CSE3506

ESSENTIALS OF DATA ANALYTICS

J COMPONENT REPORT

A project report titled

SONG RECOMMENDATION SYSTEM

By

M N M SASIDHAR -18BEC1198

ADITYA SIRANGU -18BEC1252

BACHELOR OF TECHNOLOGY IN ELECTRONICS AND COMMUNICATION ENGENEERING

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

Submitted to

Dr. R. KARTHIK

JUNE 2021

SCHOOL OF ELECTRONICS ENGINEERING

DECLARATION BY THE CANDIDATES

We hereby declare that the Report entitled **"SONG RECOMMENTATION SYSTEM"** submitted by M N M SASIDHAR (18BEC1198), ADITYA SIRANGU (18BEC1252) to VIT Chennai is a record of bonafide work undertaken by all three under the supervision of **Dr. R. Karthik, Senior Assistant Professor, SENSE, VIT Chennai.**

SIGNATURES OF CANDIDATES:

M N M SASIDHAR

S V M ADITYA KUMAR

CHENNAI

05/06/2021.

ACKNOWLEDGEMENT

We wish to express our sincere thanks and deep sense of gratitude to our project guide, **Dr. R. Karthik**, School of Electronics Engineering for his consistent encouragement and valuable guidance offered to us in a pleasant manner throughout the course of the project work.

We are extremely grateful to **Dr. Sivasubramanian**. A,Dean of the School of Electronics Engineering (SENSE), VIT University Chennai, for extending the facilities of the School towards our project and for his unstinting support.

We express our thanks to our Head of The Department **Dr. Vetrivelan. P** (B.Tech-ECE) for his support throughout the course of this project.

We also take this opportunity to thank all the faculty of the School for their support and their wisdom imparted to us throughout the courses till date.

We thank our parents, family, and friends for bearing with us throughout the course of our project and for the opportunity they provided us in undergoing this course in such a prestigious institution.

BONAFIDE CERTIFICATE

Certified that this project report entitled **"SONG RECOMMENTATION SYSTEM"** submitted by M N M SASIDHAR (18BEC1198), ADITYA SIRANGU (18BEC1252). Who have carried out the "J"-Project work under my supervision and guidance for **CSE3506- Essentials of Data Analytics.**

DR.R.KARTHIK

SENIOR ASSISTANT PROFESSOR

SCHOOL OF ELECTRONICS ENGINEERING VIT UNIVERSITY, CHENNAI

CHENNAI – 600 127.

ABSTRACT

Songs, as a medium, have always been a popular choice to depict human emotions and the human face plays a great role in knowing the emotional status or the mood of the person. The emotion of the person can be predicted, up to a certain accuracy, with the help of certain features visible on the face. By identifying the mood of the person we can recommend a song. People/users usually have a huge number of songs in their collection or playlists. So to avoid difficulty choosing a song, most people only pick a song from their playlist at random and some of the songs might disappoint the user as they aren't suited for their current mood, so finding a song which relates to their current mood becomes a tedious and time consuming job. In order to make the job more easy, we have developed an GUI, which takes an image as input and finds the mood of the person/user by face detection and based on the emotion/mood of the user/person the GUI recommends a song which is lies or relates to their current emotion/mood.

For face detection/facial expression recognition we have used CNN(convolution neural networks).we have designed a Convolutional Neural Network model which can classify the input image into 7 different emotions and the emotions we are going to classify are angry, fear, happy, surprise, sad,disgust and neutral. In order to classify these emotions, we are implementing Convolutional Neural Networks (CNNs) which can efficiently and accurately extract the information coming from the faces in an automated manner. We have also applied some data augmentation techniques in order to intercept overfitting and underfitting problems. We have taken the FER-2013 dataset from kaggle competition and used it to evaluate the designed CNN model. Our model has achieved an accuracy rate of 72.8%.

Keywords: Convolutional Neural Networks(CNN), face detection, facial expression recognition, GUI.

TABLE OF CONTENTS

1. INTRODUCTION

Music is an important medium of entertainment and relaxation for music listeners and has even proved to have a therapeutic weightage. Current research in the field of music has shown that music induces a clear emotional response in its listeners. Musical preferences have been demonstrated to be highly correlated with personality traits and moods of an individual. The automatic analysis and understanding of music by the computer is the new possibility in the field of music information retrieval and Content-based music recommendation is one the feasible application that can be provided. From the context information, we can achieve more intelligent context-based music recommendation and emotion detection is one way we can get content information and based on the information we can build a music recommendation system.

Face expression recognition is used to recognize the essential human emotions. People usually have a great number of songs in their collection or playlists. So to avoid difficulty choosing a song, most people only pick a song from their playlist at random and some of the songs might disappoint the user. As a result, some of the songs are not matching to the user's current emotion and in addition to it there's no widely used model that can play songs based on the user's current emotions. So in this paper, our aim is to remove this manual method of selecting songs, as it's a tedious, time consuming job and to build an emotion based song recommendation system. We have used deep learning techniques- CNN(Convolution Neural Networks) for facial emotion detection, because CNN will learn to recognize components of an image and work better than other neural networks. Our model will take an image of a human face as input and return the emotion that the face expresses. To determine the best classifiers for recognizing particular emotions, single and multi-layered networks will be tested and emotion is classified accordingly. Once the emotion is successfully detected from the image, it proceeds into the next step, where a song is recommended using the emotion which we detected from the image.

1.2 PROBLEM STATEMENT

People usually have a large number of songs in their collection or playlists and find it difficult to choose a song which relates their mood/emotion at the current time and finding a song becomes a tedious and time consuming job. In order to not go through all this time consuming searching, we can develop a model which captures the image at that instant and using face recognition we can find the emotion and build a recommendation system which recommends songs according the users current emotion.

1.3 LITERATURE SURVEY

Emotion based music recommendation systems are the need of the hour and will be benefit to the fields of Emotion Intelligence, Medical Science and Psychology. In recent times, techniques such as Neural Networks (NN), Support Vector Machines (SVM) have been used. We have analyzed the techniques which are in association with our application.

[1].Rajesh Kumar G A, Ravi Kant Kumar, GoutamSanyal contributed on “Facial Emotion Analysis using Deep Convolution Neural Network”, 2017. They proposed that, human emotions/moods are mental states of feelings that take place on impulse rather than through conscious effort and are accompanied by physiological changes in facial muscles which implies expressions on face. The emotions are happy, sad, anger, disgust, fear, surprise, neutral. Facial expressions play a key role in nonverbal communication which appears due to internal feelings of a person that reflects on the faces.In this system, they are providing better approach to predict human emotions using deep Convolution Neural Network (CNN) and how emotion intensity changes on a face from low level to high level of emotion. In this algorithm, FER-2013 database has been used for training. The evaluated proposed model confers quite good result and obtained accuracy may give encouragement to the researchers for future model of computer-based emotion recognition system.

[2]. Yong-Hwan Lee, Woori Han and Young Kim contributed on "Emotional Recognition from Facial Expression Analysis Using Bezier Curve Fitting,". They proposed a system based on Bezier curve fitting in. This algorithm uses a

multiple phase process for face expression and emotions. The first one is detection and processing of facial area from the input original image. The second step is the verification of emotion in the region of interest. The first step of face detection uses color still image based on skin color pixel by initialized spatial filtering, based on the result of lighting compassion. Later it uses the Feature Map To measure eyes and lips face location and facial shape. After extracting regions of interest, this method extracts points from the feature map in order to apply Bezier curve to the eye and mouth techniques to study.Advantages - The size of the dataset isn't restricted. The algorithm shows high efficiency even on large datasets and the algorithm can also be used for 3D pictures. Thus a 3D picture can also be exploited for information. Disadvantages - Each point of study in the data has a global influence. There are no outliers and this can cause over fitting. which leads to efficiency can decrease, if the data set is distorted due to over fitting.

[3] Mostafa Mohammadpour, Seyyed Mohammad.RHashemi gave “Facial Emotion Recognition using Deep Convolutional Networks” stated that Facial emotion recognition is an emerging field which use in many now a day’s application including social robots, neuro marketing and games. Non-verbal communication methods like facial expressions, eye movement and gestures are used in many applications of human computer interaction, which among them facial emotion is widely used because it conveys the emotional states and feelings of persons. The emotion recognition is tedious task because there is no landmark distinction between the emotions on the face and also there are a lot of complexity and variability. In the traditional machine learning algorithm, some important extracted features used for modelling the face, so, it cannot achieve high accuracy rate for recognition of emotion because the features are hand-engineered and depend on prior knowledge. Convolutional neural networks (CNN) have developed in this work for recognition facial emotion expression and classify the min to seven basic categories. Instead of calculating hand 16 engineered features, CNN calculates features by learning automatically. Then the proposed method is using facial action units (AUs) of the face which first these units are recognized by CNN and in corporate to

recognizing these basic emotion states. Cohn-Kanade database is used to evaluate the proposed model, and the model achieves the accuracy rate 97.01 by incorporating AU while other works in the literature used a direct CNN and achieve accuracy rate 95.75.

[4]C. Maoui and Pruski proposed a system of total facial emotion classification system based on three phases. Face identification, properties extraction and classification of expressions. This algorithm gives a model for detecting the point of facial feature combined with shi and thomasi process. In this method, the 21 distance variation that describes the facial feature from the neutral face and the classification based on Support Vector Machine (SVM). Support Vector Machines (SVM) considers the classification problem as a quadratic optimization problem. It standardizes classification risks and also medical diagnosis The vital points on face are considered as the input for SVM training following which the emotion is recognized efficiently. Advantages - Provides one of the most detailed analysis in facial analysis. The efficiency is more, among all the other conventional methods. Disadvantages - The algorithm has a larger time complexity compared to other conventional algorithms.

[5]Xue Mei Zhao, Cheng Bing Wei proposed “A Real-time Face Recognition System” and that is based on the Improved LBPH Algorithm. They found that The Local Binary Pattern Histogram (LBPH) algorithm is a simple/easy solution on face recognition problem, which can recognize both front face and side face. However, the recognition rate of LBPH algorithm under the conditions of light diversification, expression variation and attitude deflection are decreased. To resolve this problem, a modified LBPH algorithm based on pixel neighborhood grey median (MLBPH) is proposed. Median value of its neighborhood is placed in place of grey value of the pixel and then the feature value is extracted by the sub blocks and the statistical histogram is used to form the MLBPH feature dictionary, which is used to recognize the human face identity compared with test image. The model carries out experiments on FERET standard face database and the creation of new face database, and the

results show that MLBPH algorithm is superior to LBPH algorithm in recognition rate.

To summarize from the above study, each of the techniques or the methods used consists its own flaws in it. Comparatively, the deep CNN which has been proposed found to be the best. It is considered to be the best algorithm to recognize the facial expressions.

2 REQUIREMENTS AND PROPSED SYSTEM

2.1 REQUIREMENTS

2.1.1 DATASET

The dataset which we have used is the FER2013 dataset. The data consists of 48x48 pixel gray scale images of faces and the faces have been automatically registered so that the face is more or less centered and occupies about the same amount of space in each image. This dataset consists of 35,887 gray scale, 48x48 sized face images with various emotions -7 emotions[1]. The emotions are labeled as follows:0 is Angry, 1 is Disgust, 2 is Fear, 3 is Happy, 4 is Sad, 5 is Surprise, 6 is Neutral. The train test split is done in a manner so that 80% of the data is used for training purposes[2]. That is, 28,709 tuples are used for training purposes and the rest (7178) are used for testing. Fig 1, Fig2 and Fig3 are the descriptions of the dataset. The numbers of images in the dataset for each emotion are as follows:

0: -4593 images- Angry
1: -0547 images- Disgust
2: -5121 images- Fear
3: -8989 images- Happy
4: -6077 images- Sad
5: -4002 images- Surprise
6: -6198 images- Neutral

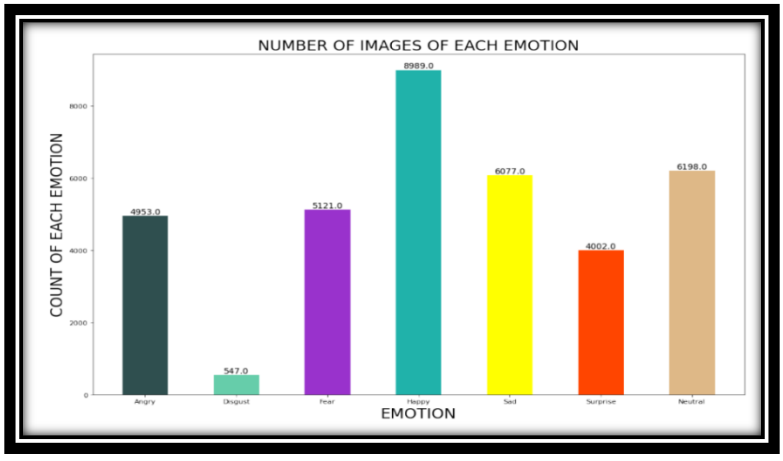

Fig 1: The bar graph displays data distribution of number of images for each emotion

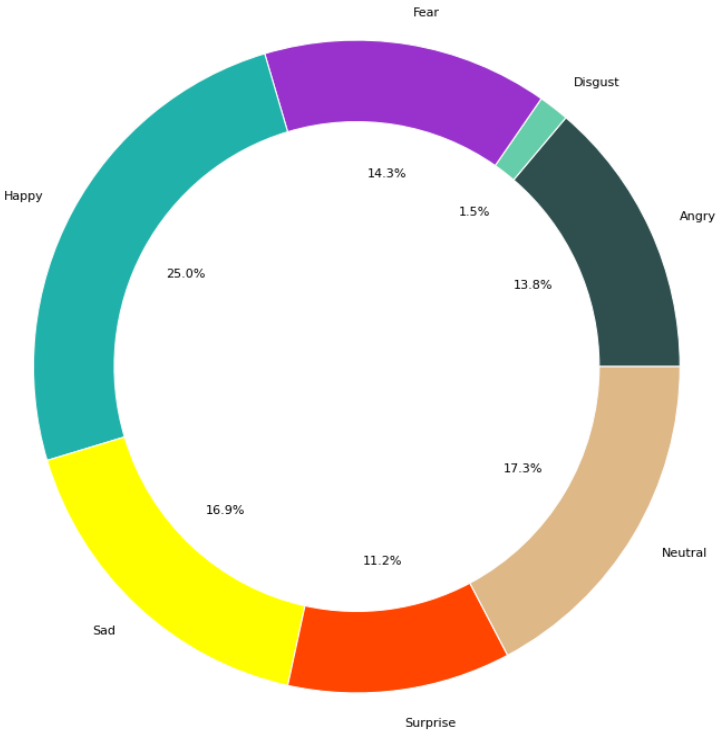

Fig 2: The Doughnut chart displays the percentage of data distribution of number of images for each emotion

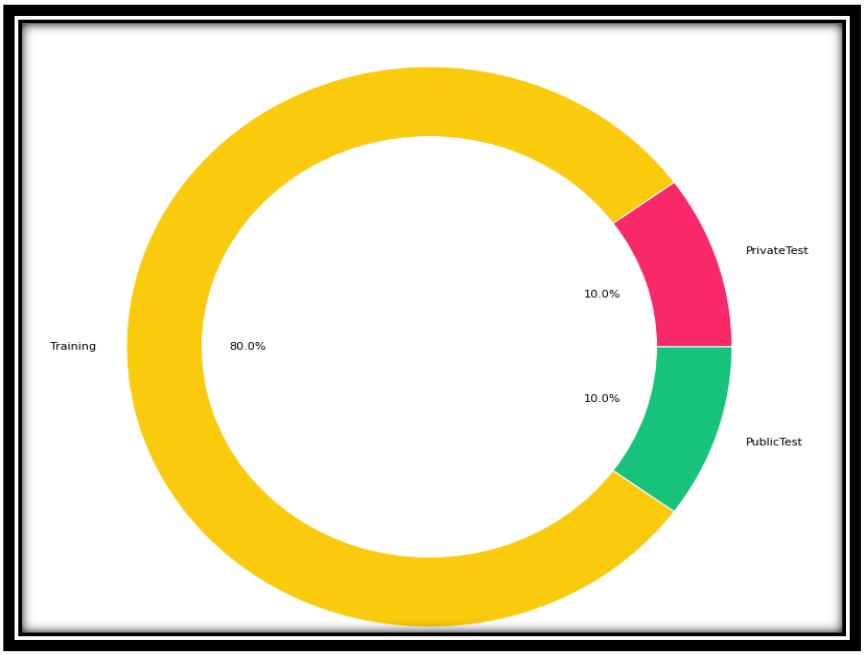

Fig 3: The Doughnut chart displays the data distributed for training, public test, private test

2.1.2 SONGS DATABASE

We have created a database containing songs of 4 different languages (Telugu, Tamil, English, Hindi) and in each language we have categorized songs based on the emotion/mood of the users most probably listening to them. Based on this the songs can be divided into 6 categories (Angry, Sad, Happy, Neutral, Surprise, Disgust)for each language[3]. We have collected 100 songs per each emotion in each language and we will update the dataset with another set of songs in the future.

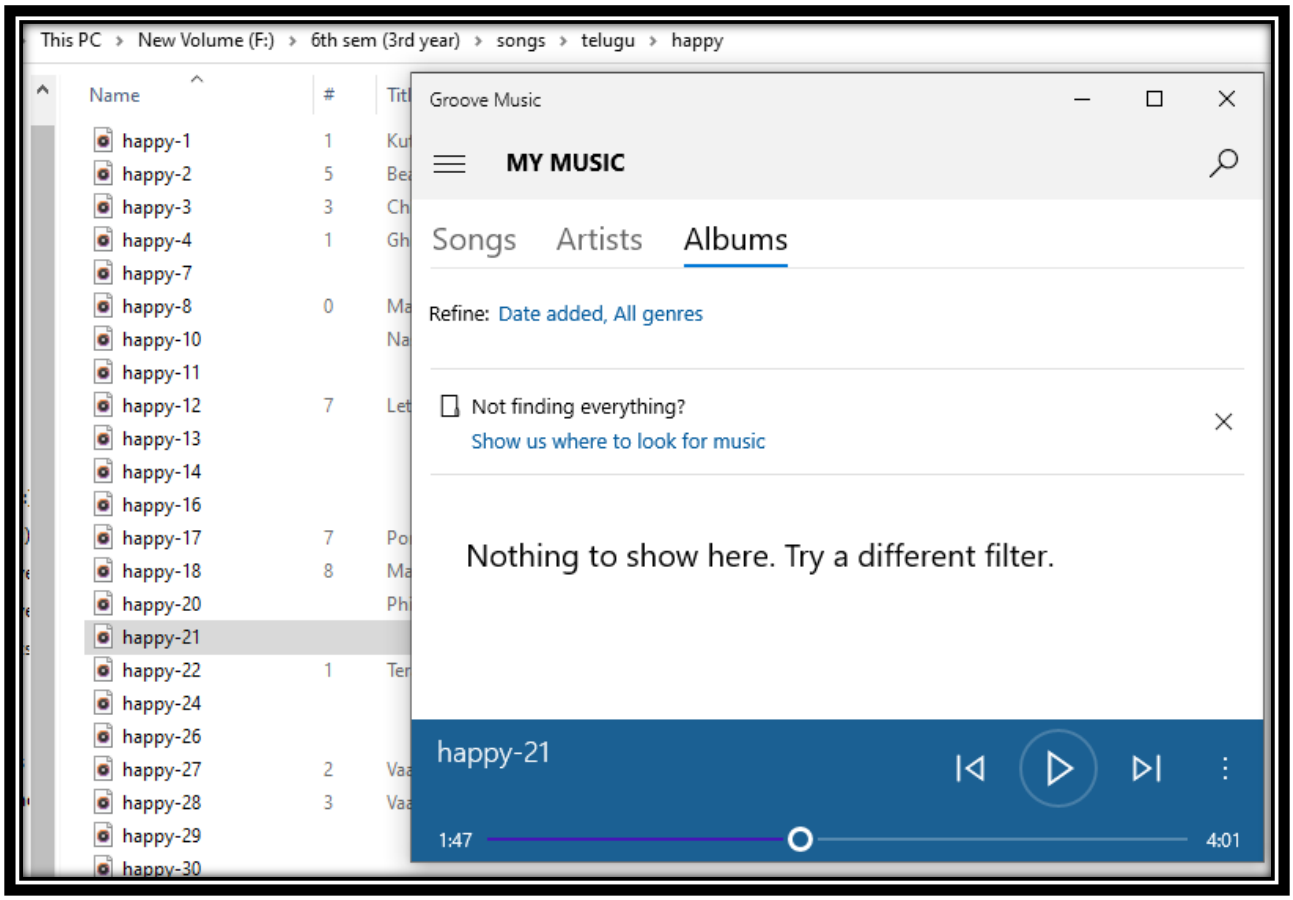

Fig 4: PLAYING OF A SONG FROM HAPPY DATASET OF SONGS IN TELUGU DATABASE.

2.2 PROPOSED SYSTEM

2.2.1 MULTI LEVEL CNN:

The below given 18 layer CNN model in Fig 5 is inspired by VGG net. The architecture not only uses high level features but also considers mid-level features for classification[5]. These feature maps are extracted from 2^{nd}, 3^{rd}, and 4^{th} convolutional layers. Since the First layer includes trivial filters we don't consider its features at the end.

In addition, according to the observation on the mid-level features carried out in the previous section, in the same block, there are a number of unimportant filters for facial feature even though the number of filters in low and mid-level layers is higher compared to high-level layer[4]. Therefore, we insert a fully connected layer with 256 units to reserve useful filters as well as eliminate insignificant ones.

Another issue that most of deep learning researchers have to deal with is how to choose the depth of the network[6]. Here we have used KERAS TUNER library which does the hyper parameter tuning to the given Deep learning model and returns the depth of the network with the size of the filters.

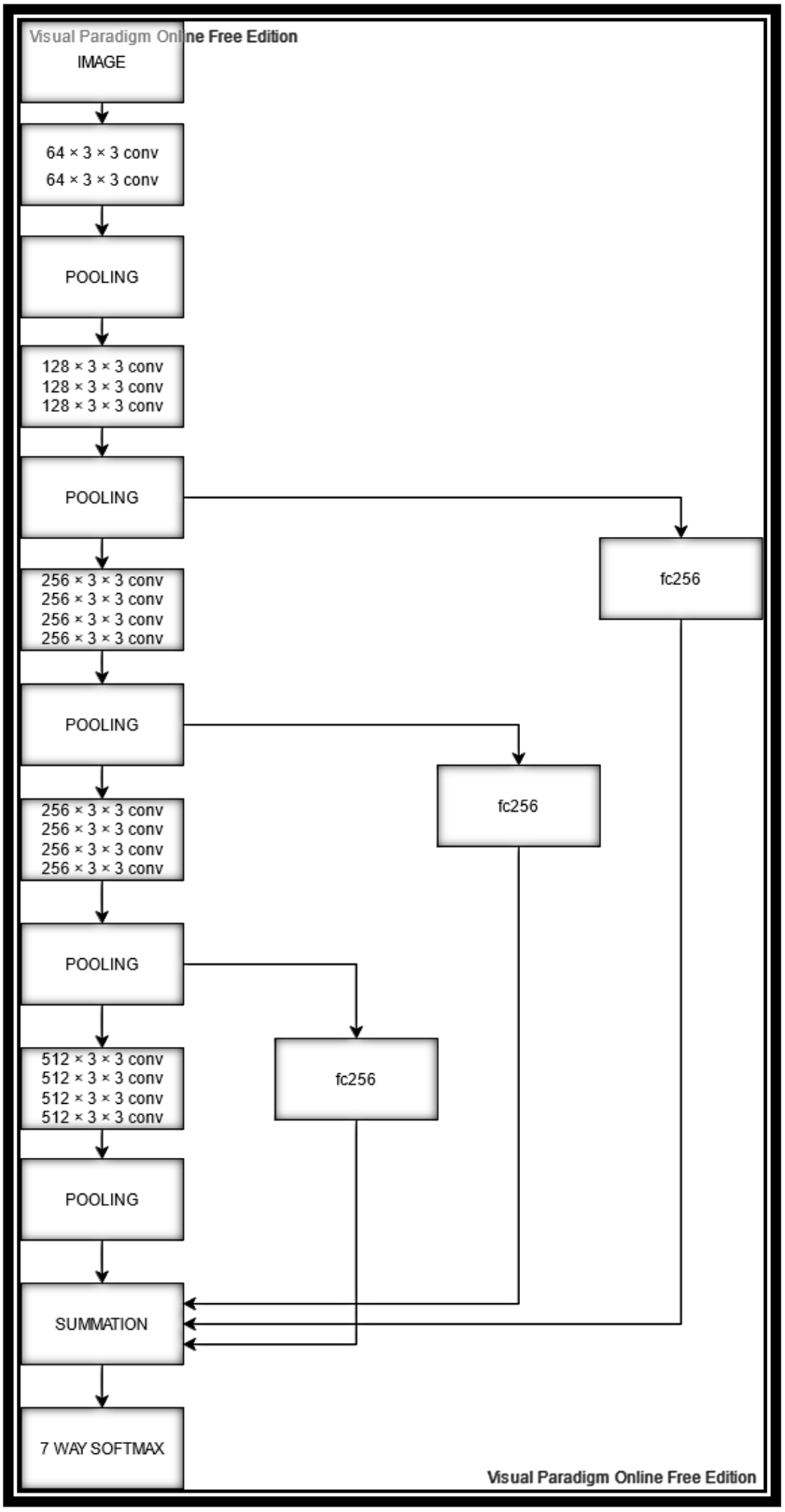

Fig 5: Representation of Multi level CNN used

2.2.2 GUI

In Fig 6 we can see the basic GUI model without inputs.

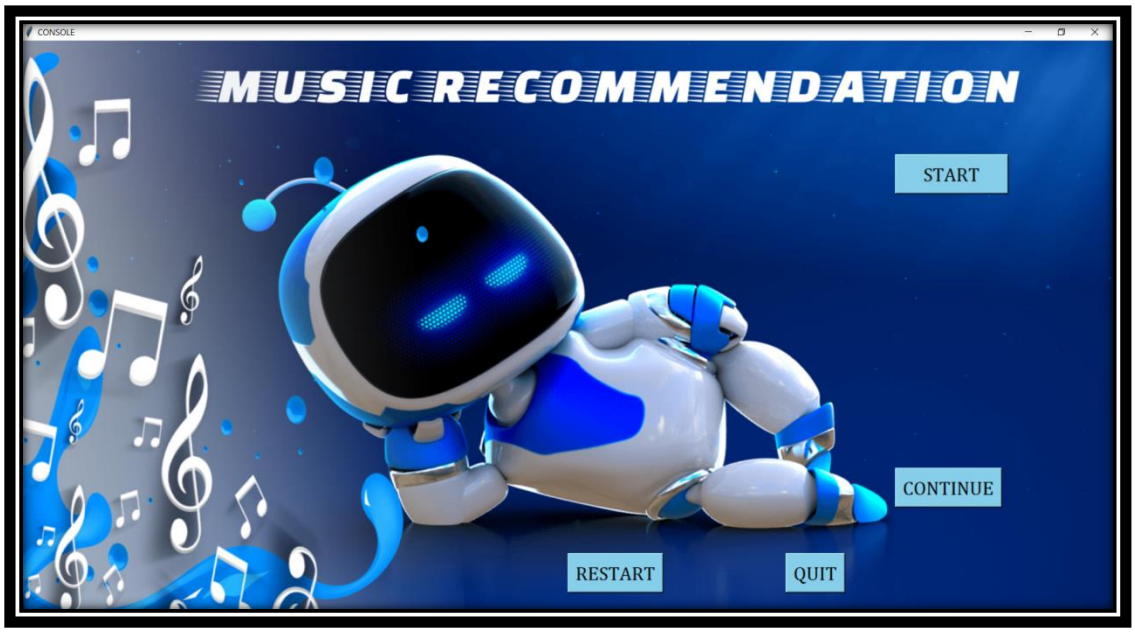

Fig 6: A basic screenshot of the GUI

In Fig 7 The GUI displays the languages to be selected after giving our name to the console

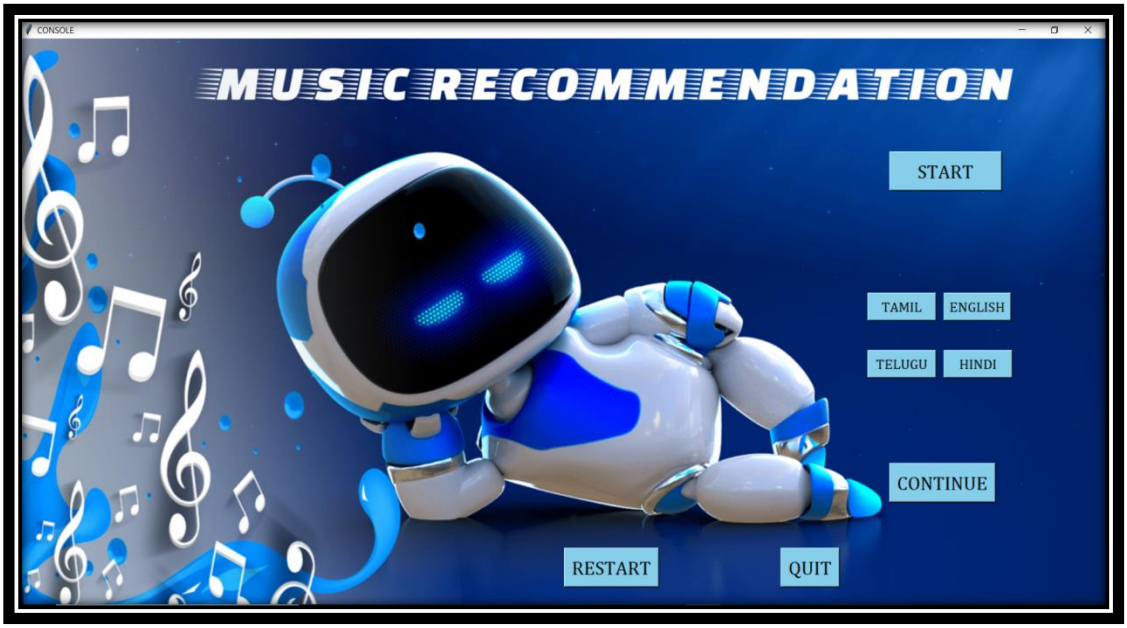

Fig 7: Languages displayed when details are entered

Once we select a language the camera pops up and captures the image is shown in Fig 8.

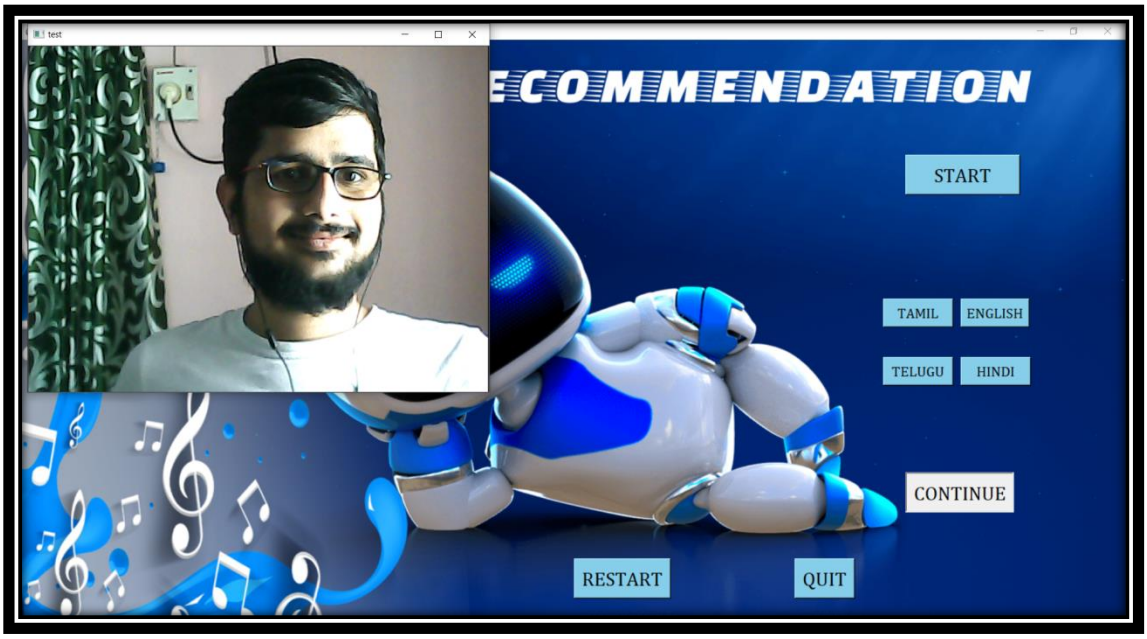

Fig 8: Camera popped when language is selected

Once the image is captured emotion from the image is captured and a song from the database of that emotion is played is shown in Fig 9.

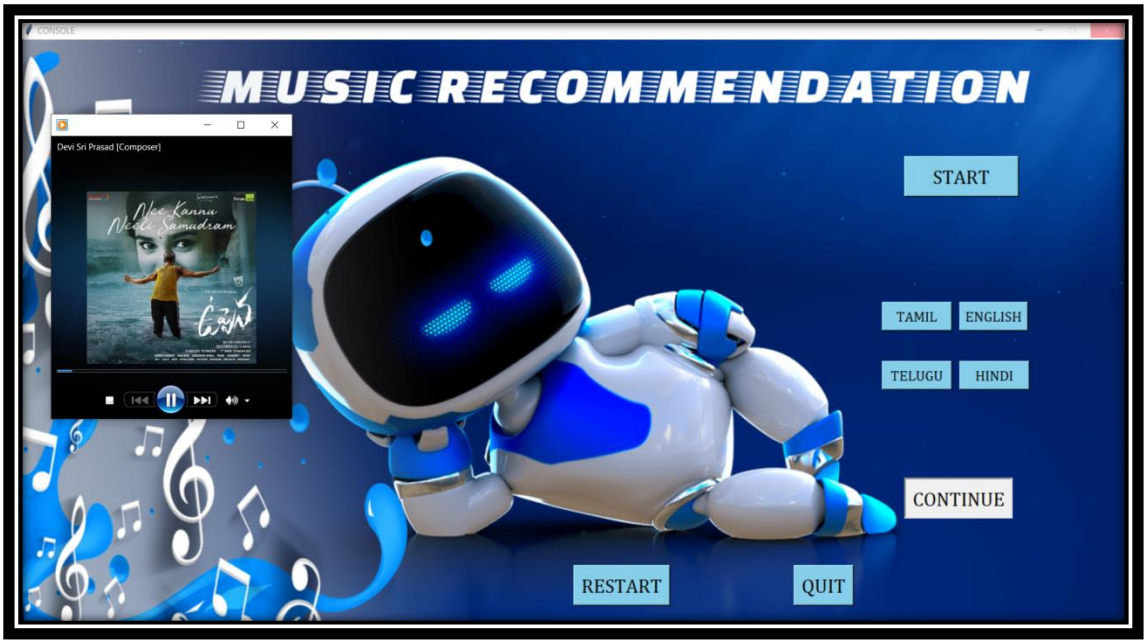

Fig 9: A song played based on the emotion predicted

2.2.3 BLOCK DIAGRAM

Fig 10 represents the block diagram of our complete model.Firstly the FER2013 dataset is fed to Multi level CNN and the result is saved as MODEL.json.

Now when the GUI starts it asks the user to enter details.If already present it moves to the webcam else it stores the data and then opens the webcam.Once the webcam is open, it captures the image of the user and recognizes the emotion of the user from the MODEL.json and predicts the emotion.

Once the emotion is recognized Song from the songs database related to same emotion is played to the user.

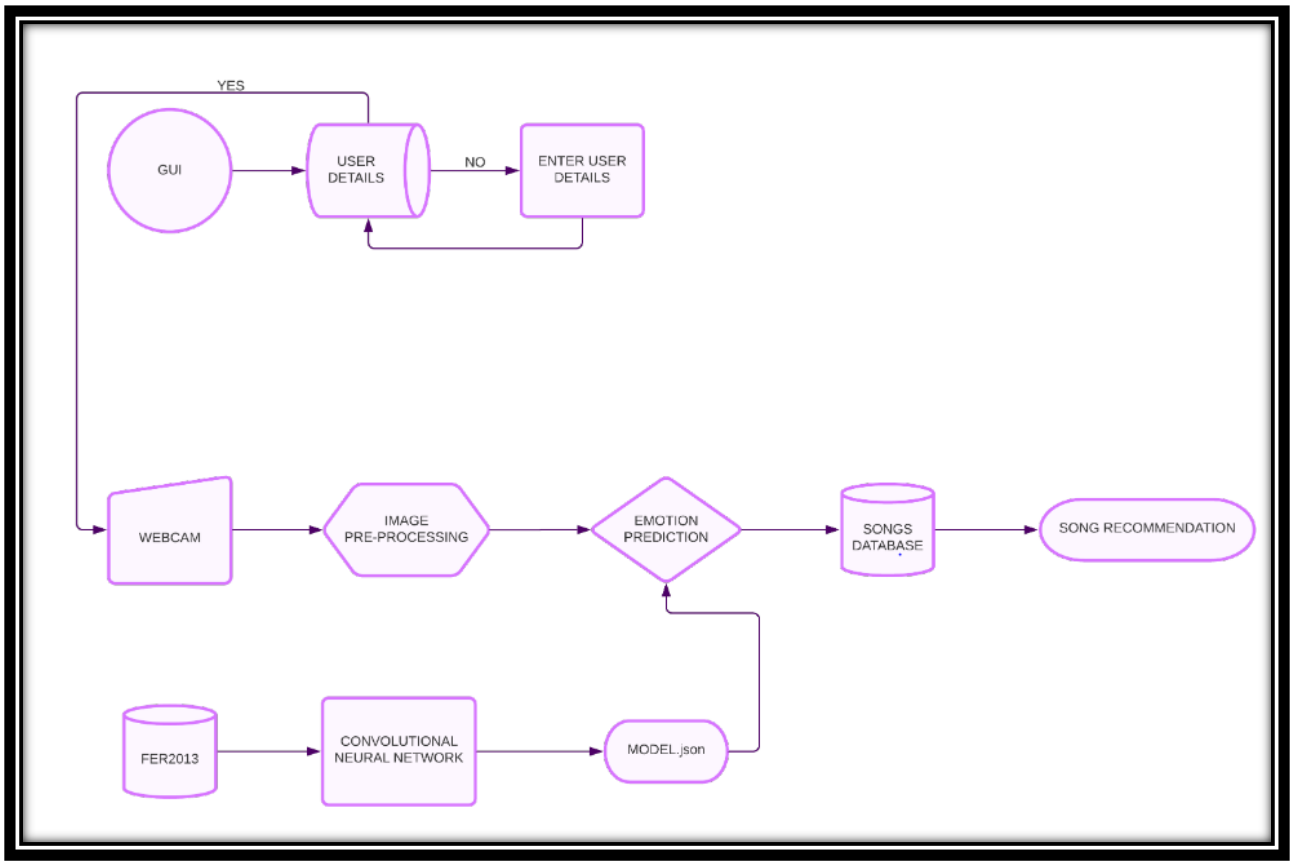

Fig 10: BLOCK DIAGRAM OF THE MODEL

3 MODULE DESCRIPTION

Face detection is a computer technology that determines the location and size of human face in arbitrary (digital) image. The facial features are detected and any other objects like trees, buildings and bodies etc are ignored from the digital image[7]. Basically there are two types of approaches to detect facial part in the given image i.e. feature base and image base approach. Feature base approach tries to extract features of the image and match it against the knowledge of the face features. While image base approach tries to get best match between training and testing images. Convolutional Neural Networks(CNN) comes under the category of image based approach.

There are main operations in the Convolution Neural Network and our project are:

3.1 CONVOLUTION:

Convolution is a mathematical operation on two functions that produces a third function that expresses how the shape of one is modified by the other. The main purpose of convolution in this case of a CNN is to extract features from the input image[9]. Convolution preserves the spatial relationship between pixels by learning image features using small squares of input data. The convolution layer's parameters consist of a set of learnable filters.

Convolutional neural networks are composed of multiple layers of artificial neurons. Artificial neurons, a rough imitation of their biological counterparts, are mathematical functions that calculate the weighted sum of multiple inputs and outputs an activation value[8]. When you input an image in a CNN, each layer generates several activation functions that are passed on to the next layer.

The first layer usually extracts basic features such as horizontal or diagonal edges. This output is passed on to the next layer which detects more complex features such as corners or combinational edges. As we move deeper into the network it can identify even more complex features such as objects, faces, etc. Based on the activation map of the final convolution layer, the classification

layer outputs a set of confidence scores (values between 0 and 1) that specify how likely the image is to belong to a "class."

An example of how the convolution in CNN of an image takes place is shown in the Fig 11

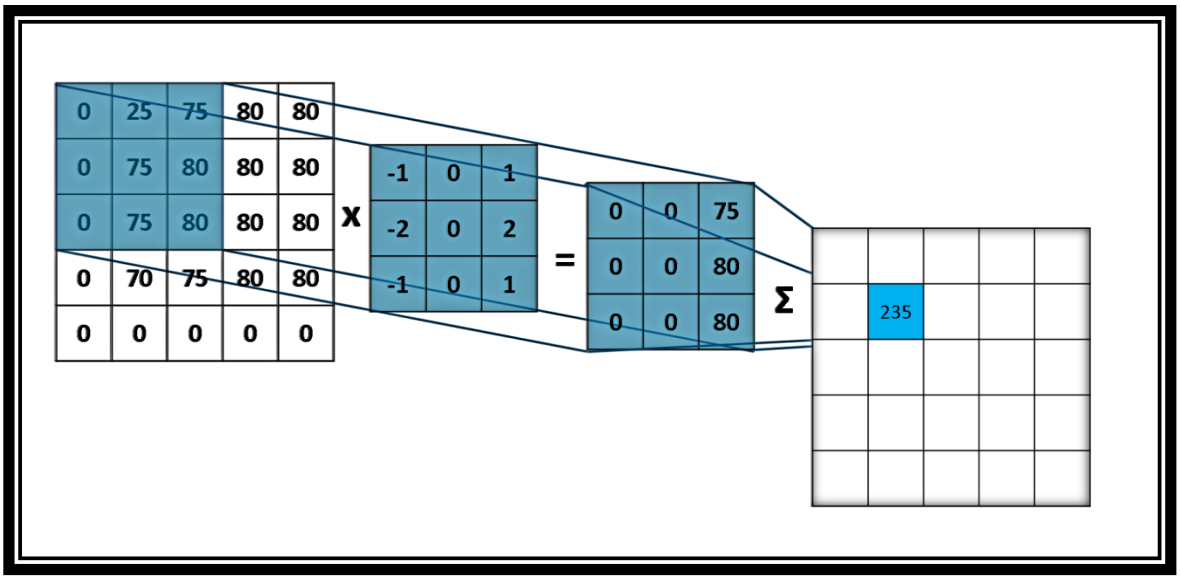

Fig 11: Depicts convolution inside a layer

3.2 RECTIFIED LINEAR UNIT(RELU):

In a neural network, the activation function is responsible for transforming the summed weighted input from the node into the activation of the node or output for that input.

The rectified linear activation function or Relu, that will output the input directly if it is positive, otherwise, it will output zero. It has become the default activation function for many types of neural networks because a model that uses it is easier to train and often achieves better performance. Relu has been used after every Convolution operation[10]. A Rectified Linear Unit (Relu) is a cell of a neural network which uses the following activation function to calculate its output given x:

$R(x) = Max(0,x)$

A graphical representation of Relu activation function is given in Fig 12.

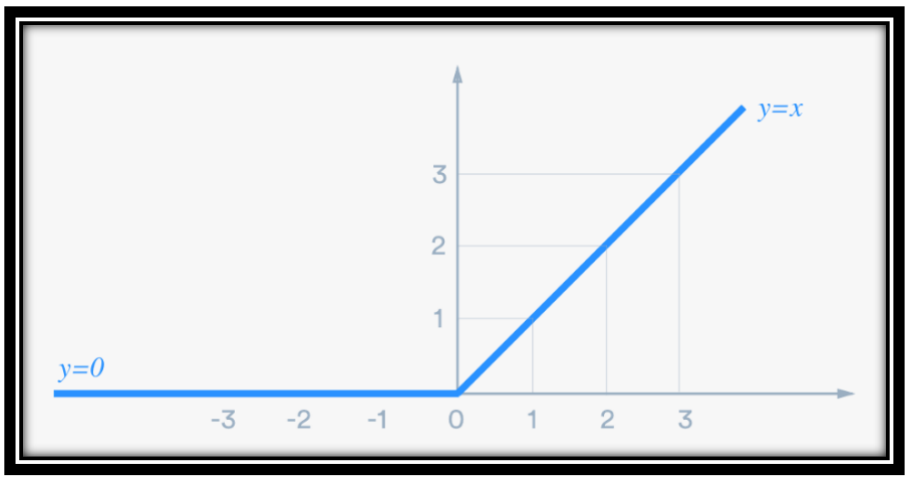

Fig 12: Function of RELU activation function

3.3 MAX-POOLING:

Similar to the Convolutional Layer, the Pooling layer is responsible for reducing the spatial size of the Convolved Feature. This is to decrease the computational power required to process the data by reducing the dimensions.

In Max Pooling is we find the maximum value of a pixel from a portion of the image covered by the kernel. Max Pooling also performs as a Noise Suppressant. It discards the noisy activations altogether and also performs de-noising along with dimensionality reduction.

A representation of kernel 2x2 and stride 2x2 of maxpooling layer is depicted in Fig 13.

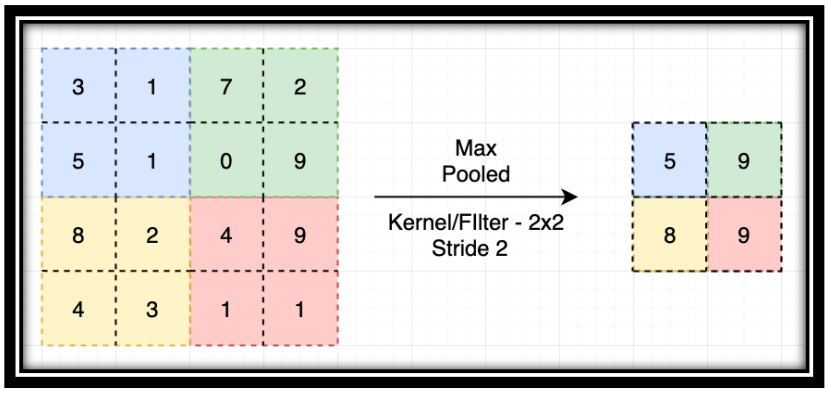

Fig 13: Max Pooling Layer description

3.4 CLASSIFICATION:

The Fully Connected layer is a traditional Multi-Layer Perceptron that uses a softmax activation function in the output layer. The term “Fully Connected” implies that every neuron in the previous layer is connected to every neuron on the next layer. The output from the convolutional and pooling layers represent high-level features of the input image.

The purpose of the Fully Connected layer is to use these features for classifying the input image into various classes based on the training dataset. Softmax is used for activation function[11]. It treats the outputs as scores for each class. In the Softmax, the function mapping stayed unchanged and these scores are interpreted as the un normalized log probabilities for each class.

A complete walkthrough of CNN model is given in the Fig 14.

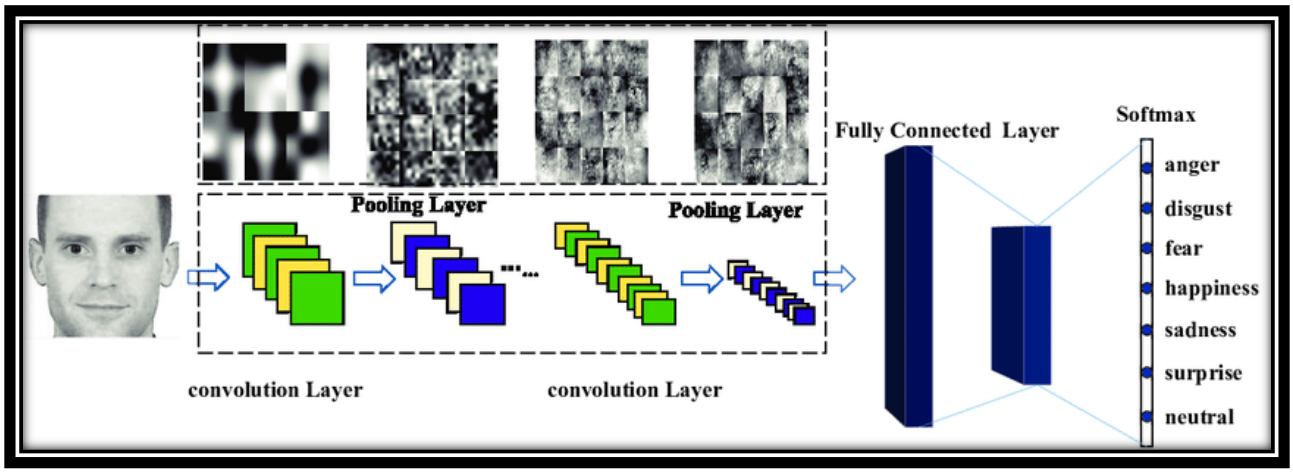

Fig 14: A complete block diagram of CNN model

3.5 MODEL SUMMARY

The size of each image is (54, 54, 3),the Keras then appends an extra dimension for processing multiple batches, i.e., to train multiple images in every step of a single epoch. Since batch size can vary, its size is represented by None. Hence, the input shape becomes (None, 54, 54, 3).

Convolving a (54, 54) image with a (3, 3) filter, with strides and dilation rate of 1, and 'valid' padding, results in an output of size (54-3+1, 54-3+1) = (52, 52). Since you have 64 such filters, the output shape becomes (52, 52, 64).

The default MaxPooling kernel has a shape of (2, 2) and strides of (2, 2). Applying that to a (29, 29) image results in an image of shape (((52 - 2)//2) + 1, ((52- 2)//2) + 1)) = (26, 26). Since you have 64 such filters, the output shape becomes (26, 26, 64)[12].

The later process of convolution is followed as in Table I.

Table I Sequential model analysis

Model: "sequential"

Layer (type)	Output Shape	Param #
zero_padding2d (ZeroPadding2	(None, 52, 52, 1)	0
conv2d (Conv2D)	(None, 52, 52, 64)	640
conv2d_1 (Conv2D)	(None, 52, 52, 64)	36928
batch_normalization (BatchNo	(None, 52, 52, 64)	256
max_pooling2d (MaxPooling2D)	(None, 26, 26, 64)	0
dropout (Dropout)	(None, 26, 26, 64)	0
conv2d_2 (Conv2D)	(None, 26, 26, 128)	73856
conv2d_3 (Conv2D)	(None, 26, 26, 128)	147584
conv2d_4 (Conv2D)	(None, 26, 26, 128)	147584
batch_normalization_1 (Batch	(None, 26, 26, 128)	512
max_pooling2d_1 (MaxPooling2	(None, 13, 13, 128)	0
dropout_1 (Dropout)	(None, 13, 13, 128)	0
conv2d_5 (Conv2D)	(None, 13, 13, 256)	295168
conv2d_6 (Conv2D)	(None, 13, 13, 256)	590080
conv2d_7 (Conv2D)	(None, 13, 13, 256)	590080
conv2d_8 (Conv2D)	(None, 13, 13, 256)	590080

batch_normalization_2 (Batch	(None, 13, 13, 256)	1024
max_pooling2d_2 (MaxPooling2	(None, 6, 6, 256)	0
dropout_2 (Dropout)	(None, 6, 6, 256)	0
conv2d_9 (Conv2D)	(None, 6, 6, 256)	590080
conv2d_10 (Conv2D)	(None, 6, 6, 256)	590080
conv2d_11 (Conv2D)	(None, 6, 6, 256)	590080
conv2d_12 (Conv2D)	(None, 6, 6, 256)	590080
batch_normalization_3 (Batch	(None, 6, 6, 256)	1024
max_pooling2d_3 (MaxPooling2	(None, 3, 3, 256)	0
dropout_3 (Dropout)	(None, 3, 3, 256)	0
conv2d_13 (Conv2D)	(None, 3, 3, 512)	1180160
conv2d_14 (Conv2D)	(None, 3, 3, 512)	2359808
conv2d_15 (Conv2D)	(None, 3, 3, 512)	2359808
conv2d_16 (Conv2D)	(None, 3, 3, 512)	2359808
batch_normalization_4 (Batch	(None, 3, 3, 512)	2048
max_pooling2d_4 (MaxPooling2	(None, 1, 1, 512)	0
dropout_4 (Dropout)	(None, 1, 1, 512)	0
conv2d_17 (Conv2D)	(None, 1, 1, 512)	262656
flatten (Flatten)	(None, 512)	0
dense (Dense)	(None, 7)	3591

3.6 OPEN CV:

OpenCV is a Python open-source library, which is used for computer vision in Artificial intelligence, Machine Learning, face recognition, etc.InOpenCV, the CV is an abbreviation form of a computer vision, which is defined as a field of study that helps computers to understand the content of the digital images such as photographs and videos.The purpose of computer vision is to

understand the content of the images[13]. It extracts the description from the pictures, which may be an object, a text description, and three-dimension model, and so on. Computer vision allows the computer to perform the same kind of tasks as humans with the same efficiency. There are a two main task which are defined below:

- **Object Classification -** In the object classification, we train a model on a dataset of particular objects, and the model classifies new objects as belonging to one or more of your training categories.
- **Object Identification -** In the object identification, our model will identify a particular instance of an object

3.7 SPEECH RECOGNITION:

Speech recognition system basically translates the spoken utterances to text. There are various real life examples of speech recognition system.

The advantage of using a speech recognition system is that it overcomes the barrier of literacy. A speech recognition model can serve both literate and illiterate audience as well, since it focuses on spoken utterances. Speech recognition can be approached in many ways. It began with using simple templates to detect beeps, and slowly moved towards understanding its frequency components.

A flowchart of how Google Recognizer module recognizes the audio is given in the Fig 15.

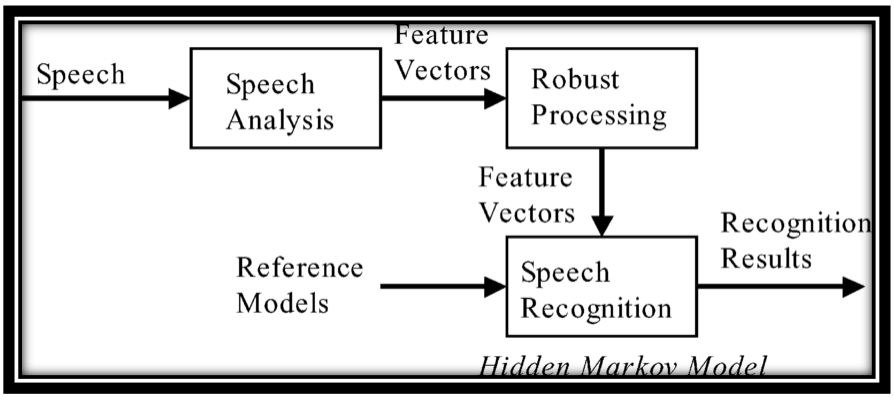

Fig 15: Flow Chart of Speech recognition

4. RESULTS AND DISCUSSION

We trained the our neural network using Keras with TensorFlow backend and the training data from FER2013 dataset. The weights were initialized randomly before updating by Adam optimizer using mini batch size of 1024. The learning rate started from 10^-3 and the minimum value was 10^-8. To prevent over fitting, data augmentation was performed on training data.
Data preprocessing techniques include normalization, translation, rotation, scaling, and mirroring, and augmentation. In addition, we added dropout after maxpooling layers in order to improve the network generalization capability. To train our model, we initialized it with the weights of the plain architecture before training for around 30 iterations and the model with the best performance was selected. The loss function that we used to train our models is cross entropy and the activation function is ReLU.
Fig 16 shows different emotions captured and we can observe that there are some mis-predictions which are due to lack of efficient data.

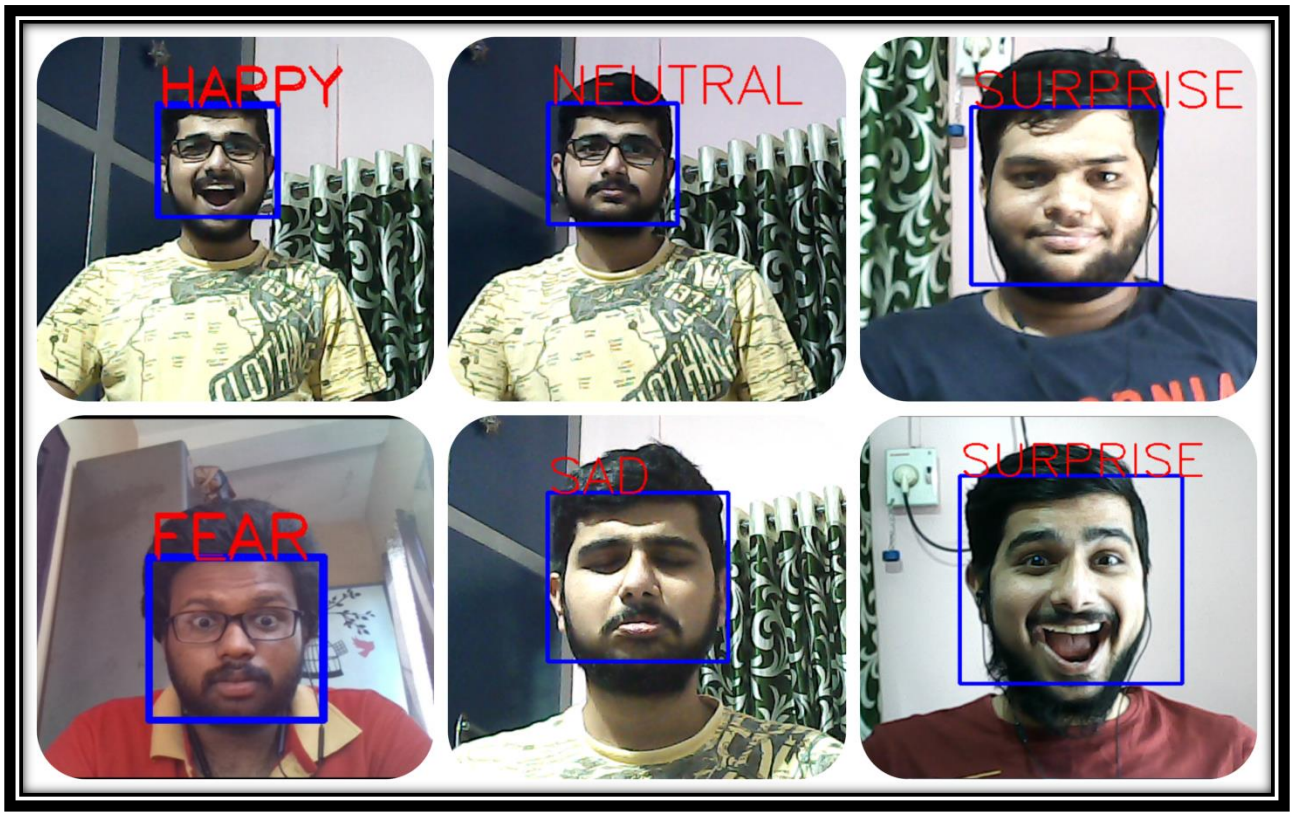

Fig 16: Depicts different emotions detected

Fig 17 shows the confusion matrix of our test set. Even though the dataset is imbalanced we got a better result by adding up features from different layers. We have compared our model with different existing research papers which has been shown in Table II.

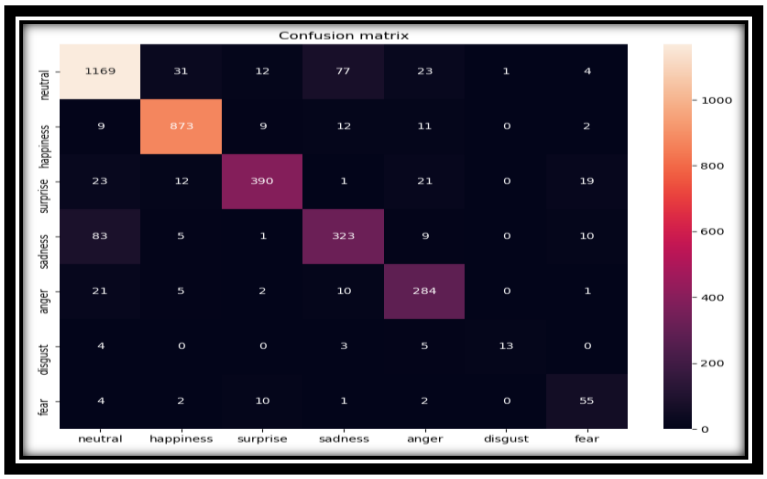

Fig 17: Confusion Matrix of test set.

Table II Performance comparison on FER2013 dataset

Proposed Method	Test Accuracy
CNN + Batch Normalization + Varying number of filters	65%
CNN + Batch Normalization + GAP	66%
New architecture + polynomial learning rate	66.40%
Custom CNN	66.67%
New architecture based on attentional CNN	70.02%
SBNN + L2-SVMs (DLSVM)	71.20%
Multi-Level CNN (MLCNN)	73.03%
OUR MODEL	72.8%

Our model is more robust since it extracts features from different layers and hence it has an accuracy of 72.8% and f1 score of 0.8. Many other models extract the features only from final layer which might have different unwanted filter outputs. Hence our model resolves all these problems.

Although our model performs better it involves double training for extracting features into summing layers and into pooling layer which is the only drawback of our model.

5 CONCLUSION

In this project, we presented a model to recommend music based on emotion based on facial expression. This project proposes designing & developing an emotion based music recommendation system using emotion detected from the face recognition System. We have proposed a model which detects emotion of a group of persons in a given image by their facial expression using convolutional neural networks. It primarily comprises three parts i.e., face representation, emotion pattern recognition and classification. CNN detects the patterns of emotion as we trained the model with a large amount of data. The proposed model detects seven major emotions: they are happy, sad, angry, fear, disgust, surprise and neutral. We achieved a performance of detecting expressions with 72.8% accuracy.

We have included music corresponding to the emotions captured from the user. The song is played automatically after detecting the emotions. Each folder contains a playlist of songs which the emotion based smart music player plays simultaneously; after emotion is matched the song is played. We have created a GUI to make the model more feasible.

6 CODE:

```
import sys, os
import pandas as pd
import numpy as np

from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation, Flatten
from keras.layers import Conv2D, MaxPooling2D,
BatchNormalization,AveragePooling2D
from keras.losses import categorical_crossentropy
from keras.optimizers import Adam
from keras.regularizers import l2
from keras.utils import np_utils
# pd.set_option('display.max_rows', 500)
# pd.set_option('display.max_columns', 500)
# pd.set_option('display.width', 1000)

# In[2]:

df=pd.read_csv('fer2013.csv')

# print(df.info())
# print(df["Usage"].value_counts())

# print(df.head())
X_train,train_y,X_test,test_y=[],[],[],[]

# In[3]:

for index, row in df.iterrows():
  val=row['pixels'].split(" ")
  try:
    if 'Training' in row['Usage']:
      X_train.append(np.array(val,'float32'))
      train_y.append(row['emotion'])
    elif 'PublicTest' in row['Usage']:
      X_test.append(np.array(val,'float32'))
```

```
            test_y.append(row['emotion'])
        except:
            print(f"error occured at index :{index} and row:{row}")

# In[4]:
X_train = np.array(X_train,'float32')
train_y = np.array(train_y,'float32')
X_test = np.array(X_test,'float32')
test_y = np.array(test_y,'float32')

train_y=np_utils.to_categorical(train_y, num_classes=7)
test_y=np_utils.to_categorical(test_y, num_classes=7)

#cannot produce
#normalizing data between oand 1
X_train -= np.mean(X_train, axis=0)
X_train /= np.std(X_train, axis=0)

X_test -= np.mean(X_test, axis=0)
X_test /= np.std(X_test, axis=0)

X_train = X_train.reshape(X_train.shape[0], 48, 48, 1)

X_test = X_test.reshape(X_test.shape[0], 48, 48, 1)

# print(f"shape:{X_train.shape}")
##designing the cnn


# In[5]:


X_train.shape

# In[6]:

import tensorflow as tf

# In[7]
```

```
#1st convolution layer
model = Sequential()
model.add(tf.keras.layers.ZeroPadding2D(padding=(2, 2)))
model.add(Conv2D(64, kernel_size=(3, 3), activation='relu',input_shape =
(48,48,1),padding="same"))
model.add(Conv2D(64,kernel_size= (3, 3), activation='relu',padding="same"))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2), strides=(2, 2)))
model.add(Dropout(0.5))

#2nd convolution layer
model.add(Conv2D(128, (3, 3), activation='relu',padding="same"))
model.add(Conv2D(128, (3, 3), activation='relu',padding="same"))
model.add(Conv2D(128, (3, 3), activation='relu',padding="same"))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2), strides=(2, 2)))
model.add(Dropout(0.5))

#3rd convolution layer
model.add(Conv2D(256, (3, 3), activation='relu',padding="same"))
model.add(Conv2D(256, (3, 3), activation='relu',padding="same"))
model.add(Conv2D(256, (3, 3), activation='relu',padding="same"))
model.add(Conv2D(256, (3, 3), activation='relu',padding="same"))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2), strides=(2, 2)))
model.add(Dropout(0.5))

#4rd layer
model.add(Conv2D(256, (3, 3), activation='relu',padding="same"))
model.add(Conv2D(256, (3, 3), activation='relu',padding="same"))
model.add(Conv2D(256, (3, 3), activation='relu',padding="same"))
model.add(Conv2D(256, (3, 3), activation='relu',padding="same"))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2), strides=(2, 2)))
model.add(Dropout(0.5))

#5th layer
model.add(Conv2D(512, (3, 3), activation='relu',padding="same"))
```

```
model.add(Conv2D(512, (3, 3), activation='relu',padding="same"))
model.add(Conv2D(512, (3, 3), activation='relu',padding="same"))
model.add(Conv2D(512, (3, 3), activation='relu',padding="same"))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2), strides=(2, 2)))
model.add(Dropout(0.5))
model.add(Conv2D(512,(1,1),activation = "relu"))
model.add(Flatten())
model.add(Dense(7, activation='softmax'))

# model.summary()

#Compliling the model

# In[8]:


model.compile(loss=categorical_crossentropy,
          optimizer=Adam(),
          metrics=['accuracy'])

#Training the model
model.fit(X_train, train_y,
        batch_size=64,
        epochs=30,
        validation_data=(X_test, test_y),
        shuffle=True)

#Saving the  model to  use it later on

# In[9]:

fer_json4 = model.to_json()
with open("fer8.json", "w") as json_file:
   json_file.write(fer_json4)
model.save_weights("fer8.h5")
model.summary()
```

```
import pandas as pd
import numpy as np
import seaborn as sns
from IPython.display import HTML, Javascript, display
import IPython
import tkinter.font as font
import speech_recognition as sr
import tkinter as tk
from tkinter import ttk
from tkinter import *
from tkinter import simpledialog
from PIL import Image, ImageTk
from win32com.client import constants, Dispatch
import pyttsx3


# In[2]:


import os
import cv2
import numpy as np
from tensorflow.keras.models import model_from_json
from tensorflow.keras.preprocessing import image
import subprocess
import random


# In[3]:


mp = 'C:/Program Files (x86)/Windows Media Player/wmplayer.exe'


# In[4]:


#load model
model = model_from_json(open("fer8.json", "r").read())
```

```
#load weights
model.load_weights('fer8.h5')
face_haar_cascade =
cv2.CascadeClassifier('haarcascade_frontalface_default.xml')


# In[5]:


root = Tk()
root.title("CONSOLE")
root.geometry("5000x2000")
root.configure(background="black")
engine = pyttsx3.init()

class Example(Frame):
   def __init__(self, master, *pargs):
      Frame.__init__(self, master, *pargs)



      self.image = Image.open("501.png")
      self.img_copy= self.image.copy()


      self.background_image = ImageTk.PhotoImage(self.image)

      self.background = Label(self, image=self.background_image)
      self.background.pack(fill=BOTH, expand=YES)
      self.background.bind('<Configure>', self._resize_image)

   def _resize_image(self,event):

      new_width = event.width
      new_height = event.height

      self.image = self.img_copy.resize((new_width, new_height))

      self.background_image = ImageTk.PhotoImage(self.image)
```

```
        self.background.configure(image =  self.background_image)

myFont1 = font.Font(family = "Cambria",size=15)
myFont = font.Font(family = "Cambria",size=12)
myFont2 = font.Font(family = "Cambria",size=20)

e = Example(root)
e.pack(fill=BOTH, expand=YES)
def start():
   global text
   engine.say("HI!....Welcome to Music Recommendation!..")
   engine.runAndWait()

   r = sr.Recognizer()

   # Reading Microphone as source
   # listening the speech and store in audio_text variable

   with sr.Microphone() as source:
      print("Talk")
      Msg = " May I know your Name?"
      engine.say(Msg)
      engine.runAndWait()
      audio_text = r.listen(source)
      print("Time over, thanks")
   # recoginize_() method will throw a request error if the API is unreachable,
hence using exception handling

      try:
         # using google speech recognition
         print("Text: "+r.recognize_google(audio_text))
         text = r.recognize_google(audio_text)
      except:
          print("Sorry, I did not get that")

   Msg = "Hi!.."+text+"!..Please Select a language"
   engine.say(Msg)
   engine.runAndWait()
```

```
    def chosingNumbers():
        global text
        global text2
        text = "Thank You for choosing telugu"
        text2 = "G:/songs/telugu"
    button = Button(root, text = "TELUGU", command = chosingNumbers,font =
myFont1,width = 8,bg='sky blue',fg = 'black')
    button.place(relx = 0.78,rely = 0.55)
    def chosingNumbers2():
        global text
        global text2
        text = "Thank You for choosing HINDI"
        text2 = "G:/songs/hindi"
    button2 = Button(root, text = "HINDI", command = chosingNumbers2,font =
myFont1,width = 8,bg='sky blue',fg = 'black')
    button2.place(relx = 0.85,rely = 0.55)
    def chosingNumbers3():
        global text
        global text2
        text = "Thank You for choosing ENGLISH"
        text2 = "G:/songs/english"
    button3 = Button(root, text = "ENGLISH", command =
chosingNumbers3,font = myFont1,bg='sky blue',fg = 'black')
    button3.place(relx = 0.85,rely = 0.45)
    def chosingNumbers4():
        global text
        global text2
        text = "Thank You for choosing TAMIL"
        text2 = "G:/songs/tamil"
    button4 = Button(root, text = "TAMIL", command = chosingNumbers4,width
= 8,font = myFont1,bg='sky blue',fg = 'black')
    button4.place(relx = 0.78,rely = 0.45)


button7 = Button(root, text = "START",font = myFont2, command =
start,width = 10,bg='sky blue',fg = 'black')
button7.place(relx = 0.8,rely = 0.2)
global text3
def continuous():
```

```
    global text3
    Msg = text
    engine.say(Msg)
    engine.runAndWait()

    Msg = "Would you like to use emotion detection?.. "

    engine.say(Msg)
    engine.runAndWait()

    r = sr.Recognizer()

    with sr.Microphone() as source:
        print("Talk")
        Msg = "If yes reply open else reply random "
        engine.say(Msg)
        engine.runAndWait()
        audio_text = r.listen(source)
        print("Time over, thanks")
    # recoginize_() method will throw a request error if the API is unreachable,
hence using exception handling


    print("Text: "+r.recognize_google(audio_text))
    text3 = r.recognize_google(audio_text)
    text3 = text3.lower()
    if("open" in text3):
        print("Emotion Recognition")
        engine.say("Emotion Reconition selected")
        engine.runAndWait()
        cam = cv2.VideoCapture(0)

        cv2.namedWindow("test")

        img_counter = 0

        while True:
            ret, frame = cam.read()
            if not ret:
```

```
            print("failed to grab frame")
            break
        cv2.imshow("test", frame)

        k = cv2.waitKey(1)
        if k%256 == 27:
            # ESC pressed
            print("Escape hit, closing...")
            break
        elif k%256 == 32:
            # SPACE pressed
            img_name = "image{}.png".format(img_counter)
            cv2.imwrite(img_name, frame)
            print("{} written!".format(img_name))
            img_counter += 1

    cam.release()
    cv2.destroyAllWindows()
    sourceimage = cv2.imread('image0.png')
    test_img = np.copy(sourceimage)
    gray_img= cv2.cvtColor(test_img, cv2.COLOR_BGR2GRAY)
    faces_detected = face_haar_cascade.detectMultiScale(gray_img,1.25,5)
    print(faces_detected)
    for (x,y,w,h) in faces_detected:
        cv2.rectangle(test_img,(x,y),(x+w,y+h),(255,0,0),thickness=4)
        roi_gray=gray_img[y:y+w,x:x+h]#cropping region of interest i.e. face
area from  image
        roi_gray=cv2.resize(roi_gray,(48,48))
        img_pixels = image.img_to_array(roi_gray)
        img_pixels = np.expand_dims(img_pixels, axis = 0)
        img_pixels /= 255
    print(img_pixels)
    predictions = model.predict(img_pixels)
    print(predictions)
                #find max indexed array
    max_index = np.argmax(predictions[0])

    emotions = ('angry', 'disgust', 'fear', 'happy', 'sad', 'surprise', 'neutral')
    predicted_emotion = emotions[max_index]
```

```
    cv2.putText(test_img, predicted_emotion.upper(), (int(x), int(y)),
cv2.FONT_HERSHEY_SIMPLEX, 2, (0,0,255), 2)

    resized_img = cv2.resize(test_img, (1000, 700))
    cv2.imshow('Facial emotion analysis ',resized_img)
    while True:
      k = cv2.waitKey(1)
      if k%256 == 27:
            # ESC pressed
        print("Escape hit, closing...")
        break
    cv2.destroyAllWindows()
    text7 = predicted_emotion
    global dirc
    dirc = text2
    if text7 == 'angry':
      randomfile = random.choice(os.listdir(dirc+"/"+text7+"/" ))
      engine.say('You are angry !!!!... please calm down ,I will play song for
you :'+randomfile)
      engine.runAndWait()
      file = (dirc+"/"+text7+"/" + randomfile)
      subprocess.call([mp, file])

    if text7 == 'happy':
      randomfile = random.choice(os.listdir(dirc+"/"+text7+"/" ))
      engine.say('You are smiling.... ,I am playing special song for you: ' +
randomfile)
      engine.runAndWait()
      file = (dirc+"/"+text7+"/" + randomfile)
      subprocess.call([mp, file])

    if text7 == 'fear':
      randomfile = random.choice(os.listdir(dirc+"/"+text7+"/" ))
      engine.say('You have fear of something..... ,I am playing song for you: ' +
randomfile)
      engine.runAndWait()
      file = (dirc+"/"+text7+"/"+ randomfile)
      subprocess.call([mp, file])
```

```
    if text7 == 'sad':
        randomfile = random.choice(os.listdir(dirc+"/"+text7+"/" ))
        engine.say('You are sad,.....dont worry.... ,I am playing song for you: ' +
randomfile)
        engine.runAndWait()
        file = (dirc+"/"+text7+"/"+ randomfile)
        subprocess.call([mp, file])

    if text7 == 'neutral':
        randomfile = random.choice(os.listdir(dirc+"/"+text7+"/" ))
        engine.say('You are neutral.......,enjoy this song: ' + randomfile)
        engine.runAndWait()
        file = (dirc+"/"+text7+"/" + randomfile)
        subprocess.call([mp, file])
    if text7 == 'surprise':
        randomfile = random.choice(os.listdir(dirc+"/"+text7+"/" ))
        engine.say('You are neutral.......,enjoy this song: ' + randomfile)
        engine.runAndWait()
        file = (dirc+"/"+text7+"/" + randomfile)
        subprocess.call([mp, file])
    if text7 == 'disgust':
        randomfile = random.choice(os.listdir(dirc+"/"+text7+"/" ))
        engine.say('You are neutral.......,enjoy this song: ' + randomfile)
        engine.runAndWait()
        file = (dirc+"/"+text7+"/" + randomfile)
        subprocess.call([mp, file])

  else:
    print("Random song from language")
    engine.say("Random Song being played")
    engine.runAndWait()
    randomfile = random.choice(os.listdir(text2+"/"+"random"+"/" ))
    file = (text2+"/"+"random"+"/"+ randomfile)
    subprocess.call([mp, file])
button10 = Button(root, text = "CONTINUE", command = continuous,font =
myFont2,bg='sky blue',fg = 'Black')
button10.place(relx = 0.8,rely = 0.75)
def initialize():
```

```
    display(HTML(
    '''
      <script>
        code_show = false;

          IPython.notebook.kernel.restart();

            IPython.notebook.execute_all_cells();


      </script>
    '''
  ))
button10 = Button(root, text = "RESTART", command = initialize,bg='sky
blue',fg = 'black')
button10['font'] = myFont2
button10.place(relx=0.5, rely=0.9)

def close_window():
  root.destroy()
button5 = Button(root)
button5['text'] ="QUIT"
button5['command'] = close_window
button5['font'] = myFont2
button5['bg'] = 'sky blue'
button5.place(relx = 0.7,rely = 0.9)

root.mainloop()
```

7 REFERENCES

[1] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge:A retrospective," International Journal of Computer Vision, vol. 111,pp. 98–136, Jan. 2015.

[2] A.Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," tech. rep., University of Toronto, 2009.

[3] P. Carrier and A. Courville, "Challenges in representation learning: Facial expression recognition challenge." https://goo.gl/kVzT48, 2013.

[4] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," in Proceedings of the IEEE 86(11), pp. 2278–2324, 1998.

[5] A.Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in Advances in Neural Information Processing Systems 25 (F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, eds.), pp. 1097–1105, Curran Associates, Inc.,2012.

[6] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in International Conference on Learning Representations (ICLR), 2014.

[7] https://www.researchgate.net/publication/330611486_Using_CNN_for_facial_expression_recognition_a_study_of_the_effects_of_kernel_size_and_number_of_filters_on_accuracy

[8] https://ieeexplore.ieee.org/document/9299633

[9] https://ieeexplore.ieee.org/document/7477450

[10] https://www.researchgate.net/publication/336287978_Real_Time_Emotion_Recognition_from_Facial_Expressions_Using_CNN_Architecture

[11] https://arxiv.org/pdf/1902.01019.pdf

[12] https://www.cs.toronto.edu/~tang/papers/dlsvm.pdf

[13] https://www.researchgate.net/publication/329948265_Facial_Emotion_Recognition_Using_an_Ensemble_of_Multi-Level_Convolutional_Neural_Networks