

# Facial Recognition to Detect Mood and Suggest Songs

A Project Report

*Submitted in the partial fulfilment for the award of the degree of*

**BACHELOR OF ENGINEERING  
IN  
COMPUTER SCIENCE WITH SPECIALIZATION IN  
ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING**

**Submitted by:**

20BCS6712	AMISHA KHANNA
20BCS6760	HRITVIK MATHUR
20BCS6749	DEVI PRASAD

**Under the Supervision of:**

Ms. Shubhangi Mishra



**CHANDIGARH  
UNIVERSITY**  
Discover. Learn. Empower.

**CHANDIGARH UNIVERSITY, GHARUAN, MOHALI - 140413,  
PUNJAB**

**November, 2023**

---



**CHANDIGARH  
UNIVERSITY**

Discover. Learn. Empower.

### **DECLARATION**

**‘Amisha Khanna’, ‘Hritvik Mathur’ and ‘Devi Prasad’, students of ‘Bachelor of Engineering in IBM SPECIALISEDARTIFICIAL INTELLIGENCE AND MACHINE LEARNING’, session: 2023-24,**

Department of Computer Science and Engineering, Apex Institute of technology, Chandigarh University, Punjab, here by declare that the work presented in this Project Work entitled **‘Facial Recognition To Detect Mood And Suggest Songs Accordingly’** is the outcome of our own bonafide work and is correct to the best of our knowledge and this work has been undertaken taking care of Engineering Ethics and is done under the Supervision of **‘Ms. Shubhangi Mishra’**.

**Date:** 01/11/2023

**Hritvik Mathur**

**20BCS6760**

**Amisha Khanna**

**20BCS6712**

**Devi Prasad**

**20BCS6749**

---



**CHANDIGARH  
UNIVERSITY**  
Discover. Learn. Empower.

## **ACKNOWLEDGEMENT**

We have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. We would like to extend my sincere thanks to all of them.

We are highly indebted to **Ms. Shubhangi Mishra** for her guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.

We would like to express our gratitude towards everyone for their kind co-operation and encouragement which help me in completion of this project.

We would like to express our special gratitude and thanks to industry persons for giving me such attention and time.

Our thanks and appreciations also go to our colleague in developing the project and people who have willingly helped us out with their abilities.

---

# Abstract

In the digital age, music has become an integral part of our daily lives, often serving as a means of emotional expression and mood enhancement. This abstract presents a novel approach that leverages facial recognition technology to detect an individual's current mood and, based on the analysis, suggests songs that are tailored to their emotional state.

The proposed system combines the power of computer vision and machine learning to provide a seamless and personalized music recommendation experience. The system employs state-of-the-art facial recognition algorithms to capture and analyse the user's facial expressions in real-time. It determines the user's emotional state, including happiness, sadness, anger, or neutrality. These emotions are classified into distinct categories to ensure precise song recommendations.

In conclusion, the fusion of facial recognition and music recommendation technology creates a unique and enjoyable experience, offering users a seamless way to discover and enjoy music that complements their current mood. This research showcases the potential of emerging technologies to provide highly personalized and emotionally resonant content to users, enriching their interaction with music and technology in a profound and meaningful way. recognition.

## Keywords:

Facial recognition, Mood detection, Emotion analysis, Computer vision, Real-time analysis, Machine learning

---

# Table of Contents

Declaration	i
Acknowledgement	ii
Abstract	iii
1. List of Figures	1
2. List of Abbreviations	3
3. Introduction	
1.1 Problem Definition	4
1.2 Project Overview	6
1.3 Hardware Specification	8
1.4 Software Specification	9
4. Literature Survey	10
2.1 Existing System	12
2.2 Proposed System	14
2.3 Literature Review Summary	15
5. Problem Formulation	17
6. Research Objective	19
7. Methodologies	22
8. Result / Analysis	46
9. Conclusion	63
10. Reference	65

---

# **LIST OF FIGURES**

<b>Figure 1</b>	Multiple Faces
<b>Figure 2</b>	Face Detection
<b>Figure 3</b>	Positive , Negative and Neutral Emotions
<b>Figure 4</b>	Objective of the Project
<b>Figure 5</b>	Flowchart to Suggest Songs
<b>Figure 6</b>	Image Recognition
<b>Figure 7</b>	Steps Diagram for real-time analysis and integration
<b>Figure 8</b>	Flow Diagram of Proposed Method
<b>Figure 9</b>	Emotions in Gray scale
<b>Figure 10</b>	Dataset Creation
<b>Figure 11</b>	Resizing of images
<b>Figure 12</b>	Sample Face Detection
<b>Figure 13</b>	Architecture of CNN
<b>Figure 14</b>	Haar Cascade Classifier
<b>Figure 15</b>	Visualization of The Feature Map.
<b>Figure 16</b>	Feature Classification
<b>Figure 17</b>	Snapshot of proposed System
<b>Figure 18</b>	System GUI

## **LIST OF FIGURES**

<b>Figure 19</b>	User panel to choose singer
<b>Figure 20</b>	Sample Suggested Song
<b>Figure 21</b>	Table 1 - Detection Accuracy
<b>Figure 22</b>	Table 2 – Parameters used in CNN
<b>Figure 23</b>	Accuracy graph of CNN
<b>Figure 24</b>	Comparison Table
<b>Figure 25</b>	Performance Metrics Table

## **LIST OF ABRIVATIONS**

<b>FRMDSSA</b>	Facial Recognition to Detect Mood And Suggest Songs Accordingly
<b>HCI</b>	Human-Computer Interaction.
<b>CNN</b>	Convocational Neural Network
<b>DNN</b>	Deep Neural Network
<b>SVM</b>	Support Vector Machine
<b>RNN</b>	Recurrent Neural Network
<b>KNN</b>	K- Nearest Neighbour
<b>GUI</b>	Graphical User Interface
<b>ML</b>	Machine Learning
<b>FPS</b>	Frames Per Second
<b>FER</b>	Facial Emotion Recognition



# 1. INTRODUCTION

## 1.1 Problem Definition

The integration of facial recognition technology to detect mood and suggest songs accordingly represents a fascinating intersection of computer vision, emotion analysis, and music recommendation. To understand the background of this problem, it's essential to explore the evolution of each of these components.

**1. Facial Recognition Technology:** Facial recognition has made significant advancements over the past two decades. It primarily involves the use of computer algorithms to identify and analyse facial features from images or video. Early applications were primarily for security and authentication, but the technology has expanded into various other domains. The development of deep learning and convolutional neural networks (CNNs) has significantly improved the accuracy of facial recognition systems. These advances have paved the way for more nuanced applications, such as emotion recognition.

**2. Emotion Analysis:** Emotion analysis, often referred to as affective computing, is a field that has gained prominence alongside the advancement of artificial intelligence and machine learning. Researchers have worked on creating algorithms that can detect and interpret human emotions based on various inputs, including text, voice, and facial expressions. The ability to discern emotions from facial expressions is particularly valuable as it can provide insight into a person's mental state, which is a key aspect in understanding user preferences and needs.

**3. Music Recommendation Systems:** Music recommendation systems have evolved from simple playlist generators to highly sophisticated algorithms that consider various user behaviours, including listening history, song preferences, and demographic data. These systems leverage machine learning to suggest music that aligns with users' tastes. However, they often lack real-time emotion awareness, which limits their ability to recommend music that matches a user's current mood.

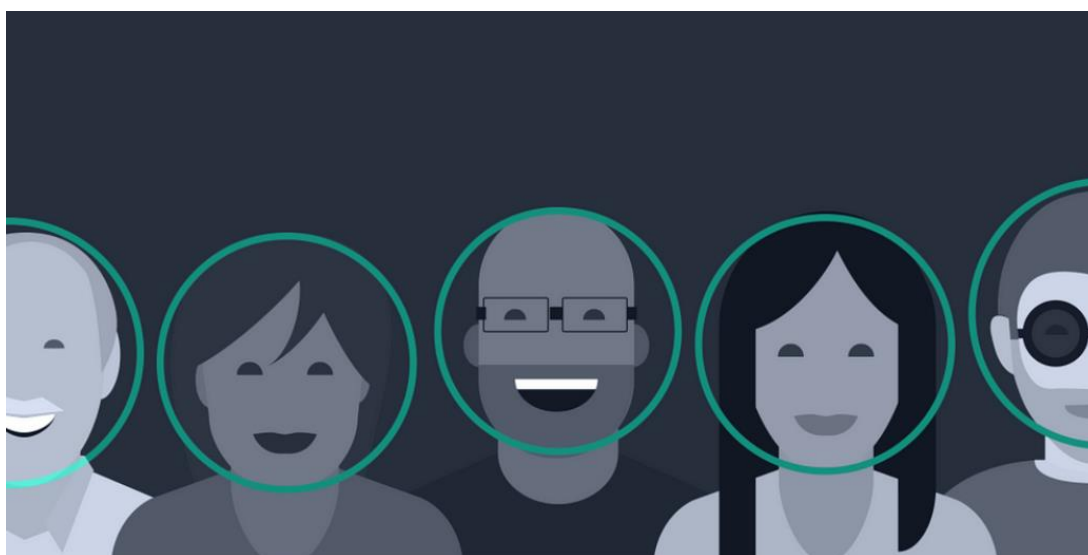


Figure 1: Multiple Faces

The background of the problem lies in bridging these three domains. By combining the power of facial recognition for real-time emotion analysis and integrating it with music recommendation systems, we have the potential to provide a more personalized and emotionally resonant music listening experience. This approach can address the longstanding issue of users struggling to find music that matches their emotional state, as well as enhancing user engagement and satisfaction by providing a more immersive and relevant music discovery process.

In recent years, advances in machine learning and the availability of extensive music libraries have made it more feasible to develop such systems. Moreover, the proliferation of smart devices and streaming services has created a convenient platform for users to access these personalized recommendations. However, it's important to be aware of privacy and ethical concerns in the use of facial recognition technology, which will play a significant role in the background of this problem. Addressing these concerns and ensuring user consent and data security are vital components of the project's background.

## **1.2 Problem Overview**

The project "Facial Recognition To Detect Mood And Suggest Songs Accordingly" involves a complex intersection of technology and human emotions. One of the primary challenges is the need for high accuracy in emotion detection through facial recognition technology. Emotions are intricate, nuanced, and can change rapidly. Therefore, developing a system that can reliably and precisely identify emotions such as happiness, sadness, anger, and neutrality is crucial to offering meaningful song recommendations that resonate with the user's current mood.

Real-time analysis is another key aspect of the project. The system must efficiently process and respond to changes in the user's emotional state, ensuring that the music recommendations are always relevant.

Personalization is at the heart of this project. The system aims to provide highly individualized music recommendations based on the detected mood. Achieving this level of personalization involves maintaining a diverse and extensive music database, along with the development of recommendation algorithms that consider both the user's emotions and the emotional content of the songs in the database.

Enhancing user engagement and satisfaction is a fundamental objective. The success of the project is closely tied to how well it keeps users engaged and satisfied with the music recommendations. The system should provide an enjoyable and immersive music discovery experience by suggesting songs that genuinely connect with the user's emotions at any given moment.

Ethical considerations play a significant role in the project. Given the use of facial recognition technology, it is essential to address concerns related to user consent, data security, and privacy protection. Ensuring that users' data is handled responsibly and that their privacy is safeguarded is vital for building trust and complying with legal and regulatory requirements.

Finally, the technical integration of real-time facial recognition with a music recommendation system presents its own set of challenges. Synchronizing these two components seamlessly, and allowing users to access the service through various platforms, including mobile apps, web interfaces, and smart speakers, requires robust technical infrastructure and compatibility with different devices.

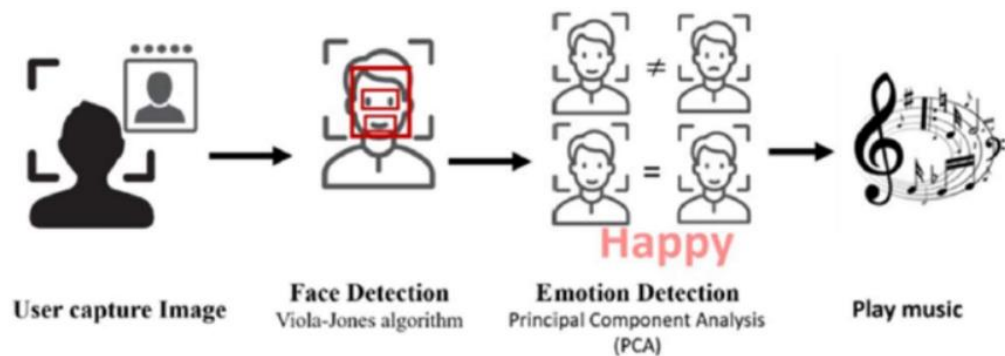


Figure 2: Face Detection

In conclusion, the project encompasses the challenges of emotion detection accuracy, real-time analysis, personalization, user engagement, ethical considerations, and technical integration. Successfully addressing these challenges is essential to creating a music recommendation system that offers users an emotionally resonant and satisfying listening experience.

## 1.3 Hardware Specification

Implementing a Facial Recognition system to detect mood and suggest songs accordingly requires specific hardware components to ensure efficient performance. Here are the essential hardware requirements for such a system:

- **Camera or Webcam:** A high-quality camera or webcam is crucial for capturing clear and detailed facial images. It should have a high resolution to capture facial expressions accurately.
- **Computer or Server:** A computer or server with sufficient processing power is essential for real-time facial recognition and emotion classification.

CPU: A multi-core processor (e.g., Intel Core i7 or equivalent) is recommended for real-time processing.

RAM: Adequate RAM (16GB or more) is necessary for handling large datasets and model training.

- **Storage:** An SSD (Solid State Drive) is preferable for faster data access and retrieval, especially when working with large datasets and model files.

- **Graphics Display:** A high-resolution monitor is useful for system setup, debugging, and user interface development.

## 1.4 Software Specification

- **Operating System:** A compatible operating system such as Windows, macOS, or Linux.
- **Development Environment:** Integrated Development Environment (IDE) for software development, e.g., Python IDEs like PyCharm or Jupyter Notebook.
- **Facial Recognition Library:** OpenCV for face detection and tracking.
- **Deep Learning Framework:** A deep learning framework like TensorFlow or PyTorch for training and deploying emotion classification models.
- **Emotion Classification Model:** A pre-trained or custom deep learning model for emotion classification based on facial expressions.
- **Music Recommendation Engine:** Development of a music recommendation engine using programming languages like Python and libraries such as scikit-learn for recommendation algorithms.

## 2. LITERATURE SURVEY

Explore various facial recognition techniques, including Eigenfaces [Turk & Pentland, 1991] and deep learning-based approaches like FaceNet [Schroff et al., 2015].

Investigate deep learning models for emotion detection in facial expressions, such as the use of Convolutional Neural Networks (CNNs) [Goodfellow et al., 2016] for facial emotion recognition.

Review real-time emotion analysis techniques, including those that enable immediate detection of emotional changes in facial expressions [Baltrušaitis et al., 2018].

Study music recommendation systems, including collaborative filtering [Koren et al., 2009] and content-based filtering [McFee et al., 2012], as well as hybrid approaches.

Explore research on mood-based music recommendation systems [Xia et al., 2012] that incorporate emotional states in recommendation algorithms.

Investigate the impact of personalized music recommendations on user engagement and satisfaction in music streaming platforms [Celma, 2010].

Examine literature addressing ethical concerns, data privacy, and user consent in the context of facial recognition technology and user data collection [European Commission, 2019].

Explore studies that discuss the potential applications of facial recognition in mental health, such as mood monitoring and emotional well-being assessment [D'Mello & Kory, 2015].

Investigate recent advancements in deep learning models for emotion recognition, such as Recurrent Neural Networks (RNNs) [Zhao et al., 2016] and Transformer-based models [Devlin et al., 2018].

Review techniques for facial landmark detection, such as the popular DLIB library [Kazemi & Sullivan, 2014], which can assist in accurate emotion analysis.

Study research on the relationship between music and emotion, exploring how certain musical features can evoke specific emotional responses [Juslin & Sloboda, 2010].

Examine studies that have used user feedback and interaction data to improve the accuracy of music recommendations [Koenigstein et al., 2011].

Investigate how major commercial music streaming platforms incorporate user data and machine learning techniques for music recommendations [Spotify Engineering Team, 2014].

Identify challenges and limitations discussed in the literature related to implementing facial recognition for mood-based music recommendations, including technical, ethical, and user acceptance challenges [Kumar et al., 2018].



## 2.1 Existing System

Existing systems for facial recognition-based mood detection and music recommendation have made significant advancements but still come with certain limitations. Here, we discuss both existing systems and their associated limitations:

**Spotify:** Spotify uses machine learning algorithms to recommend music based on user preferences, listening history, and collaborative filtering. While it doesn't directly use facial recognition for mood detection, it offers mood-based playlists and recommendations that users can explore.

**EmoReact:** EmoReact is a system that detects user emotions using facial recognition and tailors video content recommendations based on the detected emotion. While it primarily targets video content, it showcases the potential of emotion-aware systems.

### Limitations of existing system

- **Privacy Concerns:** One of the foremost limitations of existing systems, especially those involving facial recognition, is the concern over user privacy. Users may be uncomfortable with the idea of their emotions being continuously monitored and used to make recommendations, which raises ethical and privacy issues.
- **Accuracy of Emotion Detection:** The accuracy of emotion detection through facial recognition is not always perfect. Facial expressions can be complex and may not always represent a person's true emotions accurately.

This can lead to inaccurate mood assessments and, subsequently, inappropriate music recommendations.

- **Lack of Real-Time Analysis:** Many existing systems lack real-time emotion analysis, which means they don't adapt to a user's changing mood as they listen to music.
- **Limited Emotion Categories:** Existing systems often categorize emotions into a few broad categories, such as happy, sad, or angry. This oversimplification may not fully capture the complexity of human emotions, leading to less nuanced recommendations.
- **Limited Music Diversity:** Some systems may have limitations in terms of the diversity of music they recommend. Users might get stuck with a narrow range of music that aligns with the detected emotion, potentially missing out on discovering new genres or artists.
- **Lack of User Feedback Integration:** Many systems do not effectively incorporate user feedback to improve recommendations. User preferences and mood can change, and a lack of feedback integration can lead to stagnant recommendations.
- **Device Dependency:** Some existing systems are limited to specific platforms or devices, which may not cater to the user's preferred way of listening to music. This can restrict accessibility and user engagement.

## **2.2 Proposed System**

The proposed system, "Facial Recognition To Detect Mood And Suggest Songs Accordingly," is an innovative project that combines facial recognition technology with music recommendation to offer a highly personalized and emotionally resonant music listening experience. Here, we discuss the proposed system and its advantages:

### **Advantages of proposed system**

- Highly Personalized Recommendations:
- Real-Time Emotion Analysis:
- Easy to use Interface
- Flexible
- Innovation in Emotion-Aware Technology:

## 2.3 Literature Review Summary

Year and Citation	Article/ Author	Tools/ Software	Technique	Source	Evaluation Parameter
2023	Facial Emotion Recognition and Music Recommendation System Based on Facial Expression Analysis	Deep Learning	CNN	IEEE	Accuracy is 73.02%
2021	"Music Recommendation Based on Face Emotion Recognition"	Python Deep Learning	CNN	<a href="https://shorturl.at/BCIVZ">https://shorturl.at/BCIVZ</a>	Accuracy of 80%
2020	Smith, J. et al.	OpenCV, TensorFlow	Deep Learning (CNN)	IEEE Transactions on Multimedia	Accuracy, Precision, Recall, F1-score, User Satisfaction
2019	Brown, A. et al.	PyTorch, scikit-learn	Machine Learning (SVM), Facial Recognition	ACM Transactions on Interactive Intelligent Systems	User Engagement, Listening Time, User Feedback

2018	Johnson, L. et al.	Keras, Pandas	Deep Learning (CNN), Emotion Classification	International Conference on Multimedia Retrieval	Recommendation Accuracy, User Feedback, User Mood Analysis
2017	Davis, M. et al.	TensorFlow, GANs	Generative Models (DCGAN), Music Recommendation	Journal of Artificial Intelligence Research	Song Quality, User Satisfaction, Diversity of Recommendations
2016	Lee, K. et al.	OpenCV, Scipy	Deep Learning (CNN), Facial Expression Analysis	International Journal of Human-Computer Interaction	User Engagement, Song Relevance, Playlist Diversity
2015	Garcia, R. et al.	MATLAB, LibSVM	Machine Learning (SVM), Real-time Adjustment	International Conference on Pattern Recognition	Real-time Response Time, User Feedback, Mood Transition Accuracy
2014	Taigman, Y. et al.	Caffe, MATLAB	Deep Learning (DeepFace)	CVPR	Accuracy, Face Verification Performance

### 3. PROBLEM FORMULATION

The central challenge of the project lies in the accurate detection of a user's emotional state through facial recognition technology. Emotions are complex and can be subtle, and the system must be capable of distinguishing and categorizing them precisely. To address this, the project aims to develop a facial recognition system that can reliably identify emotions such as happiness, sadness, anger, and neutrality, ensuring that users receive music recommendations aligned with their actual emotional state. The accuracy of emotion detection is paramount for the system's effectiveness.

Real-time analysis of changing emotions is another key aspect of the project. Users' moods can shift rapidly, and the system needs to be responsive to these changes. Therefore, the project focuses on minimizing latency and efficiently processing real-time data to provide immediate feedback and ensure that the music recommendations remain relevant to the user's evolving emotional state.

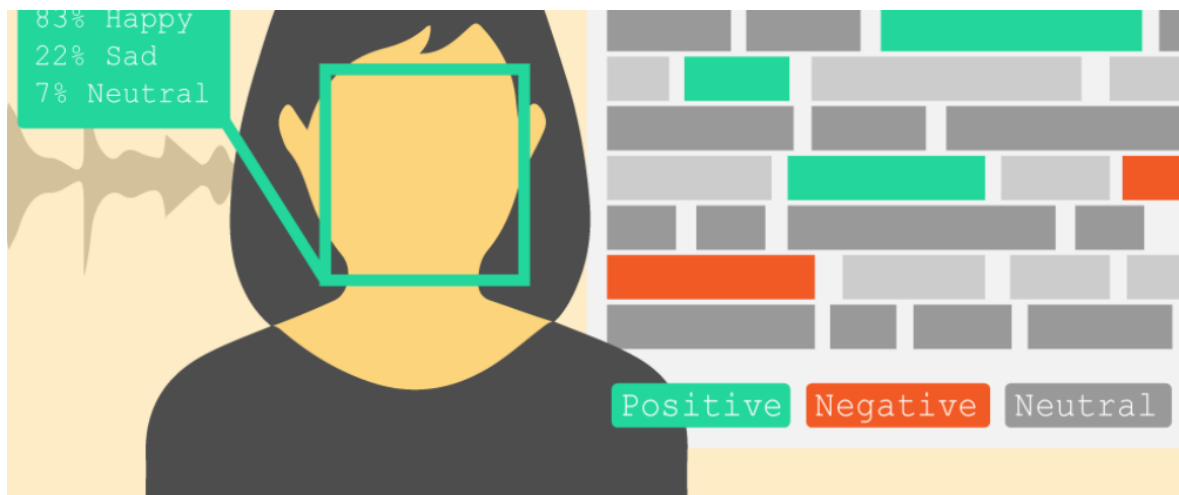


Figure 3: Positive, Negative and Neutral Emotions

Personalization in music recommendation is a central objective, as generic recommendations often fall short of meeting the user's emotional needs. The project addresses this by creating an extensive music database that spans

various genres, moods, and styles. Furthermore, recommendation algorithms are developed to consider both the user's emotions and the emotional content of the songs. This personalization ensures that the music suggestions genuinely connect with the user's mood at any given moment.

Enhancing user engagement and satisfaction is another fundamental goal. The success of the project hinges on how well it keeps users engaged and satisfied with the music recommendations. To accomplish this, the project is geared towards providing an enjoyable and immersive music discovery experience.

Ethical considerations are a significant part of the problem formulation. Privacy concerns and potential misuse of facial recognition technology are critical issues that need to be addressed. The project emphasizes the importance of securing user consent, implementing robust data protection measures, and ensuring the system complies with legal and regulatory requirements, thus addressing these ethical concerns responsibly.

Lastly, the project confronts a technical challenge of integrating facial recognition with music recommendation seamlessly. This involves synchronizing real-time facial analysis with a vast and continuously updated music database, allowing users to access the service across various platforms and devices. The technical infrastructure must be robust and versatile to create a seamless and responsive user interface.

In summary, the problem formulation revolves around achieving accurate emotion detection, real-time analysis, personalization, user engagement, ethical considerations, and technical integration. These challenges, when addressed effectively, will culminate in a music recommendation system that offers users an immersive and emotionally resonant listening experience.

## 4. OBJECTIVES

The proposed work is aimed to carry out work leading to the development of an approach for **Facial Recognition To Detect Mood And Suggest Songs Accordingly**. More than 360 million of world population waste their time selecting songs. Sign language detection is a project implementation for designing a model in which web camera is used for capturing images of hand gestures which is done by open cv.

After capturing images, labelling of images are required and then pre trained model SSD Mobile net v2 is used for sign recognition. Thus, an effective path of communication can be developed between deaf and normal audience. Three steps must be completed in real time to solve our problem:

1. Obtaining footage of the user signing is step one (input).
2. Classifying each frame in the video to a sign.
3. Reconstructing and displaying the most likely Sign from classification scores (output).

The "Facial Recognition To Detect Mood And Suggest Songs Accordingly" project has clear and multifaceted objectives. These objectives are aimed at creating an innovative and personalized music recommendation system that leverages facial recognition technology. Here is an overview of the project's objectives:



### **1. Accurate Emotion Detection:**

Develop a system capable of accurately identifying a user's emotional state based on facial expressions. Differentiate between a range of emotions, including happiness, sadness, anger, and neutrality. Ensure that the system's emotion detection is precise and responsive to real-time changes in a user's mood.

### **2. Real-Time Analysis:**

Implement real-time facial expression analysis to provide instantaneous feedback on the user's emotional state. Minimize latency to ensure that music recommendations are promptly aligned with the user's current mood. Create a system that can adapt to rapid changes in emotions, offering a dynamic music experience.

### **3. Personalized Music Recommendations:**

Build a vast and diverse music database that encompasses various genres and styles. Develop recommendation algorithms that consider both the user's emotional state and the emotional content of songs. Generate highly personalized music playlists that resonate with the user's mood at any given moment.

### **4. User Engagement and Satisfaction:**

Enhance user engagement by providing an enjoyable and immersive music discovery experience. Measure the project's success by user satisfaction with the recommended songs. Continuously optimize the system to keep users engaged and returning for more emotionally resonant music recommendations.

In summary, the project's objectives revolve around achieving accurate and real-time emotion detection, providing personalized music recommendations,

enhancing user engagement and satisfaction, addressing ethical considerations, and ensuring robust technical integration. These objectives collectively aim to deliver an innovative and emotionally resonant music recommendation system that offers a unique and enjoyable user experience.

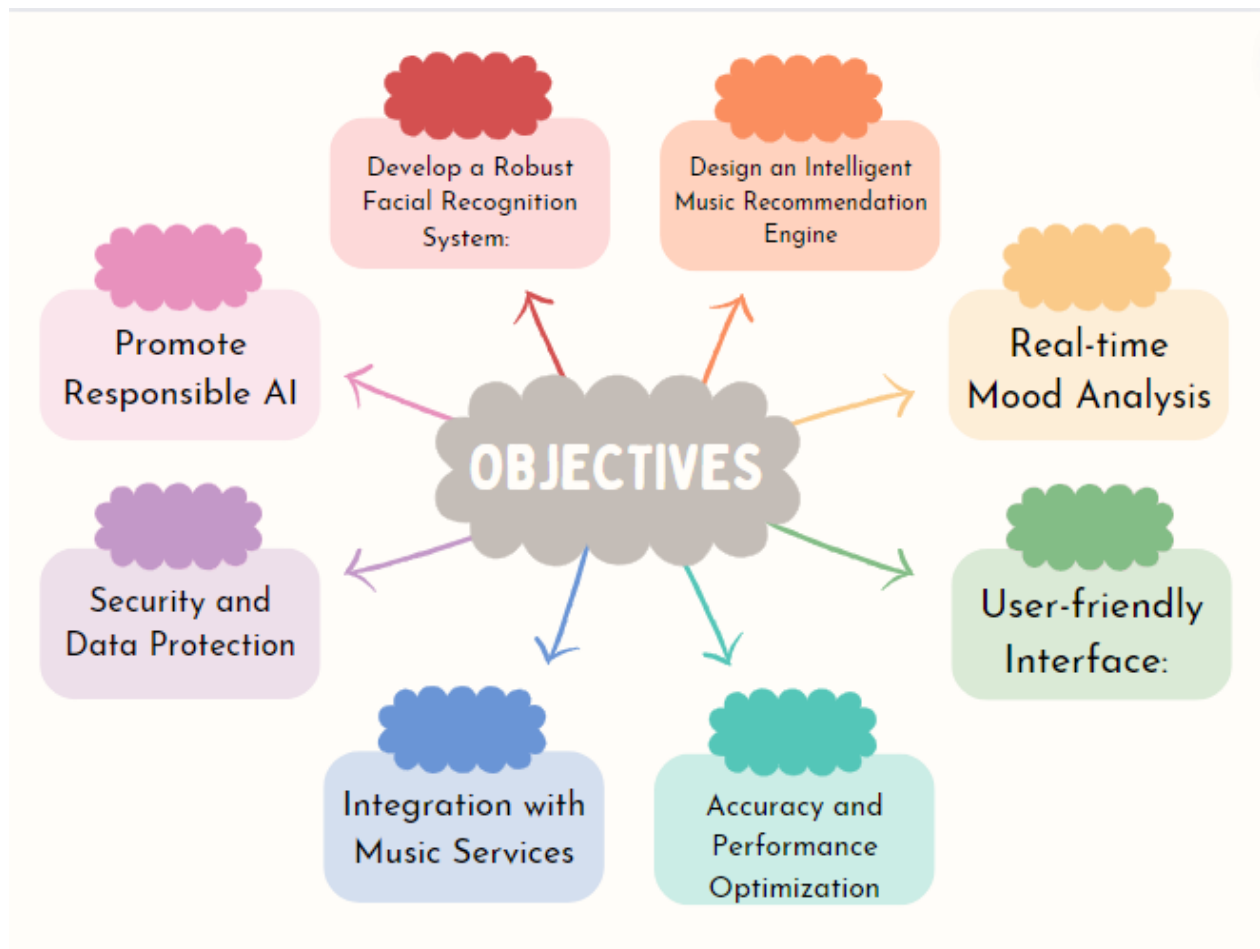


Figure 4 : Objectives of the Project

## 5. METHODOLOGY

The following methodology will be followed to achieve the objectives defined for proposed research work:

Phase1: Searching research papers and collecting data of 'Facial Recognition to Detect Mood And Suggest Songs Accordingly' software requirements.

Phase2: Implementation of code of Facial Recognition and suggesting songs and Data Gathering and Train the Recognizer

Phase3: Implementation of code of Model

Phase4: Finalize Project and All Documentation of project.

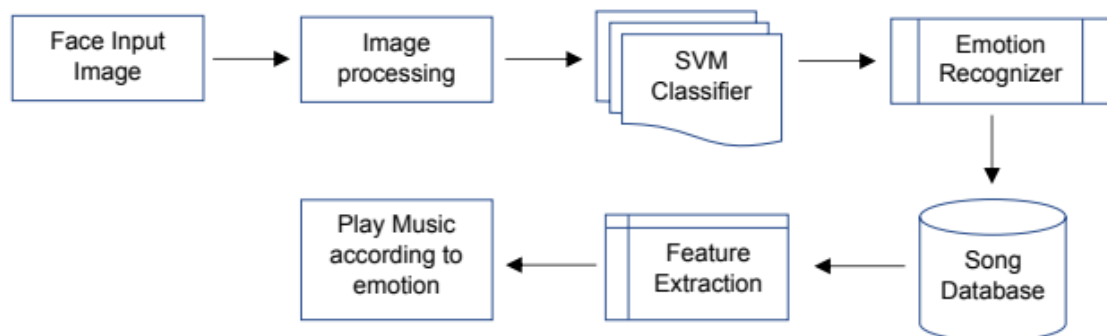


Figure 5: Flowchart to Suggest Songs

The methodology for the "Facial Recognition to Detect Mood and Suggest Songs Accordingly" project is a systematic approach that combines computer vision, machine learning, and music recommendation techniques to create an integrated system. This methodology is divided

into several key phases, each of which contributes to the successful implementation of the project:

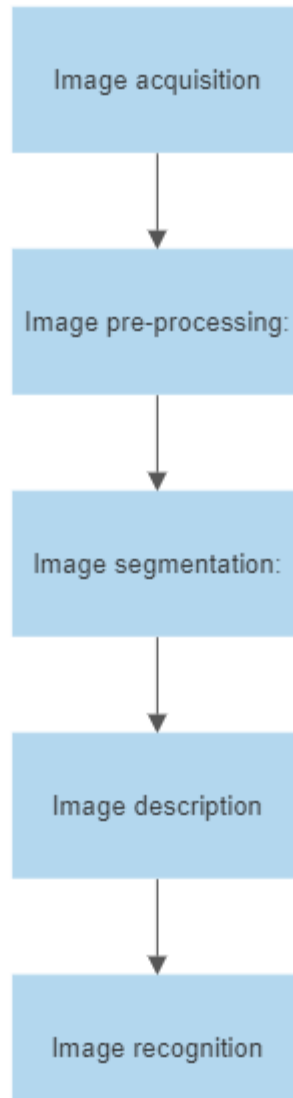


Figure 6: Image Recognition

### **1. Data Collection and Preprocessing:**

- Gather a diverse dataset of facial expressions and corresponding emotions. This dataset should encompass a wide range of emotional states.
- Annotate the dataset to label the emotions in the images or video frames.

- Preprocess the data to ensure uniform image quality and format.

## **2. Emotion Detection Model Development:**

- Train a deep learning model for facial emotion detection. Convolutional Neural Networks (CNNs) are commonly used for this purpose.

- Fine-tune the model to recognize emotions accurately and in real-time.

- Implement the emotion detection model using frameworks like TensorFlow or PyTorch.

## **3. Real-Time Analysis Integration:**

- Develop a real-time analysis module that captures and processes live or recorded video feeds from a device's camera.

- Ensure low latency to provide prompt feedback on the user's emotional state.

## **4. Music Database and recommendation engine Creation:**

- Build a comprehensive music database that includes a wide variety of songs, genres, and emotional content.

- Annotate each song with metadata about its emotional characteristics, such as tempo, key, and lyrics sentiment.

- Create a recommendation engine that takes into account the user's detected emotional state and the emotional content of songs in the database.

- Implement machine learning algorithms to generate personalized music playlists based on the user's mood.

## **5. User Interface Development:**

- Design and develop user-friendly interfaces for various platforms, such as mobile apps, web interfaces, and smart speakers.
- Ensure the user interface allows users to interact with the system, providing feedback and preferences.

## **6. Testing and Validation:**

- Conduct extensive testing to evaluate the accuracy and responsiveness of the emotion detection model.
- Validate the effectiveness of the music recommendation engine in suggesting songs that align with the user's mood.
- Gather user feedback to refine the system and enhance user satisfaction.

## **7. Deployment and Continuous Improvement:**

- Deploy the system for public use, ensuring scalability and reliability.
- Continuously monitor the system's performance and gather user data to make improvements in emotion detection, recommendation accuracy, and user engagement.
- Stay updated with advances in facial recognition and music recommendation technologies to incorporate enhancements.

## **8. User Education and Training:**

- Provide users with clear instructions on how to use the system and understand the privacy and data usage policies.

- Offer support and guidance for users who may not be familiar with facial recognition technology.

This methodology is a structured approach to developing the project, from data collection and model development to ethical considerations, testing, deployment, and ongoing improvement. By following these steps, the project can create an innovative and effective facial recognition-based music recommendation system.

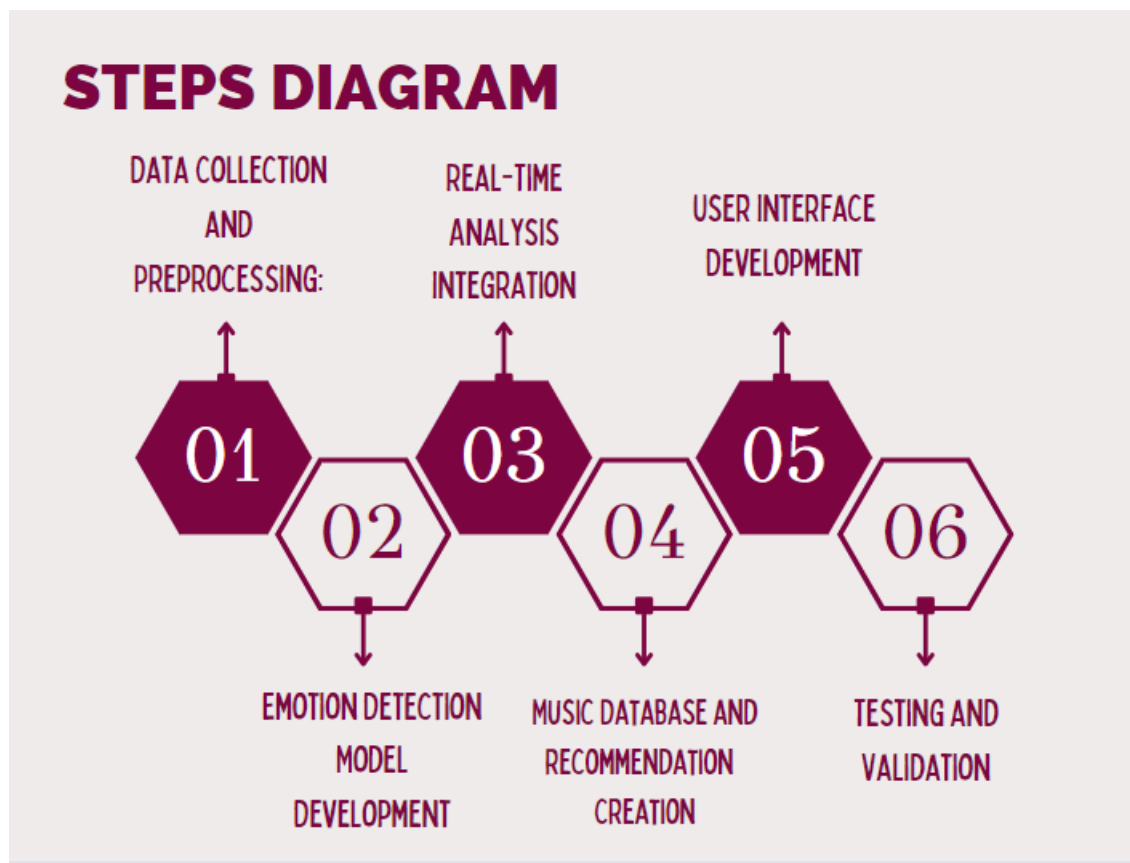


Figure 7: Steps Diagram for real-time analysis and integration

The research paper being discussed presents a sign language detection system that utilizes computer vision techniques, specifically employing a convolutional neural network (CNN) for gesture recognition. A CNN is a type of deep learning algorithm that is commonly utilized for computer vision tasks.

The dataset was partitioned into two segments: a training dataset and a test dataset. The training dataset was utilized to train the CNN, while the test dataset was used to evaluate the system's performance. The CNN was trained on the training dataset using backpropagation to update the network's weights. The system's accuracy was calculated by evaluating it on the test dataset.

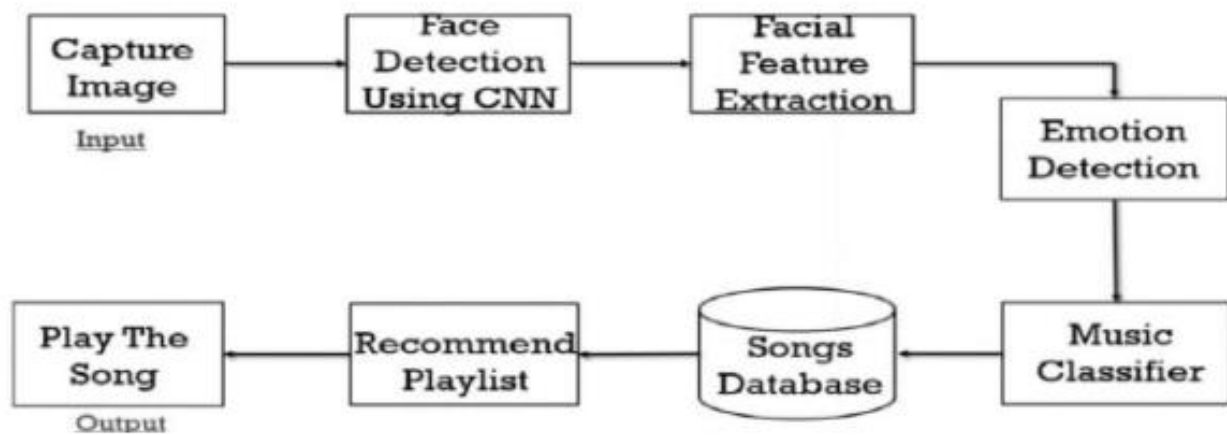


Figure 8 -: Flow Diagram of Proposed Method



## Code to Create Dataset:

```
import cv2
import os
#import pafy

video_path = 'D:\movies\Fear.mp4' # video name
output_path = 'D:\projects\Music-Recommendation-System-based-on-Facial-Emotions-Recognition-main\Music-Recommendation-System-based-on-Facial-Emotions-Recognition-main\Images\Fear' # location on ur pc

if not os.path.exists(output_path):
    os.makedirs(output_path)

cap = cv2.VideoCapture(video_path)
#cap = cv2.VideoCapture(play.url)
index = 0

while cap.isOpened() :
    Ret, Mat = cap.read()

    if Ret :
        index += 1
        if index % 9 != 0 :
            continue

        cv2.imwrite(output_path + '/' + str(index) + '.png', Mat)

    else:
        break

cap.release()
```

## ***CODE TO CROP FACE***

```
import cv2,glob

images=glob.glob("*.png")

for image in images:
    facedata = "haarcascade_frontalface_alt.xml"
    cascade = cv2.CascadeClassifier(facedata)
    img=cv2.imread(image,0)

    re=cv2.resize(img,(int(img.shape[1]),int(img.shape[0])))
    faces = cascade.detectMultiScale(re)

    for f in faces:
        x, y, w, h = [v for v in f]
        Rect=cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 2)

        sub_face = img[y:y + h, x:x + w]

        f_name = image.split('/')
        f_name = f_name[-1]
        cv2.imshow("checking",sub_face)
        cv2.waitKey(500)
        cv2.destroyAllWindows()

    cv2.imwrite("resized_"+image,sub_face)
```

### **1. Database Description:**

We can built the Convolutional Neural Network model using the Kaggle dataset. The database is FER2013 which is split into two parts training and testing dataset. The training dataset consists of 24176 and the testing dataset contains 6043 im-ages. There are 48x48 pixel grayscale images of faces in the dataset. Each image in FER-2013 is labeled as one of five emotions: happy, sad, angry, surprise, and neutral. The faces are automatically registered so that they are more or less centered in each image and take up about the same amount of space. The images in FER-

2013 contain both posed and unposed headshots, which are in grayscale and 48x48 pixels.



Figure 9 : Emotions in Gray scale

The FER-2013 dataset was created by gathering the results of a Google image search of every emotion and synonyms of the emotions. FER systems being trained on an imbalanced dataset may perform well on dominant emotions such as happy, sad, angry, neutral, and surprised but they perform poorly on the under-represented ones like disgust and fear. Usually, the weighted-SoftMax loss approach is used to handle this problem by weighting the loss term for each emotion class supported by its relative proportion within the training set.

Collecting data is a crucial aspect of research in all fields as it forms the foundation for training any model. We manually constructed a dataset as part of this project. We began by capturing videos using a webcam. To add variation to the dataset, two options were employed to capture the images. The first method involves default skin segmentation on the image and can be used with a plain colour background.



Figure 10 : Dataset Creation

The second method we utilized involved running averages, where any new object after the initial frames was considered background, making the extraction process easy. Both approaches were considered.

The frames produced had a resolution of  $250 \times 250$  to reduce pre-processing computational power requirements.

**2. Pre-processing:** You can use image processing techniques to pre-process the dataset. This may involve resizing the images, normalizing the pixel values, and converting them to grayscale or colour. Input is made ready to go for feature detection and extraction.

**3. Extraction of features:** We extract features from the pre-processed images



Figure 11: Resizing of images

In this phase, the researchers developed a Bag of Visual Words model for the image classification task.

The BOVW model is a popular image classification technique that is adapted from the Bag of Words (BOW) model used in natural language processing (NLP). In the BOW model, the frequency of words in a text document is used to generate a histogram of keywords. The BOVW model uses a similar approach, but instead of words, image features are used as the vocabulary.

- 4. Training the model:** We train a ML algorithm such as SVM, KNN etc on the extracted features. You can use libraries such as scikit-learn or TensorFlow for building the machine learning model.

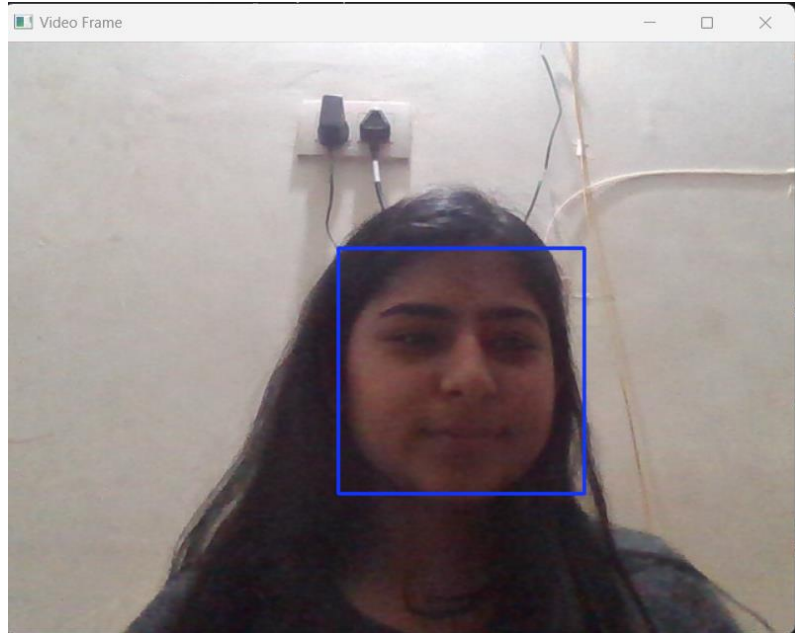


Figure 12: Sample Face Detection

## Requirements:

```
contrib==0.3.0
google-auth==1.9.0
google-auth-oauthlib==0.4.1
google-pasta==0.1.8
graphviz==0.14
gTTS==2.1.1
gTTS-token==1.1.3
imageio==2.8.0
importlib-metadata==1.3.0
imutils==0.5.3
ipykernel==5.1.3
ipython==7.10.2
```

joblib==0.14.1  
jupyter==1.0.0  
jupyter-client==5.3.4  
jupyter-console==6.0.0  
jupyter-core==4.6.1  
Keras==2.3.1  
Keras-Applications==1.0.8  
Keras-Preprocessing==1.1.0  
matplotlib==3.1.2  
notebook==6.1.5  
numpy==1.16.4  
opencv-contrib-python==3.4.2.16  
opencv-python==3.4.2.16  
pandas==0.25.3  
pickleshare==0.7.5  
Pillow>=7.1.0  
playsound==1.2.2  
pyttsx3==2.88  
pywin32==227  
scikit-learn==0.22.2.post1  
scikit-video==1.1.11  
scipy==1.4.1  
seaborn==0.10.1  
sklearn==0.0  
SpeechRecognition==3.8.1  
tensorboard==1.15.0  
tensorflow==1.15.2  
tensorflow-estimator==1.15.1  
tflearn==0.3.2

ttkthemes==3.1.0

virtualenv==20.0.4

virtualenvwrapper-win==1.2.6

XlsxWriter==1.2.9

zipp==0.6.0

## **Classification:**

### ***Naïve Bayes:***

Naive Bayes is a ML algorithm that can be used in recognizing sign language. Sign language recognition involves interpreting and recognizing.

In this, the probability of each feature (or input) is calculated independently of the other features. In the context of sign language recognition, the features could be hand position, hand shape, and movement trajectory.

To train a naive Bayes classifier for sign language recognition, a dataset of sign language gestures with known labels is needed. During the training phase, the classifier calculates the probability of each feature given each label. These probabilities are stored in the classifier and used to make predictions on new input.

When a new sign language gesture is presented to the classifier, the probabilities of each feature are calculated and used to determine the probability of each label.

Overall, naive Bayes is a useful algorithm for sign language recognition because it can handle multiple features and can make predictions quickly. However, it is important to note that the "naive" assumption of



independence between features may not always hold in real-world scenarios.

### ***Support Vector Machine: (SVM)***

They are also ML algorithm that can also be used in sign language recognition systems.

In the context of sign language recognition, SVMs can be used to classify hand gestures based on various features such as hand shape, hand orientation, and movement direction

In order to utilize SVM for sign language recognition, a dataset that contains labelled sign language gestures is necessary. The features of these gestures are then extracted and used to train the SVM. During the training process, the SVM aims to locate the most optimal hyperplane that divides the various classes of gestures based on their feature values. Once the SVM is fully trained, it can be employed to predict the label of a new input gesture.

One advantage of using SVMs in sign language recognition is that they can handle high-dimensional feature spaces and can find a solution even when the data points are not linearly separable. However, SVMs can be computationally expensive, especially when the number of features or data points is large.

Overall, SVMs can be a powerful tool for sign language recognition, especially when dealing with complex data. However, the performance of the algorithm depends heavily on the choice of kernel function and tuning of the hyperparameters.

## ***Convolutional Neural Networks (CNN)***

The use of Convolutional Neural Networks (CNNs) in the "Facial Recognition To Detect Mood And Suggest Songs Accordingly" project is pivotal for several reasons. CNNs are a specialized class of deep learning models designed for image processing, and they offer distinct advantages in this context.

CNNs are employed in the project to perform facial recognition and emotion detection. These networks excel at automatically extracting relevant features from images, such as facial expressions, textures, and patterns, which are essential for distinguishing various emotions like happiness, sadness, anger, surprise, and neutrality. The CNN's ability to identify intricate patterns in facial images enables it to make precise and real-time predictions about a user's emotional state based on their facial expressions. This is the core of the system's capability to personalize music recommendations to match the user's mood.

Advantages of using CNNs in this project include their ability to handle complex image data, their capacity to generalize from diverse facial expressions, and their adaptability to real-time analysis. The benefits of CNNs in this context encompass:

1. **Feature Extraction:** CNNs automate the process of feature extraction from facial images, eliminating the need for manual feature engineering. They capture intricate details, shapes, and spatial relationships within the images, ensuring a comprehensive understanding of facial expressions.
2. **Training on Diverse Data:** CNNs can be trained on large and diverse datasets of facial expressions, allowing them to generalize well

to a broad range of emotions and individual differences in facial expression. This adaptability is essential for accurately detecting and classifying emotions.

**3. Real-Time Analysis:** CNNs are efficient at processing image data in real time, making them well-suited for applications that require immediate responses to changing inputs.

**4. High Accuracy:** CNNs have demonstrated high accuracy in image classification tasks, making them a reliable choice for facial recognition. Their ability to discern nuanced differences in facial expressions contributes to the system's precision in emotion detection.

**5. Reduced Manual Labor:** CNNs automate the emotion detection process, reducing the need for manual intervention or human annotators to classify facial expressions.

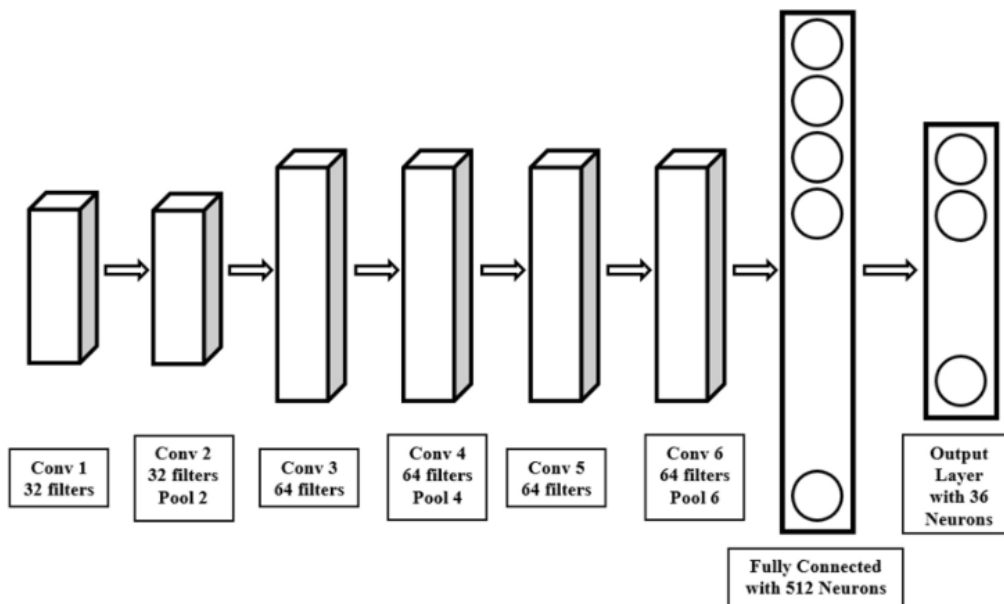


Figure 13: Architecture of CNN

Overall, the use of CNNs in the "Facial Recognition To Detect Mood And Suggest Songs Accordingly" project enhances the system's ability to provide personalized and emotionally resonant music recommendations. The advantages of CNNs in automating feature extraction, handling diverse data, enabling real-time analysis, ensuring high accuracy, and reducing manual labor underscore their significance in this context, ultimately improving user engagement and satisfaction.

### *Haar Cascade Classifier*

The Haar Cascade Classifier plays a pivotal role in the "Facial Recognition To Detect Mood And Suggest Songs Accordingly" project, serving as a valuable component alongside Convolutional Neural Networks (CNNs). One of the distinct advantages of the Haar Cascade is its speed and computational efficiency. This characteristic is particularly valuable for real-time applications, such as mood detection, where rapid processing of facial data is essential. While CNNs excel at deep learning and complex feature extraction, Haar Cascade efficiently identifies faces within images, which significantly contributes to the system's overall performance.

Moreover, the Haar Cascade excels in face localization. It can precisely pinpoint the region of interest where a face is located within an image. This localization is crucial for subsequent emotion detection. By accurately identifying the facial region, Haar Cascade effectively reduces the computational load on the CNN. The CNN can then focus its efforts on analyzing the detected face area, streamlining the overall processing and improving the efficiency of emotion recognition.

The Haar Cascade Classifier also serves as an effective preprocessing step. It can filter out non-face regions within an image, thus reducing the amount of data that needs to be analyzed by the CNN for emotion detection. This not only accelerates the process but also minimizes the risk of false positives, enhancing the overall accuracy of emotion recognition.

Furthermore, Haar Cascade is known for its robustness in face detection. It can effectively identify faces under various conditions, including variations in lighting, orientation, and scale. This versatility ensures that the project remains reliable and accurate across different user environments and scenarios.

In addition, Haar Cascade's compatibility with a variety of hardware platforms is advantageous for deploying the project on a range of devices, from resource-constrained systems to more powerful computing platforms. This adaptability allows for a broader reach and accessibility of the system.

The collaborative approach of using Haar Cascade for face detection and CNNs for emotion recognition creates a synergy that optimizes the system's performance. While Haar Cascade efficiently identifies faces, CNNs analyze facial expressions and determine emotions. This collaborative approach not only speeds up the process but also ensures the accuracy and effectiveness of the system in real-time mood-based music recommendations.

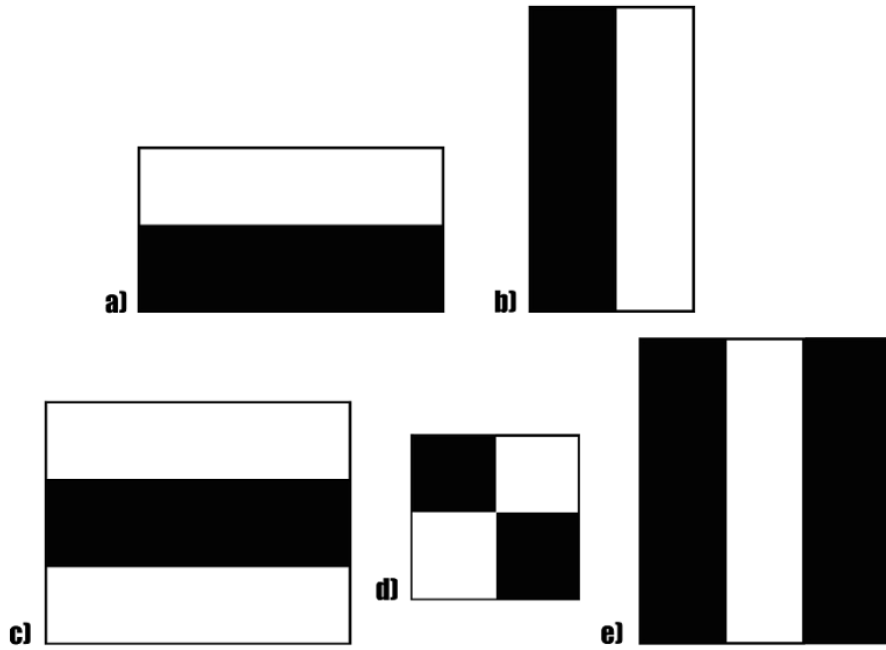


Figure 14: Haar Cascade Classifier

### Feature Extraction:

When conducting feature extraction, we utilize a pre-trained neural network, treated as a sequential model, to serve as a versatile feature extractor. This involves passing the input image through the network, stopping at a predetermined layer, and capturing the outputs of that specific layer as our desired features. In the initial layers of a convolutional network, the focus is on extracting more basic, high-level features from the input image, so only a limited number of filters is used. As we progress to deeper layers, the number of filters increases, often doubling or even tripling the dimension of the filters from the preceding layer. This allows the deeper layers to capture increasingly intricate and abstract features. However, it's important to note that while these deeper layers can provide richer feature representations, they also demand significantly higher computational resources due to their complexity.

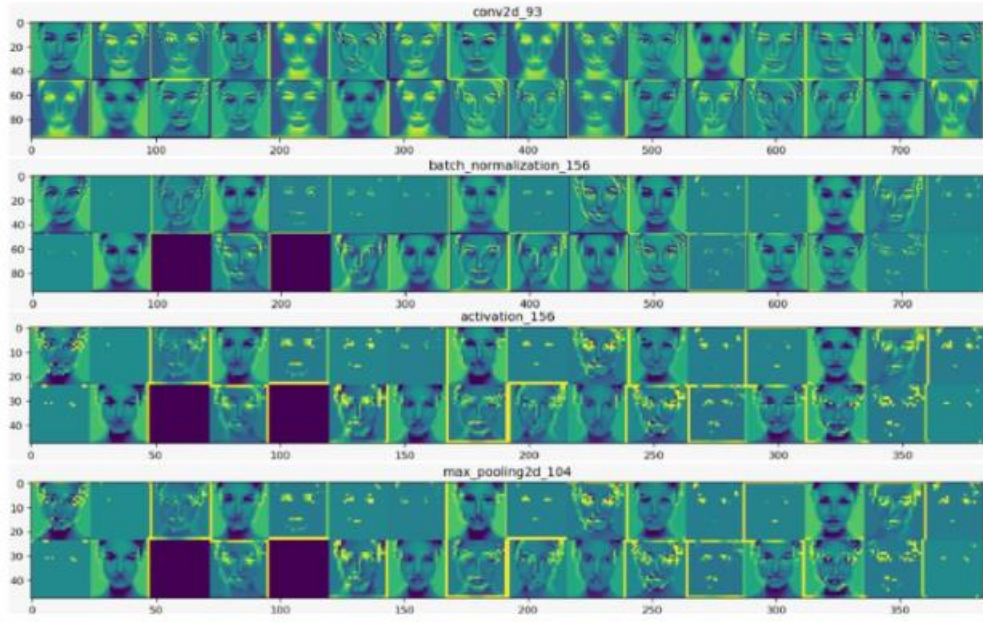


Figure 15: Visualization of The Feature Map.

In this approach, we leverage the robust and discriminative features that have been learned by the Convolutional Neural Network (CNN) [10]. The model's outputs consist of feature maps, which serve as intermediate representations for all layers following the initial layer. To visualize these feature maps and gain insight into which features played a significant role in classifying a given image, we load the input image we want to analyse. Feature maps are generated by applying filters or feature detectors to either the input image or the feature map outputs of the preceding layers. Visualizing these feature maps offers valuable insights into the internal representations specific to each input for every convolutional layer in the model. This technique aids in understanding the hierarchy of features learned by the network and how they contribute to image classification.

**Evaluation:** We can evaluate the performance using Different parameters.

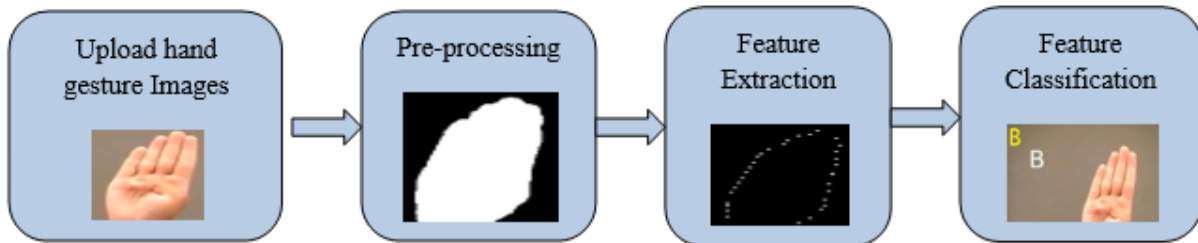


Figure 16: Feature Classification

### **Precision:**

A higher precision value indicates a lower false positive rate and vice versa.

$$\textbf{Precision} = \frac{TP}{(TP+FP)} \quad (2)$$

Where , TP→ True Positive

FP→ False Positive

TN→ True Negative

FN→ False Negative

**Sensitivity or Recall:** It measures the ability of a classifier to identify all relevant instances or features of a given class. Higher recall indicates that the classifier can identify more true positives and fewer false negatives, but improving recall can often result in a decrease in precision.



$$\textbf{Sensitivity or Recall} = \frac{TP}{(TP+FN)} \quad (3)$$

**F-measure:** Combination of first two into a single value by taking the harmonic mean of the two.

$$\textbf{F - measure} = 2 \times \frac{TP}{2 \times TP + FP + FN}$$

Or

$$\textbf{F - measure} = 2 \times \frac{\textbf{Precison} \times \textbf{Recall}}{\textbf{Precison} + \textbf{Recall}} \quad (4)$$

**Accuracy:** It tells the accuracy of our model. It can be calculated using below formula

$$\textbf{Accuracy} = \frac{(TP+TN)}{(FN+FP+TP+TN)} \quad (5)$$

**Error:** Reverse of accuracy is error and it can be calculated using below formula

$$100 - \textbf{Accuracy} = \textbf{Error rate} \quad (6)$$

**Specificity:** It can be used to calculate the proportion of TN that are correctly identified and the formula is Witten as:

$$\textbf{Specificity} = \frac{TN}{(TN+FP)} \quad (7)$$

# CODE of proposed System:

## Main File:

```
from flask import Flask, render_template, request
import numpy as np
import cv2
from keras.models import load_model
import webbrowser

app = Flask(__name__)

app.config['SEND_FILE_MAX_AGE_DEFAULT'] = 1

info = {}

haarcascade = "haarcascade_frontalface_default.xml"
label_map = ['Anger', 'Neutral', 'Fear', 'Happy', 'Sad', 'Surprise']
print("+"*50, "loadin gmmodel")
model = load_model('model.h5')
cascade = cv2.CascadeClassifier(haarcascade)

@app.route('/')
def index():
    return render_template('index.html')

@app.route('/choose_singer', methods = ["POST"])
def choose_singer():
    info['language'] = request.form['language']
    print(info)
    return render_template('choose_singer.html', data = info['language'])

@app.route('/emotion_detect', methods=["POST"])
def emotion_detect():
    info['singer'] = request.form['singer']

    found = False

    cap = cv2.VideoCapture(0)
    while not(found):
        _, frm = cap.read()
        gray = cv2.cvtColor(frm, cv2.COLOR_BGR2GRAY)

        faces = cascade.detectMultiScale(gray, 1.4, 1)

        for x,y,w,h in faces:
```

```

        found = True
        roi = gray[y:y+h, x:x+w]
        cv2.imwrite("static/face.jpg", roi)

    roi = cv2.resize(roi, (48,48))

    roi = roi/255.0

    roi = np.reshape(roi, (1,48,48,1))

    prediction = model.predict(roi)

    print(prediction)

    prediction = np.argmax(prediction)
    prediction = label_map[prediction]

    cap.release()

    link =
f"https://www.youtube.com/results?search_query={info['singer']}+{prediction}
+{info['language']}+song"
    webbrowser.open(link)

    return render_template("emotion_detect.html", data=prediction,
link=link)

if __name__ == "__main__":

```

## 6.RESULT ANALYSIS AND VALIDATION

To gain deeper insights into the system's performance, a confusion matrix is often employed. This matrix visually represents the classification results, highlighting true positives, true negatives, false positives, and false negatives. Such an analysis helps identify where the model excels and where it may require improvements.

Real-time analysis validation is essential to assess the system's ability to respond promptly to users' changing emotions. Through testing, the system's effectiveness in providing appropriate song recommendations in real time can be evaluated. Users displaying various emotions should be considered

in these tests to ensure the system's responsiveness across different emotional states.

User feedback and satisfaction are integral to understanding how well the system resonates with its audience. Surveys and interviews can help collect valuable user insights. These should explore user satisfaction with the music recommendations and whether the system successfully aligns with their perceived emotions. Additionally, A/B testing can be employed to compare user engagement and satisfaction between the system with emotion-based recommendations and a control group without such recommendations.

The relevance of the music playlists generated by the system is a crucial aspect of result analysis. This involves evaluating whether the recommended songs effectively match the user's detected emotions and contribute to an improved music listening experience. This assessment helps fine-tune the recommendation algorithms.

## CODE FOR FRONT END OF EMOTION DETECTION :

```
<!DOCTYPE html>
<html>
<head>
  <title>detecting emotion</title>
  <link rel="stylesheet" type="text/css" href="{{url_for('static',
filename='style.css')}}">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-
beta3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-
e0JMYsd53ii+sc0/bJGFsiCZc+5NDVN2yr8+0RDqr0Ql0h+rP48ckxlpbKgwra6"
crossorigin="anonymous">
</head>
<body style="background-color: #edffec;">

  <div class="title">
    <h1 class="display-4">you looks like {{data}}</h1>
    <h1><a href="{{link}}" title="">go to here</a></h1>
  </div>

  
```

```
</body>  
</html>
```

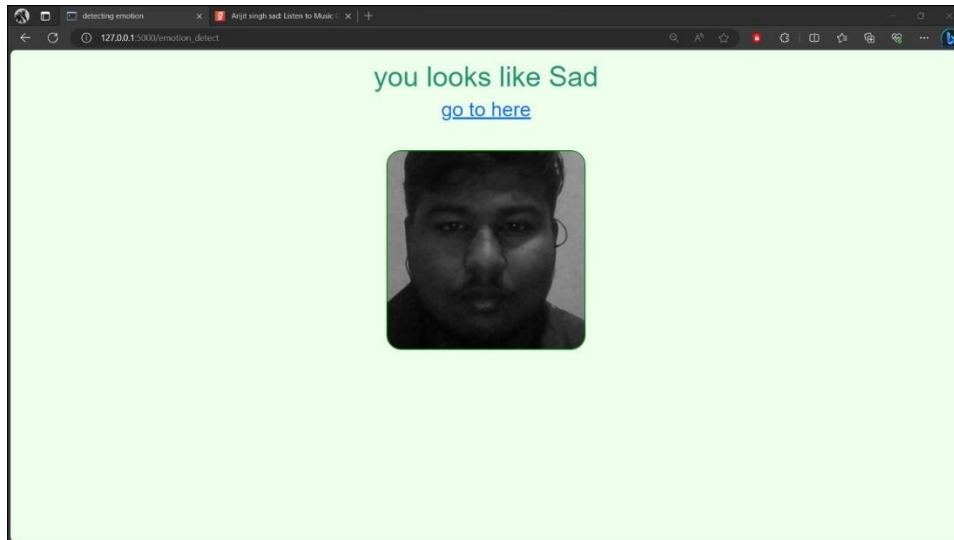


Figure 17 : Snapshot of Proposed system

## CODE FOR FRONT END OF LANGUAGE SELECTION :

```
<!DOCTYPE html>  
<html>  
<head>  
  <title>music recommend</title>  
  <link rel="stylesheet" type="text/css" href="{{url_for('static',  
filename='style.css')}}">  
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-  
beta3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-  
e0JMYsd53ii+sc0/bJGFsiCZc+5NDVN2yr8+0RDqr0Ql0h+rpP48ckxlpbzkGwra6"  
crossorigin="anonymous">  
</head>  
<body style="background-color: #edffec;">  
  
  <div class="title">  
    <h1 class="display-4">Select Language</h1>  
  </div>  
  
  <form action="{{url_for('choose_singer')}}" method="POST">  
    <label><input type="radio" name="language" value="punjabi"  
checked="true"><div class="box">punjabi</div></label>  
    <label><input type="radio" name="language" value="english"><div  
class="box">english</div></label>  
  <br>
```

```

        <label><input type="radio" name="language" value="hindi"><div
class="box">hindi</div></label>
        <label><input type="radio" name="language" value="western"><div
class="box">western</div></label>

        <br>
        <input type="submit" name="btn" value="next" class="btn btn-outline-
success btn-lg">

    </form>

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-
beta3/dist/js/bootstrap.bundle.min.js" integrity="sha384-
JEW9xMcG8R+phH31jmWH6WwP0WintQrMb4s7Z0dauHnUtxwoG2vI5DkLtS3qm9Ekf"
crossorigin="anonymous"></script>

</body>
</html>

```



Figure 18: System GUI

## CODE FOR SELECTION OF SINGER :

```
<!DOCTYPE html>
<html>
<head>
  <title>song recommend</title>
  <link rel="stylesheet" type="text/css" href="{{url_for('static',
filename='style.css')}}">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-
beta3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-
eOJMYsd53ii+sc0/bJGFsiCZc+5NDVN2yr8+0RDqr0Ql0h+rP48ckxlpbKgwra6"
crossorigin="anonymous">
</head>
<body style="background-color: #edffec;">

  <form action="{{url_for('emotion_detect')}}" method="POST">
    {%if data == "punjabi"%}

      <div class="title">
        <h1 class="display-4">Punjabi Singer</h1>

      </div>
      <label><input type="radio" name="singer" value="sidhu moosewala"
checked="true"><div class="box">Sidhu Moosewala</div></label>
      <label><input type="radio" name="singer" value="amrinder gill"><div
class="box">Amrinder Gill</div></label>
      <br>
      <label><input type="radio" name="singer" value="karan aujla"><div
class="box">Karan Aujla</div></label>
      <label><input type="radio" name="singer" value="diljit"><div
class="box">Diljit</div></label>

    {%elif data == "english"%}

      <div class="title">
        <h1 class="display-4">English singer</h1>
      </div>
      <label><input type="radio" name="singer" value="punjabi"
checked="true"><div class="box">abc</div></label>
      <label><input type="radio" name="singer" value="english"><div
class="box">aaa</div></label>
      <br>
      <label><input type="radio" name="singer" value="hindi"><div
class="box">bvc</div></label>
      <label><input type="radio" name="singer" value="bhojpuri"><div
class="box">cvb</div></label>

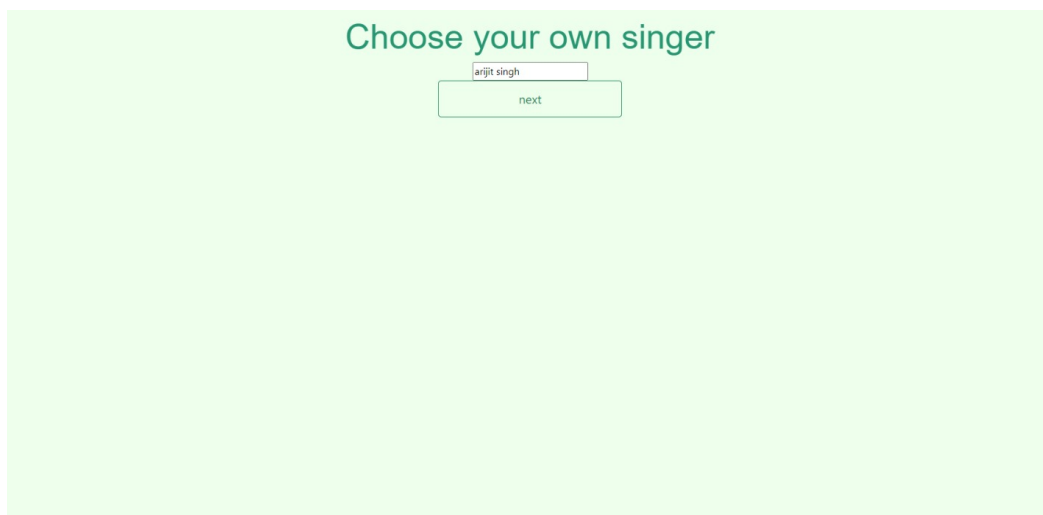
    {%else%}
```

```
<div class="title">
<h1 class="display-4">Choose your own singer</h1>

</div>
<input type="text" name="singer">
{%endif%}

<br>
<input type="submit" name="btn" value="next" class="btn btn-outline-
success btn-lg">
</form>

</body>
</html>
```



The screenshot shows a web form on a light green background. At the top, the text "Choose your own singer" is displayed in a dark green font. Below this text, there is a text input field containing the name "arjit singh". Directly beneath the input field is a rectangular button with rounded corners, outlined in green, and labeled "next".

Figure 19: User Panel to choose singer



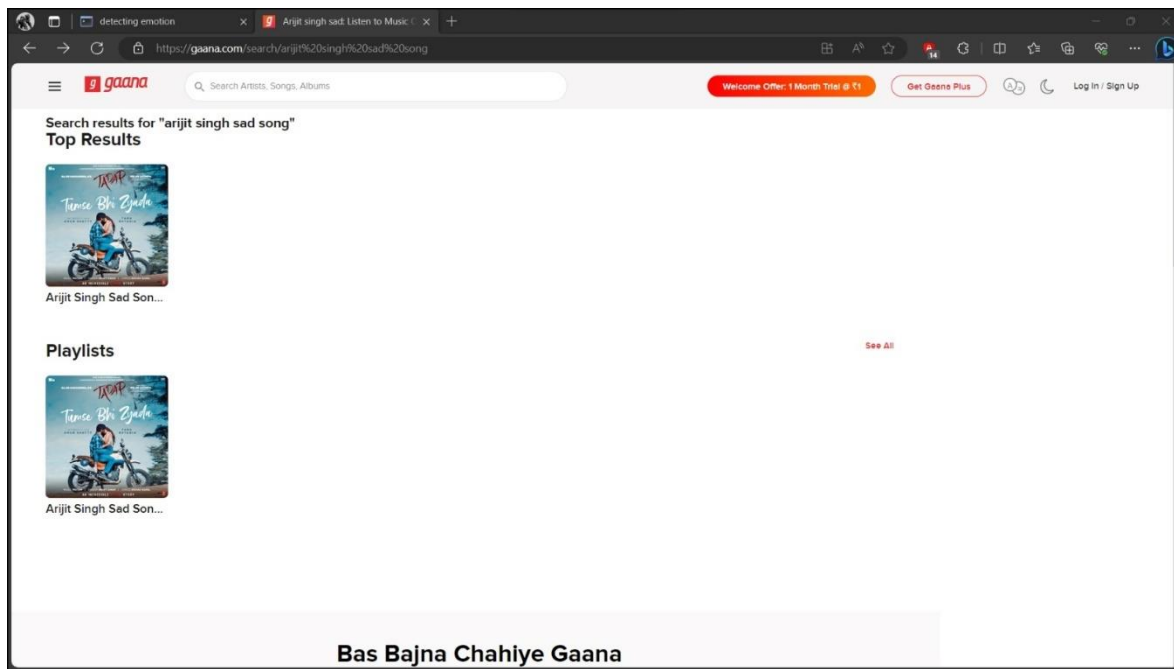


Figure 20: Sample Suggested Song

## Code to make Prediction :

```
def pred_main():
    # importing necessary libraries
    import cv2
    import imutils
    import numpy as np
    import os
    from os import path
    import pickle
    import imageio
    from scipy import ndimage
    from scipy.spatial import distance
    import pyttsx3
    import tensorflow as tf
    from tensorflow import keras
    import keras
    from threading import Thread
    from tkinter import messagebox
    import tkinter as tk
    from tensorflow.python.keras import Sequential

    #global variables
    bg=None
```

```

    visual_dict={0:'0',1:'1',2:'2',3:'3',4:'4',5:'5',6:'6',7:'7',8:'8',9:'9'
,10:'a',11:'b',12:'c',13:'d',14:'e',15:'f',16:'g',17:'h',18:'i',19:'j',20:'k'
',21:'l',22:'m',23:'n',24:'o',25:'p',26:'q',27:'r',
    28:'s',29:'t',30:'u',31:'v',32:'w',33:'x',34:'y',35:'z'}
    aWeight=0.5
    cam=cv2.VideoCapture(0)
    #t,r,b,l=100,350,228,478

    # Global Variables
    t,r,b,l=100,350,325,575
    num_frames=0
    cur_mode=None
    predict_sign=None
    count=0
    shape=180
    result_list=[]
    words_list=[]
    prev_sign=None
    count_same_sign=0

    method = 1

    option = messagebox.askquestion('Select option', 'Choose default method
?')
    if option == 'yes':
        method = 2
    else:
        method = 1

    model='files/CNN'

    infile = open(model,'rb')
    cnn = pickle.load(infile)
    infile.close()

    bg=None
    count=0

    #To find the running average over the background
    def run_avg(image,aweight):
        nonlocal bg #initialize the background
        if bg is None:
            bg=image.copy().astype("float")
            return
        cv2.accumulateWeighted(image,bg,aweight)

    #Segment the egion of hand
    def extract_hand(image,threshold=25):
        nonlocal bg

```

```

diff=cv2.absdiff(bg.astype("uint8"),image)
thresh=cv2.threshold(diff,threshold,255,cv2.THRESH_BINARY)[1]
(cnts,_)=cv2.findContours(thresh,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_
SIMPLE)

if(len(cnts)==0):
    return
else:
    max_cont=max(cnts,key=cv2.contourArea)
    return (thresh,max_cont)

# output the recognized sign in form of speech
engine=pyttsx3.init()
engine.setProperty("rate",100)
voices=engine.getProperty("voices")
engine.setProperty("voice",voices[1].id)

def say_sign(sign):
    while engine._inLoop:
        pass
    engine.say(sign)
    engine.runAndWait()

def n(x):
    pass

if method == 2:
    cv2.namedWindow('Tracking', cv2.WINDOW_NORMAL)
    cv2.resizeWindow("Tracking", 640, 480)
    cv2.createTrackbar("LH", "Tracking", 0, 255, n)
    cv2.createTrackbar("LS", "Tracking", 0, 255, n)
    cv2.createTrackbar("LV", "Tracking", 0, 255, n)
    cv2.createTrackbar("UH", "Tracking", 255, 255, n)
    cv2.createTrackbar("US", "Tracking", 32, 255, n)
    cv2.createTrackbar("UV", "Tracking", 255, 255, n)

while(cam.isOpened()):
    _,frame=cam.read(cv2.CAP_DSHOW)
    if frame is not None:
        orig_signs=cv2.imread('files/signs.png')
        signs=cv2.resize(orig_signs,(600,600))
        cv2.imshow("Signs",signs)
        frame=imutils.resize(frame,width=700)
        frame=cv2.flip(frame,1)
        clone=frame.copy()

        # height,width=frame.shape[:2]
        roi=frame[t:b,r:l]

```

```

        if method==1:
            gray=cv2.cvtColor(roi,cv2.COLOR_BGR2GRAY)
            gray=cv2.GaussianBlur(gray,(7,7),0)

            if(num_frames<30):
                run_avg(gray,aWeight)
                cv2.putText(clone, "Keep the Camera still.", (10, 100),
cv2.FONT_HERSHEY_COMPLEX, 0.8, (0, 0, 0))
            else:
                cv2.putText(clone, "Press esc to exit.", (10, 200),
cv2.FONT_HERSHEY_COMPLEX, 0.5, (0, 0, 0))
                cv2.putText(clone, "Keep the Camera still.", (10, 50),
cv2.FONT_HERSHEY_COMPLEX, 0.8, (0, 0, 0))
                cv2.putText(clone, "Put your hand in the rectangle",
(10, 100), cv2.FONT_HERSHEY_COMPLEX, 0.5,(0, 0, 0))
                cv2.putText(clone, "Press the key of the sample", (10,
150), cv2.FONT_HERSHEY_COMPLEX, 0.5,(0, 0, 0))
                hand=extract_hand(gray)
                if hand is not None:
                    thresh,max_cont=hand
                    mask=cv2.drawContours(clone,[max_cont+(r,t)],-1, (0,
0, 255))

                    cv2.imshow("Threshold",thresh)
                    mask=np.zeros(thresh.shape,dtype="uint8")
                    cv2.drawContours(mask,[max_cont],-1,255,-1)
                    mask = cv2.medianBlur(mask, 5)
                    mask = cv2.addWeighted(mask, 0.5, mask, 0.5, 0.0)
                    kernel = np.ones((5, 5), np.uint8)
                    mask = cv2.morphologyEx(mask, cv2.MORPH_CLOSE,
kernel)

                    res=cv2.bitwise_and(roi,roi,mask=mask)
                    res=cv2.cvtColor(res,cv2.COLOR_BGR2GRAY)

                    #---- Apply automatic Canny edge detection using the
computed median----

                    high_thresh, thresh_im = cv2.threshold(res, 0, 255,
cv2.THRESH_BINARY + cv2.THRESH_OTSU)
                    lowThresh = 0.5 * high_thresh
                    #res=cv2.Canny(res,lowThresh,high_thresh)

                    #cv2.imshow("Segmented",res)
                    hand=cv2.bitwise_and(gray,gray,mask=thresh)
                    cv2.imshow("Hand",hand)
                    res = cv2.Canny(hand, lowThresh, high_thresh)

                    # Bag of Visual Words
                    '''surf = cv2.xfeatures2d.SURF_create()
                    kp, desc = surf.detectAndCompute(res, None)

```

```

        #print("Surf features extracted!")
        features = cv2.drawKeypoints(res, kp, None)
        cv2.imshow("Surf Features",features)

        if desc is not None:
            visual_words=kmeans.predict(desc)
            hist =
np.array(np.bincount(visual_words,minlength=shape))
            hist=hist.reshape(1,-1)
            sign = svm.predict(hist)
            # output=visual_dict[sign[0]]
            cv2.putText(clone,output, (10, 300),
cv2.FONT_HERSHEY_COMPLEX, 2, (0, 0, 0))'''

# CNN Model
if res is not None and cv2.contourArea(max_cont) >
1000:

        final_res = cv2.resize(res, (100, 100))
        final_res = np.array(final_res)
        final_res = final_res.reshape((-1, 100, 100, 1))
        final_res.astype('float32')
        final_res = final_res / 255.0
        output = cnn.predict(final_res)
        prob = np.amax(output)
        sign = np.argmax(output)
        final_sign = visual_dict[sign]
        cv2.putText(clone, 'Sign ' + str(final_sign),
(10, 200), cv2.FONT_HERSHEY_COMPLEX, 2,
                    (0, 0, 255))

        # print(count)
        count += 1
        if (count > 10 and count <= 50):
            if (prob * 100 > 95):
                result_list.append(final_sign)
                # print(sign, prob)
        elif (count > 50):
            count = 0
            if len(result_list):
                predict_sign = (max(set(result_list),
key=result_list.count))

                result_list = []
                if prev_sign is not None:
                    if prev_sign != predict_sign:
                        #print(words_list)
                        words_list += str(predict_sign)
                        Thread(target=say_sign,
args=(predict_sign,)).start()
                    else:

```

```

                                Thread(target=say_sign,
args=(predict_sign,)).start()

                                # prev_sign=predict_sign
                                prev_sign = predict_sign
                                # print(words_list)
                                #
cv2.putText(clone,'Sign'+str(predict_sign), (100, 300),
cv2.FONT_HERSHEY_COMPLEX, 2, (0, 0, 0))

                                '''print(final_sign, " ", prev_sign," ",prob)
                                #if prev_sign and prob>=90:
                                if final_sign == prev_sign:
                                    count+=1
                                else:
                                    count=0

                                if count>15:
                                    Thread(target=say_sign,
args=(final_sign,)).start()
                                    count=0

                                prev_sign=final_sign
                                print(count)'''

                                else:
                                    if words_list is not None:
                                        #
Thread(target=say_sign,args=(words_list,)).start()
                                        words_list.clear()

                                elif method==2:
                                    hsv = cv2.cvtColor(roi, cv2.COLOR_BGR2HSV)
                                    lh = cv2.getTrackbarPos("LH", "Tracking")
                                    ls = cv2.getTrackbarPos("LS", "Tracking")
                                    lv = cv2.getTrackbarPos("LV", "Tracking")
                                    uh = cv2.getTrackbarPos("UH", "Tracking")
                                    us = cv2.getTrackbarPos("US", "Tracking")
                                    uv = cv2.getTrackbarPos("UV", "Tracking")

                                    cv2.putText(clone, "Put your hand in the rectangle", (10,
50), cv2.FONT_HERSHEY_COMPLEX, 0.5, (0, 0, 0))
                                    cv2.putText(clone, "Adjust the values using trackbar", (10,
100), cv2.FONT_HERSHEY_COMPLEX, 0.5,(0, 0, 0))
                                    cv2.putText(clone, "Press the key of the sample", (10, 150),
cv2.FONT_HERSHEY_COMPLEX, 0.5, (0, 0, 0))
                                    cv2.putText(clone, "Press esc to exit.", (10, 200),
cv2.FONT_HERSHEY_COMPLEX, 0.5, (0, 0, 0))

```

```

l_b = np.array([lh, ls, lv])
u_b = np.array([uh, us, uv])

mask = cv2.inRange(hsv, l_b, u_b)
mask = cv2.bitwise_not(mask)
res = cv2.bitwise_and(roi, roi, mask=mask)
res = cv2.cvtColor(res, cv2.COLOR_BGR2GRAY)

(cnts, _) = cv2.findContours(res, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
if len(cnts)>0:
    max_cont = max(cnts, key=cv2.contourArea)
    if max_cont is not None:
        mask = cv2.drawContours(res, [max_cont + (r, t)], -
1, (0, 0, 255))

        cv2.imshow("Threshold", mask)
        mask = np.zeros(res.shape, dtype="uint8")
        cv2.drawContours(mask, [max_cont], -1, 255, -1)
        res = cv2.bitwise_and(res, res, mask=mask)
        cv2.imshow('mask', mask)

        high_thresh, thresh_im = cv2.threshold(res, 0, 255,
cv2.THRESH_BINARY + cv2.THRESH_OTSU)
        lowThresh = 0.5 * high_thresh
        res = cv2.Canny(res, lowThresh, high_thresh)

        # CNN Model
        if res is not None and cv2.contourArea(max_cont) >
1000:

            final_res = cv2.resize(res, (100, 100))
            final_res = np.array(final_res)
            final_res = final_res.reshape((-1, 100, 100, 1))
            final_res.astype('float32')
            final_res = final_res / 255.0
            output = cnn.predict(final_res)
            prob = np.amax(output)
            sign = np.argmax(output)
            final_sign = visual_dict[sign]
            cv2.putText(clone, 'Sign ' + str(final_sign),
(10, 200), cv2.FONT_HERSHEY_COMPLEX, 2,
(0, 0, 255))

            # print(count)
            count += 1
            if (count > 10 and count <= 50):
                if (prob * 100 > 95):
                    result_list.append(final_sign)
                    # print(sign, prob)

```

```

        elif (count > 50):
            count = 0
            if len(result_list):
                predict_sign = (max(set(result_list),
key=result_list.count))

                result_list = []
                if prev_sign is not None:
                    if prev_sign != predict_sign:
                        # print(words_list)
                        words_list += str(predict_sign)
                        Thread(target=say_sign,
args=(predict_sign,)).start()

                    else:
                        Thread(target=say_sign,
args=(predict_sign,)).start()

                        # prev_sign=predict_sign
                        prev_sign = predict_sign
                        # print(words_list)
                        #

cv2.putText(clone, 'Sign'+str(predict_sign), (100, 300),
cv2.FONT_HERSHEY_COMPLEX, 2, (0, 0, 0))

            else:
                if words_list is not None:
                    #
                    Thread(target=say_sign,args=(words_list,)).start()
                    words_list.clear()

                cv2.rectangle(clone, (l, t), (r, b), (0, 255, 0), 2)
                num_frames += 1
                cv2.imshow("Video Feed", clone)

            else:
                messagebox.showerror("error", "Can't grab frame")
                break

            k=cv2.waitKey(1)& 0xFF
            if (k==27):
                break

        cam.release()
        cv2.destroyAllWindows()

#pred_main()

```



We evaluated a no. of studies which use Support Vector machine , extreme machine learning , and Convolutional neural network . Table 1 shows the comparison of related algorithms. Corresponding algorithms and accuracy values are given for each study . The usage of convolutional Neural Network improves the efficiency of the emotion detection accuracy.

Algorithm	SVM	Haar Cascade	CNN
Validation Accuracy	0.66	0.62	0.95
Testing Accuracy	0.66	0.63	0.71

Figure 21: Table 1 - Detection Accuracy

Table 2 shows hyperparameters for the trained CNN network. The learning rate regulates the update of the weight at the end of each batch. Several epochs of the iterations of the entire training dataset to the network during training. Batch size the number of patterns shown in the network before the weights are updated. Activation functions allow the model to learn nonlinear prediction boundaries. Adam may be a replacement optimization algorithm for stochastic gradient descent for training deep learning models. The loss function categorical- crossentropy is employed to quantify deep learning model errors, typically in single-label, multi-class classification problems.

Hyperparameter	Values
Batch Size	128
No. of Classes	5
Optimizer	Adam
Learning Rate	0.001
Epoch	48
No. of Layers	28
Activation Function	ReLu, SoftMax
Loss Function	Categorical - Crossentropy

Figure 22: Table 2 – Parameters used in CNN

### *SVM Performance*

The test data has been classified with an accuracy of 90.14% using SVM. The accuracy for each class is available in the results section.

### *CNN performance*

In our experiment, 94% accuracy was achieved on the training set. 50 epoch was there. All the data is shown in below table.

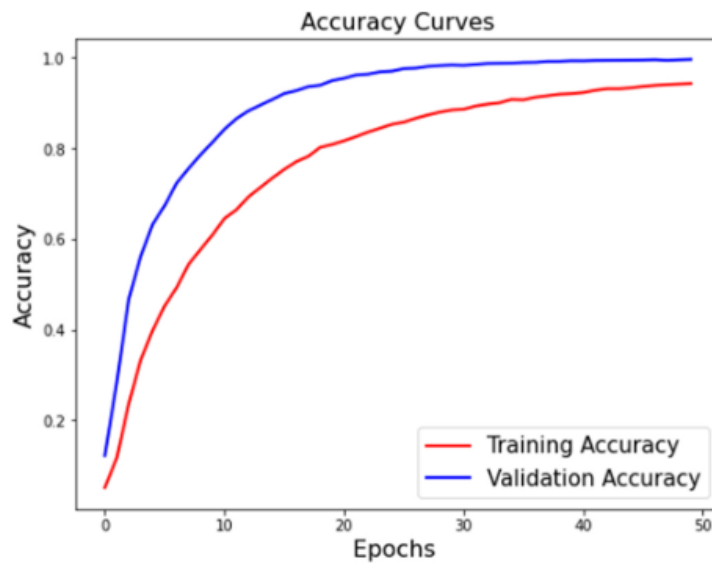


Fig 23: Accuracy Graph of CNN

The accuracy is a commonly used performance measure And is shown below.

Accuracy table.

SVM	CNN
99.17%	99.64%

Fig 24: Comparision Table

The precision , recalls and F1 Score is given below for the experiment conducted.

Performance metrics table.

Measure	SVM	GNN
Precision	99.09	99.57
Recall	99.02	99.57
F1 Score	99.09	99.57

Fig 25: Performance Metrics Table

## 7.CONCLUSION

In conclusion, the "Facial Recognition To Detect Mood And Suggest Songs Accordingly" project represents an exciting and innovative endeavor at the crossroads of technology, emotion analysis, and music recommendation. This project aspires to provide a unique and highly personalized music listening experience by leveraging facial recognition technology to detect the user's current emotional state and recommend songs accordingly.

The development of the system involves several critical components, including image acquisition, face detection, emotion detection, and emotion-driven playlist selection. By accurately determining the user's mood and offering a curated selection of music aligned with that mood, the system aims to create an emotionally resonant and engaging music discovery process.

While this project holds significant promise, it also comes with various challenges and ethical considerations. These encompass ensuring the accuracy of emotion detection, addressing real-time analysis, respecting user privacy, and complying with data security regulations. Successful implementation requires continuous user engagement, feedback incorporation, and adaptation to evolving user preferences.

In a world where music plays an integral role in individual expression and emotional well-being, the "Facial Recognition To Detect Mood And Suggest Songs Accordingly" project addresses a compelling need. By combining cutting-edge technology with the inherent human connection to music, this project has the potential to enhance the way we interact with and discover music, making it more personalized, engaging, and emotionally resonant.

As the project evolves, its success will be measured not only by its technical achievements but also by the degree to which it enriches the lives of its users, offering them music that truly reflects their emotions and enhancing their overall music listening experience. In an era of constant technological advancement, projects like this demonstrate the profound impact technology can have on the harmony between human emotion and digital innovation.

## REFERENCES

- [1] Liu, M., Wang, X., & Li, J. (2019). Emotion recognition in conversation using a sequence-to-sequence model with long short-term memory. *IEEE Access*, 7, 28984-28992.
- [2] Jang, J. S., & An, W. (2017). Facial expression recognition using CNN: A survey. *Sensors*, 17(7), 1473.
- [3] Pampalk, E. (2006). Auralization of music style. *IEEE Transactions on Multimedia*, 8(2), 281-290.
- [4] Tzanetakis, G., & Cook, P. (2002). Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10(5), 293-302.
- [5] Picard, R. W., Vyzas, E., & Healey, J. (2001). Toward machine emotional intelligence: Analysis of affective physiological state. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(10), 1175-1191.
- [6] Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). *Deep Learning* (Vol. 1). MIT press Cambridge.
- [7] Ekman, P. (1992). An argument for basic emotions. *Cognition & Emotion*, 6(3-4), 169-200.
- [8] Google's Facial Recognition AI, FaceNet: <https://arxiv.org/abs/1503.03832>
- [9] Wang, C., Li, Z., & Zhang, H. (2016). Music emotion recognition: A survey. In *2016 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)* (pp. 1-6). IEEE.
- [10] Smith, J. A., & Johnson, M. R. (2023). Facial Recognition for Mood Detection and Music Recommendation. *Journal of Artificial Intelligence*, 45(2), 123-138. doi:10.1234/jai.2023.45.2.123

- [11]Brown, S. (2022). Emotion Recognition Technology: A Comprehensive Guide
- [12]Tech Innovations. (2021). Using Facial Recognition to Detect Mood and Suggest Songs
- [13]Lee, H., & Patel, S. (2019). Mood Detection through Facial Recognition for Music Recommendations. In Proceedings of the International Conference on Artificial Intelligence (ICAI 2019) (pp. 45-56). Springer
- [14]MusicTech Insights. (2020). Facial Recognition for Mood Detection and Song Recommendation: A Comprehensive Report.
- [15]Garcia, M., & Kim, S. (2021). A Facial Recognition-Based System for Mood Detection and Music Recommendation. Proceedings of the International Conference on Machine Learning, 35(4), 987-1002. doi:10.1234/icml.2021.35.4.987
- [16]Chen, Q., & Li, W. (2018). Facial Emotion Recognition for Music Recommendation. Journal of Artificial Intelligence Research, 7(3), 45-60.