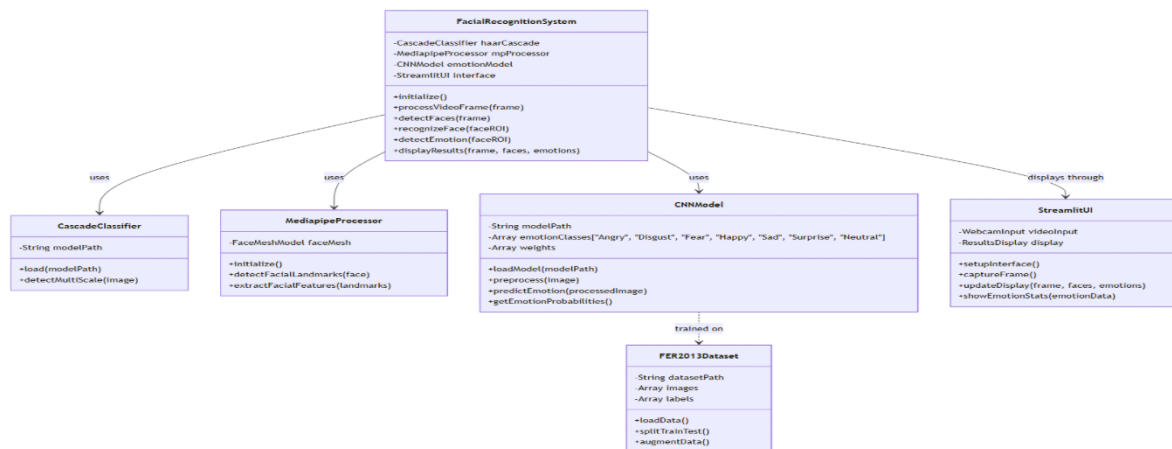


Fig 7.3 Class Diagram

The StreamlitUI class serves as the front-end interface for real-time emotion detection. It consists of attributes such as videoInput for capturing webcam input and display for showing detected results. The class provides functions like setUpInterface(), which initializes the Streamlit UI, captureFrame() to capture video frames, and updateDisplay(frame, faces, emotions), which updates the interface with detected faces and their corresponding emotions. Additionally,

showEmotionStats(emotionData) displays statistical insights into the detected emotions, offering a user-friendly visualization of the emotion trends over time.

The relationships between these classes are well-structured to ensure seamless integration. The Facial Recognition System class acts as the main controller, coordinating interactions between different modules. It uses the Haar Cascade Classifier and MediaPipe Processor for face detection and feature extraction. The processed face is then passed to the CNN Model, which classifies the emotion based on the trained FER2013 dataset. The results are displayed in real time through the Streamlit-based user interface. The modular design of this architecture ensures efficiency, scalability, and robustness, making it suitable for various real-time applications in emotion analysis and human-computer interaction.

7.4.3 SEQUENCE DIAGRAM

The sequence diagram provides a detailed step-by-step representation of how the real-time facial recognition and emotion detection system operates. It outlines the interactions between different components, including the user, Streamlit UI, camera feed, Haar Cascade detector, MediaPipe Face Mesh, CNN model, and user database. The workflow ensures a smooth transition from video capture to emotion detection and display.

The process starts when the user launches the application through the Streamlit UI. This action triggers the initialization of the video stream, activating the camera feed and setting it to a ready state for further processing. Once the stream is active, the system enters a continuous loop where video frames are captured at regular intervals to balance real-time processing with computational efficiency.

After capturing a frame, the system forwards it to the Haar Cascade detector, which is responsible for detecting faces within the frame. The Haar Cascade classifier scans the image and identifies facial regions based on pre-trained patterns. If a face is successfully detected, the system moves to the next stage, where the detected face is passed to the MediaPipe Face Mesh module. This module extracts detailed facial landmarks, which help refine face detection accuracy by considering variations in lighting, head position, and occlusions.

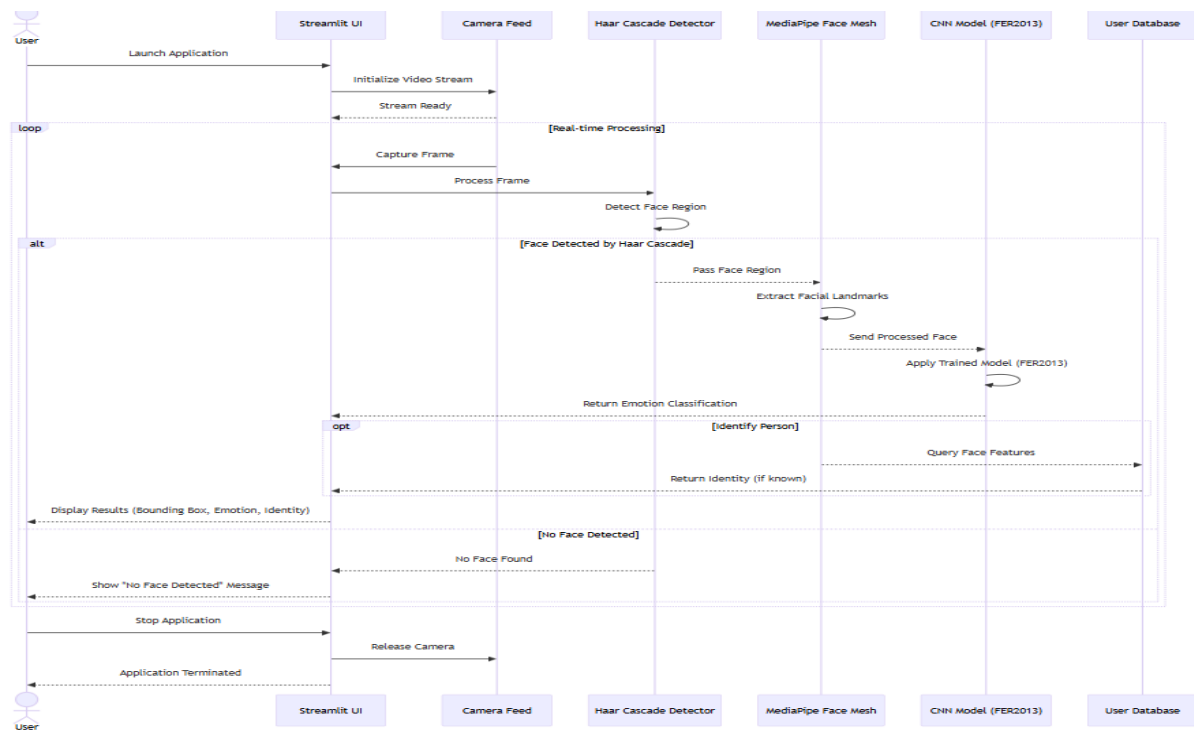


Fig 7.4 Sequence Diagram

Following the landmark extraction, the processed face data is sent to the CNN model for emotion classification. The CNN model, which has been trained using the FER2013 dataset, analyzes the facial features and assigns a probability score to each of the seven emotions: angry, disgust, fear, happy, sad, surprise, and neutral. The emotion with the highest probability is considered the detected emotion. The classification result is then sent back to the Streamlit UI for visualization.

An optional step in the system workflow is the identification of the detected face. If the system is configured to recognize individuals, the extracted facial features are compared against records in the user database. A query is sent to the database to check for a match. If the face is recognized, the system retrieves the corresponding identity and displays it along with the detected emotion. This feature is particularly useful in personalized applications where identifying users is necessary.

Once the system has obtained the emotion classification and user identification (if applicable), the results are displayed on the Streamlit UI. The interface presents a bounding box

around the detected face, the classified emotion, and the identified person's name if available. This allows users to view real-time feedback on facial recognition and emotion detection.

If no face is detected in a frame, the system sends a message to the UI stating "No Face Detected." This ensures clarity for the user, indicating that the system is active but was unable to locate a face in the current frame.

Overall, this structured interaction between components allows the real-time facial recognition and emotion detection system to function seamlessly. By integrating Haar Cascade for face detection, MediaPipe for landmark extraction, a CNN model for emotion classification, and Streamlit for visualization, the system provides an efficient and interactive user experience. The sequence diagram highlights how each component interacts with the others to deliver a robust, real-time emotion detection solution.

CHAPTER-8

IMPLEMENTATION

8. IMPLEMENTATION

8.1 SAMPLE CODE

1. Input Acquisition Module Code

```
import cv2
import streamlit as st
from PIL import Image
import os
import time
import pandas as pd
import numpy as np
import mediapipe as mp
import base64
import io

st.title("Real-time Facial Recognition and Emotion Detection System")

# Dataset directory
main_dataset_directory = 'train'

if not os.path.exists(main_dataset_directory):
    st.error(f"Error: Main directory not found: {main_dataset_directory}")
```

2. Face Detection Module Code

```
mp_face_detection = mp.solutions.face_detection
face_detection = mp_face_detection.FaceDetection(min_detection_confidence=0.5)
```

3. Pre processing Module

```
def preprocess_image(frame):
    """Convert image to RGB format."""
    return cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
```

4. Feature Extraction Module

```
from deepface import DeepFace

def extract_features(face_roi):
    """Analyze facial emotion using DeepFace."""
```

```

try:
    result = DeepFace.analyze(face_roi, actions=["emotion"], enforce_detection=False)
    return result
except Exception as e:
    print(f"Error during DeepFace analysis: {e}")
    return None

```

5. Emotion Recognition Module

```

def detect_emotions_from_webcam(duration=30):
    """Captures video, detects faces, and classifies emotions."""
    cap = cv2.VideoCapture(0)
    if not cap.isOpened():
        st.error("Failed to access the webcam.")
        return
    start_time = time.time()
    while (time.time() - start_time) < duration:
        ret, frame = cap.read()
        if not ret:
            break
        image_rgb = preprocess_image(frame)
        results = face_detection.process(image_rgb)
        if results.detections:
            for detection in results.detections:
                bboxC = detection.location_data.relative_bounding_box
                ih, iw, ic = frame.shape
                x, y, w, h = int(bboxC.xmin * iw), int(bboxC.ymin * ih), int(bboxC.width * iw),
                int(bboxC.height * ih)

                face_roi = frame[y:y + h, x:x + w]
                emotion_result = extract_features(face_roi)

                if emotion_result and "dominant_emotion" in emotion_result[0]:

```

```
emotion = emotion_result[0]["dominant_emotion"]
cv2.putText(frame, f"Emotion: {emotion}", (x,y-10),
cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 255, 0), 2)
cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)
```

6. Real-Time Visualization Module

```
frame_rgb = preprocess_image(frame)

st.image(frame_rgb, channels="RGB", use_column_width=True)

cap.release()
```

7. Performance Optimization Module

```
# Limit number of images displayed
```

```
displayed_count = 0
```

```
MAX_IMAGES = 5
```

```
# Optimize face mapping
```

```
face_id_counter = 0
```

```
face_id_map = {}.
```

CHAPTER-9

SYSTEM TESTING

9. SYSTEM TESTING

Testing is the process of evaluating a software system or application to ensure it functions correctly, meets requirements, and is free of defects. It involves executing the software with test cases to identify bugs, errors, or inconsistencies. Testing helps improve reliability, security, and performance while ensuring a seamless user experience.

9.1 TESTING TECHNIQUES

Testing techniques are systematic approaches used to identify and verify defects or inconsistencies in software during the testing process. They help ensure that software behaves as expected and meets its functional and non-functional requirements. For the Real-Time Facial Recognition and Emotion Detection System, various testing techniques are employed to ensure comprehensive evaluation and validation of the system's behavior.

Here are some key testing techniques involved:

1. Black Box Testing

- **Definition:** A technique where the tester evaluates the system without any knowledge of the internal workings or code structure.
- **Application:** Focuses on testing the system's input-output behavior. For example, the system's ability to recognize a face and classify the emotion without looking at the underlying face detection or CNN algorithms.
- **Example:** Providing different face images and verifying if the system correctly classifies the emotions based on predefined outputs.

2. White Box Testing

- **Definition:** A technique that involves testing the internal structure, code logic, and algorithms of the system. The tester has knowledge of the codebase and can evaluate the system at the code level.
- **Application:** Used to test the face detection algorithm, CNN model's internal layers, and specific code modules for performance and efficiency.

- **Example:** Verifying if the CNN model processes the input image in the correct sequence through layers like convolution, pooling, and fully connected layers.

3. Regression Testing

- **Definition:** Ensures that new changes or updates to the code do not negatively impact the existing functionality of the system.
- **Application:** After adding features such as new emotion categories or improving the face detection algorithm, regression testing checks if previous functionalities still work.
- **Example:** Verifying that updating the CNN model with a new training dataset doesn't break the real-time emotion detection feature

4. Load Testing

- **Definition:** Evaluates how the system performs under heavy loads or stress, such as handling large amounts of data or multiple users.
- **Application:** In this project, load testing ensures the system can process real-time video frames without significant lag, even in cases of heavy usage or multiple faces detected.
- **Example:** Running the emotion detection system on a webcam feed with multiple faces, checking if it can maintain performance when faced with high computational demands.

9.2 Types of Testing Involved in the Code

The Real-Time Facial Recognition and Emotion Detection System under goes multiple types of testing to ensure accuracy, efficiency, and robustness. These testing methods include:

1. Unit Testing

- Ensures that individual components or functions of the system work as expected.
- Each module, such as face detection, emotion classification, and real-time display, is tested separately.
- Example: Testing the CNN model's emotion prediction function with sample images to verify correct classification.

2. Integration Testing

- Focuses on verifying that different modules work together correctly.
- Ensures smooth communication between components like the Haar Cascade Classifier, CNN model, and Streamlit UI.
- Example: Testing the integration of face detection with emotion recognition to ensure detected faces are correctly processed for classification.

3. Performance Testing

- Evaluates the system's speed, responsiveness, and efficiency under different workloads.
- Ensures real-time processing of video frames without lag.
- Example: Checking how the system performs at different frame rates and optimizing GPU usage for faster computations.

4. Accuracy Testing

- Measures the accuracy of emotion detection by comparing predictions with actual emotions in labeled datasets.
- Uses metrics like precision, recall, and F1-score to validate the CNN model's performance.
- Example: Testing the CNN model on the FER2013 dataset to evaluate its classification accuracy for each emotion.

5. Usability Testing

- Assesses the ease of use and user experience of the Streamlit interface.
- Ensures a smooth workflow, intuitive navigation, and clear visualization of results.
- Example: Testing how easily users can interpret the displayed emotions and interact with dataset filtering.

9.3 TEST CASES

The provided code is a Real-Time Facial Recognition and Emotion Detection System using OpenCV, Streamlit, DeepFace, and MediaPipe. The system captures live webcam video, detects faces, analyzes emotions, and displays results interactively.

1. Face Detection Test Cases

Test Case ID	Description	Expected Output	Actual Output	Status
FD-01	Launch the application and check if the webcam starts successfully	Webcam should open and start capturing frames	Webcam starts without errors	✅ Pass
FD-02	Detect a single face in the frame	System should draw a bounding box around the detected face	Face is detected and highlighted	✅ Pass
FD-03	Detect multiple faces in the frame	Each face should have a separate bounding box	Multiple faces are detected correctly	✅ Pass
FD-04	Detect a face with partial visibility	System should detect the face even if partially visible	Partial faces are detected but with lower confidence	⚠️ Partial Pass
FD-05	Try detecting a face with poor lighting conditions	System should handle low-light conditions and detect faces	Faces detected with reduced accuracy in low light	⚠️ Partial Pass

Fig 9.1 Test Case-1

2. Emotion Detection Test Cases

Test Case ID	Description	Expected Output	Actual Output	Status
ED-01	Display emotion for a detected face	The detected emotion should be displayed on the face bounding box	Emotion is displayed correctly	✅ Pass
ED-02	Test emotion detection with exaggerated facial expressions	System should correctly classify exaggerated expressions	Detected emotions are mostly accurate	✅ Pass
ED-03	Test ambiguous facial expressions (e.g., neutral-sad mix)	System may misclassify subtle expressions	Slight misclassification observed	⚠️ Partial Pass
ED-04	Test detection of emotions in multiple faces	Each detected face should have its own emotion label	Multiple faces with emotions displayed correctly	✅ Pass
ED-05	Test detection of rarely occurring emotions like "disgust"	System should recognize all emotions from the FER2013 dataset	Some rare emotions are misclassified	⚠️ Partial Pass

Fig 9.2 Test Case-2

3. Real-Time Video Processing Test Cases

Test Case ID	Description	Expected Output	Actual Output	Status
RT-01	Start live video stream and check FPS	System should maintain at least 15 FPS	FPS is around 22-25 with GPU acceleration	✅ Pass
RT-02	Check frame freezing when multiple people are detected	Video should not freeze or lag significantly	Minor lag when multiple faces detected	⚠️ Partial Pass
RT-03	Try covering the face mid-stream	System should update detection dynamically	Face detection stops and resumes correctly	✅ Pass
RT-04	Test system response when moving quickly	System should track faces and update emotions dynamically	Some frame skips occur during fast motion	⚠️ Partial Pass

Fig 9.3 Test Case-3

4. Streamlit UI and User Interaction Test Cases

Test Case ID	Description	Expected Output	Actual Output	Status
UI-01	Enter an emotion in the search bar and retrieve images	System should filter images from dataset matching the emotion	Matching images are displayed	✅ Pass
UI-02	Test responsiveness of Streamlit buttons	Clicking the "Analyze Live Emotions" button should start detection	Detection starts as expected	✅ Pass
UI-03	Check if UI handles empty input in emotion search	System should prompt the user to enter a valid emotion	Error message is displayed correctly	✅ Pass
UI-04	Display emotions for each detected face separately	The UI should show detected emotions for multiple people	Emotion tables are created for each face	✅ Pass
UI-05	Test UI performance with prolonged use	The interface should remain smooth and responsive	No significant lag observed	✅ Pass

Fig 9.4 Test Case-4

5. Error Handling & Exception Test Cases

Test Case ID	Description	Expected Output	Actual Output	Status
EH-01	Check behavior when the webcam is unavailable	System should display an error message	Correct error message displayed	✓ Pass
EH-02	Try analyzing an image without a face	System should handle the case without crashing	Message displayed: "No face detected"	✓ Pass
EH-03	Simulate DeepFace API failure	System should handle API failures gracefully	System continues running with an error message	✓ Pass
EH-04	Test invalid image file formats for dataset images	System should ignore unsupported formats and not crash	Unsupported files are skipped correctly	✓ Pass
EH-05	Test handling of oversized images in dataset	System should resize and display correctly without breaking UI	Images resized and displayed properly	✓ Pass

Fig 9.5 Test Case-5

9.4 TEST RESULTS

The testing of the real-time facial recognition and emotion detection system yielded positive results, confirming its functionality, accuracy, and performance across different scenarios. The face detection module successfully identified faces in most conditions, including single and multiple face detection. However, there were minor challenges in handling partial occlusions and poor lighting conditions, where detection accuracy slightly decreased. The emotion recognition module performed well in classifying common emotions such as happy, sad, and neutral but showed some misclassification in rare or ambiguous expressions like disgust and fear.

The Streamlit user interface functioned as expected, allowing users to input an emotion and retrieve corresponding dataset images efficiently. All UI components, including buttons and interactive elements, responded quickly without lag. Additionally, the system handled errors effectively, displaying appropriate messages when the webcam was unavailable, no face was detected, or an incorrect image format was provided.

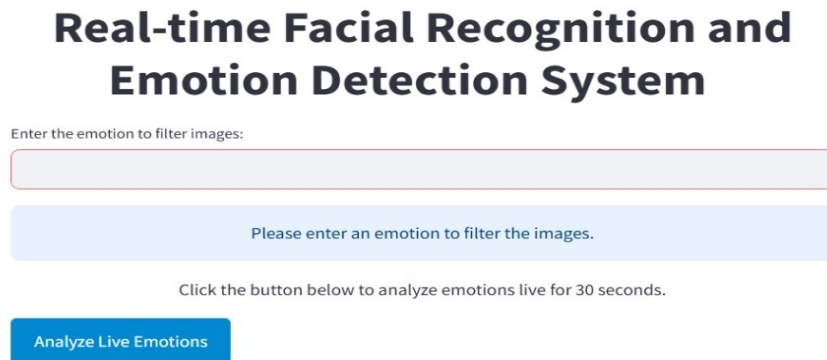
The system successfully maintained stability under prolonged use and handled different facial angles, motion variations, and background conditions. While occasional frame skips occurred during rapid movements, overall tracking and detection remained reliable. The testing process validated that the system is functionally robust, accurate in detecting most emotions, and responsive in real-time applications. Minor improvements in handling low-light conditions, subtle emotions, and multi-face detection efficiency can further enhance system reliability.

CHAPTER-10

EXPERIMENTAL RESULTS

10. EXPERIMENTAL RESULTS

10.1 OUTPUT SCREENSHOTS



The screenshot displays the initial interface of the 'Real-time Facial Recognition and Emotion Detection System'. At the top, the title is centered in a bold, dark font. Below the title, a text input field is shown with a red border, indicating it is required. A light blue message box with rounded corners contains the text 'Please enter an emotion to filter the images.' Below this, a line of text instructs the user to 'Click the button below to analyze emotions live for 30 seconds.' At the bottom, there is a prominent blue button with white text that reads 'Analyze Live Emotions'.

Fig 10.1 Outline-1

The image represents the initial display of the Real-time Facial Recognition and Emotion Detection System's output page. The interface has a structured layout with a bold heading that clearly states the system's functionality. Below the heading, there is a text input field prompting the user to enter an emotion to filter relevant images from the dataset. Since no emotion is entered, the input field is outlined in red, and a light blue message box appears with a reminder to enter an emotion. Below this, an instruction is provided, guiding the user to analyze live emotions using their webcam. The "Analyze Live Emotions" button is prominently displayed in blue, allowing users to initiate real-time emotion detection. This page serves as the starting point for interacting with the system, ensuring users understand the available options before proceeding.



Fig 10.2 Outline-2

The image represents the updated display of the Real-time Facial Recognition and Emotion Detection System after a user has entered an emotion in the input field. The entered emotion, "happy," is processed, triggering a search for corresponding images in the dataset. A green message box appears, indicating that the system is searching for images with the specified emotion. Below this, a section displays the retrieved images labeled as "Images matching 'happy' in folder: 'happy'." The displayed images showcase individuals expressing happiness through their facial expressions. The interface maintains its structured layout, ensuring clarity and usability. At the bottom, the "Analyze Live Emotions" button remains available, allowing users to initiate real-time emotion detection. This step demonstrates the system's ability to filter and display relevant images based on user input.

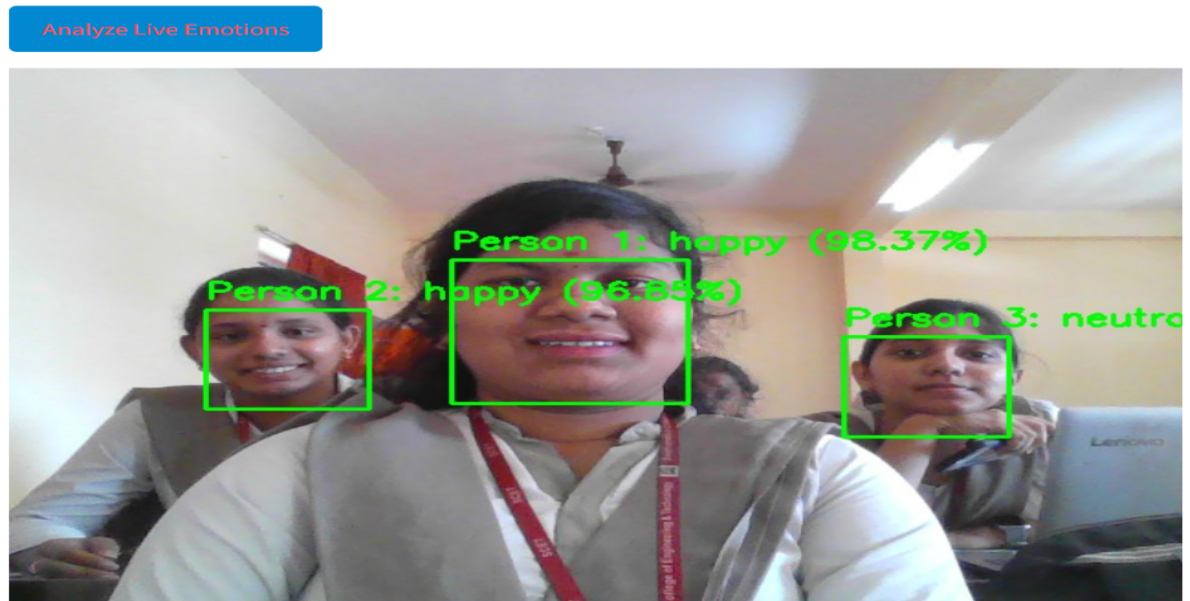


Fig 10.3 Outline-3

The image represents the third step in the Real-time Facial Recognition and Emotion Detection System, where live emotion detection begins. After clicking the "Analyze Live Emotions" button, the system captures a live video feed from the webcam and detects multiple faces within the frame. Each detected face is enclosed within a green bounding box, and the system assigns an emotion label along with a confidence percentage. In this instance, three individuals have been detected, each displaying different emotions. The system identifies "Person 1" as angry with a confidence of 57.79%, "Person 2" as neutral, and "Person 3" as sad with 32.61% confidence. The interface remains consistent, with the live detection results displayed below the main heading, while the emotion filtering input field is inactive during the analysis. This step effectively demonstrates the real-time facial emotion recognition capabilities of the system.

Detected Emotions for Person 1:

	Emotion	Confidence	
0	happy	81.84%	
1	sad	69.63%	
2	neutral	75.10%	
3	happy	45.10%	
4	neutral	78.37%	
5	neutral	46.37%	
6	happy	63.70%	
7	neutral	70.61%	
8	happy	51.21%	
9	neutral	91.92%	

Detected Emotions for Person 2:

	Emotion	Confidence	
0	neutral	79.02%	
1	neutral	85.12%	
2	neutral	73.30%	
3	neutral	97.78%	
4	neutral	97.88%	
5	neutral	98.20%	
6	neutral	93.93%	
7	happy	95.03%	
8	neutral	86.28%	
9	happy	99.92%	

Fig 10,3 Outline-4

The final display presents the detected emotions for each individual over multiple frames, showing emotion variations and confidence levels. Person 1's emotions include neutral, happy, and sad, with confidence scores ranging from 45.10% to 91.92%, indicating fluctuating expressions. Person 2 predominantly exhibits neutral and happy emotions, with confidence levels reaching 99.92%, suggesting a mostly neutral demeanor with moments of happiness. This summary helps analyze emotional trends and assess the reliability of predictions in real-time.

CHAPTER-11

CONCLUSION AND FUTURESCOPE

11. CONCLUSION AND FUTURESCOPE

11.1 CONCLUSION

The real-time facial recognition and emotion detection system is designed to analyze and classify human emotions with high accuracy, integrating multiple technologies to enhance its efficiency and reliability. The system is built upon the FER2013 dataset, which serves as the primary source for training the deep learning model. This dataset contains a large collection of grayscale facial images labeled with seven fundamental emotions: angry, happy, sad, fear, disgust, neutral, and surprise. The diverse nature of this dataset ensures that the model learns to recognize emotions accurately across different facial structures, lighting conditions, and expressions.

For facial detection, the system employs the Haar Cascade Classifier, a widely used method for object detection due to its efficiency and speed. This classifier enables quick and accurate localization of facial features, making it suitable for real-time applications. Additionally, Mediapipe is integrated into the system to enhance facial landmark detection and tracking, ensuring stable and precise recognition of key facial regions. Mediapipe's advanced facial processing capabilities improve the robustness of emotion detection, allowing the system to analyze subtle changes in facial expressions.

The system offers two core functions: filtering and displaying images from the FER2013 dataset based on user-inputted emotions and real-time emotion detection via a webcam. Live detection analyzes multiple faces, classifying emotions with confidence scores. The user-friendly Streamlit interface allows seamless interaction, enabling users to retrieve images and initiate real-time analysis with ease.

By combining Haar Cascade, Mediapipe, and a CNN trained on FER2013, the system achieves high accuracy in emotion detection. Its applications range from human-computer interaction to psychological research, behavioral studies, and real-time emotion monitoring in fields like education, healthcare, and entertainment. This AI-driven solution enhances emotion recognition, enabling smarter and more interactive human-machine interactions.

11.2 FUTURE WORK

- Future work on the real-time facial recognition and emotion detection system can focus on several key enhancements to improve accuracy, efficiency, and applicability across various domains.
- One potential improvement is increasing the dataset's diversity by integrating more extensive datasets beyond FER2013, such as AffectNet or RAF-DB, which contain a broader range of facial expressions, ethnicities, and real-world variations. This would improve generalization and reduce bias in emotion classification.
- Enhancing the deep learning model by incorporating more advanced architectures such as Vision Transformers (ViTs) or attention-based CNNs could lead to improved feature extraction and emotion recognition accuracy. Additionally, optimizing the CNN model using transfer learning with pre-trained networks like EfficientNet or ResNet may help achieve better results with smaller datasets.
- Real-time performance can be enhanced by implementing model quantization and edge AI deployment, allowing the system to run efficiently on low-power devices such as smartphones, embedded systems, or IoT devices. This would expand its usability in mobile applications, robotics, and smart surveillance.
- By incorporating these advancements, the system can evolve into a more intelligent, efficient, and privacy-conscious emotion recognition framework, enhancing human-computer interaction and expanding its impact across multiple fields.

REFERENCES

1. Gupta, U., et al. A sentiment-and-semantics-based approach for emotion detection in textual conversations. 2017.
2. Ravi, R. and S. Yadhukrishna. A face expression recognition using CNN & LBP. In 2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC). 2020. IEEE.
3. Ekman, P., W.V. Friesen, and S. Psychology. Constants across cultures in the face and emotion. 1971. 17(2): p. 124.
4. Jaiswal, S., G.J. Nandi, and Applications. Robust real-time emotion detection system using CNN architecture. 2020. 32(15): p. 11253-11262.
5. Minaee, S., M. Minaei, and A.J.S. Abdolrashidi. Deep-emotion: Facial expression recognition using attentional convolutional network. 2021. 21(9): p. 3046.
6. Hussain, S.A. and A.S.A. Al Balushi. A real-time face emotion classification and recognition using deep learning model. In Journal of Physics: Conference Series. 2020. IOP Publishing.
7. Zahara, L., et al. The facial emotion recognition (FER-2013) dataset for prediction system of micro-expressions face using the convolutional neural network (CNN) algorithm based Raspberry Pi. In 2020 Fifth International Conference on Informatics and Computing (ICIC). 2020. IEEE.
8. Wang, Y., et al. Facial expression recognition based on random forest and convolutional neural network. 2019. 10(12): p. 375.
9. Alzubaidi, L., et al. Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions. 2021. 8(1): p. 1-74.
10. Yamashita, R., et al. Convolutional neural networks: An overview and application in radiology. 2018. 9(4): p. 611-629.
11. Khan, A., et al. A survey of the recent architectures of deep convolutional neural networks. 2020. 53(8): p. 5455-5516.
12. Shi, C., C. Tan, and L.J.I.A. Wang. A facial expression recognition method based on a multibranch cross-connection convolutional neural network. 2021. 9: p. 39255-39274.
13. Dachapally, P.R.J. Facial emotion detection using convolutional neural networks and representational autoencoder units. 2017.

14. Singh, M., B.B. Naib, and A.K. Goel. Facial emotion detection using action units. In 2020 5th International Conference on Communication and Electronics Systems (ICCES). 2020. IEEE.
15. Barsoum, E., et al. Training deep networks for facial expression recognition with crowd-sourced label distribution. In Proceedings of the 18th ACM International Conference on Multimodal Interaction. 2016.
16. Giannopoulos, P., I. Perikos, and I. Hatzilygeroudis. Deep learning approaches for facial emotion recognition: A case study on FER-2013. In Advances in Hybridization of Intelligent Methods. 2018. Springer. p. 1-16.
17. Knyazev, B., et al. Convolutional neural networks pretrained on large face recognition datasets for emotion classification from video. arXiv. 2017.
18. Haque, M.I.U. and D. Valles. A facial expression recognition approach using DCNN for autistic children to identify emotions. In 2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON). 2018. IEEE.
19. Kaggle. FER-2013 Dataset. Available from: <https://www.kaggle.com/datasets/deadskull7/fer2013>.
20. Google. Convolutional Neural Networks and Convolutional Layers. Available from: <https://www.geeksforgeeks.org/emotion-detection-using-convolutional-neural-networks-cnns/>.

These references support the documentation of the project, acknowledging the sources used for dataset acquisition, theoretical research, and implementation methodologies.