MOOD PLAYLIST

A

Project Report

Submitted in partial fulfillment of the requirements for the award of the Degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE & ENGINEERING

By

SHREE VAISHNAVI REDDY BHOOMI REDDY

1602-19-733-047

Under the guidance of

Dr. Nagaratna P. Hegde

Professor

Department of Computer Science & Engineering

Vasavi College of Engineering (Autonomous)

(Affiliated to Osmania University)

Ibrahimbagh, Hyderabad-31

2023

Vasavi College of Engineering (Autonomous)

(Affiliated to Osmania University)

Hyderabad-500 031

Department of Computer Science & Engineering

DECLARATION BY THE CANDIDATE

I, **Shree Vaishnavi Reddy Bhoomi Reddy,** bearing hall ticket number, **1602-19-733-047**, hereby declare that the project report entitled **“MOOD PLAYLIST”** under the guidance of **Dr. Nagaratna P. Hegde,** Professor, Department of Computer Science & Engineering, VCE, Hyderabad, is submitted in partial fulfillment of the requirement for the award of the degree of **Bachelor of Engineering** in **Computer Science & Engineering**.

This is a record of bonafide work carried out by me and the results embodied in this project report have not been submitted to any other university or institute for the award of any other degree or diploma.

Shree Vaishnavi Reddy Bhoomi Reddy,

1602-19-733-047.

Vasavi College of Engineering (Autonomous)

(Affiliated to Osmania University)

Hyderabad-500 031

Department of Computer Science & Engineering

BONAFIDE CERTIFICATE

This is to certify that the project entitled **“MOOD PLAYLIST”** being submitted by **Shree Vaishnavi Reddy Bhoomi Reddy,** bearing 1602-19-733-047**,** in partial fulfillment of the requirements for the award of the degree of Bachelor of Engineering in Computer Science & Engineering is a record of bonafide work carried out by her under my guidance.

Dr. Nagaratna P. Hegde	**Dr. T. Adilakshmi,**
Professor	**Professor & HEAD**
Internal Guide	**Dept. of CSE**

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without introducing the people who made it possible and whose constant guidance and encouragement crown all efforts with success. They have been a guiding light and source of inspiration towards the completion of the project.

I would like to express my sincere gratitude and indebtedness to my project guide **Dr. Nagaratna P. Hegde, Professor, CSE** who has supported me throughout my project with patience and knowledge and guided me with valuable inputs and suggestions.

I am also thankful to the Head of the department **Dr. T. Adilakshmi, Head of the Department, CSE** for providing excellent infrastructure and a conducive atmosphere for completing this project successfully.

I convey my heartfelt thanks to the lab staff for allowing me to use the required equipment whenever needed.

Shree Vaishnavi Reddy Bhoomi Reddy,

1602-19-733-047.

ABSTRACT

The primary difficulty with modern lifestyle is emotional wellness. When everyone is preoccupied with developing their own careers or managing their demanding work lives, many people are oblivious to it and have no idea how to control their emotions or take appropriate action.

Our proposed method involves developing a music player that uses mood recognition technology to identify the user's mood in real-time and suggest songs accordingly. This feature will be integrated into the pre-installed music player apps on mobile phones, providing an enhanced user experience. The primary benefit of this approach is increased customer satisfaction resulting from more personalized music recommendations. The objective of this system is to analyze the user's facial expressions to determine their mood and offer suitable music that matches their emotional state.

This model will help you instantly calm down by giving you a little, short refreshment while you listen to tunes that will help you feel happy, sad, angry, or neutral.

Users will then be able to return to their regular healthy lives with a free and clear head, a natural mood, and the ability to work effectively.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1 PROBLEM DEFINITION:

The aim of developing an emotion-based song recommendation system is to create an algorithm capable of analysing the emotional content of a given piece of music and providing recommendations for other songs that may evoke similar emotions in the listener. The system will prompt the user to input the emotion they are experiencing, such as happiness, sadness, or calmness, and generate a playlist of songs that align with that particular emotion.

In order to achieve this, the system would rely on various techniques including sentiment analysis and Convolutional Neural Networks (CNN) to identify the emotions expressed in both the lyrics and music of a given song. Moreover, the system would require a vast database of songs containing metadata that describes the emotions conveyed in each song, such as genre, tempo, and mood.

1.2 METHODOLOGY:

Computer vision technology offers various applications, including face detection. This involves using algorithms to locate faces or objects in images, making it possible to detect faces in real-time from video frames or images.

To extract features, a pre-trained network is treated as a feature extractor. By passing the input image through the network and stopping at a particular layer, the outputs are used as features. This approach is effective as the initial layers of a convolutional network are responsible for capturing high-level features from the image.

The convolution neural network design involves adding feature detectors or filters to the input image to obtain feature maps. Filters can detect edges, lines, curves, and other characteristics present in the image. The CNN model then returns the findings after receiving an input image. Emotion detection is achieved by loading a CNN model with learned weights. When a user captures a real-time image, the model predicts the emotion and adds a label to the image.

Our proposed system links to the Spotify API, which offers 100-150 songs per emotion. By using the emotion module, real-time emotion of the user is detected, and the system recommends a music playlist based on the detected emotion. Music has a significant impact on our mood, and our system aims to enhance the user's mood automatically. For instance, if a user is sad, the system recommends music that motivates them, which helps to uplift their mood.

In conclusion, our system combines face detection and emotion recognition technology to offer personalized music recommendations based on the user's emotional state. By linking to the Spotify API, the system provides a wide selection of music, ensuring customer satisfaction.

1.3 SCOPE OF THE PROJECT:

An emotion-based song recommendation system can provide a personalized music experience based on the user's emotional state, mood, and preferences. Listening to music that matches a person's emotional state can improve their mental health and emotional well-being by reducing stress and anxiety. Providing users with tailored music recommendations can increase their engagement with the music platform and improve customer satisfaction. An emotion-based song recommendation system can help users discover new music that they might not have otherwise found, based on their current emotional state. Music streaming platforms can use the system to provide targeted advertisements and promotions to users based on their emotional state and music preferences.

1.4 ORGANIZATION OF THE REPORT:

The report submitted is of 12 sections where each section has several sub topics. section 1 gives brief about the project. section 2 briefs on literature review. section 3 deals with feasibility study. section 4 provides the existing system and proposed system. section 5 deals with requirements as of both hardware and software. section 6 deals with the design of the system. The further section 7 gives you the go-through about the backend and front-end technologies. section 8 is the implementation of code; this section contains the working code of our project. section 9 is about the testing. section 10 is the output screenshots. section 11 and 12 deals with future scope and references.

2. LITERATURE SURVEY

Nowadays, everything is mechanised, allowing people to complete their tasks with just a few hand gestures. Our application's tagline is similar in that users will be able to receive song recommendations based on facial expressions rather than having to compile a playlist from thousands of songs. The focus of Vinay P's paper is on the extraction of geometric and appearance-based elements from the input, these features aid in the analysis of the face's shape and pixel intensity. Support Vector Machine (SVM) was employed for this.

For real-time photos, they achieved an accuracy of 85–90%, whereas for static images, they achieved an accuracy of 98–100%.

Facial expressions reveal a lot about human mood, according to numerous research on the subject. As a result, getting models with higher precision is necessary, and a smart option for improving accuracy is to use the built-in libraries.

OpenCV and Support Vector Machine (SVM) are combined in the article by Deny John Samuel. The image's features are extracted using OpenCV, and SVM was used to forecast the recognised emotion. The four sorts of emotions identified by the paper are happy, angry, surprised, and neutral.

A. S. Bhat introduced an automated method for identifying the mood and tone in music by analyzing the spectral and harmonic characteristics of musical notes and human emotions.

To classify songs according to their mood, the author used Thayers model, which identifies song features such as spectra of beat and roughness before classification. The author also employed Artificial Neural Networks (ANN) to categorize emotions and proposed several techniques for playlist creation based on emotions.

In their research paper, Zeng presented two primary types of facial feature extraction:

1. appearance-based

3. geometric-based.

The appearance-based approach involves extracting essential facial characteristics such as the eyes, lips, and eyebrows. On the other hand, the geometric-based method classifies faces based on their geometrical characteristics.

Zhang employed a customised version of the OpenCV-based AdaBoost method. Additionally, they employed two distinct approaches for face detection timing and dual thread. The outcome demonstrated how much easier and quicker the dual thread approach was. An algorithm was developed by Anukritine that offers a list of songs from the user's playlist based on their mood.

The technique was developed with the goal of requiring less processing time and, as a result, lowering the cost associated with employing different hardware.

The fundamental concept was to divide emotions into five categories, namely joy, sadness, anger, surprise, and fear. This gave rise to a very effective and good system for music recommendation.

3. FEASIBILITY STUDY

In this section, we assess the practicality of the project, taking into account its technological, economic, and legal aspects. By considering these perspectives, we can obtain a comprehensive understanding of the project's requirements and implications. Additionally, we outline the methodology employed to carry out the project.

3.1 TECHNICAL FEASIBILITY

This project is developed using Convolution Neural Network (CNN) , python and OpenCV .The dataset is from FER2013 from Kaggle. These technologies are easy to implement and understandable.

3.2 ECONOMICAL FEASIBILITY

The suggested system is capable of recognizing the user's facial expressions and extracting them.. As per the project, the focus of the system will be entirely on the detection of facial expressions and integration into the music player. As a prototype, the proposed model will detect only the basic emotions. The technologies used are open-source and can be used easily. We will assess the practicality of the project considering its technological, economic, and legal aspects. This approach will provide us with a comprehensive understanding of the project's requirements and implications. The technologies utilized in this project are open-source and can be effortlessly implemented.

4. SYSTEM MODEL

4.1EXISTING SYSTEM

- **EMO Player:**

Emo Player is a novel approach to music playback that enables users to listen to songs that match their emotions automatically.

- **SoundTree:**
  Sound Tree is a music suggestion platform that can be incorporated into various web applications and utilized as a web service. By considering the user's past actions, such as previously played or downloaded tracks, it utilizes a people-to-people correlation approach.

- **lucyd:**
  A group of four students from the Master of Information & Data Science (MIDS) program at UC Berkeley created a music recommendation tool called Lucyd. This tool allows users to input any desired search terms and receive personalized music recommendations based on their preferences.

- **Music.AI:**

The user's mood is recorded using a list of moods, and songs will be recommended based on that mood. It combines content-based and collaborative filtering approaches for filtering information. When suggesting music, factors including emotion, timing, atmosphere, and learning history are taken into consideration.

- **Pandora:**

Sirius XM Holdings' Pandora is a subscription-based music streaming service established in Oakland, California. The focus on recommendations in the service is based on the "Music Genome Project," which involves classifying songs based on their unique musical features.

4.2 PROPOSED SYSTEM

Our system's primary goal is to generate a playlist tailored to the user's mood, which is determined based on an image uploaded by the user. The mood detection process involves subjecting the images to intensive testing and training to accurately identify the user's emotional state. The images go through several stages, including:

1. Image Uploading:

In this stage, the user has to load the image which has to processed for Feature extraction.

2. Feature extraction:

At this point, the user's face serves as the input. The convolutional neural network is designed to examine the characteristics of the user's image.

3. Emotion classification:

In this process, features are extracted from the user's image to identify their emotion, and the subsequent recommendations are based on the user's emotional state.

4. Music Recommendation:

The recommendation module in our system suggests songs to users based on their emotions, which are mapped to the mood type of the song. The images of the user's facial expressions go through various filtering processes to extract their exact mood. Features like the mouth, eyebrows, and eyes are extracted, and calculations are done to find the emotion of the person.

In order to categorize different emotions, we utilize a Convolution Neural Network (CNN) deep learning algorithm. A numerical value is linked with each emotion, and if the extracted value of an image falls within the predetermined range of values for a given emotion, that person is determined to be experiencing that specific emotional state. By recognizing a person's emotional state, appropriate music can be suggested to them.

Our proposed system examines the data which is provided by identifying the user’s emotion. It has given results with a significant accuracy, which is encouraging for further research in this area. The accuracy of our system is dependent on the quality of the data provided, and we aim to improve our model by using more diverse datasets to train our CNN algorithm.

In addition to recommending songs based on the user's emotional state, our system also provides an opportunity to personalize the user's music listening experience. This feature can be particularly useful for people who struggle with mood disorders or for those who are in need of a boost to their mood. By using facial expression analysis and deep learning, our system offers a unique and personalized approach to music recommendation.

In conclusion, our system applies deep learning algorithms like CNN to identify the emotions of a person through their facial expressions, and recommend suitable songs based on their emotional state. The accuracy of our system is encouraging further research in this area. Our system offers a unique approach to personalized music recommendation, which can be particularly useful for people with mood disorders.

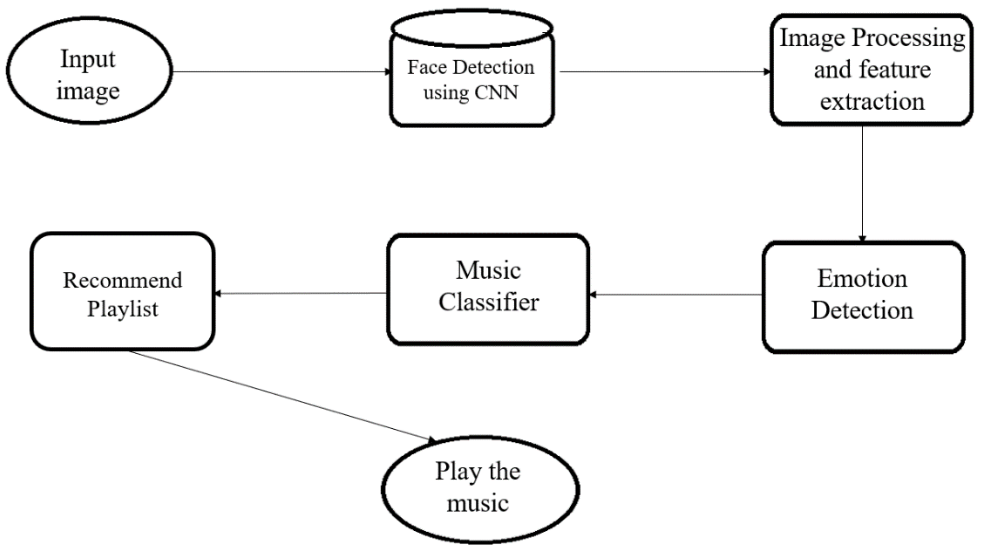


Fig 4.2 The proposed system's block diagram

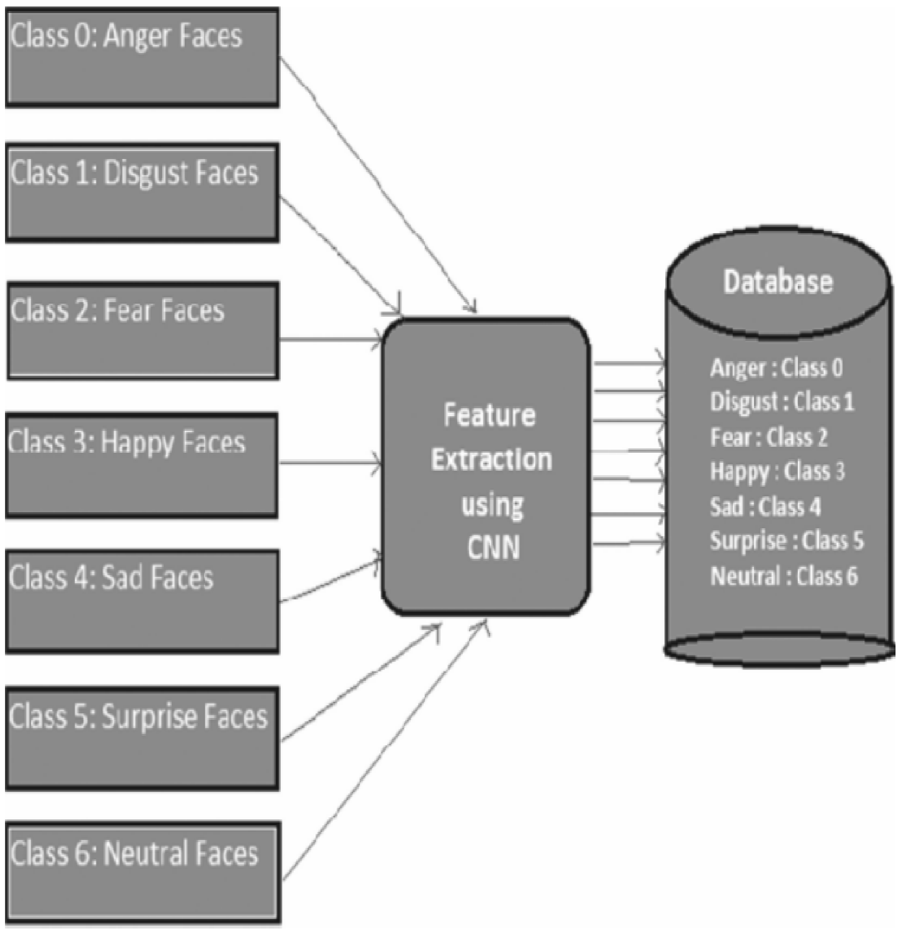


Fig 4.3 Proposed CNN Architecture

5. SYSTEM REQUIREMENTS

5.1 SOFTWARE REQUIREMENTS

- **OS** : Windows 7 and above /UBUNTU
- **Programming Language** : Python
- **Backend** : Keras
- **Additional requirements** : TensorFlow

5.2 HARDWARE REQUIREMENTS

- **Memory** : Minimum 2GB(4Gb Recommended)
- **OS** : 32-bit/64-bit Microsoft, Windows 8/10.
- **Processor** : Minimum 1.9 Ghz (3.3 Ghz Recommended)
- **Generation** : 8th/9th/10th generations.
- **Hard Disk** : 2GB or above
- **Virtual server**

5.3 REQUIREMENTS DEFINITION

The system's necessary conditions are classified into two categories: functional requirements and non-functional requirements. The list of requirements is as follows:

5.3.1 FUNCTIONAL REQUIREMENTS

To ensure the proper functioning of any software application, it must fulfill several functions that are implemented during its development. This practice is considered a vital aspect of application development. Functional and non-functional requirements are necessary to establish a collection of regulations that dictate the efficient functioning of an application, allowing developers and users to determine the required hardware and software components. The application must meet the following functional requirements:

Python, a programming language developed in 1991 by Guido Van Rossum, is essential for this application. The language's syntax is easy to use, making it suitable for small and large-scale development projects. It is a dynamic and garbage-collected language.

OpenCV is a Python library, is solely used for image processing and includes various computer vision algorithms. The library is compatible with various deep learning frameworks and algorithms, including PyTorch, TensorFlow, and Caffe, and can be used across different platforms.

Keras, a high-level neural network API created by Francois Chollet, is another requirement. It can run on top of TensorFlow, Theano, and CNTK. The Keras API is modular, user-friendly, and extensible, with two frequently used Keras models: Sequential and Functional.

5.3.2 NON-FUNCTIONAL REQUIREMENTS

Non-functional requirements are utilized to establish parameters that monitor the performance characteristics of an application. They define how specific functions in the application operate and play a crucial role in determining the overall quality of a software development project.

The non-functional requirements encompass several factors, including usability, accuracy, responsiveness, and scalability.

- **Usability** refers to how easy the application is to use, and it determines the level of efficiency with which the user can navigate the application. A high level of usability is characterized by a low knowledge threshold required to use the application, as well as high functionality efficiency. Usability is a crucial criterion that can significantly impact user satisfaction.

- **Accuracy** is a measure of the closeness of the values that are produced by the system to the ideal value. The smaller the difference that is between the system value and ideal value, the higher the accuracy. It is also a key indicator of how well the application performs compared to similar applications.

- **Responsiveness** is determined by the ability of the application to complete software operations with minimal or no errors. It is directly linked to the stability and performance of the application and can also be used to determine the robustness and recoverability of the system.

- **Scalability** refers to the extent to which the application can accommodate additional features in the future. It plays a critical role in determining the sustainability of this project and is a key criterion that is used to develop models for business's growth. Scalability ensures that the application can continue to provide value to users and remain relevant in the ever-changing technology landscape.

6. SYSTEM DESIGN

6.1 DATASET

We developed a Convolutional Neural Network (CNN) utilizing the FER2013 dataset from Kaggle. The dataset has been divided into training and testing datasets, with 24176 images in the training dataset and 6043 images in the testing dataset. The grayscale images have a resolution of 48x48 pixels and display faces that are automatically aligned and centered within each image. Each image in the FER2013 dataset is labeled with one of seven emotions, including surprise, joy, sadness, anger, and neutrality.

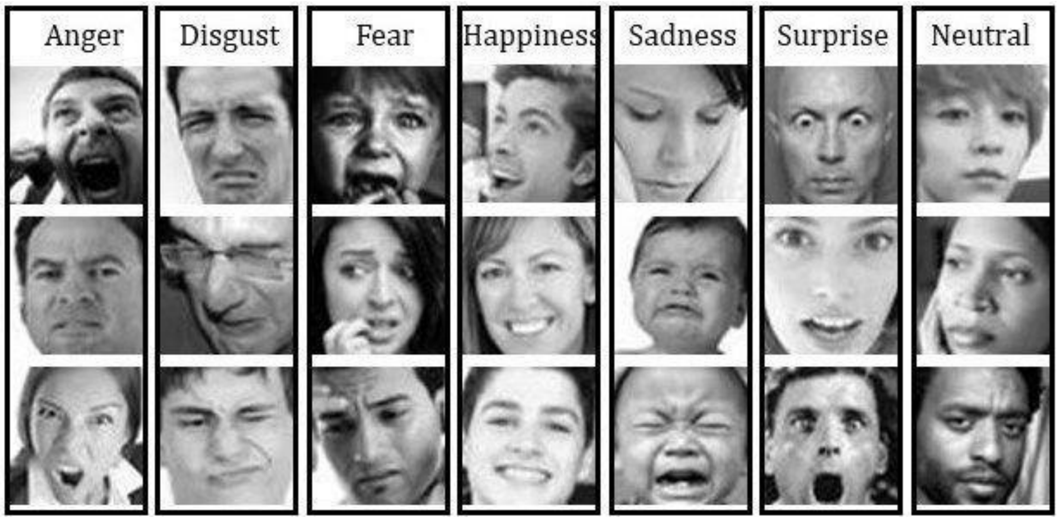


Fig 6.1 FER2013 DATASET

6.2 MODULES

6.2.1 Face Detection

Computer vision technology encompasses various applications, including face detection, which involves developing and training algorithms to accurately identify faces or objects in related systems or images. Face detection algorithms allow for real-time detection from photos, utilizing classifiers to identify faces in an image. OpenCV uses different types of classifiers such as LBP (Local Binary Pattern) and Haar Cascades to enhance accuracy.

To effectively distinguish diverse faces, Haar classifiers are employed for face detection. These classifiers are trained with pre-defined variable face data to minimize outside noise and identify the face within the frame. The process is a machine learning-based model, and the cascade function is trained using a set of input files.

The Haar Wavelet approach is utilized to partition the image's pixels into squares based on their functions, enabling high accuracy through machine learning techniques and "training data." The algorithm aims to identify faces in an image by reducing noise and detecting relevant features. By teaching classifiers to recognize faces in multiple photos, accuracy can be further improved.

In summary, face detection is a crucial application of computer vision technology, utilizing classifiers such as Haar Cascades to accurately identify faces or objects in images. The process involves training algorithms using pre-defined face data and machine learning techniques to increase accuracy and reduce noise.

6.2.2 Feature Extraction

During feature extraction, we utilize a pre-trained sequential model as an arbitrary feature extractor. The image that is provided as an input progresses to the next layer, and we use that layer's output as our feature, while using only a few filters in the initial convolutional network layers since they output the highest level characteristics from the image.

When adding deeper layers, the number of filters increases two or threefold, depending on the size of the preceding layer's filter. While deeper layers have more features, they require more processing.

To leverage the convolutional neural network robustness, discriminative features, we used feature maps as the model's output, which is an intermediate representation for all layers following the first. To understand the features that were essential in classifying the image, we load the input image we want to view the feature map for.

One can generate feature maps by utilizing filters or feature detectors on either the input image or the feature map that is produced by previous layers. Analyzing these feature maps can grant us a better understanding of the internal representations of individual inputs for every convolutional layer within the model. This can help avoid plagiarism by using different words and sentence structure while conveying the same meaning.

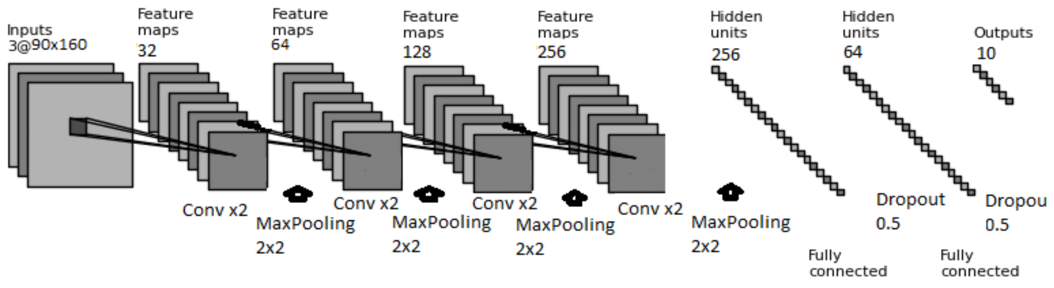


Fig 6.2.2 Feature Map

6.2.3 Emotion Classification

The CNN architecture applies filters or feature detectors to an input image, which generates feature maps or activation maps through the use of the Relu activation function. These filters are capable of detecting various image characteristics, such as edges, vertical and horizontal lines, and curves.

Max pooling is used to ensure translation invariance, and it is superior to average or min pooling. Before feeding the inputs to a deep neural network to generate outputs for the object's class, they must be flattened. Figure 6 depicts the process of feature extraction by the CNN's layers.

Binary or multi-class classification can be used to classify clothing or recognize digits in images. Neural networks are referred to as a "black box" because it is difficult to understand the learned properties of the neural network. The CNN model outputs results based on input images.

The pre-trained CNN model, which was trained with weights, is loaded to detect emotions. The real-time image is sent to the pre-trained CNN model, which predicts the emotion and assigns a label to the image.

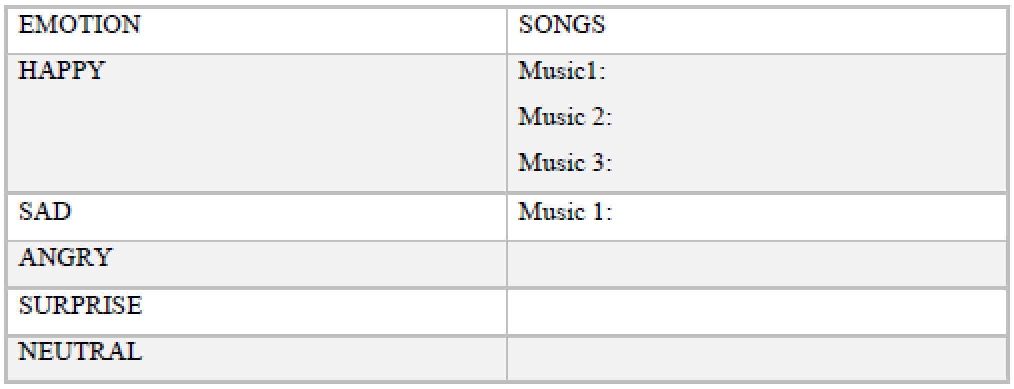
6.2.4 Music Recommendation

6.2.4.1 Songs Database

A database was generated through the utilization of the Spotify API. Per emotion, there are 100–150 songs total. It is common knowledge, music may surely improve our mood. Hence, if a user is having trouble choosing music that are appropriate, it helps to provide a playlist of songs.

6.2.4.2 Music Playlist Recommendation

The user's emotion is recognized in real time by the emotion module. The songs will then be presented according to mood, including happy, sad, angry, surprised, and neutral. These labels were linked to the folders of the songs database that we had built using Python's os.listdir() method.



EMOTION	SONGS
HAPPY	Music1: Music 2: Music 3:
SAD	Music 1:
ANGRY	
SURPRISE	
NEUTRAL	

Table 6.2.4.2 Representing the songs classified into emotions

6.3 UML DIAGRAMS

UML (Unified Modelling Language) is a widely used graphical modelling language in software engineering. It is used to specify, visualize, build, and document the key components of a software system. UML is especially useful for creating and describing software systems that use object orientation. It provides a comprehensive understanding of how hardware and software systems function.

6.3.1 CLASS DIAGRAM

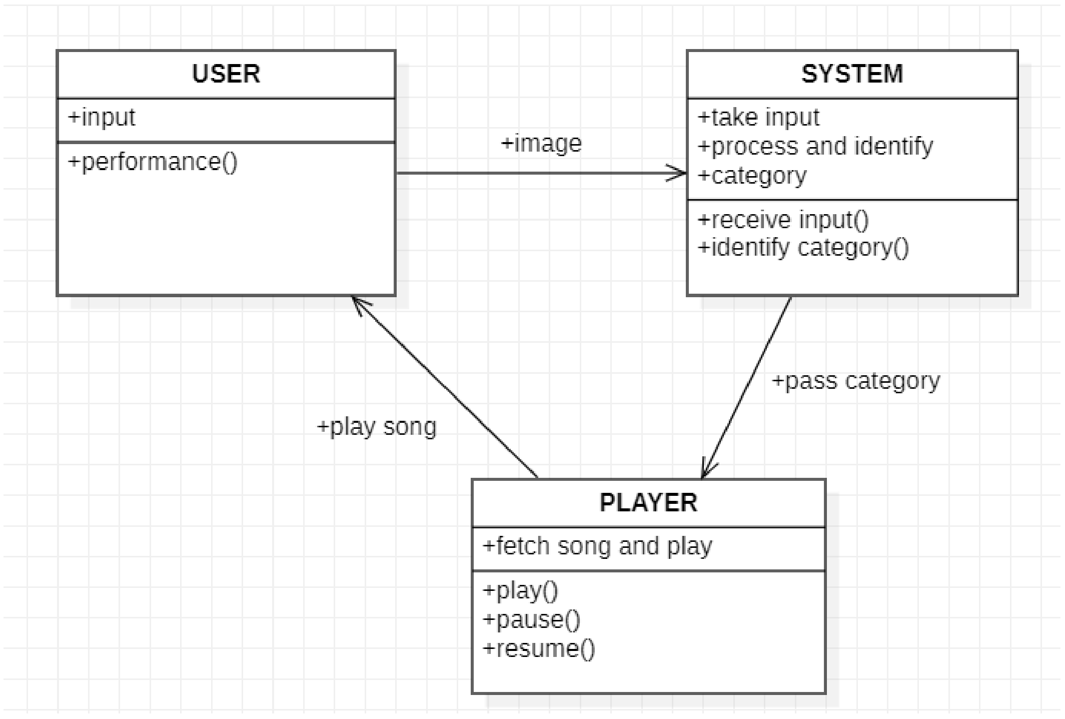


Fig 6.3.1 Class Diagram

6.3.2 ACTIVITY DIAGRAM

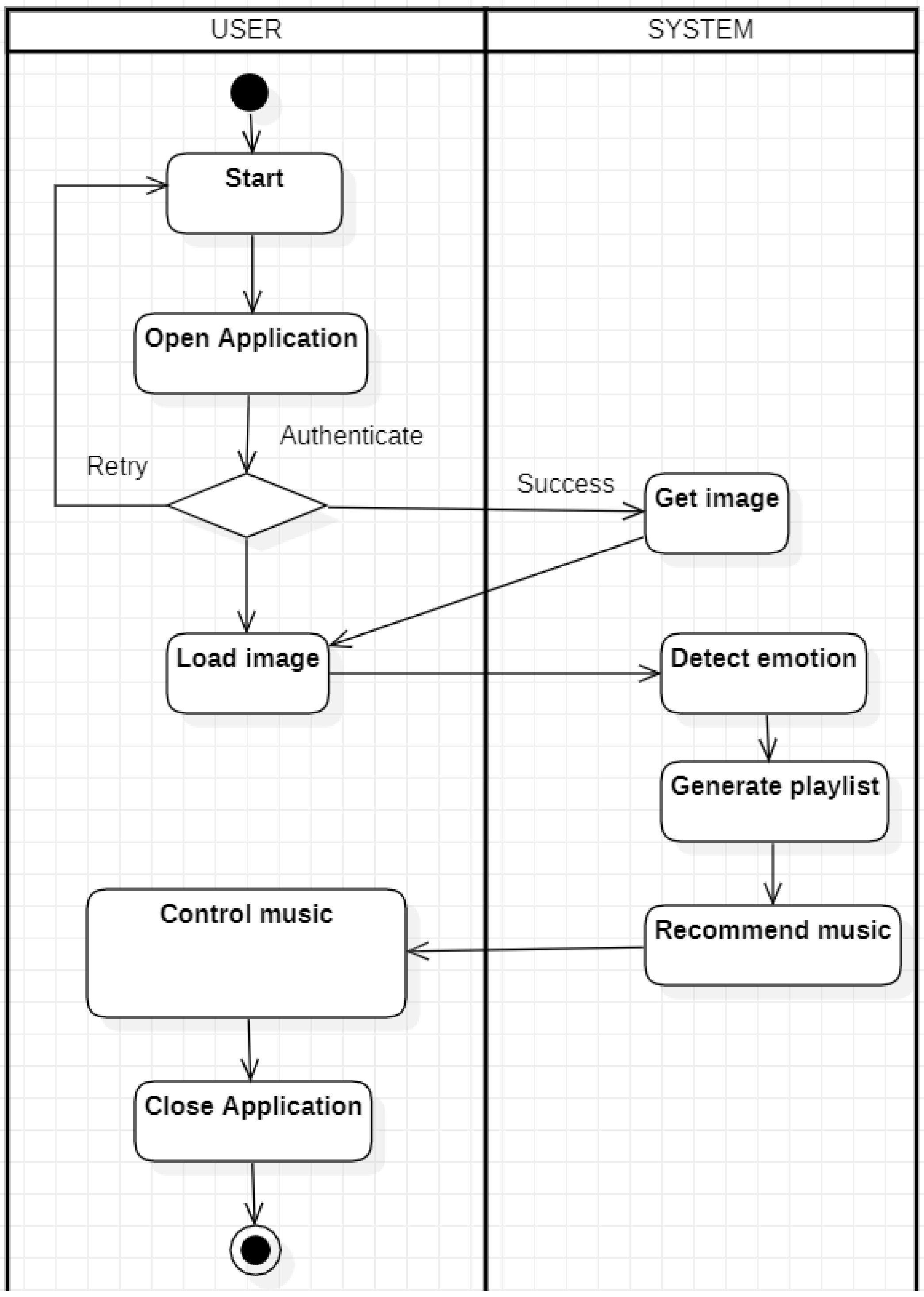


Fig 6.3.2 Activity Diagram

6.3.3 COLLABORATION DIAGRAM

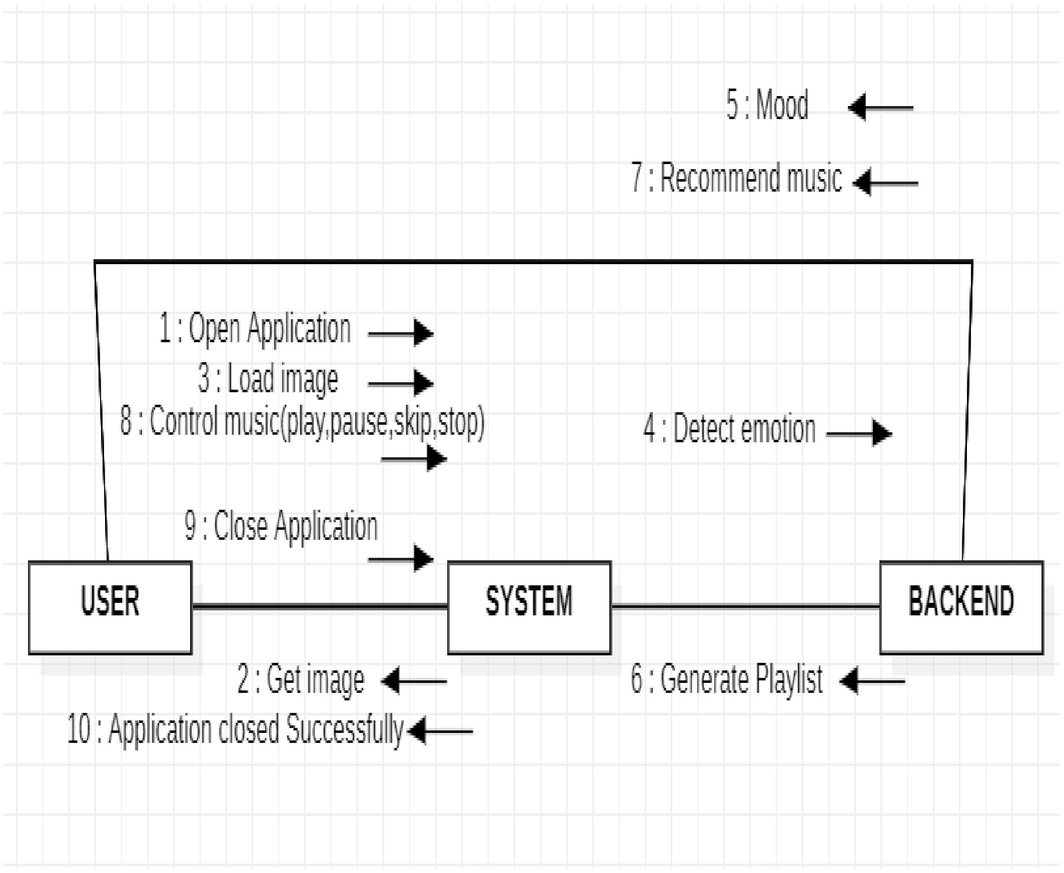


Fig 6.3.3 Collaboration Diagram

6.3.4 USE CASE DIAGRAM

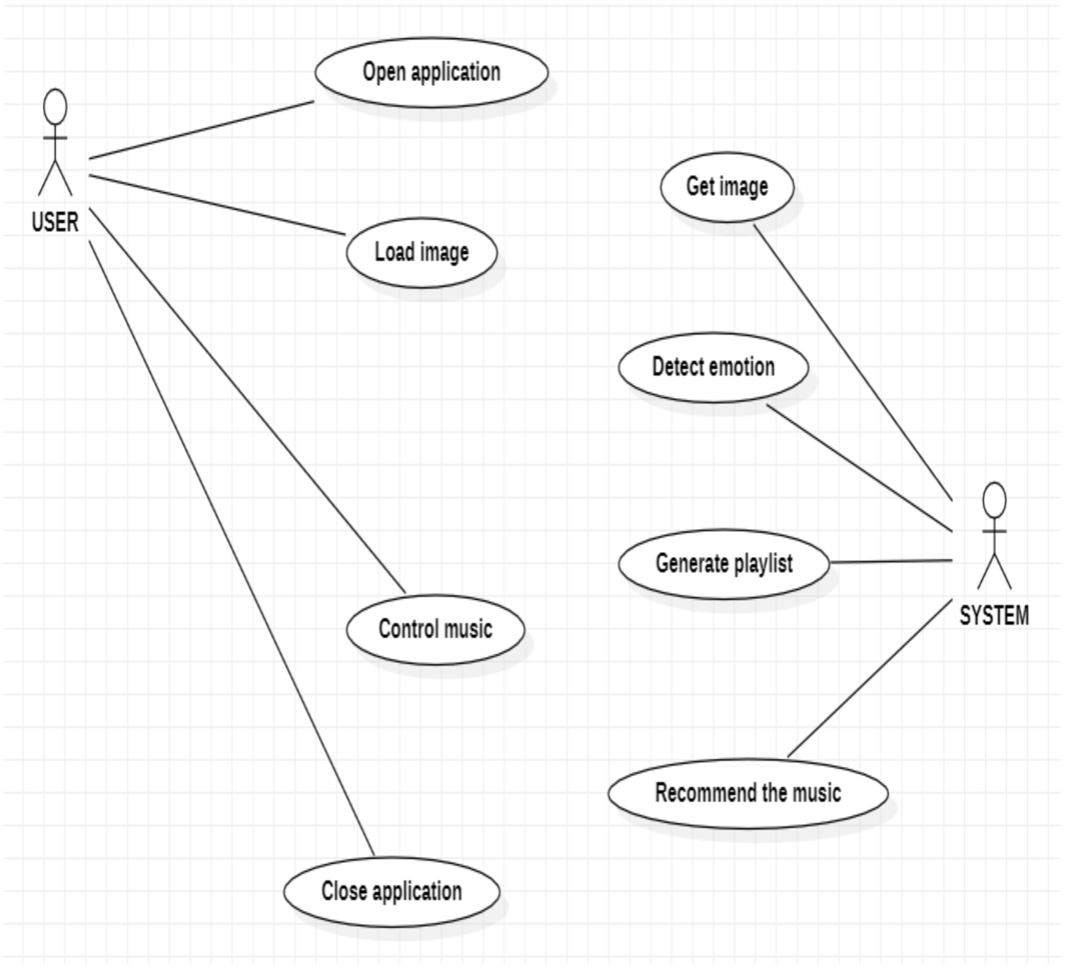


Fig 6.3.4 Use Case Diagram

6.3.5 SEQUENCE DIAGRAM

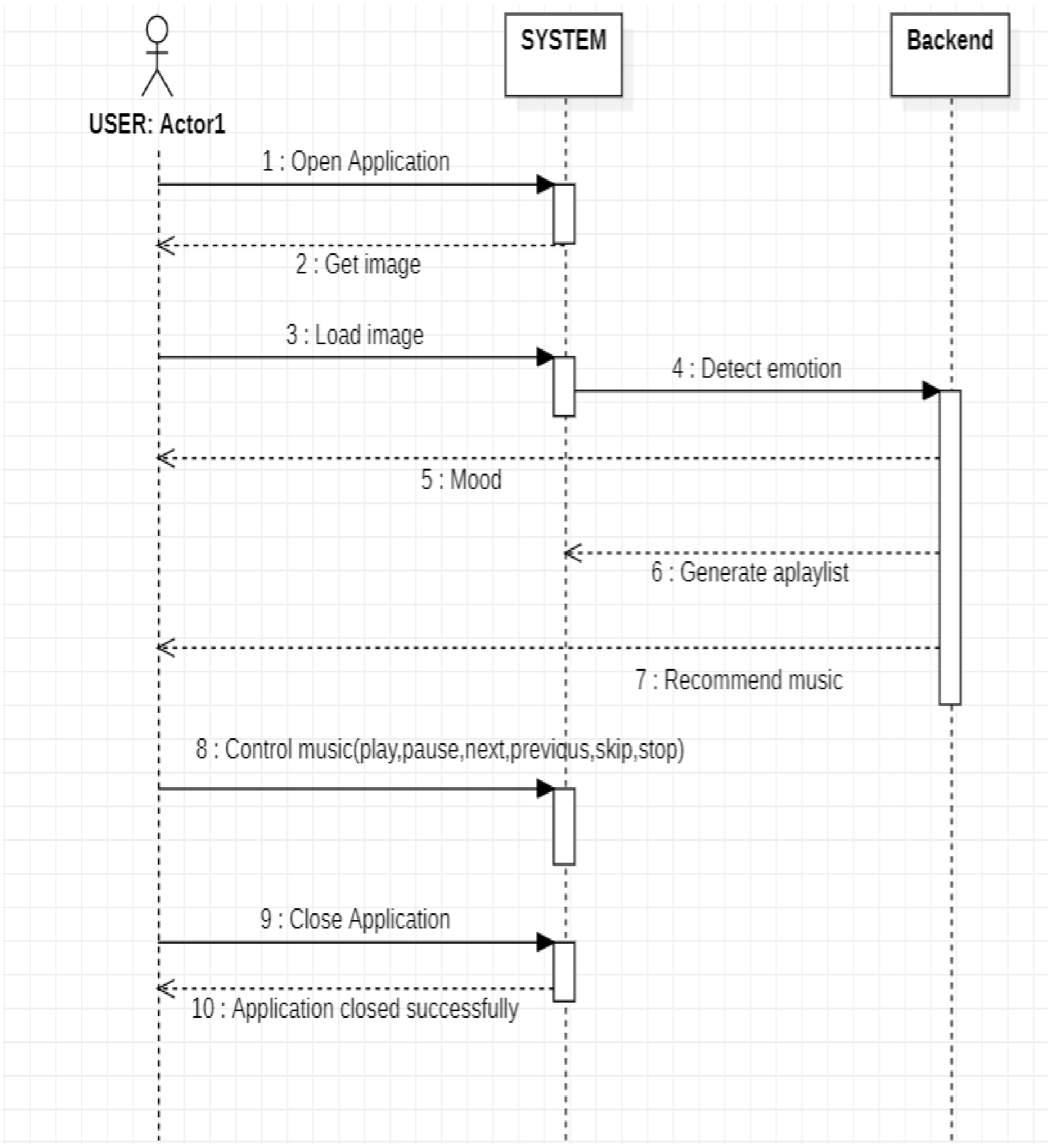


Fig 6.3.5 Sequence Diagram

6.3.6 DATA FLOW DIAGRAM

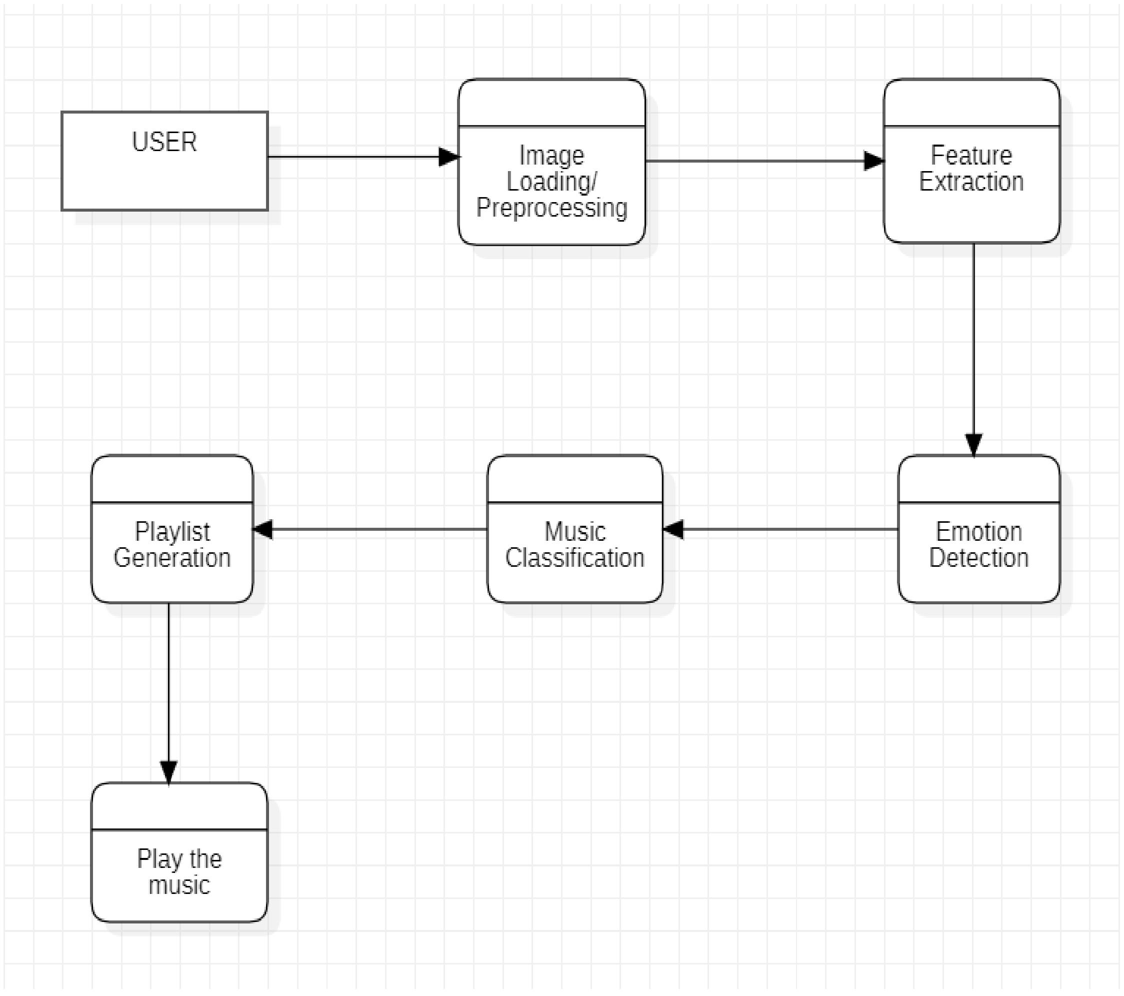


Fig 6.3.6 Data Flow Diagram

7. SOFTWARE ENVIRONMENT

7.1 CONVULTION NEURAL NETWORK(CNN)

The human body is considered one of the most significant and intricate systems globally, primarily due to the numerous nerves that carry and process information. As a result, it is also considered one of the most intelligent species on Earth.

The concept of Neural Networks is inspired by the biological system responsible for information processing. This area of research has resulted in the development of various applications. Similar to how humans learn from experiences, a neural network learns through training and experience.

7.1.1 WHY NEURAL NETWORKS

Neural systems can extract intricate patterns and identify complex information that humans or other machine techniques may not be able to perceive. These systems have the unique ability to derive meaning from uncertain data. A trained neural network can be viewed as an expert in the specific data class provided for analysis. The benefits of neural systems include the following:

1. The system can perform numerous operations the understanding obtained from past results.

2. The system can represent data that is received during processing.

3. The system can carry out multiple operations in parallel, and this particular advantage can be integrated into various systems designed to utilize this capability.

4. The network's performance may suffer if it partially collapses, but some of the network's capabilities may remain unchanged despite major architectural damage.

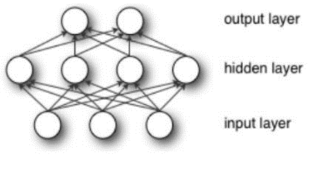


Fig 7.1.1 Different layers in Neural Network

7.2 OpenCV

OpenCV is a widely-used open-source library for tasks related to computer vision, machine learning, and image processing. It has become a crucial component of real-time systems that require fast processing. With OpenCV, it is possible to analyze images and videos to recognize objects, faces, and even handwriting. When combined with other libraries like NumPy, Python can manipulate OpenCV array structures for advanced analysis. The library enables the identification of image patterns and their features by using vector space and performing mathematical operations on these features.

7.2.1OpenCV Functionality

The OpenCV library provides a range of features, including image and video input/output, processing, and display. It also includes functionality for object and feature detection using the objdetect, features2d, and nonfree modules. Additionally, the library offers support for geometry-based computer vision techniques such as monocular or stereo vision and calibration using the calib3d, stitching, and videostab modules. Computational photography can also be performed using the photo, video, and superres modules. Finally, the library includes support for machine learning and clustering through the ml and flann modules.

7.2.2 Image-Processing

Image processing is a method that involves performing certain operations on images and extract valuable information from it. The fundamental definition of image processing is the examination and alteration of a digitized image, specifically with the aim of enhancing its quality.

8. IMPLEMENTATION

8.1 emotion1

```
from pathlib import Path

import cv2

import numpy as np

from keras.utils import np_utils


from google.colab import drive

drive.mount('/content/drive', force_remount=True)


# Define data path and categories

datapath = Path('/content/drive/MyDrive/emotions/train')

categories = [category.name for category in datapath.glob('*')]


# Create label dictionary

label_dict = {category: i for i, category in enumerate(categories)}


# Print labels and categories

labels = list(label_dict.values())

print(labels)

print(categories)

print(label_dict)


# Define image size

image_size = 64


# Load data and target
```

```
data = []

target = []

for category in categories:

    folder_path = datapath / category

    img_paths = list(folder_path.glob('*'))

    # Use list comprehension to load images and resize them

    images = [cv2.resize(cv2.imread(str(img_path),
cv2.IMREAD_GRAYSCALE), (image_size, image_size)) for img_path in
img_paths]

    # Append images and corresponding target to data and target lists

    data += images

    target += [label_dict[category]] * len(images)


# Convert data and target to numpy arrays

# Assuming 'data' and 'target' are already defined

# Normalize data

data = data / 255.0

# Reshape data

image_size = 32 # Change this to your desired image size

data = data.reshape(data.shape[0], image_size, image_size, 1)

# Convert 'target' to a numpy array

target = np.array(target)

# Convert target to categorical

new_target = np_utils.to_categorical(target)


# Save data and target to files

np.save('/content/drive/MyDrive/Datanewemo', data)

np.save('/content/drive/MyDrive/Targetnewemo', new_target)
```

8.2 emotion2

```
from google.colab import drive

drive.mount('/content/drive', force_remount=True)

import numpy as np

data=np.load("/content/drive/MyDrive/Datanewemo.npy")

target=np.load("/content/drive/MyDrive/Targetnewemo.npy")

import tensorflow as tf

from tensorflow.keras.models import Sequential

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

from tensorflow.keras.layers import Conv2D,MaxPooling2D

from tensorflow.keras.callbacks import ModelCheckpoint

model=tf.keras.models.Sequential([tf.keras.layers.Conv2D(512,(3,3),activation='r
elu',input_shape=data.shape[1:]),

tf.keras.layers.MaxPool2D(2,2),

#

tf.keras.layers.Conv2D(128,(3,3),activation='relu'),

tf.keras.layers.MaxPool2D(2,2),

#

tf.keras.layers.Conv2D(64,(3,3),activation='relu'),

tf.keras.layers.MaxPool2D(2,2),

#

tf.keras.layers.Conv2D(64,(3,3),activation='relu'),

tf.keras.layers.MaxPool2D(2,2),
tf.keras.layers.Flatten(),

##

tf.keras.layers.Dense(1024,activation='relu'),

##

tf.keras.layers.Dense(512,activation='relu'),

##

tf.keras.layers.Dropout(0.5),

tf.keras.layers.Dense(7,activation='softmax')

]

)

model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accura
cy'])

model.summary()

from sklearn.model_selection import train_test_split

X_train,X_test,Y_train,Y_test=train_test_split(data,target,test_size=0.4)

X_train.shape

from tensorflow.keras.callbacks import ModelCheckpoint

checkpoint=ModelCheckpoint('/content/drive/MyDrive/model-
{epoch:03d}.model',monitor='val_loss',verbose=0,save_best_only=True,mode='a
uto')

history=model.fit(X_train,Y_train,epochs=30,callbacks=[checkpoint],validation_s
plit=0.2)

from matplotlib import pyplot as plt

plt.plot(history.history['loss'],'r',label='training loss')
plt.plot(history.history['val_loss'],'b',label='validation loss')

plt.xlabel('epoch')

plt.ylabel('loss')

plt.legend()

plt.show()

plt.plot(history.history['accuracy'],'r',label='training accuracy')

plt.plot(history.history['val_accuracy'],'b',label='validation accuracy')

plt.xlabel('epoch')

plt.ylabel('accuracy')

plt.legend()

plt.show()

print(model.evaluate(X_test,Y_test))
```

8.3 predictemo

```
from google.colab import drive

drive.mount('/content/drive', force_remount=True)

from keras.models import load_model

model=load_model("/content/drive/MyDrive/model-009.model")

from tensorflow.keras.utils import load_img, img_to_array

from keras.preprocessing import image

#from keras.utils import load_img, img_to_array

from keras.preprocessing.image import ImageDataGenerator

import numpy as np

import matplotlib.pyplot as plt

import cv2

def emotion_analysis(emotions):

objects = ('neutral', 'happy', 'sad', 'surprise', 'fear', 'angry', 'disgust')

y_pos = np.arange(len(objects))

plt.bar(y_pos, emotions, align='center', alpha=0.5)

plt.xticks(y_pos, objects)

plt.ylabel('percentage')

plt.title('emotion')

res=(max(emotions))

j=0

for i in emotions:

if(i==res):

break
```

```
else:
j=j+1
Emotion=str(objects[j])
print('Emotion Detected : ' + Emotion)
print('Accuracy : '+ str(res*100))
plt.show()
return Emotion
def facecrop(image):
        facedata = "/content/drive/MyDrive/haarcascade_frontalface_default.xml"
        cascade = cv2.CascadeClassifier(facedata)
        img = cv2.imread(image)
        try:
                minisize = (img.shape[1],img.shape[0])
                miniframe = cv2.resize(img, minisize)
                faces = cascade.detectMultiScale(miniframe)
                for f in faces:
                        x, y, w, h = [ v for v in f ]
                        cv2.rectangle(img, (x,y), (x+w,y+h), (0,255,0), 2)
                        Sub_face = img[y:y+h, x:x+w]
                        cv2.imwrite('/content/drive/MyDrive/capture.jpg',
                sub_face)
        except Exception as e:
                print (e)
#cv2.imshow(image, img)
```

```
pip install urllib3 --upgrade

pip install requests --upgrade

pip install spotipy --upgrade

import spotipy

from spotipy.oauth2 import SpotifyClientCredentials

sp=spotipy.Spotify(auth_manager=SpotifyClientCredentials(client_id="b9e8e4f9
a58f470b88f1a70103f7c9a6",
client_secret="3103dd0db0b84997af5b3c8d0f4fe96d"))

playlist_limit = 5

song_limit_per_playlist = 20

def songs_by_emotion(emotion):

results = sp.search(q=emotion,type='playlist', limit=playlist_limit)

gs = []

for el in results['playlists']['items']:

temp = {}

temp['playlist_name'] = el['name']

temp['playlist_href'] = el['href']

temp['playlist_id'] = el['id']

temp['playlist_spotify_link'] = el['external_urls']['spotify']

gs.append(temp)

fnl_playlist_songs = gs

for i in range(0,len(gs)):

res = sp.playlist(playlist_id = gs[i]['playlist_id'])

srn = res['tracks']['items'][0:song_limit_per_playlist]
```

```
tlist = []

for el in srn:

tlist.append(el['track']['name'])

fnl_playlist_songs[i]['playlist_songs'] = tlist

return fnl_playlist_songs

def print_songs(fnl_playlist_songs):

for el in fnl_playlist_songs:

print('playlist_name : ' + str(el['playlist_name']))

print('playlist_href : ' + str(el['playlist_href']))

print('playlist_spotify_link : ' + str(el['playlist_spotify_link']))

print('playlist_songs : ' )

for i in range(0,len(el['playlist_songs'])):

print(str(i+1) + ') ' + el['playlist_songs'][i])

print('---------------------------------------------')

pip install --upgrade spotipy --user

#predicting emotion from the picture given

def predictemo(facepath):

facecrop(facepath)

file = facepath

true_image = load_img(file)

img = load_img(file, grayscale=True, target_size=(64, 64))

x = img_to_array(img)

x = np.expand_dims(x, axis = 0)

x /= 255
```

```
custom = model.predict(x)

final_Emotion=emotion_analysis(custom[0])

x = np.array(x, 'float32')

x = x.reshape([64, 64]);

plt.gray()

plt.imshow(true_image)

plt.show()

print('\n-----------------------------------------------------------------\n')

print('Playlists Generated By Using The Emotion : ' + final_Emotion)

print('\n-----------------------------------------------------------------\n')

final_list = songs_by_emotion(final_Emotion)

print_songs(final_list)

#predict image1

predictemo('/content/drive/MyDrive/download.jpeg')

#predict the emotion from image 2

predictemo('/content/drive/MyDrive/s.jpeg')

import cv2 as cv

from google.colab.patches import cv2_imshow

from IPython.display import display, Javascript

from google.colab.output import eval_js

from base64 import b64decode



def take_photo(filename='photo.jpg', quality=0.8):

  js = Javascript('''
async function takePhoto(quality) {

    const div = document.createElement('div');

    const capture = document.createElement('button');

    capture.textContent = 'Capture';

    div.appendChild(capture);

    const video = document.createElement('video');

    video.style.display = 'block';

    const stream = await navigator.mediaDevices.getUserMedia({video: true});

    document.body.appendChild(div);

    div.appendChild(video);

    video.srcObject = stream;

    await video.play();

    // Resize the output to fit the video element.

    google.colab.output.setIframeHeight(document.documentElement.scrollHeigh
t, true);

    // Wait for Capture to be clicked.

    await new Promise((resolve) => capture.onclick = resolve);

const canvas = document.createElement('canvas');

    canvas.width = video.videoWidth;

    canvas.height = video.videoHeight;

    canvas.getContext('2d').drawImage(video, 0, 0);
```

```
  from IPython.display import display
from PIL import Image
import io
import base64
import time
from google.colab import output
def take_photo():
  """Capture a photo using the device's camera and return it as a PIL Image."""
  print("Starting camera...")
  js = f"""
    async function takePhoto() {{
      const stream = await navigator.mediaDevices.getUserMedia({{video: true}});
      const track = stream.getVideoTracks()[0];
      const imageCapture = new ImageCapture(track);
      const blob = await imageCapture.takePhoto();
      const base64 = await blobToBase64(blob);
      {output._js_side_injection_manager.capture_output()}; // Capture the output
to be able to return the value
      return base64;
    }}
    async function blobToBase64(blob) {{
      return new Promise((resolve, reject) => {{
        const reader = new FileReader();
        reader.readAsDataURL(blob);
        reader.onloadend = () => {{
          const base64String = reader.result.replace('data:', '').replace(/^.+,/, '');
          resolve(base64String);}};
        reader.onerror = reject;
      }});}}"""
  output.register_callback('takePhoto', lambda _: display(Javascript(js)))
  data = output.eval_js('takePhoto()', output_js=True)
  output.clear()
  print("Image captured!")
  return Image.open(io.BytesIO(base64.b64decode(data)))
# Take a photo and display it
image = take_photo()
display(image)
except Exception as err:
  # grant the page permission to access it.
  print(str(err))
predictemo('/content/drive/MyDrive/photo.jpg')
```

9. SYSTEM TESTING

To execute test cases, testing teams make use of a testing environment, which is a combination of software and hardware configurations. It facilitates test execution by providing a configured network, software, and hardware.

The test environment is customized to meet the specific requirements of the application being tested, and in certain cases, it encompasses both the test data and the test environment.

The creation of a suitable test environment is crucial for the success of software testing. Any errors or inadequacies in this process can lead to additional expenses and efforts for the client.

9.1 Types of Tests

9.1.1 Unit Testing

Unit testing is a method of testing that concentrates on examining individual elements or functionalities of software. The main purpose of unit testing is to test each component or function separately.

A unit refers to the smallest testable component of an application, typically with one or a few inputs and a single output. Unit testing speeds up the testing process and helps to maintain and modify the code. Writing and executing good unit test cases after code changes can reveal any defects that might have been introduced.

In the long term, development will go more quickly since unit tests are more trustworthy. It simplifies debugging. When a test fails in this, only the most recent modifications need to be troubleshooted. It is necessary to scan for or discover changes produced over a long period of time while testing at a higher level or in higher phases.

9.1.2 Integration Testing

Integration testing is a method of testing that confirms the correct operation of modules/components after they have been merged. Its objective is to uncover any problems that may occur when formerly operational modules are integrated with each other.

Complexity and logical aptitude are needed for integration testing.

Each integrated module is tested for proper operation during integration. Interface issues are found during integration testing.

9.1.3 Performance Testing

Performance testing is a type of software testing that concentrates on an application's non-functional components, such as its stability, speed, scalability, and responsiveness, when subjected to different workloads.

Its main goals are to evaluate an application's efficiency in terms of processing speed, output, data transfer rate, network bandwidth usage, maximum concurrent users, memory usage, workload effectiveness, and response times to commands.

The specific performance testing procedures are determined by the performance metrics that the company values the most. But since most performance testing objectives are universal, there is a specific procedure that most testing strategies will adhere to.

9.1.4 Verification and Validation

Verification is a crucial process in software development that involves reviewing intermediate work products to ensure that they align with the specifications outlined at the beginning of the phase. This is done to ensure that the development process stays on track and is progressing towards the desired outcome.

In contrast, the process of validation refers to the assessment of the end result to guarantee that it satisfies the standards and specifications of the business.

This process typically includes various types of testing, such as smoke testing, functional testing, regression testing, system testing, and others, and is primarily a validation activity.

To sum up, the process of verification is aimed at making sure that the development process is on the right track, whereas the process of validation is aimed at making sure that the final product meets the specified criteria and functions as intended.

.

```
from matplotlib import pyplot as plt

plt.plot(history.history['loss'],'r',label='training loss')
plt.plot(history.history['val_loss'],'b',label='validation loss')
plt.xlabel('epoch')
plt.ylabel('loss')
plt.legend()
plt.show()
```

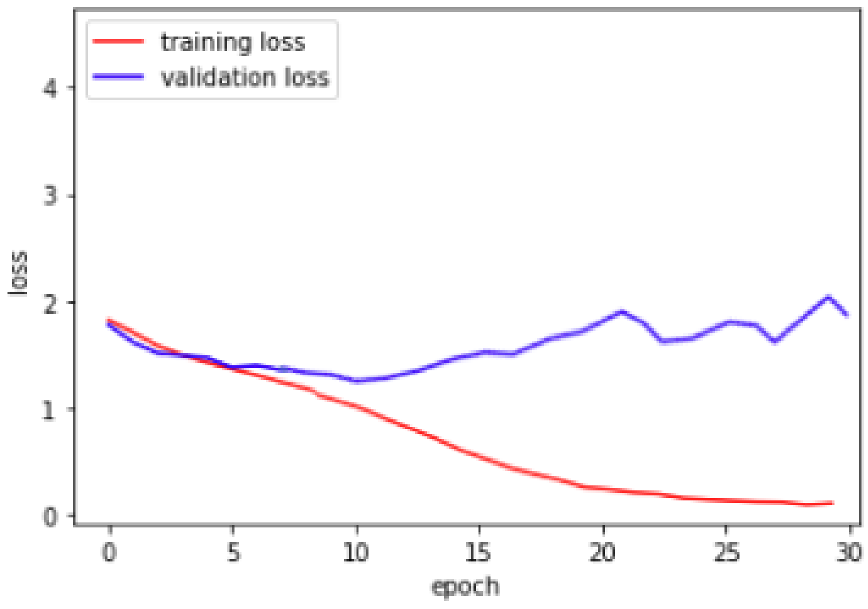
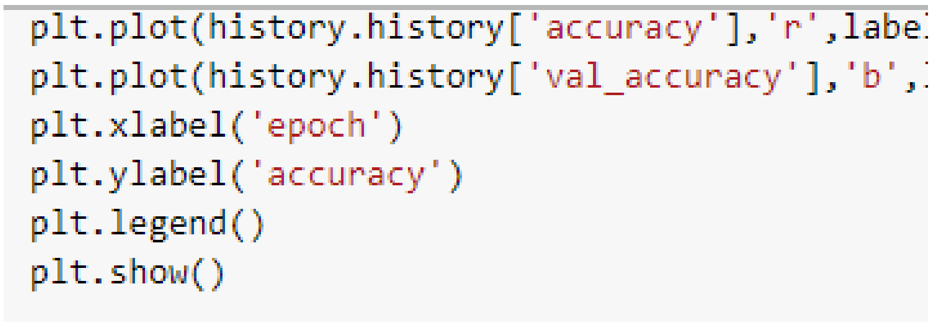


Fig 9.1 Training loss vs Validation loss graph



```
plt.plot(history.history['accuracy'],'r',labe
plt.plot(history.history['val_accuracy'],'b',
plt.xlabel('epoch')
plt.ylabel('accuracy')
plt.legend()
plt.show()
```

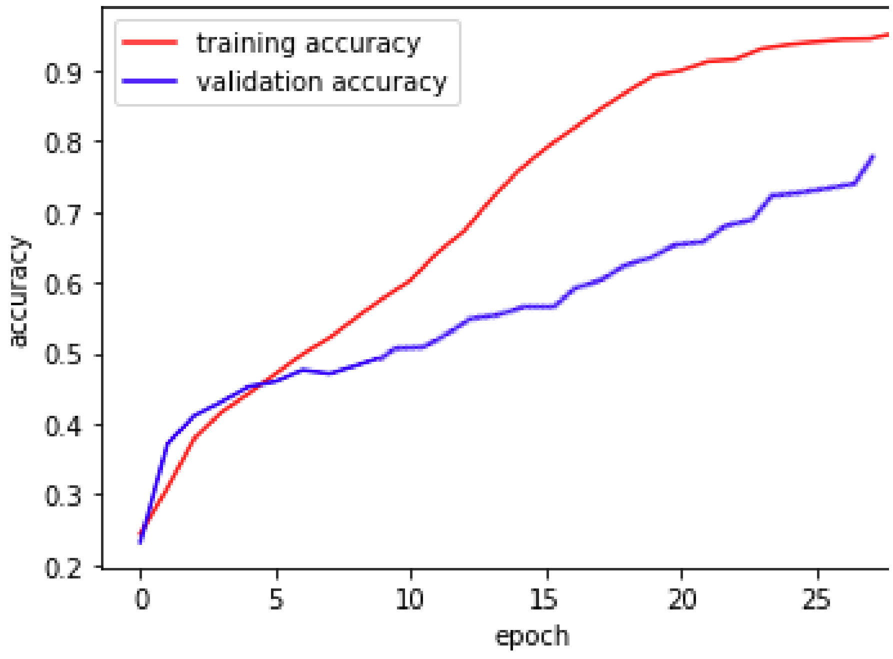
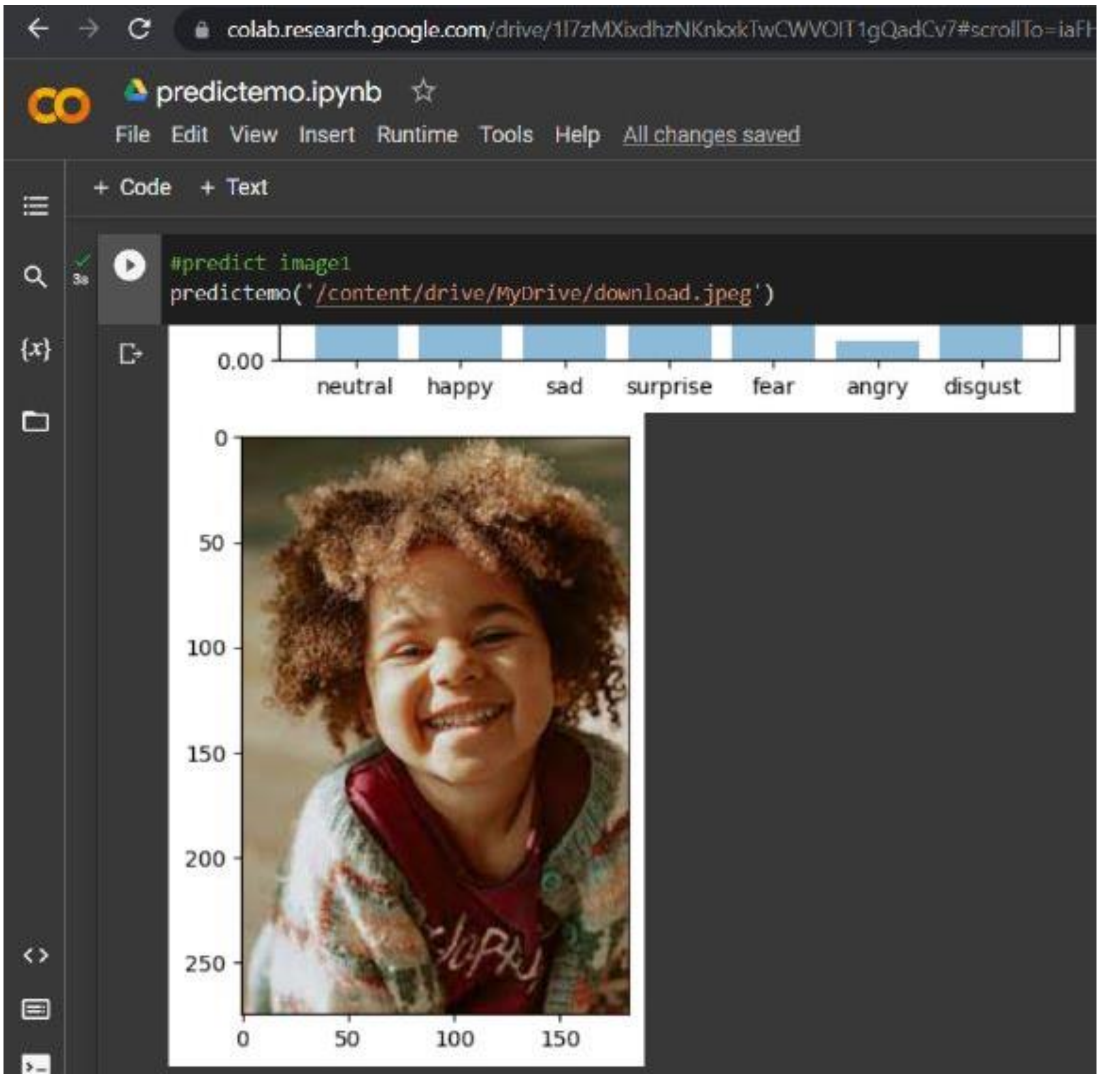


Fig 9.2 Training Accuracy vs Validation Accuracy graph

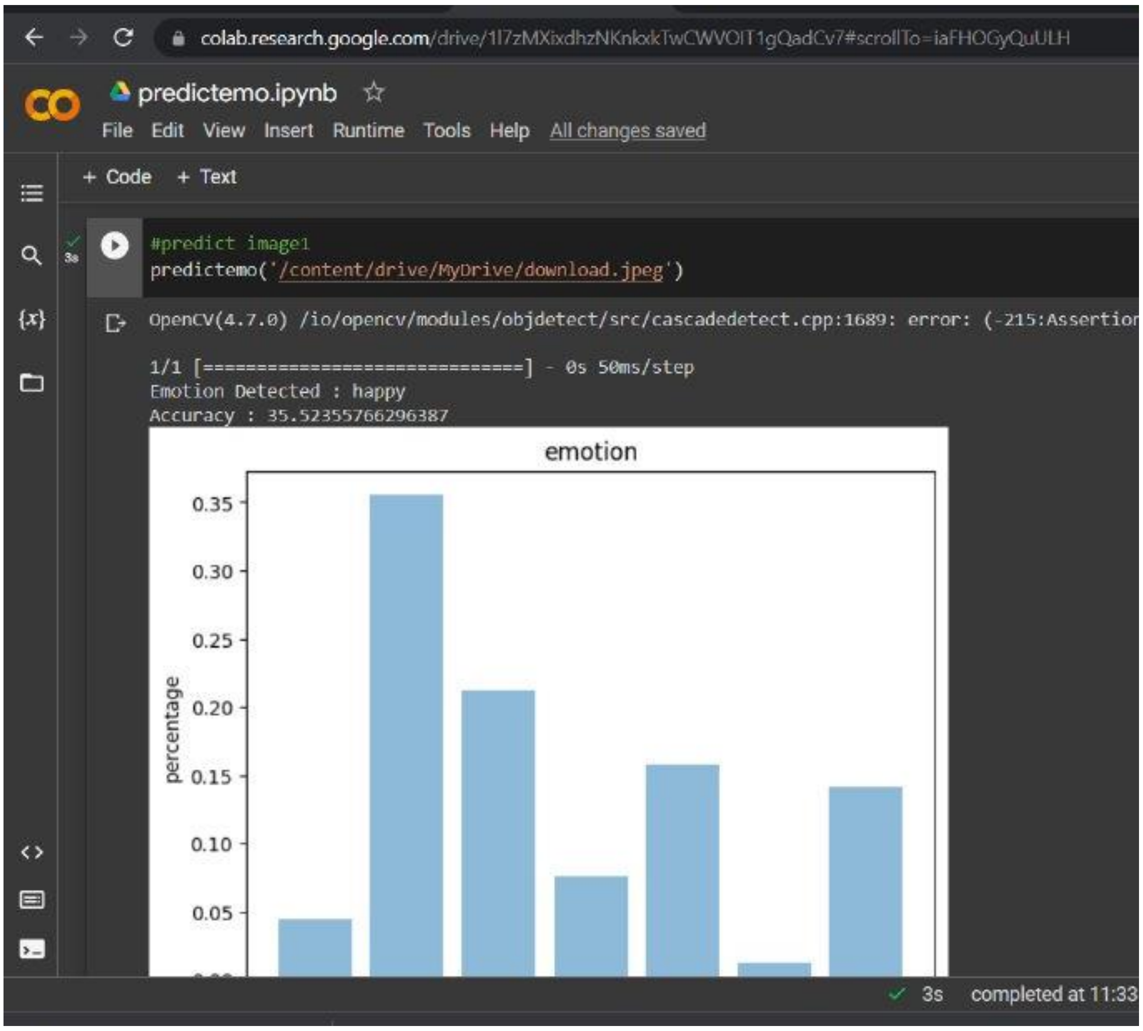
10 RESULTS AND OUTPUT SCREENS


colab.research.google.com/drive/1I7zMXixdhzNKnkxkTwCWVOIT1gQadCv7#scrollTo=iaFH
predictemo.ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text

```
#predict image1
predictemo('/content/drive/MyDrive/download.jpeg')
```

0.00
neutral
happy
sad
surprise
fear
angry
disgust
0
50
100
150
200
250
0
50
100
150

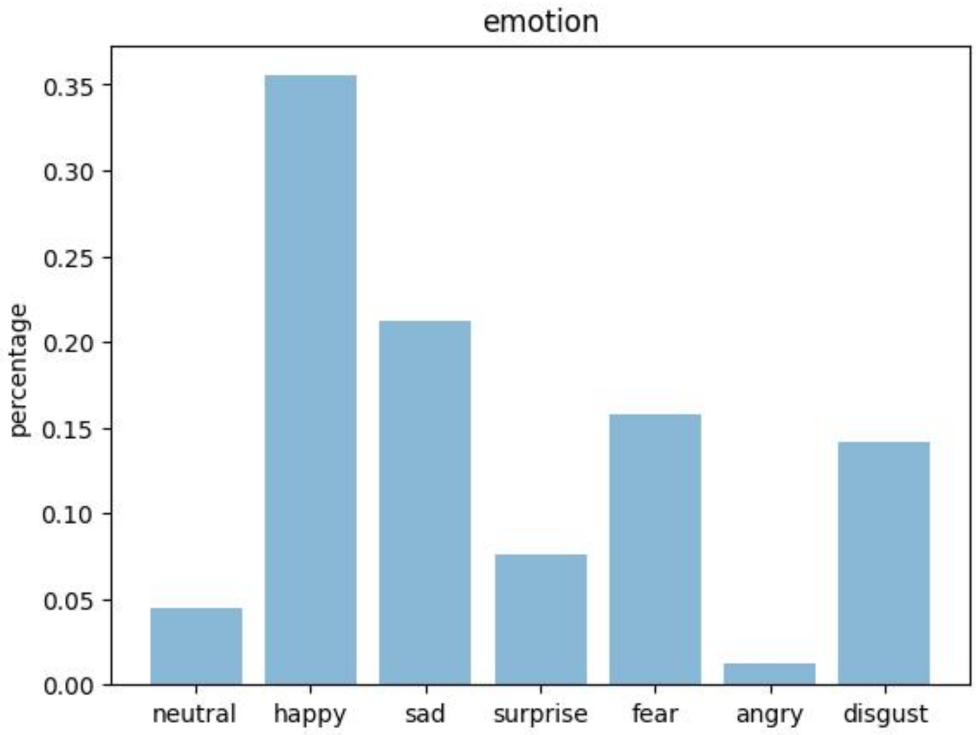
Fig 10.1 Input Image download.jpeg


colab.research.google.com/drive/1l7zMXixdhzNKnkxkTwCWVOIT1gQadCv7#scrollTo=iaFHOGyQuULH
predictemo.ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
emotion
percentage
0.35
0.30
0.25
0.20
0.15
0.10
0.05
3s completed at 11:33

```
#predict image1
predictemo('/content/drive/MyDrive/download.jpeg')
```

```
OpenCV(4.7.0) /io/opencv/modules/objdetect/src/cascadedetect.cpp:1689: error: (-215:Assertion

1/1 [==============================] - 0s 50ms/step
Emotion Detected : happy
Accuracy : 35.52355766296387
```

Fig 10.2 Emotion Detection of download.jpeg



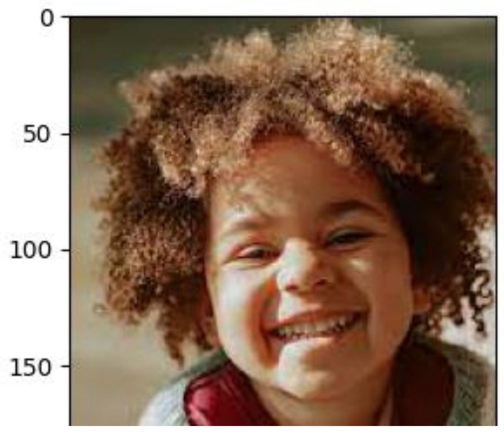
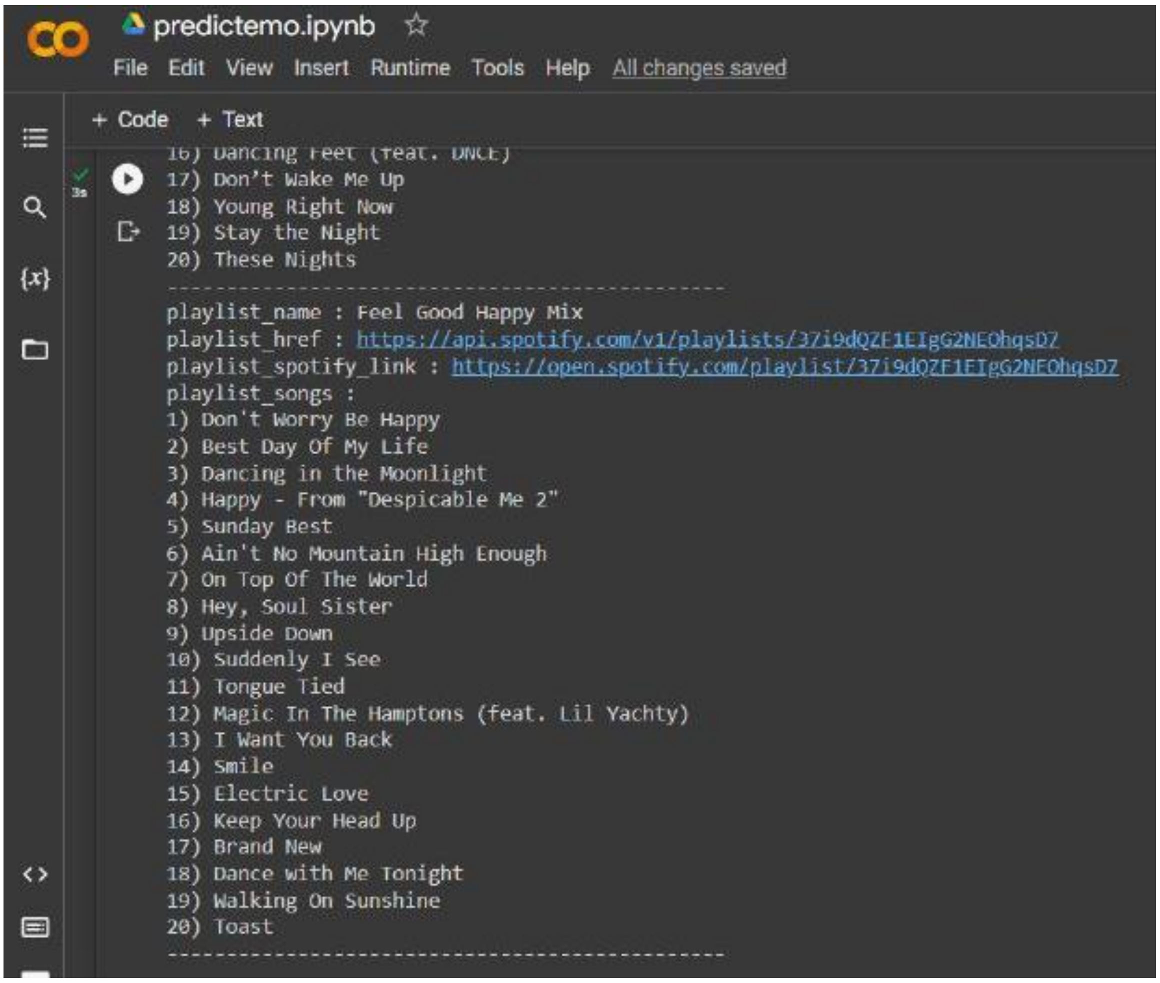


Fig 10.3 Predicted emotion download.jpeg

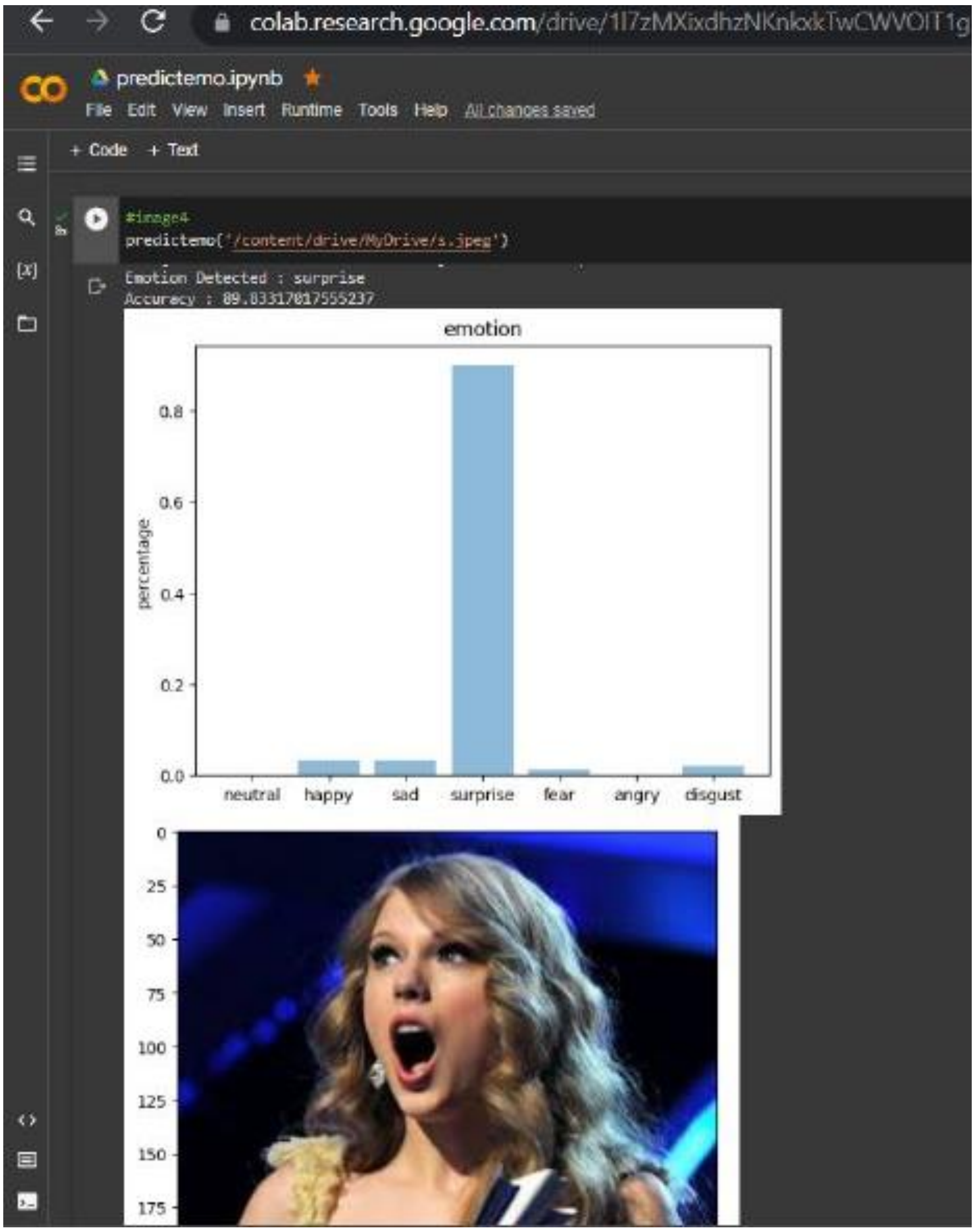

predictemo.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
16) Dancing Feet (feat. DNCE)
17) Don't Wake Me Up
18) Young Right Now
19) Stay the Night
20) These Nights
----------------------------------------------
playlist_name : Feel Good Happy Mix
playlist_href : https://api.spotify.com/v1/playlists/37i9dQZF1EIgG2NEOhqsD7
playlist_spotify_link : https://open.spotify.com/playlist/37i9dQZF1EIgG2NEOhqsD7
playlist_songs :
1) Don't Worry Be Happy
2) Best Day Of My Life
3) Dancing in the Moonlight
4) Happy - From "Despicable Me 2"
5) Sunday Best
6) Ain't No Mountain High Enough
7) On Top Of The World
8) Hey, Soul Sister
9) Upside Down
10) Suddenly I See
11) Tongue Tied
12) Magic In The Hamptons (feat. Lil Yachty)
13) I Want You Back
14) Smile
15) Electric Love
16) Keep Your Head Up
17) Brand New
18) Dance with Me Tonight
19) Walking On Sunshine
20) Toast
----------------------------------------------
```

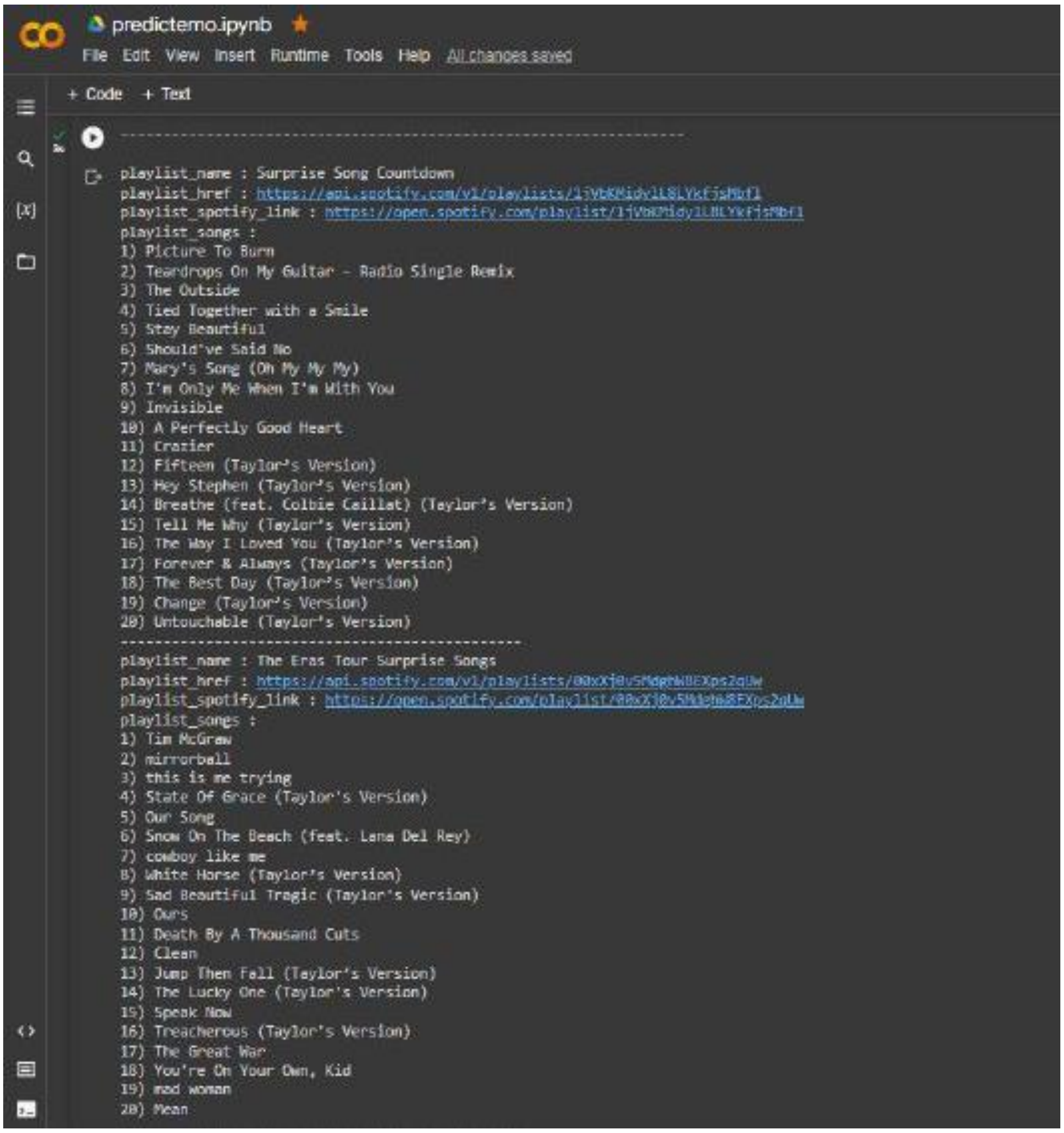
Fig 10.4 Recommended Playlist download.jpeg



```
#image4
predictemo('/content/drive/MyDrive/s.jpeg')
```

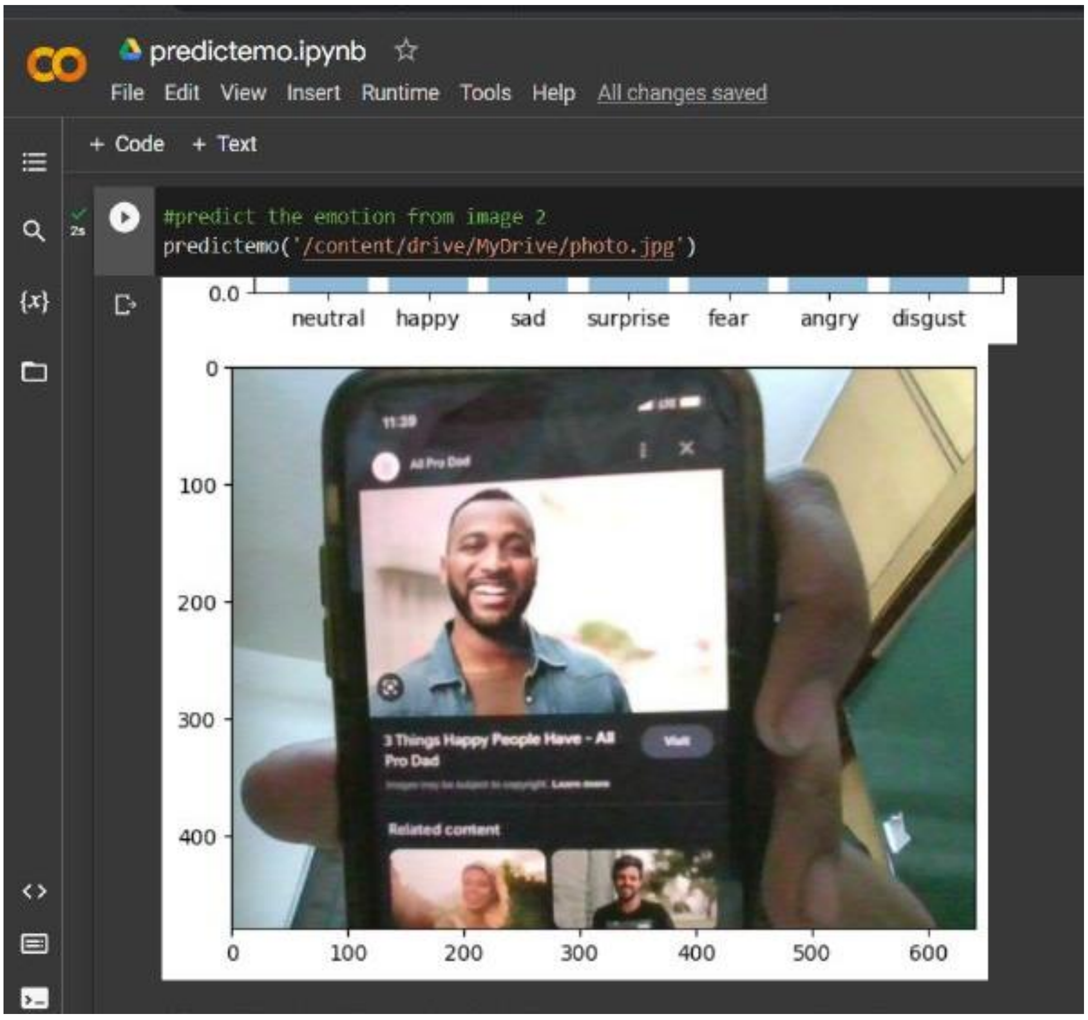
```
Emotion Detected : surprise
Accuracy : 89.83317017555237
```

Fig 10.5 Input Image s.jpeg



```
----------------------------------------------------------------------
playlist_name : Surprise Song Countdown
playlist_href : https://api.spotify.com/v1/playlists/1jVbKMidy1L8LYkFjsMbf1
playlist_spotify_link : https://open.spotify.com/playlist/1jVbKMidy1L8LYkFjsMbf1
playlist_songs :
1) Picture To Burn
2) Teardrops On My Guitar - Radio Single Remix
3) The Outside
4) Tied Together with a Smile
5) Stay Beautiful
6) Should've Said No
7) Mary's Song (Oh My My My)
8) I'm Only Me When I'm With You
9) Invisible
10) A Perfectly Good Heart
11) Crazier
12) Fifteen (Taylor's Version)
13) Hey Stephen (Taylor's Version)
14) Breathe (feat. Colbie Caillat) (Taylor's Version)
15) Tell Me Why (Taylor's Version)
16) The Way I Loved You (Taylor's Version)
17) Forever & Always (Taylor's Version)
18) The Best Day (Taylor's Version)
19) Change (Taylor's Version)
20) Untouchable (Taylor's Version)
--------------------------------------------------
playlist_name : The Eras Tour Surprise Songs
playlist_href : https://api.spotify.com/v1/playlists/00xXj0vSMdghW8EXps2qUw
playlist_spotify_link : https://open.spotify.com/playlist/00xXj0vSMdghW8EXps2qUw
playlist_songs :
1) Tim McGraw
2) mirrorball
3) this is me trying
4) State Of Grace (Taylor's Version)
5) Our Song
6) Snow On The Beach (feat. Lana Del Rey)
7) cowboy like me
8) White Horse (Taylor's Version)
9) Sad Beautiful Tragic (Taylor's Version)
10) Ours
11) Death By A Thousand Cuts
12) Clean
13) Jump Then Fall (Taylor's Version)
14) The Lucky One (Taylor's Version)
15) Speak Now
16) Treacherous (Taylor's Version)
17) The Great War
18) You're On Your Own, Kid
19) mad woman
20) Mean
```

Fig 10.6 Recommended Playlist s.jpeg


predictemo.ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
0.0
neutral happy sad surprise fear angry disgust
0
100
200
300
400
0 100 200 300 400 500 600

```
#predict the emotion from image 2
predictemo('/content/drive/MyDrive/photo.jpg')
```

Fig 10.7 Input Image photo.jpg

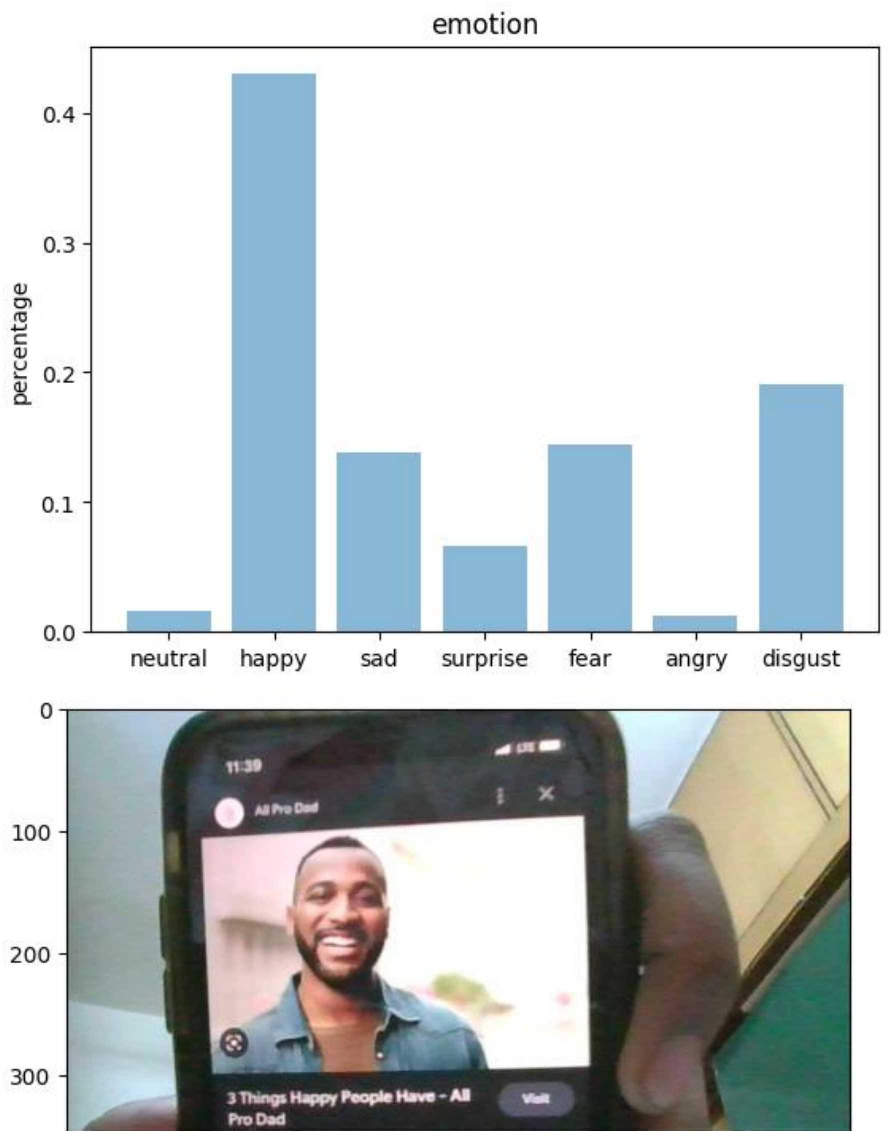
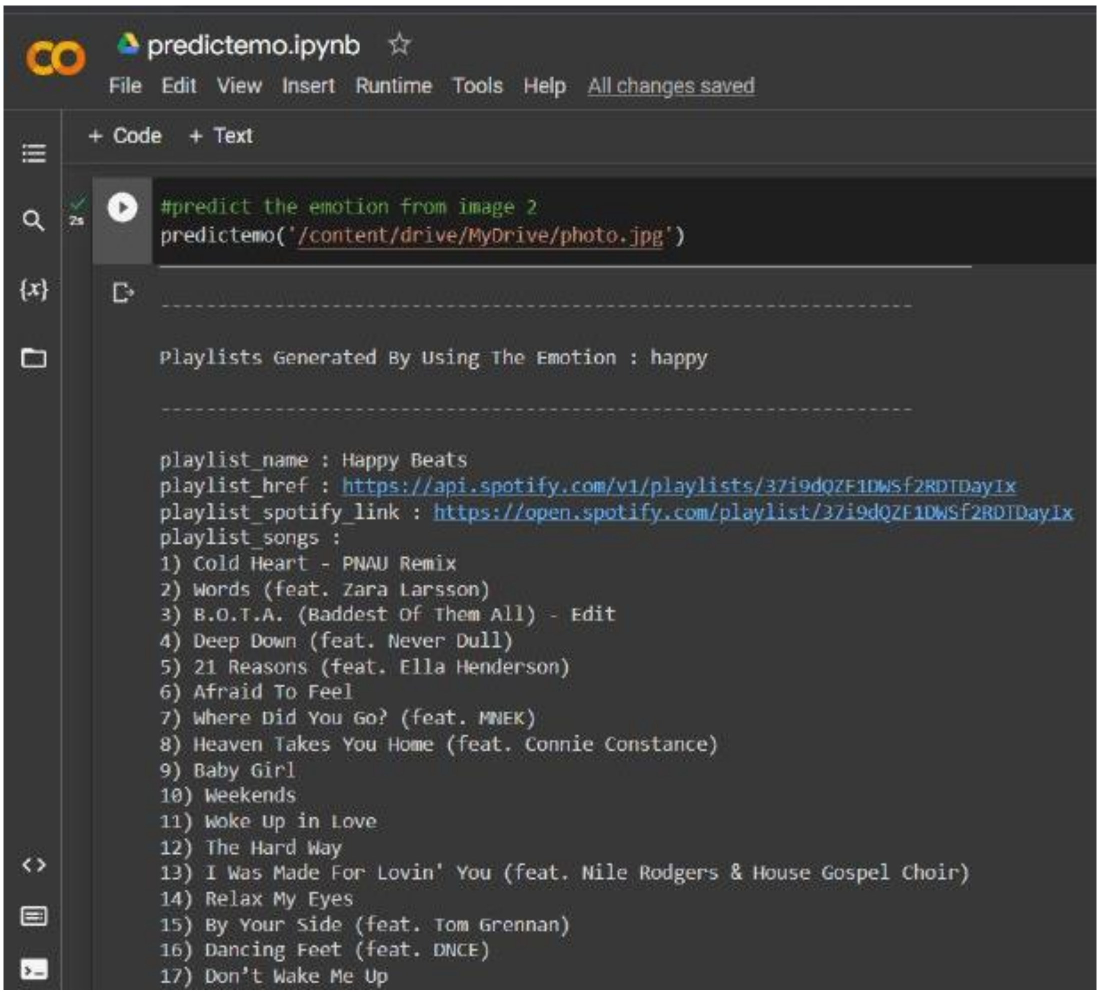


Fig 10.8 Emotion Prediction photo.jpg


predictemo.ipynb

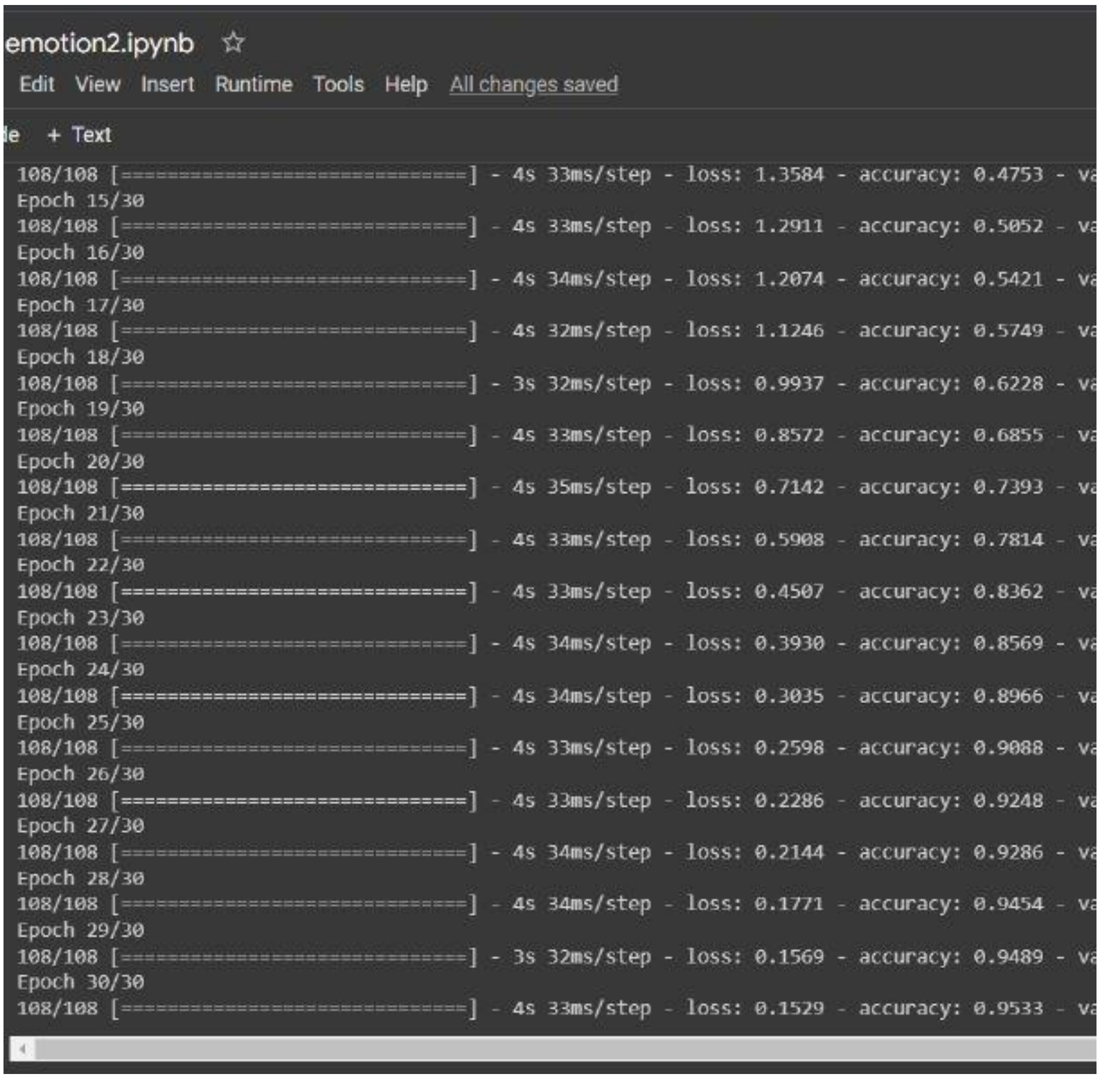
File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
#predict the emotion from image 2
predictemo('/content/drive/MyDrive/photo.jpg')
```

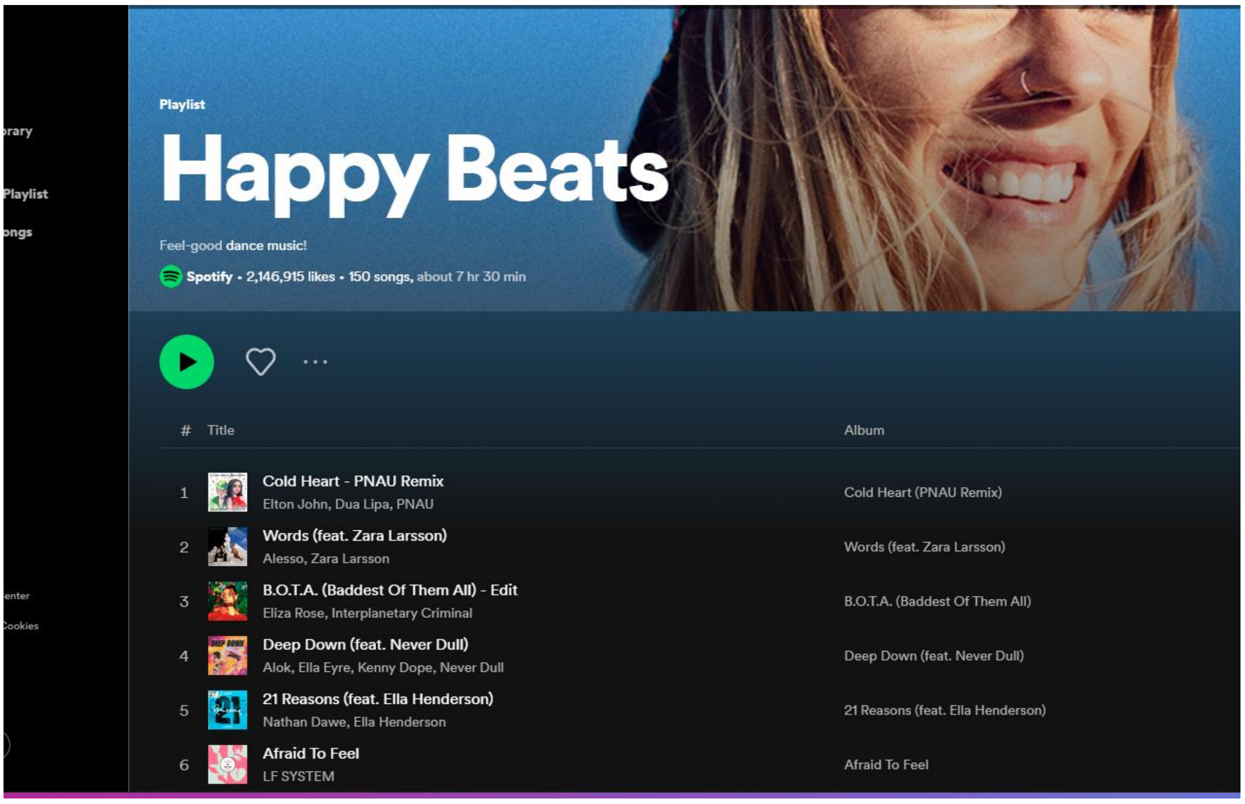
```
-------------------------------------------------------------------

Playlists Generated By Using The Emotion : happy

-------------------------------------------------------------------

playlist_name : Happy Beats
playlist_href : https://api.spotify.com/v1/playlists/37i9dQZF1DWSf2RDTDayIx
playlist_spotify_link : https://open.spotify.com/playlist/37i9dQZF1DWSf2RDTDayIx
playlist_songs :
1) Cold Heart - PNAU Remix
2) Words (feat. Zara Larsson)
3) B.O.T.A. (Baddest Of Them All) - Edit
4) Deep Down (feat. Never Dull)
5) 21 Reasons (feat. Ella Henderson)
6) Afraid To Feel
7) Where Did You Go? (feat. MNEK)
8) Heaven Takes You Home (feat. Connie Constance)
9) Baby Girl
10) Weekends
11) Woke Up in Love
12) The Hard Way
13) I Was Made For Lovin' You (feat. Nile Rodgers & House Gospel Choir)
14) Relax My Eyes
15) By Your Side (feat. Tom Grennan)
16) Dancing Feet (feat. DNCE)
17) Don't Wake Me Up
```

Fig 10.9 Recommended Playlist photo



```
108/108 [==============================] - 4s 33ms/step - loss: 1.3584 - accuracy: 0.4753 - va
Epoch 15/30
108/108 [==============================] - 4s 33ms/step - loss: 1.2911 - accuracy: 0.5052 - va
Epoch 16/30
108/108 [==============================] - 4s 34ms/step - loss: 1.2074 - accuracy: 0.5421 - va
Epoch 17/30
108/108 [==============================] - 4s 32ms/step - loss: 1.1246 - accuracy: 0.5749 - va
Epoch 18/30
108/108 [==============================] - 3s 32ms/step - loss: 0.9937 - accuracy: 0.6228 - va
Epoch 19/30
108/108 [==============================] - 4s 33ms/step - loss: 0.8572 - accuracy: 0.6855 - va
Epoch 20/30
108/108 [==============================] - 4s 35ms/step - loss: 0.7142 - accuracy: 0.7393 - va
Epoch 21/30
108/108 [==============================] - 4s 33ms/step - loss: 0.5908 - accuracy: 0.7814 - va
Epoch 22/30
108/108 [==============================] - 4s 33ms/step - loss: 0.4507 - accuracy: 0.8362 - va
Epoch 23/30
108/108 [==============================] - 4s 34ms/step - loss: 0.3930 - accuracy: 0.8569 - va
Epoch 24/30
108/108 [==============================] - 4s 34ms/step - loss: 0.3035 - accuracy: 0.8966 - va
Epoch 25/30
108/108 [==============================] - 4s 33ms/step - loss: 0.2598 - accuracy: 0.9088 - va
Epoch 26/30
108/108 [==============================] - 4s 33ms/step - loss: 0.2286 - accuracy: 0.9248 - va
Epoch 27/30
108/108 [==============================] - 4s 34ms/step - loss: 0.2144 - accuracy: 0.9286 - va
Epoch 28/30
108/108 [==============================] - 4s 34ms/step - loss: 0.1771 - accuracy: 0.9454 - va
Epoch 29/30
108/108 [==============================] - 3s 32ms/step - loss: 0.1569 - accuracy: 0.9489 - va
Epoch 30/30
108/108 [==============================] - 4s 33ms/step - loss: 0.1529 - accuracy: 0.9533 - va
```

Fig 10.10 Accuracy of epochs



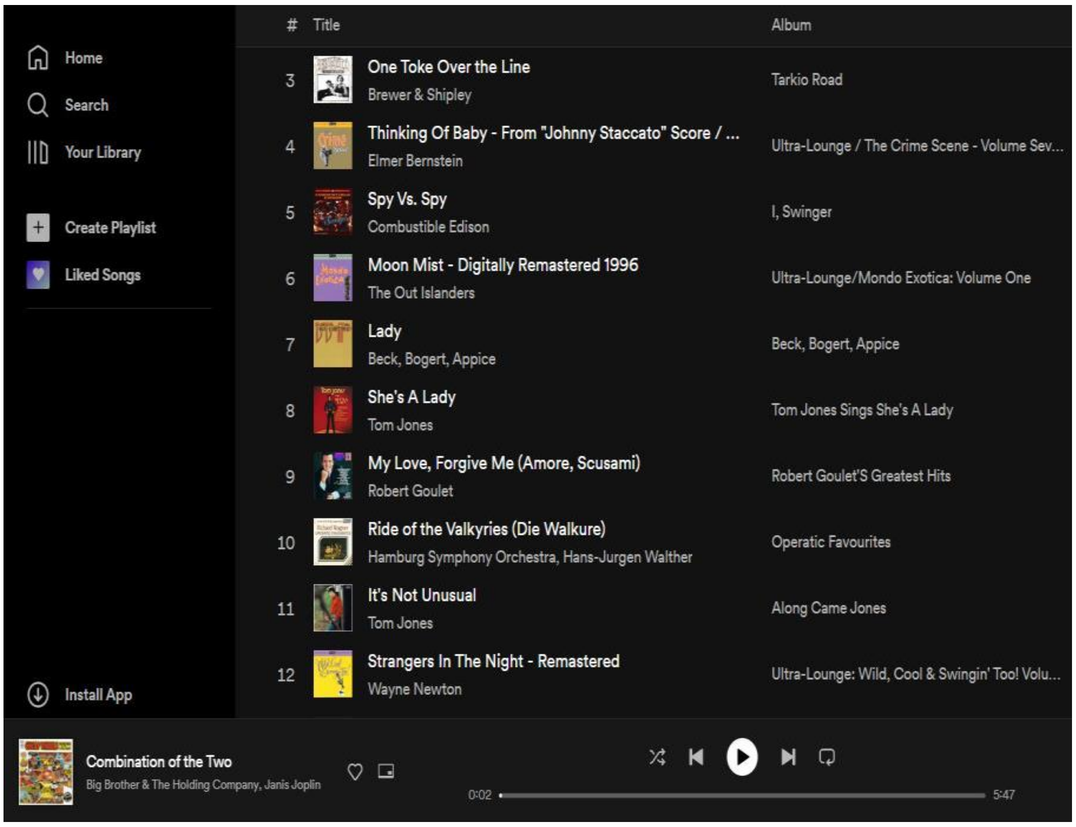
Happy Beats

Feel-good dance music!

Spotify • 2,146,915 likes • 150 songs, about 7 hr 30 min

#	Title	Album
1	Cold Heart - PNAU Remix Elton John, Dua Lipa, PNAU	Cold Heart (PNAU Remix)
2	Words (feat. Zara Larsson) Alesso, Zara Larsson	Words (feat. Zara Larsson)
3	B.O.T.A. (Baddest Of Them All) - Edit Eliza Rose, Interplanetary Criminal	B.O.T.A. (Baddest Of Them All)
4	Deep Down (feat. Never Dull) Alok, Ella Eyre, Kenny Dope, Never Dull	Deep Down (feat. Never Dull)
5	21 Reasons (feat. Ella Henderson) Nathan Dawe, Ella Henderson	21 Reasons (feat. Ella Henderson)
6	Afraid To Feel LF SYSTEM	Afraid To Feel

Fig 10.11 Redirection to spotify on clicking playlist link



#	Title	Album
3	One Toke Over the Line Brewer & Shipley	Tarkio Road
4	Thinking Of Baby - From "Johnny Staccato" Score / ... Elmer Bernstein	Ultra-Lounge / The Crime Scene - Volume Sev...
5	Spy Vs. Spy Combustible Edison	I, Swinger
6	Moon Mist - Digitally Remastered 1996 The Out Islanders	Ultra-Lounge/Mondo Exotica: Volume One
7	Lady Beck, Bogert, Appice	Beck, Bogert, Appice
8	She's A Lady Tom Jones	Tom Jones Sings She's A Lady
9	My Love, Forgive Me (Amore, Scusami) Robert Goulet	Robert Goulet'S Greatest Hits
10	Ride of the Valkyries (Die Walkure) Hamburg Symphony Orchestra, Hans-Jurgen Walther	Operatic Favourites
11	It's Not Unusual Tom Jones	Along Came Jones
12	Strangers In The Night - Remastered Wayne Newton	Ultra-Lounge: Wild, Cool & Swingin' Too! Volu...

Combination of the Two
Big Brother & The Holding Company, Janis Joplin
0:02 5:47

Fig 10.12 Playing songs from recommended playlist

11. CONCLUSION AND FUTURE ENHANCEMENT

11.1 CONCLUSION

The system uses a general model to suggest music based on the user's mood since music can elicit emotions. Its primary objective is to detect any changes in the emotional state of the user and play music accordingly by surfing through various tracks.

The system utilizes the CNN algorithm for emotion classification by analyzing changes in facial expressions like the size, shape, and movement of the eyes, eyebrows, and mouth. Based on the six basic emotions - happiness,sadness, anger, disgust, fear, and surprise, the algorithm categorizes emotions and generates a playlist.

The CNN (Convolutional Neural Network) algorithm is favored over SVM (Support Vector Machine) algorithm due to its ability to detect the crucial features in an image without any human intervention. In contrast, SVM's prediction accuracy is comparatively lower than CNN's accuracy. The CNN system produces highly accurate results, and it outperforms the SVM algorithm in recognizing important features in images.

However, achieving 100% accuracy with the CNN algorithm is challenging due to the inconsistent nature of human emotions and their susceptibility to internal and external factors. Despite its advanced features, the CNN algorithm still faces difficulties in capturing the complex nature of human emotions accurately. This limitation implies that the algorithm's predictions may not always be correct and that further research is necessary to enhance its accuracy.

In summary, CNN's superiority over SVM in identifying critical features in images has made it a popular choice for emotion recognition systems. However, due to the complex and dynamic nature of human emotions, achieving 100% accuracy in the system remains a challenge. Researchers need to continue to explore innovative approaches and techniques to enhance the accuracy of these systems and make them more reliable in real-world scenarios.

To conclude, although the CNN algorithm offers significant advantages over the SVM algorithm in recognizing essential image features, the system's accuracy in detecting human emotions remains imperfect. Further research is necessary to enhance the system's accuracy, and novel techniques must be developed to address the dynamic and complex nature of human emotions.With advanced algorithms and further research, an ideal emotion-detection based music recommendation system can be developed, and the cost of developing such a system is relatively low.

a. Happy songs are classified as energetic, playful, and cheerful while sad and depressing songs fall under the Sad category.

b. Songs that express anger, patriotism, and revenge are grouped under the Anger category.

c. Joy-Anger is related to the tunes that have outrage in an energetic mode.

d. The Sad-anger category contains of songs that revolve around being extremely angry and depressed .

e. If a user is experiencing fear or surprise, the system suggests songs from the fear category.

11.2 FUTURE ENHANCEMENTS

The area of emotion classification presents a promising research opportunity with numerous potential applications. A critical aspect of emotion classification is facial recognition, which has recently been adopted by Apple for device unlocking. However, its capabilities extend beyond device unlocking and could potentially be used for enhanced facial authentication in ATMs and offices to increase resource security and safety.

Marketing researchers are utilizing emotion recognition to add value to their brands by collecting and studying customer data to deliver personalized goods, ultimately leading to increased customer loyalty. In the future, this technology could even be used to automatically answer customer queries, providing a valuable asset for businesses.

In major supermarkets, customers are typically asked for feedback through buttons, but emotion recognition technology could be used to scan customers' faces and identify their feelings while waiting in lines. This data could enable supermarkets to provide better customer services.

Drunk driving is a major cause of accidents, and alcohol consumption can significantly affect drivers' abilities to stay focused while driving. Emotion detection technology could serve as a warning system to alert drivers when it is unsafe to drive or as a feature in automated cars to ensure drivers reach their destination safely without driving.

Facial recognition and emotion detection present numerous possibilities, and it is important to harness this potential to maximize its benefits.

12. REFERENCES

• Raut, Nitisha, “Facial Emotion Recognition Using Machine Learning” (2018). Master’s Projects. 632. https://doi.org/10.31979/etd.w5fs-s8wd

• Tim Spittle, lucyd, GitHub, , April 16, 2020. Accessed on: [Online], Available at: https://github.com/timspit/lucyd

• D Priya, Face Detection, Recognition and Emotion Detection in 8 lines of code!, towards data science, April 3, 2019. Accessed on: July 12, 2020 [Online], Available at: https://towardsdatascience.com/face- detection-recognition-and-emotion-detection-in-8-lines-of-code- b2ce32d4d5de

• bluepi, \Classifying Different Types of Recommender Systems, November 14, 2015. Accessed on: July 7, 2020. [Online],

Available on: https://www.bluepiit.com/blog/classifying-recommender systems/#:~:text=There%20are%20majorly%20six%20types,system %20and%20Hybrid%20recommender%20system

• 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

• Manas Sambare, FER2013 Dataset, Kaggle, July 19, 2020. Accessed on: September 9, 2020. [Online], Avalilable at: https://www.kaggle.com/msambare/fer2013