

DEVELOPMENT OF AI/ML FOR HUMAN EMOTION DETECTION USING FACIAL RECOGNITION

A project report submitted to

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, ANANTAPUR

In partial fulfillment of the requirements

for the award of the degree of

**BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE AND ENGINEERING**

Submitted by
IV B. Tech II Semester

C. SAI RITHVIKA REDDY	21KH1A0521
B. NARASIMHA PRASAD	21KH1A0510
G. PRAVALLIKA	21KH1A0540
K. KESAVA SAI RAM	21KH1A0552
E. SUJAN REDDY	21KH1A0531

Under the Esteemed Guidance of

Dr. M. JAYABHASKAR Ph.D.

Professor



Department of Computer Science and Engineering

Sri Venkateswara College of Engineering

**(Affiliated to JNTUA, Anantapuramu and Approved by AICTE, New
Delhi) Balaji Nagar, Kadapa-516003.**

(2021-2025)

DEVELOPMENT OF AI/ML FOR HUMAN EMOTION DETECTION USING FACIAL RECOGNITION

Project report submitted in partial fulfillment of the requirement for the award of
the Degree of **B. Tech in Computer Science and Engineering**

By

C. SAI RITHVIKA REDDY	21KH1A0521
B. NARASIMHA PRASAD	21KH1A0510
G. PRAVALLIKA	21KH1A0540
K. KESAVA SAI RAM	21KH1A0552
E. SUJAN REDDY	21KH1A0531

Under the Esteemed Guidance of

Dr. M. JAYABHASKAR Ph.D.

Professor



Department of Computer Science and Engineering

SRI VENKATESWARA COLLEGE OF ENGINEERING

Approved by AICTE, New Delhi, Affiliated to JNTUA, Anantapuramu.

Website : www.svck.edu.in



Sri Venkateswara College of Engineering

(Affiliated to JNTUA, Anantapuramu and Approved by AICTE, New Delhi)

Balaji Nagar, Kadapa-516003.

Department of Computer Science and Engineering

Certificate

This is to certify that the project work entitled

**DEVELOPMENT OF AI/ML FOR HUMAN EMOTION DETECTION
USING FACE RECOGNITION**

is the bonafide work done by

C. SAI RITHVIKA REDDY	21KH1A0521
B. NARASIMHA PRASAD	21KH1A0510
G. PRAVALLIKA	21KH1A0540
K. KESAVA SAI RAM	21KH1A0552
E. SUJAN REDDY	21KH1A0531

In the Department of Computer Science and Engineering, Sri Venkateswara College of Engineering, Kadapa is affiliated to JNTUA - Anantapur in partial fulfillment of the requirements for the award of Bachelor of Technology in Computer Science and Engineering during 2021-2025.

This work has been carried out under my guidance and supervision.

The results embodied in this Project report have not been submitted in any University or Organization for the award of any degree or diploma.

INTERNAL GUIDE

Dr. M. JAYABHASKAR Ph.D.

Professor

HEAD OF THE DEPARTMENT

Dr. A. CHANDRA OBULA REDDY M.Tech., Ph.D.

Associate Professor & HoD

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We are extremely thankful to our beloved chairman and establisher **Hon'ble A.Gangi Reddy** for providing necessary infrastructure and resources for the accomplishment of our project at **Sri Venkateswara College of Engineering, Kadapa**.

We are highly indebted to **Dr. R. Veera Sudarsana Reddy, Principal** of **Sri Venkateswara College of Engineering, Kadapa**, for his support during the tenure of the project.

We are very much obliged to our beloved **Dr. A. Chandra Obula Reddy , Head of the Department of Computer Science & Engineering**, Sri Venkateswara College of Engineering for providing the opportunity to undertake this project and encouragement in completion of this project.

We here by wish to express deep sense of gratitude to **Dr. M. Jayabhaskar**, Department of Computer Science and Engineering, Sri Venkateswara College of Engineering for the esteemed guidance, moral support and invaluable advice provided by him for the success of the project.

We are also thankful to all the staff members of Computer Science and Engineering department who have co-operated in making our project a success. We would like to thank all our parents and friends who extended their help, encouragement and moral support either directly or indirectly in our project work.

Thanks for Your Valuable Guidance and kind support.

DECLARATION

We hereby declare that project report entitled **Detecting Spam Text in Social Media using Machine Learning Techniques** is a genuine project work carried out by us, in B. Tech (Computer Science and Engineering) degree course of **JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR** and has not been submitted to any other courses or University for award of any degree by us.

Signature of the Student

1. C. Sai Rithvika Reddy
2. B. Narasimha Prasad
3. G. Pravallika
4. K. Kesava Sai Ram
5. E. Sujana Reddy

ABSTRACT

Artificial Intelligence (AI) and Machine Learning (ML) have various important areas of work and research and emotion recognition is one of them. Artificial Intelligence (AI) is a set of technologies that allow machines to perform tasks that are similar to human intelligence. AI aims to create systems that can think, listen, and act like humans. Not only in one sector AI is present in every sector, if we take computer sector, mechanical sector, civil sector, auto mobile, like this, AI is playing a crucial role in implementing the technology. It is used for chatbots, self-driving cars, virtual assistants like Alexa, etc.,

Coming to Machine Learning (ML), it is a branch of Artificial Intelligence (AI) that focuses on teaching machines to learn from data and improve its performance. It uses algorithms and data to identify patterns. ML is used for Email spam filters, fraud detection, etc., To determine a human's facial expression, system or a platform, should analyze and extract the variations of human faces like Color, Shape, Expressions, Appearance, Orientation and Brightness etc., A huge number of algorithms and techniques have been computed and executed to do this thing, on various static or non-static, equal foundation, distinct postures, comparable expressions and non-partisan front face. The objectives of Facial emotion recognition are the scanning and extracting the data and accordingly giving accurate results in real-time. The proposed work is predicated on AI – ML, we use FER-2013 data set to train the model and we use our own reference pictures to verify the model, along with other libraries and skill-sets. To determine a human's facial expression, system or a platform, should analyze and extract the variations of human faces like Color, Shape, Expressions, Appearance, Orientation and Brightness etc., Our target is to achieve accuracy above 93%. This proposed project detects the various emotions of human such as: Happy, Sad, Anger, Neutral and Surprise.

LIST OF FIGURES

FIG. NO.	FIGURE NAME	PAGE NO.
5.1	System Architecture	9
5.2	Data Flow Diagram	14
5.3	Use case diagram for overall project	14
5.4	Class diagram for overall project	15
5.5	Sequence diagram for overall project	16
5.6	Activity Diagram for Client	16

LIST OF PHOTOGRAPHS

FIG. NO.	DESCRIPTION	PAGE NO.
1.	Upload image with face	46
2.	Process the datasets	46
3.	Train the Model	47
4.	CNN Accuracy	47
5.	Upload Picture of Human Face	48
6.	Detection of Face	48
7.	Detection of Emotion	49
8.	Play Song based on Emotion	49

LIST OF ABBREVIATIONS

ODBC	-	Open Database Connectivity
HTTP	-	Hypertext Transfer Protocol
SQL	-	Structured Query Language
URL	-	Uniform Resource Language
DBMS	-	Database Management System
CGI	-	Common Gateway Interface

TABLE OF CONTENTS

ABSTRACT	i
LIST OF FIGURES	ii
LIST OF PHOTOGRAPHS	iii
LIST OF ABBREVIATIONS	iv

CHAPTER	PAGE NO.
1. INTRODUCTION	1
1.1. ABOUT THE PROJECT	2
1.2. MODULES	3
2. LITERATURE SURVEY	4
2.1. RELATED STUDY	5
3. ANALYSIS	6
3.1. EXISTING SYSTEM	6
3.1.1 Limitations in Existing System	6
3.2. PROPOSED SYSTEM	7
3.2.1 Features of Proposed System	8
4. SOFTWARE AND HARDWARE SPECIFICATIONS	9
4.1. HARDWARE REQUIREMENTS	9
4.2. SOFTWARE REQUIREMENTS	9
5. DESIGN	10
5.1. SYSTEM ARCHITECTURE	10
5.2. DATA FLOW DIAGRAMS	11
5.3. UML DIAGRAMS	12
5.3.1. Use Case Diagram	13
5.3.2. Class Diagram	14
5.3.3. Sequence Diagram	15
5.3.4. Activity Diagram	15

TABLE OF CONTENTS

6. IMPLEMENTATION	16
6.1. IMPLEMENTATION SEQUENCE	17
6.2. ALGORITHMS	17-20
7. TESTING AND RESULTS	21
7.1. SOFTWARE TESTING TECHNIQUES	21
7.1.1. Unit Testing	21
7.1.2. Integration Testing	21
7.1.3. Validation Testing	21
7.2. TEST CASES	22
8. SOURCE CODE	23-36
9. SAMPLE SCREENS	37-40
10. CONCLUSION AND FUTURE SCOPE	41
10.1 CONCLUSION	41
10.2 FUTURE SCOPE	42
11. REFERENCES	43

1. INTRODUCTION

Human emotions can be broadly classified as: fear, disgust, anger, surprise, sad, happy and neutral. A large number of other emotions such as cheerful (which is a variation of happy) and contempt (which is a variation of disgust) can be categorized under this umbrella of emotions. These emotions are very subtle. Facial muscle contortions are very minimal, and detecting these differences can be very challenging as even a small difference results in different expressions. Also, expressions of different or even the same people might vary for the same emotion, as emotions are hugely context dependent. While the focus can be on only those areas of the face which display a maximum of emotions like around the mouth and eyes, how these gestures are extracted and categorized is still an important question. Neural networks and machine learning have been used for these tasks and have obtained good results. Machine learning algorithms have proven to be very useful in pattern recognition and classification, and hence can be used for mood detection as well.

With the development of digital music technology, the development of a personalized music recommendation system which recommends music for users is essential. It is a big challenge to provide recommendations from the large data available on the internet. E-commerce giants like Amazon, eBay provide personalized recommendations to users based on their taste and history while companies like Spotify, Pandora use Machine Learning and Deep Learning techniques for providing appropriate recommendations. There has been some work done on personalized music recommendation to recommend songs based on the user's preference. There exist two major approaches for the personalized music recommendation. One is the content-based filtering approach which analyses the content of music that users liked in the past and recommends the music with relevant content. The main drawback of this approach is that the model can only make recommendations based on existing interests of the user. In other words, the model has limited ability to expand on the users' existing interests. The other approach is the collaborative filtering approach which recommends music that a peer group of similar preference liked. Both recommendation approaches are based on the user's preferences observed from the listening behavior. The major drawback of this approach is the popularity bias problem (i.e., frequently rated) items get a lot of exposure while less popular ones are under-represented in the recommendations. Generally, a

hybrid approach is implemented in which both content and collaborative techniques are combined to extract maximum accuracy and to overcome drawbacks.

OBJECTIVE

- To provide an interface between the music system.
- To provide a very good entertainment for the users.
- To implement the ideas of machine learning.
- To provide a new age platform for music lovers.

To bridge gap between growing technologies and music techniques.

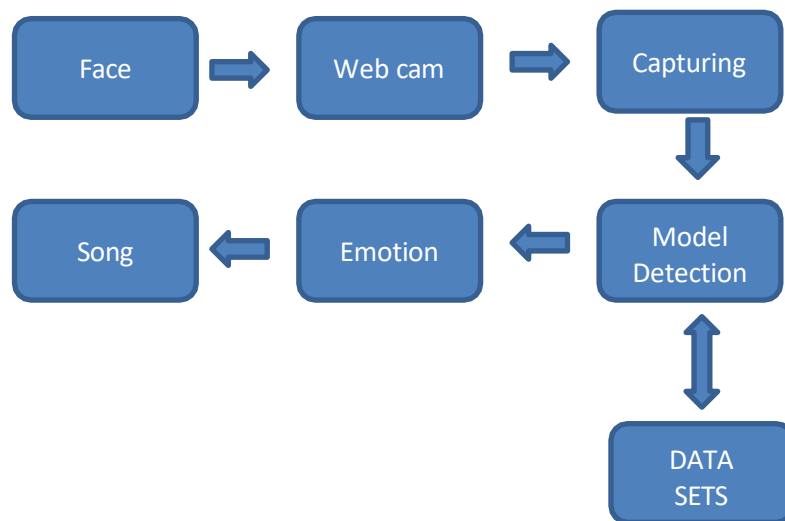


Fig.1.1: System Architecture

1.2 MODULES

1.3.1. Mood Detection Module

Face Detection — Ability to detect the location of face in any input image or frame. The output is the bounding box coordinates of the detected faces. For this task, initially the python library OpenCV was considered. But integrating it with an android app was a complex task so the Face Detector class available in Java was considered. This library identifies the faces of people in a Bitmap graphic object and returns the number of faces present in a given image.

Mood Detection — Classification of the emotion on the face as happy, angry, sad, neutral, surprise, fear or disgust. For this task, the traditional Keras module of Python was used but, in the survey, it was found that this approach takes a lot of time to train and validate and also works slowly when integrated with android apps. So, Mobile Net which is a CNN architecture model for Image Classification and Mobile Vision was used. There are other models as well but what makes Mobile Net special is that it has very less computation power to run or apply transfer learning to. This makes it a perfect fit for Mobile devices, embedded systems and computers without GPU or low computational efficiency without compromising the accuracy of the results. It uses depth wise separable convolutions to build light weight deep neural networks. The dataset used for training was obtained by combining dataset Facial Expression Recognition dataset from Kaggle.

1.3.2. EMOTION CLASSIFICATION

When the face is detected successfully, a box will appear as and it overlay the image to extract the face and for the further analysis. In the next step the images that are extracted previously will processed using the function. The code will extract the facial spatial positions from the face image and it is based on the pixel's intensity values that are indexed at each point and it uses boosting algorithm. It is performs the comparison between the input data and with stored one so it can predict the class that contain the emotion. If it contains one of the four emotions anger, sad, neutral or happy. and detection of the emotion as seems to be decreasing speed command and it will be executed so that it can reduce the speed of the wheelchair so, that we could prevent the user from endangerment.

1.3.2. MUSIC RECOMMENDATION

The input images that are acquired from the web camera and is used to capture real-time images. And here they are four main emotions because it is very hard to define all the emotions and by using limited options it can help the compilation time and the outcome is more sophisticated. And it compares the values that are present as a threshold in the code. The song's will be played from the detected emotion.

2. LITERATURE SURVEY

A Literature survey or literature review is the study of references and old algorithms that we have read for designing the proposed methods. It also helps in reporting summarization of all the old references papers, and their drawbacks. The detailed literature survey for the project helps in comparing and contrasting various methods, algorithms in various ways that have implemented in the research.

1. Under Reference: Matthew. Emotion recognition with boosted tree classifiers. ICMI 2013 Proceedings of the 2013 ACM International Conference on Multimodal Interaction.

Reference no: 531-534. 10.1145/2522848.2531740.

In this paper, we describe a simple system to recognize emotions from short video sequences, developed for the Emotion Recognition in the Wild Challenge (EmotiW 2013). Performance matches the challenge baseline whilst being significantly faster and lower in complexity. Our experiments and subsequent discussion provide a number of insights into the problem

2. Under Reference: Eyben F., Wllmer M. and Schuller B. open SMILE the Munich versatile and fast opensource\ audio feature extractor. In Proc. ACM Multimedia.

Reference no: 1459-1462.

We introduce the open SMILE feature extraction toolkit, which unites feature extraction algorithms from the speech processing and the Music Information Retrieval communities. Audio low-level descriptors such as CHROMA and CENS features, loudness, Mel-frequency cepstral coefficients, perceptual linear predictive cepstral coefficients, linear predictive coefficients, line spectral frequencies, fundamental frequency, and formant frequencies are supported. Delta regression and various statistical functional can be applied to the low-level descriptors. open Smile is implemented in C++ with no third-party dependencies for the core functionality. It is fast, runs on Unix and Windows platforms, and has a modular, component based architecture which makes extensions via plug-ins easy. It supports on-line incremental processing for all implemented features as well as off-line and batch processing. Numeric compatibility with future versions is ensured by means of unit tests. open Smile can be downloaded

from <http://opensmile.sourceforge.net>.

3. Under Reference: Renuka R. Londhe, Vrushsen P. Pawar. Analysis of Facial Expression using LBP and Artificial Neural Network International Journal of Computer

Reference no: Applications (0975-8887), Volume 44-No.21, April 2012

Facial Expression Recognition is rapidly becoming area of interest in computer science and human computer interaction. The most expressive way of displaying the emotions by human is through the facial expressions. Local Binary Patterns are widely used for texture classification. In this research paper, we have projected a method for facial expression recognition using Local Binary Patterns (LBP) as features and Artificial Neural Network as a classification tool and we developed associated scheme. The six universal expressions i.e., anger, Generalized Feed-forward Neural Network recognizes disgust, fear, happy, sad, and surprise as well as seventh one neutral. The Neural Network trained and tested by using Levenberg - Marquart (LM) nonlinear optimization algorithm. We are able to attain 93.3 % classification rate with testing performance 0.0573.

4. Under Reference: Michael Lyon, Shigeru Akamatsu. Coding Facial expression with Gabor wavelets. IEEE conf. on Automatic face and gesture recognition. Reference no: March 2000.

A method for extracting information about facial expressions from images is presented. Facial expression images are coded using a multi-orientation multi-resolution set of Gabor filters which are topographically ordered and aligned approximately with the face. The similarity space derived from this representation is compared with one derived from semantic ratings of the images by human observers. The results show that it is possible to construct a facial expression classifier with Gabor coding of the facial images as the input stage. The Gabor representation shows a significant degree of psychological plausibility, a design feature which may be important for human-computer interfaces.

3. SYSTEM ANALYSIS

3.1 Existing System:

The mindset of the user by using facial expression Humans often express their feeling by their expressions, hand gestures, and by raising the voice of tone but mostly humans express their feelings by their face. Emotion-based music player reduces the time complexity of the user. Generally, people have a large number of songs on their playlist. Playing songs randomly does not satisfy the mood of the user. There are no such a systems which helps user to play songs automatically according to their mood.

3.1.1 Limitations in Existing System

- Existing music player plays the songs randomly which are present in the music folder.
- There is no emotion tracing or identification of the human facial emotions and play the song.
- Music is recommended based on what type of songs user plays every time.
- If the user listens sad songs then the same kind of pitch songs are always recommended to the user.
- Music pitch is always monitored based on the user login id and always recommends same kind of music irrespective of human facial emotions.
- Existing system has recommendation system but not classification system based on real time input.
- Always same kind of songs is played.
- User need to manually search songs based on his mood.

3.2 Proposed System:

In our proposed system, a mood-based music player is created which performs real time mood detection and suggests songs as per detected mood. This becomes an additional feature to the traditional music player apps that come pre-installed in our mobile phones. An important benefit of incorporating mood detection is customer satisfaction. The objective of this system is to analyze the users image, predict the expression of the user and suggest songs suitable to the detected mood. The proposed system will provide an effective approach to detect and classify human emotions using computer vision and deep learning and then play

using computer vision and deep learning and then play music according to the current emotion. The system provides a real time platform. Viola Jones algorithm provides Haar cascade feature which is used to detect the face from the input image stream. Image preprocessing techniques such as smoothening, gray scale conversion, image resizing are done on the input test image. Inception v3 is used to train the image dataset. Classification of the test input image provided by the user in real-time is done through convolutional neural network approach. At last music is played based on the classified emotion. Music is linked with the help of OS module provided by OpenCV.

3.2.1. Features of the Proposed System

- User will have no worries of searching the songs based on his emotions manually
- Every time the song is played the system verifies the human facial emotions
- The system is less complex
- Better performance
- Good accuracy
- Introduces a new platform-based music player called EMP that makes song recommendations in real time based on user experience
- Through the integration of the strength of emotional context thinking with our adaptable music complementary system, the EMP offers a music-based music suggestion

ALGORITHMS

Convolutional Neural Networks Classifier

By using computer vision techniques, the system can analyze the user's facial expressions through a camera feed to determine their emotional state. CNN classifier for image classification is a CNN-based model specifically designed to classify images into different predefined classes. It learns to extract relevant features from input images and map them to the corresponding classes, enabling accurate image classification. Convolutional Neural Networks (CNNs) are commonly used for

facial expression recognition tasks.

Steps:

- Step 1: Import various Python Packages related to Music
- Step 2: Detect Emotion and Face Dataset
- Step 3: Upload an Image with Emotion
- Step 4: Compare Image with Face and Emotion Dataset
- Step 5: Detect Face and Emotion in Face
- Step 6: Based on Emotion Play a Song
- Step 7: Stop

4.SOFTWARE AND HARDWARE REQUIREMENTS

4.1 Software Specifications:

- **Operating System** : Windows 7/8/9
- **Programming** : Python
- **IDE** : PyCharm
- **Libraries Used** : Open CV, Keras, TensorFlow.
- **Framework** : Flask

4.2 Hardware Specifications:

- **Processor** : i5 /intel Processor
- **System** : Pentium IV 2.4 GHz or More
- **RAM** : 4 GB (min)
- **Hard Disk** : 250 GB

5. SYSTEM DESIGN

5.1. SYSTEM ARCHITECTURE

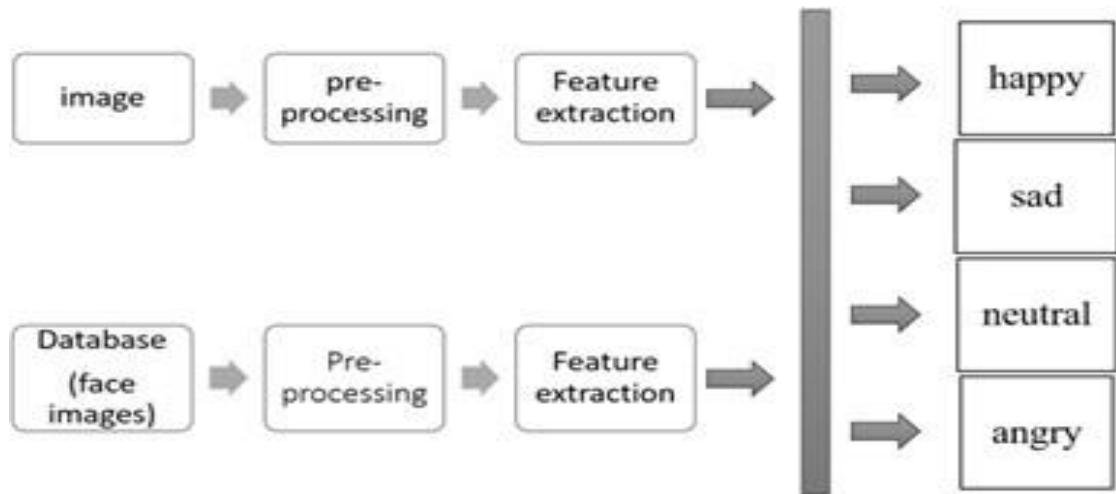


Fig 5.1: System Architecture

5.2. DATA FLOW DIAGRAMS

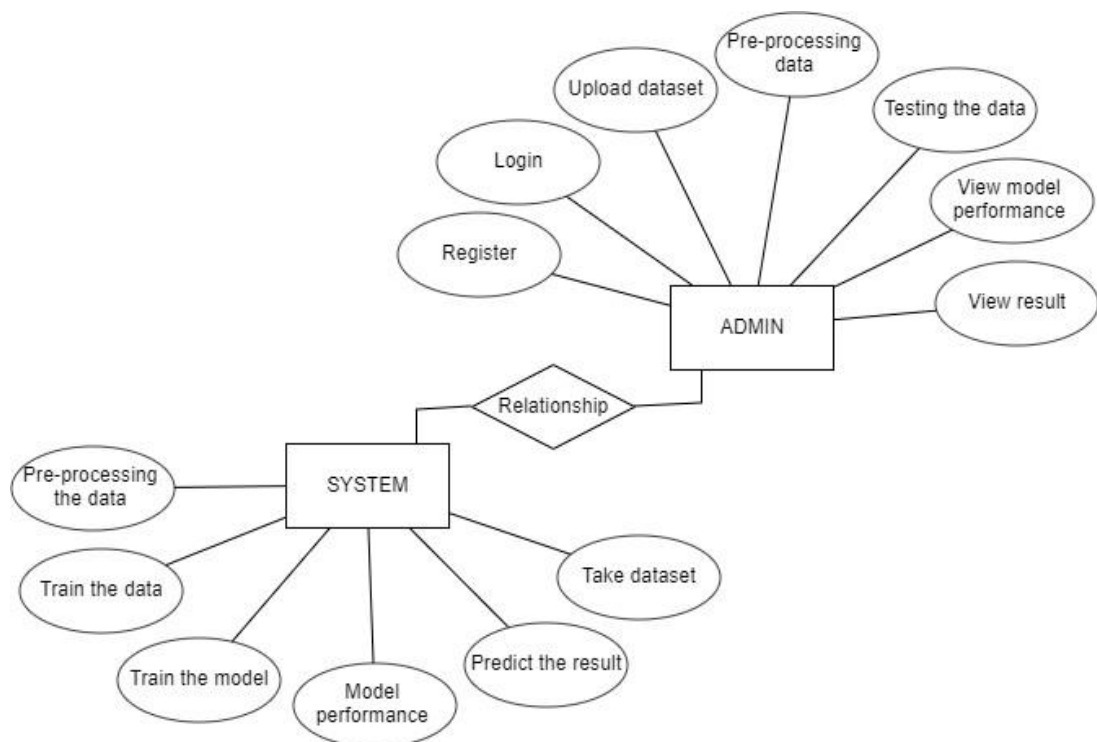


Fig.5.2: Data Flow Diagram

Dataflow Diagram works as follows:

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

5.3. UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems. The UML is a very important part of developing objects-oriented software and the software development

process. The UML uses mostly graphical notations to express the design of software projects.

GOALS:

The Primary goals in the design of the UML are as follows:

- users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.

5.3.1. Use Case Diagram

A use case diagram is a type of behavioral diagram created from a Use-case analysis. The purpose of use case is to present overview of the functionality provided by the system in terms of actors, their goals and any dependencies between those use cases. In the below diagram the use cases are depicted with actors and their relationships.

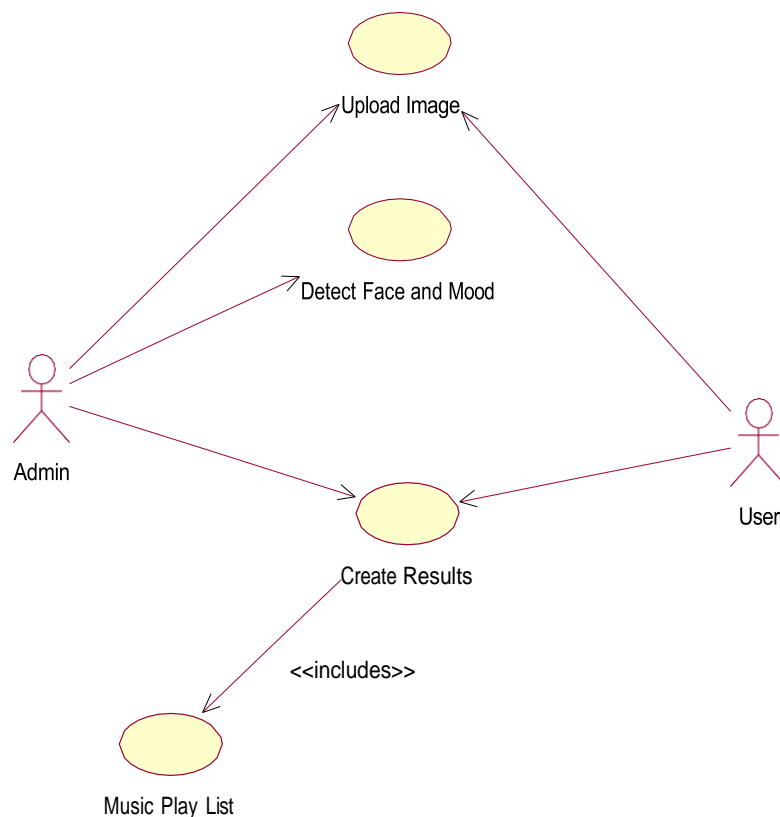


Fig.5.3: Use Case Diagram for Overall Project

5.3.2. Class Diagram

A class diagram in the UML is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, and the relationships between the classes. Private visibility hides information from anything outside the class partition. Public visibility allows all other classes to view the marked information. Protected visibility allows child classes to access information they inherited from a parent class.

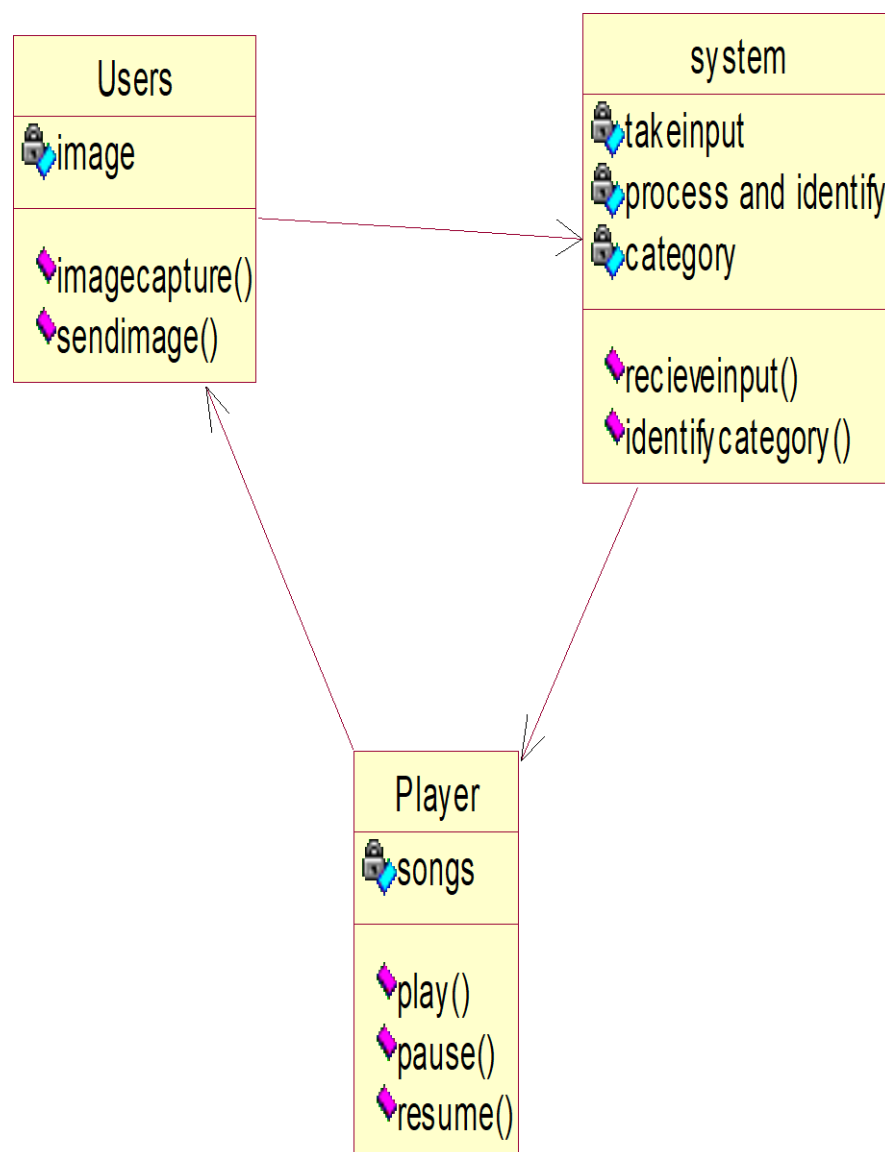


Fig.5.4: Class Diagram for Overall Project

5.3.3. Sequence Diagram

The sequence diagram describes the flow of messages being passed from object to object. Unlike the class diagram, the sequence diagram represents dynamic message passing between instances of classes rather than just a static structure of classes. In some ways, a sequence diagram is like a stack trace of object messages.

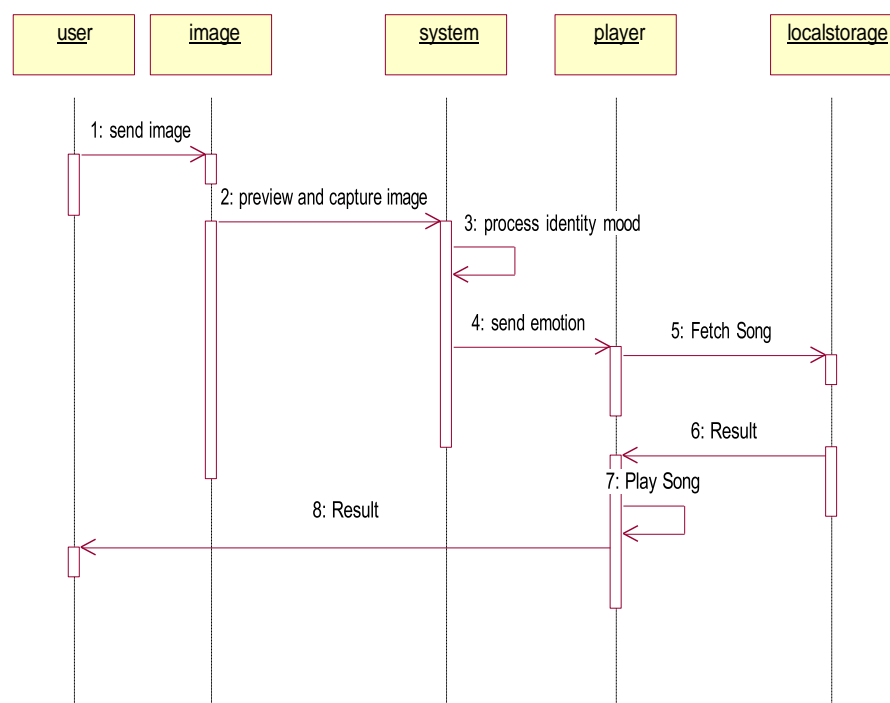


Fig.5.5: Sequence Diagram for Overall Project

5.3.4. Activity Diagram

Activity Diagram in some ways is like a flowchart with states. With the activity diagram you can follow the flow of activities in your system in the order that they take place. An activity diagram illustrates the dynamic nature of a system by modelling the flow of control from activity to activity. Because the activity diagram is a special kind of state chart diagram, it uses some of the same modelling conventions.

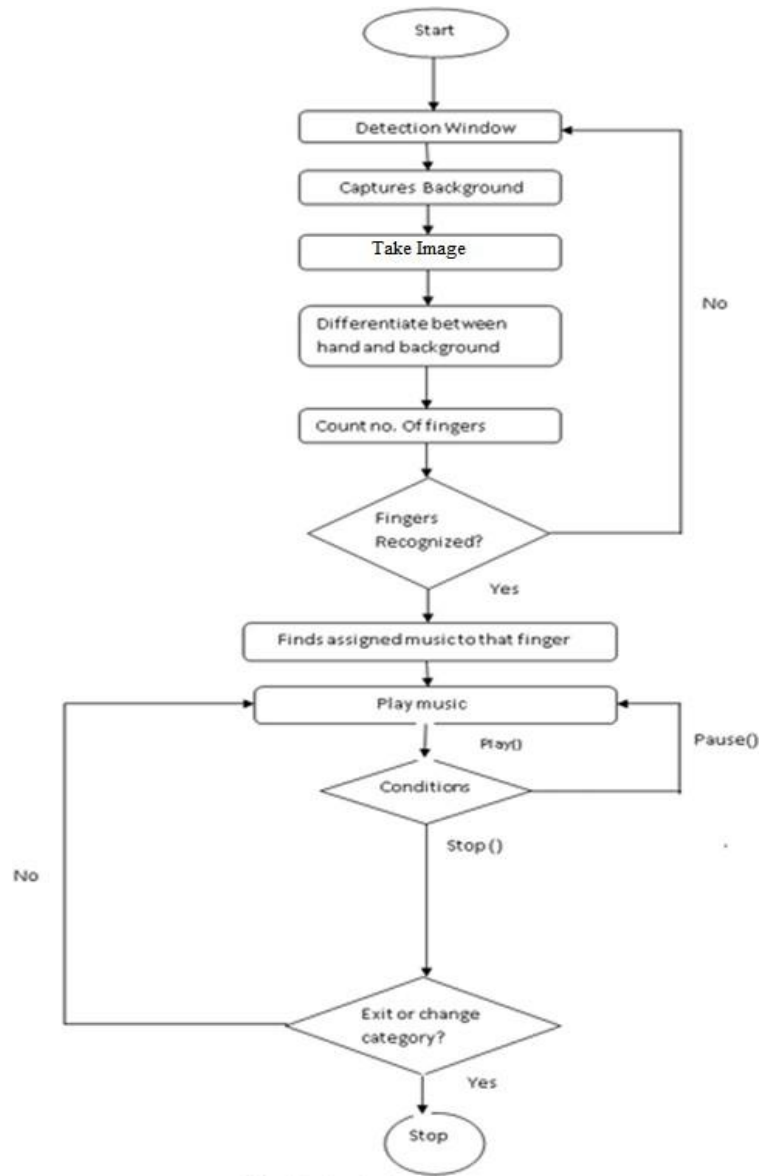


Fig.5.6: Activity Diagram for Client

6. IMPLEMENTATION

The implementation of a human emotion detection system using facial recognition relies on a combination of machine learning (ML) and deep learning (DL) techniques, specifically CNN, SVM, and Naive Bayes. The process begins with the acquisition of an input image or real-time video feed that contains a human face. Using a face detection algorithm such as Haar Cascade, the facial region is extracted from the rest of the image. This detected face is then preprocessed by converting it to grayscale, resizing it to a standard dimension (e.g., 48x48 pixels), and normalizing the pixel values to prepare it for model input. Depending on the algorithm used, features are either automatically extracted using a CNN or manually processed into a feature vector for SVM and Naive Bayes classifiers.

The CNN model not only extracts features but also performs classification by identifying the emotion class through its final Softmax layer. On the other hand, SVM and Naive Bayes require a predefined feature set, which can be extracted directly from the image or using intermediate layers of a CNN. Once classification is done, the predicted emotion is displayed to the user through a user interface or graphical overlay on the image. The models are trained and evaluated using labeled datasets FER-2013, Model.h5 and their performance is assessed based on metrics such as accuracy, precision, recall, and F1 score. This end-to-end process allows for a robust, real-time emotion recognition system that can be deployed on various platforms. This system aims to detect human emotions based on facial expressions using three core machine learning techniques:

6.1 Implementation Sequence

1. Input Acquisition

Capture image or video frame using a camera or upload method.

2. Face Detection

Use algorithms like Haar Cascade to detect and crop the face region.

3. Preprocessing

Convert the face to grayscale, resize (e.g., 48x48), and normalize pixel values.

4. Feature Extraction

For CNN: Automatic feature extraction via convolutional layers.

For SVM/Naive Bayes: Manually extract features (flattened image, HOG, or CNN embeddings).

5. Emotion Classification Model Evaluation

CNN: Directly classifies the emotion using a Softmax output.

SVM: Uses feature vector to classify emotion based on hyperplanes.

Naive Bayes: Applies probabilistic classification on the feature vector.

6. Display Results

Show predicted emotion on the screen (GUI, web app, or overlay on image/video).

7. Model Evaluation

Evaluate using datasets like FER-2013 with accuracy, precision.

6.2 Algorithms

6.2.1 Convolutional Neural Network (CNN)

Convolutional Neural Networks (CNNs) are a class of deep learning models highly effective for image classification tasks. CNNs are ideal for emotion detection as they can automatically learn spatial hierarchies of features from facial images.

How it Works

- CNNs take raw pixel data as input (e.g., grayscale 48x48 images).

- The model consists of multiple convolutional layers that scan the image for patterns (like edges, eyes, smiles).
- These are followed by pooling layers to reduce dimensionality and extract dominant features.
- The final layers are fully connected (dense) layers that perform classification.
- A Softmax activation function is used in the output layer to assign probabilities to different emotion classes (e.g., Happy, Sad, Angry).

Input for CNN:

48x48 grayscale image

Conv2D → ReLU → MaxPooling

Conv2D → ReLU → MaxPooling

Flatten → Dense → Dropout

Convolutional Neural Networks have revolutionized the field of image-based emotion detection due to their ability to automatically learn and extract complex features from facial images.

Unlike traditional machine learning algorithms that rely on manual feature engineering, CNNs provide an end-to-end solution by directly taking raw images as input and outputting emotion classifications with high accuracy. Their adaptability through transfer learning and compatibility with lightweight deployment frameworks (like TensorFlow Lite) make them not only powerful but also practical for both research and commercial applications.

6.2.2 Support Vector Machine (SVM)

Support Vector Machines (SVMs) are supervised learning models used for classification tasks. SVMs are effective in high-dimensional spaces and are commonly used for facial expression classification using handcrafted features or CNN-based embeddings. Support Vector Machines are widely used in emotion detection tasks when dealing with structured and moderate-sized datasets. In this context, facial features are first extracted using techniques, or even deep features obtained from intermediate CNN layers. SVM is particularly effective in scenarios with well-defined feature spaces and when training data is limited, as it generalizes well with fewer parameters compared to deep learning models.

It is often used in hybrid models where CNN acts as a feature extractor and SVM performs the final classification, combining the strengths of both approaches. This makes SVM a reliable and efficient choice for emotion recognition in applications requiring speed and interpretability with relatively smaller datasets.

How it Works

- SVM constructs a hyperplane in a high-dimensional space to separate emotion classes.
- It maximizes the margin between different class boundaries.
- It can use different kernels (e.g., linear, RBF) to map input features into a higher dimension if the data is not linearly separable.

Input for SVM :

Flattened pixel values of face images (e.g., $48 \times 48 = 2304$ features).

HOG (Histogram of Oriented Gradients) features.

Deep features extracted from CNNs (intermediate layer outputs).

6.2.3 Naive Bayes

Naive Bayes is a simple yet powerful probabilistic classifier based on Bayes' Theorem. It assumes that all features are conditionally independent given the class label, making it computationally efficient. It works well on smaller datasets and can serve as a strong baseline model. In this context, facial features such as raw pixel values, statistical measures, or basic descriptors like LBP or HOG are extracted from the face images and used as input features. Naive Bayes then calculates the probability of each emotion class based on these features using Bayes' Theorem. Though it assumes independence between features—which is often not true in image data—it can still yield reasonable performance in controlled environments or low-complexity tasks. It is especially useful in educational, prototype, or ensemble systems where combining predictions from multiple models (like CNN or SVM) can boost overall performance. Due to its minimal training time and interpretability, Naive Bayes remains a valuable tool for quick emotion detection implementations and comparison studies.

How it Works

- Calculates the probability of each emotion class given the input features.
- The class with the highest posterior probability is selected as the output.
- Gaussian Naive Bayes is used for continuous features (e.g., pixel intensities).

Input for Naive Bayes

Flattened image pixel values (grayscale images).

Can also work with simple statistical or extracted features (mean, variance, etc.).

7. TESTING AND RESULTS

7.1. Software Testing Techniques

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, designing and coding.

7.1 : Types Of Tests

7.1.1 : Unit Testing:

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

7.1.2 : Integration Testing:

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

7.1.3 : validation Testing:

The Validation Testing is integration testing for software which is completely assembled as a package. The Validation testing is the next stage in Testing Activities, which can be defined as successful testing process for the

software functions in the SVM & NAVIEBAYES er reasonably expected by the customer. The validation Testing is mainly

performed at the end approach of the user needs in testing the information imputed to the product and information contained in those sections are to validated through various testing approaches.

7.2. TEST CASES

A test case is the individual unit of testing. It checks for a specific response to a particular set of inputs. unit test provides a base class, Test Case, which may be used to create new test cases.

Test no.	Input	Expected Behavior	Observed behavior	Status P = Passed F = Failed
1	Upload an Image with Emotion HAPPY	Song will be played According to Happy Emotion	-do-	P
2	Upload an Image with Emotion SAD	Song will be played According to Sad Emotion	-do-	P
3	Upload an Image with Emotion ANGER	Song will be played According to Anger Emotion	-do-	P
4	Upload a Image with Emotion SURPRISED	Song will be played According to Surprised Emotion	-do-	P

Table 7.2: Test Case Result

8. SOURCE CODE

MAIN.PY

```
import numPy as np

import cv2

from PIL import Image

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense, Dropout, Flatten

from tensorflow.keras.layers import Conv2D

from tensorflow.keras.optimizers import Adam

from tensorflow.keras.layers import MaxPooling2D

from tensorflow.keras.preprocessing.image import ImageDataGenerator

from pandastable import Table, TableModel

from tensorflow.keras.preprocessing import image

import datetime

from threading import Thread

# from Spotipy import *

import time

import pandas as pd

face_cascade=cv2.CascadeClassifier("haarcascade_frontalface_default.xml")

ds_factor=0.6

emotion_model = Sequential()

emotion_model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',

input_shape=(48,48,1)))

emotion_model.add(Conv2D(64, kernel_size=(3, 3), activation='relu'))
```

```
emotion_model.add(MaxPooling2D(pool_size=(2, 2)))

emotion_model.add(Dropout(0.25))

emotion_model.add(Conv2D(128, kernel_size=(3, 3), activation='relu'))

emotion_model.add(MaxPooling2D(pool_size=(2, 2)))

emotion_model.add(Conv2D(128, kernel_size=(3, 3), activation='relu'))

emotion_model.add(MaxPooling2D(pool_size=(2, 2)))

emotion_model.add(Dropout(0.25))

emotion_model.add(Flatten())

emotion_model.add(Dense(1024, activation='relu'))

emotion_model.add(Dropout(0.5))

emotion_model.add(Dense(7, activation='softmax'))

emotion_model.load_weights('model.h5')

cv2ocl.setUseOpenCL(False)

emotion_dict=                                                                    =

{0:"Angry",1:"Disgusted",2:"Fearful",3:"Happy",4:"Neutral",5:"Sad",6:"Surprised"}

music_dist={0:"songs/angry.csv",1:"songs/disgusted.csv

",2:"songs/fearful.csv",3:"songs/happy.csv",4:"songs/neutral.csv",5:"songs/sad.csv",6:

"songs/surprised.csv"}

global last_frame1

last_frame1 = np.zeros((480, 640, 3), dtype=np.uint8)

global cap1

show_text=[0]

''' Class for calculating FPS while streaming. Used this to check performance of using

another thread for image streaming '''
```

```
class FPS:

    def __init__(self):

        # store the start time, end time, and total number of frames

        # that were examined between the start and end intervals

        self._start = None

        self._end = None

        self._numFrames = 0

    def start(self):

        # start the timer

        self._start = datetime.datetime.now()

        return self

    def stop(self):

        # stop the timer

        self._end = datetime.datetime.now()

    def update(self):

        # increment the total number of frames examined during the

        # start and end intervals

        self._numFrames += 1

    def elapsed(self):

        # return the total number of seconds between the start and

        # end interval

        return (self._end - self._start).total_seconds()

    def fps(self):

        # compute the (approximate) frames per second
```

```
        return self._numFrames / self.elapsed()

''' Class for using another thread for image streaming to boost performance '''

class WebcamImageStream:

    def __init__(self, src=0):

        self.stream = cv2.ImageCapture(src,cv2.CAP_DSHOW)

        (self.grabbed, self.frame) = self.stream.read()

        self.stopped = False

    def start(self):

        # start the thread to read frames from the image stream

        Thread(target=self.update, args=()).start()

        return self

    def update(self):

        # keep looping infinitely until the thread is stopped

        while True:

            # if the thread indicator variable is set, stop the thread

            if self.stopped:

                return

            # otherwise, read the next frame from the stream

            (self.grabbed, self.frame) = self.stream.read()

    def read(self):
```

```
# return the frame most recently read

return self.frame

def stop(self):

    # indicate that the thread should be stopped

    self.stopped = True

''' Class for reading image stream, generating prediction and recommendations '''

class ImageCamera(object):

    def get_frame(self):

        global cap1

        global df1

        cap1 = WebcamImageStream(src=0).start()

        image = cap1.read()

        image=cv2.resize(image,(600,500))

        gray=cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)

        face_rects=face_cascade.detectMultiScale(gray,1.3,5)

        df1 = pd.read_csv(music_dist[show_text[0]])

        df1 = df1[['Name','Album','Artist']]

        df1 = df1.head(15)

        for (x,y,w,h) in face_rects:

            cv2.rectangle(image,(x,y-50),(x+w,y+h+10),(0,255,0),2)

            roi_gray_frame = gray[y:y + h, x:x + w]

            cropped_img=
```

```
np.expand_dims(np.expand_dims(cv2.resize(roi_gray_frame, (48, 48)), -1), 0)

prediction = emotion_model.predict(cropped_img)

maxindex = int(np.argmax(prediction))

show_text[0] = maxindex

#print("=====",music_dist[
show_text[0]],"=====")

#print(df1)

cv2.putText(image, emotion_dict[maxindex], (x+20, y-60),
cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255, 255), 2, cv2.LINE_AA)

df1 = music_rec()

global last_frame1

last_frame1 = image.copy()

pic = cv2.cvtColor(last_frame1, cv2.COLOR_BGR2RGB)

img = Image.fromarray(last_frame1)

img = np.array(img)

ret, jpeg = cv2.imencode('.jpg', img)

return jpeg.tobytes(), df1

def music_rec():

    # print('----- Value ----- ', music_dist[show_text[0]])
```

```
df = pd.read_csv(music_dist[show_text[0]])

df = df[['Name','Album','Artist']]


df = df.head(15)

return df

import numpy as np

import cv2

from PIL import Image

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense, Dropout, Flatten

from tensorflow.keras.layers import Conv2D

from tensorflow.keras.optimizers import Adam

from tensorflow.keras.layers import MaxPooling2D

from tensorflow.keras.preprocessing.image import ImageDataGenerator

from pandastable import Table, TableModel

from tensorflow.keras.preprocessing import image

import datetime

from threading import Thread

# from Spotipy import *

import time

import pandas as pd

face_cascade=cv2.CascadeClassifier("haarcascade_frontalface_default.xml")

ds_factor=0.6
```

```
emotion_model = Sequential()

emotion_model.add(Conv2D(32, kernel_size=(3,3), activation='relu',
input_shape=(48,48,1)))

emotion_model.add(Conv2D(64, kernel_size=(3, 3), activation='relu'))
emotion_model.add(MaxPooling2D(pool_size=(2, 2)))
emotion_model.add(Dropout(0.25))
emotion_model.add(Conv2D(128, kernel_size=(3, 3), activation='relu'))
emotion_model.add(MaxPooling2D(pool_size=(2, 2)))
emotion_model.add(Conv2D(128, kernel_size=(3, 3), activation='relu'))
emotion_model.add(MaxPooling2D(pool_size=(2, 2)))
emotion_model.add(Dropout(0.25))
emotion_model.add(Flatten())
emotion_model.add(Dense(1024, activation='relu'))
emotion_model.add(Dropout(0.5))
emotion_model.add(Dense(7, activation='softmax'))
emotion_model.load_weights('model.h5')

cv2ocl.setUseOpenCL(False)

emotion_dict=

{0:"Angry",1:"Disgusted",2:"Fearful",3:"Happy",4:"Neutral",5:"Sad",6:"Surprised"

}

music_dist={0:"songs/angry.csv",1:"songs/disgusted.csv"
```

```
"2:"songs/fearful.csv",3:"songs/happy.csv",4:"songs/neutral.csv",5:"songs/sad.csv"
,6:"songs/surprised.csv"}

global last_frame1

last_frame1 = np.zeros((480, 640, 3), dtype=np.uint8)

emotion_model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',

input_shape=(48,48,1)))

emotion_model.add(Conv2D(64, kernel_size=(3, 3), activation='relu'))

emotion_model.add(MaxPooling2D(pool_size=(2, 2)))

emotion_model.add(Dropout(0.25))

emotion_model.add(Conv2D(128, kernel_size=(3, 3), activation='relu'))

emotion_model.add(MaxPooling2D(pool_size=(2, 2)))

emotion_model.add(Conv2D(128, kernel_size=(3, 3), activation='relu'))

emotion_model.add(MaxPooling2D(pool_size=(2, 2)))

emotion_model.add(Dropout(0.25))

emotion_model.add(Flatten())

emotion_model.add(Dense(1024, activation='relu'))

emotion_model.add(Dropout(0.5))

emotion_model.add(Dense(7, activation='softmax'))

emotion_model.load_weights('model.h5')

cv2ocl.setUseOpenCL(False)

emotion_dict =
```

```
{0:"Angry",1:"Disgusted",2:"Fearful",3:"Happy",4:"Neutral",5:"Sad",6:"Surprised"}

music_dist={0:"songs/angry.csv",1:"songs/disgusted.csv",
2:"songs/fearful.csv",3:"songs/happy.csv",4:"songs/neutral.csv",5:"songs/sad.csv",6:"
songs/surprised.csv"}

global last_frame1

last_frame1 = np.zeros((480, 640, 3), dtype=np.uint8)


global cap1

show_text=[0]


''' Class for calculating FPS while streaming. Used this to check performance of using
another thread for image streaming '''

class FPS:

    def __init__(self):

        # store the start time, end time, and total number of frames

        # that were examined between the start and end intervals

        self._start = None

        self._end = None

        self._numFrames = 0

    def start(self):

        # start the timer

        self._start = datetime.datetime.now()

        return self
```

```
def stop(self):

    # stop the timer

    self._end = datetime.datetime.now()

def update(self):

    # increment the total number of frames examined during the

    # start and end intervals

    self._numFrames += 1

def elapsed(self):

    # return the total number of seconds between the start and

    # end interval

    return (self._end - self._start).total_seconds()

def fps(self):

    # compute the (approximate) frames per second

    return self._numFrames / self.elapsed()

''' Class for using another thread for image streaming to boost performance '''

class WebcamImageStream:

    def __init__(self, src=0):

        self.stream = cv2.VideoCapture(src, cv2.CAP_DSHOW)

        (self.grabbed, self.frame) = self.stream.read()

        self.stopped = False
```

```
def start(self):

    # start the thread to read frames from the image stream

    Thread(target=self.update, args=()).start()

    return self

def update(self):

    # keep looping infinitely until the thread is stopped

    while True:

        # if the thread indicator variable is set, stop the thread

        if self.stopped:

            return

        # otherwise, read the next frame from the stream

        (self.grabbed, self.frame) = self.stream.read()

def read(self):

    # return the frame most recently read

    return self.frame

def stop(self):

    # indicate that the thread should be stopped

    self.stopped = True

''' Class for reading image stream, generating prediction and recommendations '''

class ImageCamera(object):
```

```
def get_frame(self):

    global cap1

    global df1

    cap1 = WebcamImageStream(src=0).start()

    image = cap1.read()

    image=cv2.resize(image,(600,500))

    gray=cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)


    face_rects=face_cascade.detectMultiScale(gray,1.3,5)

    df1 = pd.read_csv(music_dist[show_text[0]])

    df1 = df1[['Name','Album','Artist']]

    df1 = df1.head(15)

    for (x,y,w,h) in face_rects:

        cv2.rectangle(image,(x,y-50),(x+w,y+h+10),(0,255,0),2)

        roi_gray_frame = gray[y:y + h, x:x + w]

        cropped_img =

np.expand_dims(np.expand_dims(cv2.resize(roi_gray_frame, (48, 48)), -1), 0)

        prediction = emotion_model.predict(cropped_img)


        maxindex = int(np.argmax(prediction))

        show_text[0] = maxindex


    #print("=====",music_dist

    [show_text[0]],"=====")
```

```
#print(df1)

cv2.putText(image, emotion_dict[maxindex], (x+20, y-60),
cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255, 255), 2, cv2.LINE_AA)

df1 = music_rec()

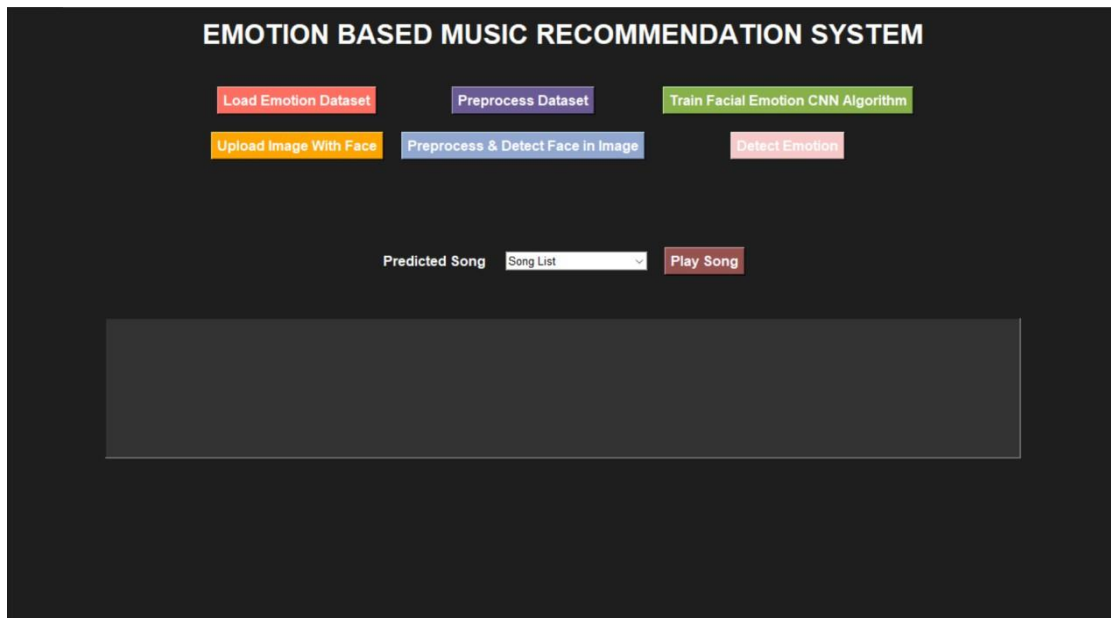
global last_frame1
last_frame1 = image.copy()
pic = cv2.cvtColor(last_frame1, cv2.COLOR_BGR2RGB)

img = Image.fromarray(last_frame1)
img = np.array(img)
ret, jpeg = cv2.imencode('.jpg', img)
return jpeg.tobytes(), df1

def music_rec():
    # print('----- Value -----', music_dist[show_text[0]])
    df = pd.read_csv(music_dist[show_text[0]])
    df = df[['Name', 'Album', 'Artist']]
    df = df.head(15)
    return df
```

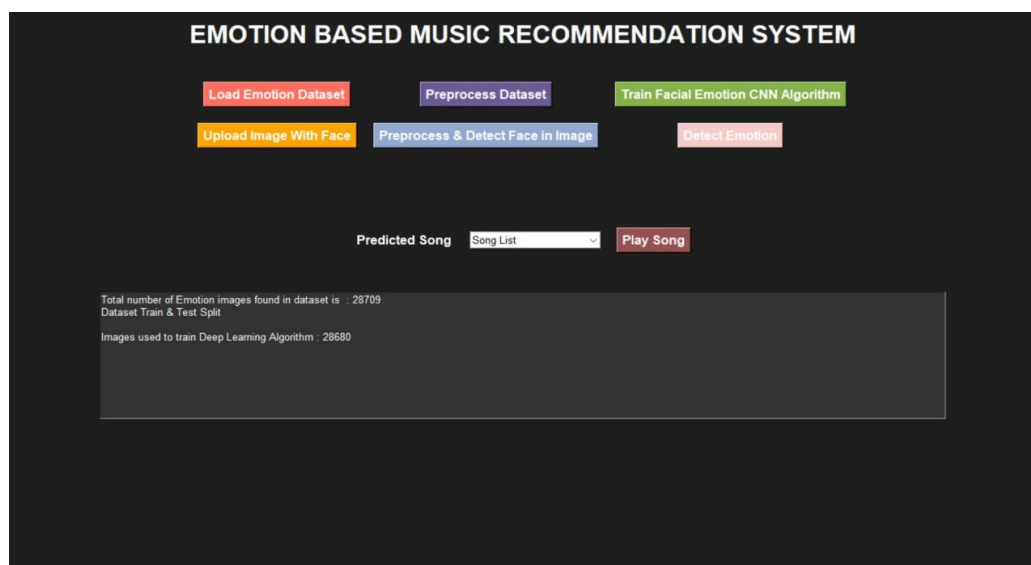
9. SAMPLE SCREENS

Screen 1: Upload Image with Face



Screen 1: In above screen click on ‘Upload Image with Face’ button to upload image

Screen 2: Process Data Set

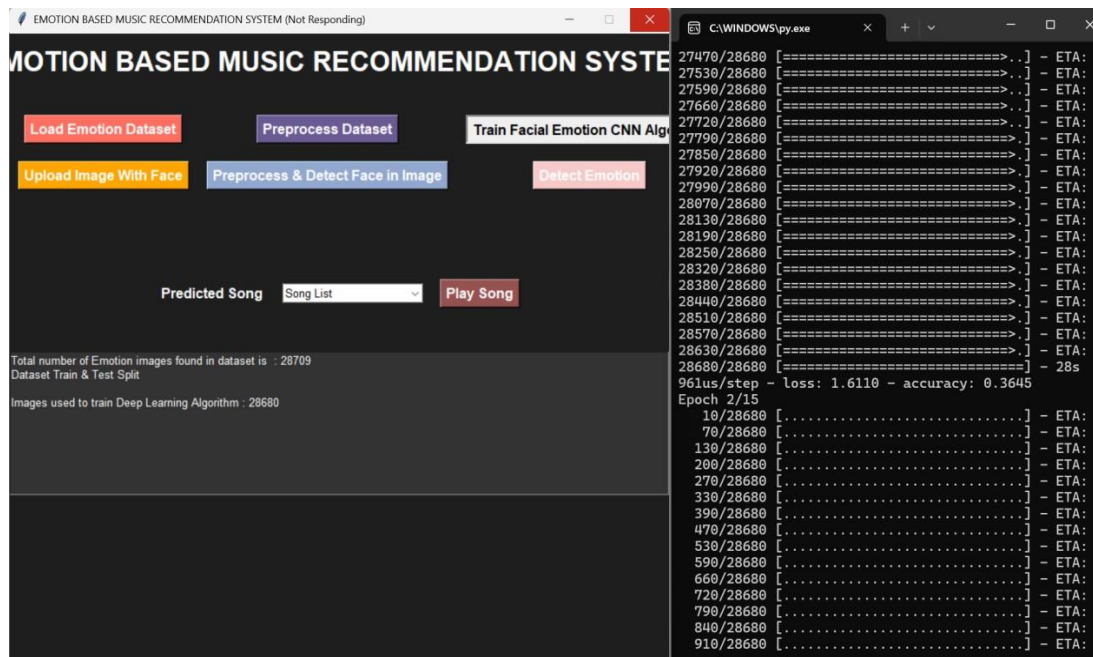


Screen 2: Process Data Set

DEVELOPMENT OF AI/ML FOR HUMAN EMOTION DETECTION USING FACIAL RECOGNITION

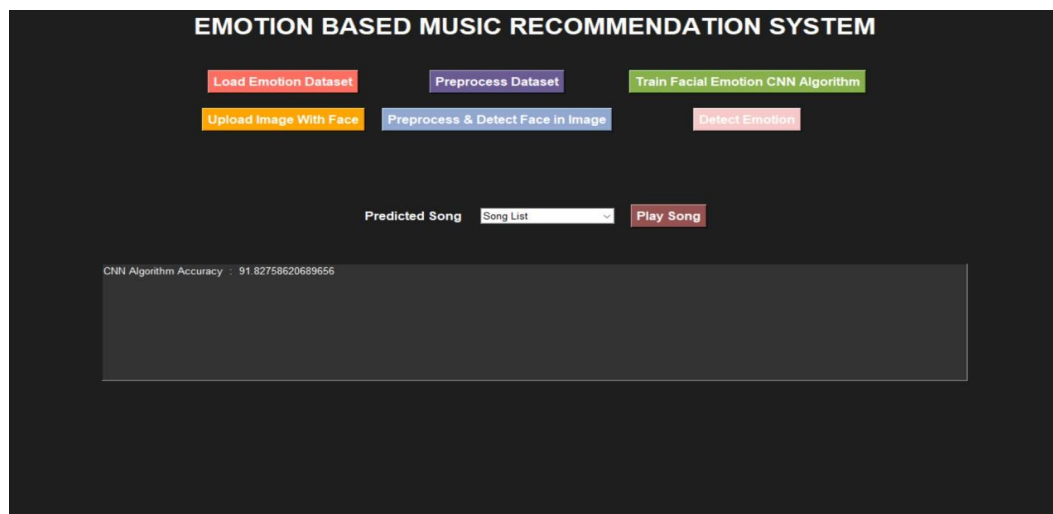
CHAPTER 9: SAMPLE SCREENS

Screen 3: Train the Model



Screen 3: Train the Model

Screen 4 : CNN Accuracy

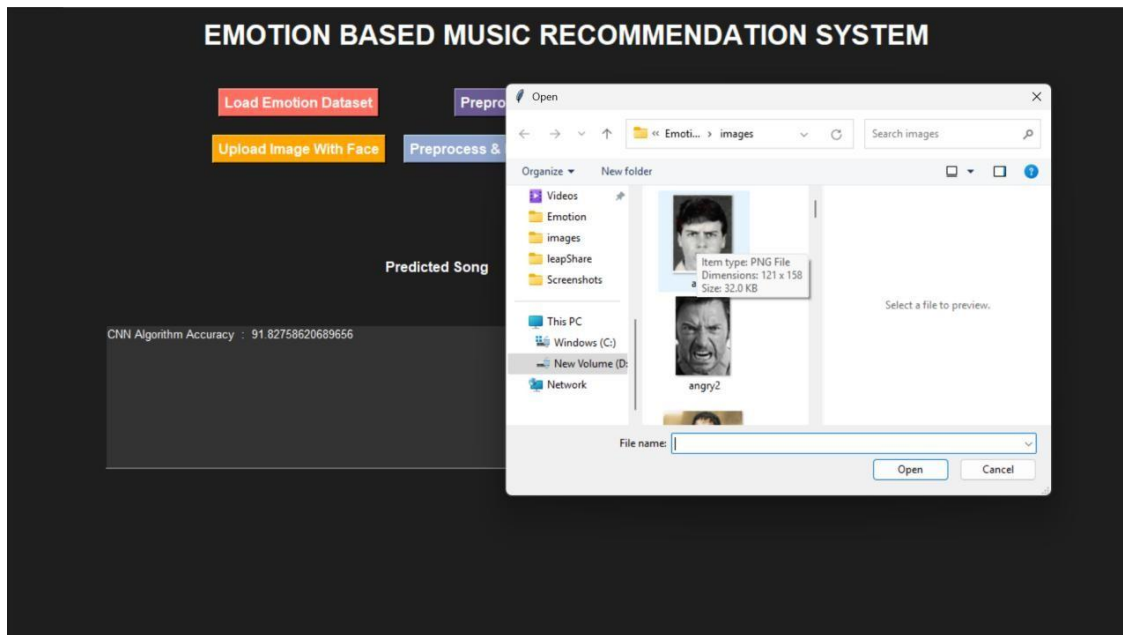


Screen 4: CNN Accuracy

DEVELOPMENT OF AI/ML FOR HUMAN EMOTION DETECTION USING FACIAL RECOGNITION

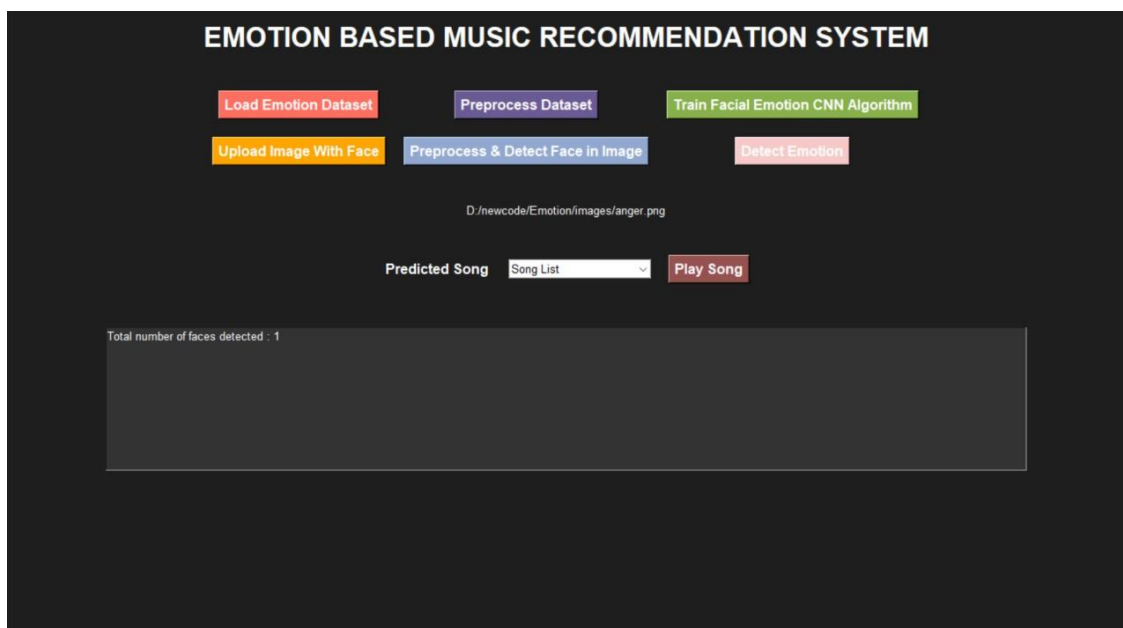
CHAPTER 9: SAMPLE SCREENS

Screen 5: Upload Random Picture of Human Face



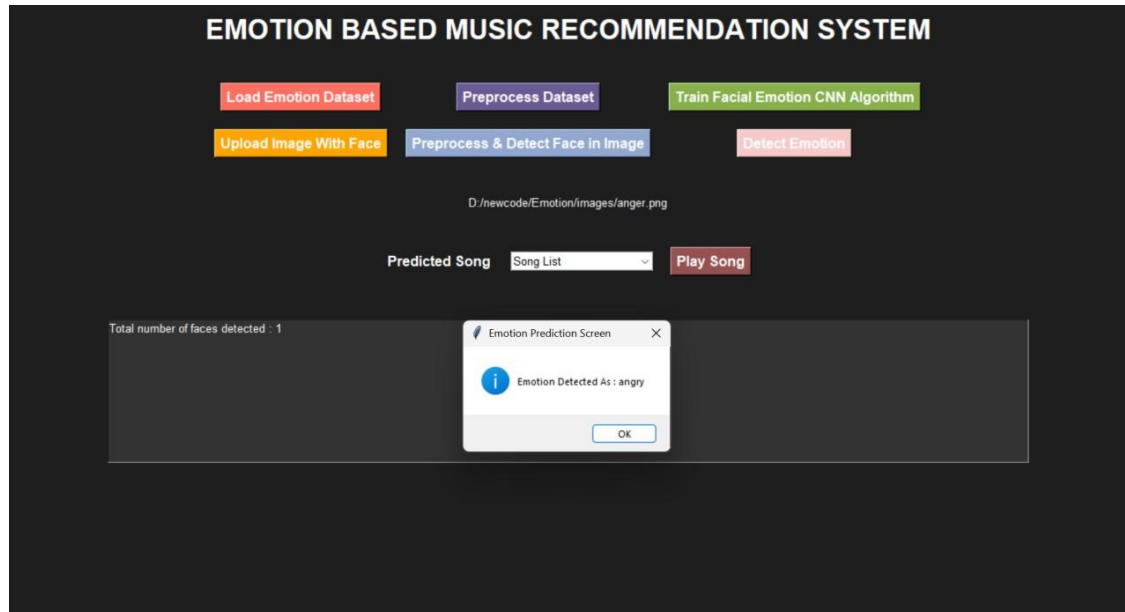
Screen 5 : Upload Random Picture of Human Face

Screen 6: Detection of Face



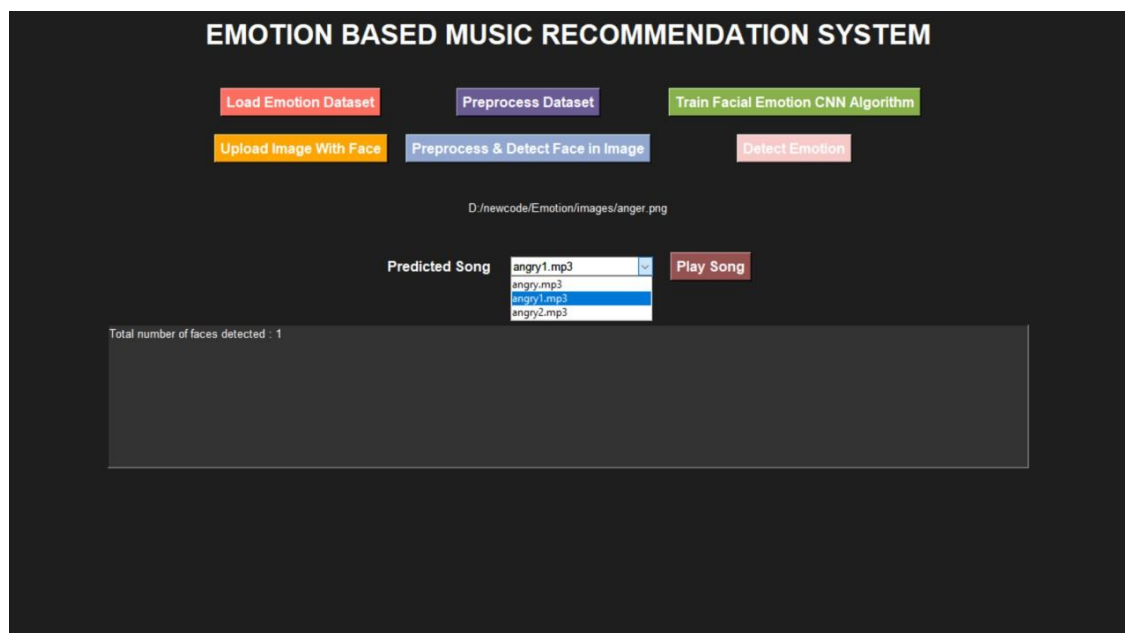
Screen 6 : Detection of Face

Screen 7: Detection of Emotion



Screen 7: Detection of Emotion

Screen 8 : Play song Based on emotion



Screen 8: Play song Based on emotion

10. CONCLUSION AND FUTURE SCOPE

10.1. CONCLUSION

Identification of human emotion based on human face emotions n has many applications in real life. It avoids the hectic work of choosing the song every time depending on the individual's mood. This paper gives the survey of various techniques and approaches have been proposed and developed to classify the human emotional state of behavior for playing music and the abstract view of the proposed system that we are going to implement. This presents Emousic, a new way of personalizing songs playlist by using machine learning techniques. Our solution works and gives better user preferable playlist. Due to the large number of classes, the performance of CNN is better than Decision tree algorithms. Since the experiment was performed on a small dataset and limited number of features, it still can be improved by adding more features like age, weather etc. More number of attributes will improvise decision making and prediction of song. Each user has it's own preferences about what kind of song is to be played for corresponding mood. some users listen sad songs when they are sad while some may prefer happy songs to change their mood. Collecting this data from every user can help us build better user specific radio application. Implementing this Emotion Detection in current music applications can provide better music experience to user.

10.2. FUTURE SCOPE

Future work involves comparison of different oversampling techniques the Music Player has changed in many different ways since its first launch. Now-a-days people like to get more out of different programs, so the design of apps and the process behind it has changed. User prefers interoperability and complexity but simplicity to use the application. The purpose of the Emotion-based music player is to introduce a well-known music player generate a playlist based on the user's feelings and in doing so provide the user with ease how to get playlists. Based on the information obtained the above content does not simply provide in depth information

**DEVELOPMENT OF AI/ML FOR HUMAN EMOTION DETECTION
USING FACIAL RECOGNITION
CHAPTER 10: CONCLUSION AND FUTURE SCOPE**

to justify software development program. The various components of the project have presented on the pages above in sufficient detail.

11. REFERENCES

1. Raut, Nitisha, “Facial Emotion Recognition Using Machine Learning” (2018). Master’s Projects. 632. <https://doi.org/10.31979/etd.w5fs-s8wd>
2. Hemanth P, Adarsh, Aswani C.B, Ajith P, Veena A Kumar, EMO PLAYER: Emotion Based Music Player, International Research Journal of Engineering and Technology (IRJET), vol. 5, no. 4, April 2018, pp. 4822-87.
3. Music Recommendation System: Sound Tree, Dcengo Unchained: Sla KAYA, BSc.; Duygu KABAKCI, BSc.; Insu KATIRCIOLU, BSc. and Koray KOCAKAYA BSc. Assistant : Dilek A–nal Supervisors: Prof. Dr. smail Hakk Toroslu, Prof. Dr. Veysi ler Sponsor Company: ARGEDOR
4. Tim Spittle, lucyd, GitHub, April 16, 2020. Accessed on: [Online], Available at: <https://github.com/timspit/lucyd>
5. Abdul, J. Chen, H.-Y. Liao, and S.-H. Chang, An Emotion-Aware Personalized Music Recommendation System Using a Convolutional Neural Networks Approach, Applied Sciences, vol. 8, no. 7, p. 1103, Jul. 2018.
6. Manas Sambare, FER2013 Dataset, Kaggle, July 19, 2020. Accessed on: September 9, 2020. [Online], Avalilable at: <https://www.kaggle.com/msambare/fer2013>
7. MahmoudiMA, MMA Facial Expression Dataset, Kaggle, June 6, 2020. Accessed on: September 15, 2020. [Online], Available at: <https://www.kaggle.com/mahmoudima/mma-facial-expression>
8. Dr. Shaik Asif Hussain and Ahlam Salim Abdallah Al Balushi, A real time face emotion classification and recognition using deep learning model, 2020 Journal. of Phys.: Conf. Ser. 1432 012087
9. P. Lucey, J. F. Cohn, T. Kanade, J. Saragih, Z. Ambadar and I. Matthews, “The Extended Cohn-Kanade Dataset (CK+): A complete dataset for action unit and emotion-specified expression,” 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition – Workshops, San Francisco, CA, USA, 2010, pp. 94-101, doi: 10.1109/CVPRW.2010.5543262.
10. Puri, Raghav & Gupta, Archit & Sikri, Manas & Tiwari, Mohit & Pathak, Nitish & Goel, Shivendra. (2020). Emotion Detection using Image Processing in Python.