# PROFESSIONAL TRAINING REPORT

**at**

**Sathyabama Institute of Science and Technology (Deemed to be University)**

Submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering Degree in Computer Science and Engineering

By

## SRAVANA SRIKANTH S

**REG. NO. 39110985**

****

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**SCHOOL OF COMPUTING**

**SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY**

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI,**

**CHENNAI – 600119, TAMILNADU**

**APRIL 2022**

|  |  |  |
| --- | --- | --- |
|  | **SATHYABAMA**  **INSTITUTE OF SCIENCE AND TECHNOLOGY** (DEEMED TO BE UNIVERSITY) **Accredited with Grade “A” by NAAC**  (Established under Section 3 of UGC Act, 1956)  JEPPIAAR NAGAR, RAJIV GANDHI SALAI  CHENNAI– 600119  [**www.sathyabama.ac.in**](http://www.sathyabama.ac.in) |  |

# DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

**BONAFIDE CERTIFICATE**

This is to certify that this Project Report is the bonafide work of **S. SRAVANA SRIKANTH (Reg. No: 39110985)** who carried out the project entitled “**EMOTION RECOGNITION**” under my supervision from Marsh 2022 to April 2022.

## Internal Guide

## Dr. S.Jancy

**Head of the Department**



## Submitted for Viva voce Examination held on

**InternalExaminer External Examiner**

**DECLARATION**

I, **S. SRAVANA SRIKANTH** hereby declare that the project report entitled **EMOTION DETECTION {REG NO:39110985}** done by me under the guidance of **Dr. S. Jancy** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering Degree in Computer Science and Engineering.

## DATE:

**PLACE: SIGNATURE OF THE CANDIDATE**

**ACKNOWLEDGEMENT**

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph.D**, **Dean**, School of Computing, **Dr. S. Vigneshwari, M.E., Ph.D. and Dr. L. Lakshmanan, M.E., Ph.D., Heads of the Department** of **Computer Science and Engineering** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. S. Jancy,** for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

**TRAINING CERTIFICATE**

# ABSTRACT

To provide better services and also better Human-machine interactions

Facial/Face Emotion Recognition(FER) is useful in most of the applications such as healthcare, teaching, criminal investigation, Human Robot Interface (HRI), etc.

Speech Emotion Recognition(SER) plays as important role in voice selection models actor training, call centers etc.

Image Emotion Recognition(IER); similar to FER is also used in this project for the better analysis. *{Here image represent to human face}*

Predicting the emotion in the best possible way and attaining it as a outcome is my key role in this project.

Humans posses different kinds of emotions. Among those I have predicted some of the major emotions. Those are; sad, happy, angry , calm, fearful, surprised, disgust.

# TABLE OF CONTENTS

# SCOPE OF THE PROJECT

# *UNIT 1* SPEECH EMOTION RECOGNITION

# 

# CHAPTER 1 INTRODUCTION

# CHAPTER 2 DATASET, LIBRARIES REQUIRED

# CHAPTER 3 METHODOLOGY, CLASSIFIERS

# CHAPTER 4 PREDICTION ANALYSIS

# CHAPTER 5 RESULT AND DISCUSSION

# CHAPTER 6 PROGRAM CODE

# *UNIT 2* IMAGE EMOTION RECOGNITION

# CHAPTER 1 INTRODUCTION

# CHAPTER 2 DATASET, LIBRARIES REQIRED

# CHAPTER 3 METHODOLOGY

# CHAPTER 4 PREDICTIONS

# CHAPTER 5 RESULT AND DISCUSSION

# CHAPTER 6 PROGRAM CODE

# *UNIT 3* VIDEO EMOTION RECOGNITION

# CHAPTER 1 INTRODUCTION

# CHAPTER 2 DATASET, LIBRARIES REQUIRED

# CHAPTER 3 METHODS, CLASSIFIERS

# CHAPTER 4 PREDICTIONS

# CHAPTER 5 RESULT AND DISCUSSION

# CHAPTER 6 PROGRAM CODE

# UNIT 4 FINAL CONCLUSION AND USE CASE

# NOTE: -

# Please make a note that in all these chapters I have included

# Some flow diagram , ideation map, work flow, abbreviations, list of

# tables, calculation part etc.

# SCOPE OF THE PROJECT

# My main objective is to detect the emotion of a certain person based on speech, image & video.

# How a machine can determine the emotion of certain person and predict the outcome.

# UNIT 1

# INTRODUCTION

# Speech recognition, or speech-to-text, is the ability of a machine or program to identify words spoken aloud and convert them into readable text. Rudimentary speech recognition software has a limited vocabulary and may only identify words and phrases when spoken clearly.

# Speech emotion recognition means recognizing the emotion for a particular audio file format and predict the nature of it.

The act of attempting to recognize human emotion and affective states from speech. This is capitalizing on the fact that voice often reflects underlying emotion through tone and pitch. This is also the phenomenon that animals like dogs and horses employ to be able to understand human emotion.

SER is tough because emotions are subjective and annotating audio is challenging.

**DATASET**

This portion of the RAVDESS contains 1440 files: 60 trials per actor x 24

actors = 1440. The RAVDESS contains 24 professional actors (12

female, 12 male), vocalizing two lexically-matched statements in a

neutral North American accent.

Each actor provide different kinds of emotions with different speech.

Each expression is produced at two levels of emotional intensity

(normal, strong), with an additional neutral expression.

**LIBRARIES REQUIRED**

Pandas, numpy, librosa, soundfile, cv2, noisereduce, soundfile, os python\_speech\_features, tensorflow, scipy, sklearn, numba, csv matplotlib, pillow, scikit-learn, glob, pickle, requests, tkinter, pyaudio, warnings, struct, wavfile, ffmpeg etc.

**METHODOLOGY**

The speech emotion detection system is implemented as a Machine

Learning (ML) model. The steps of implementation are comparable to any

other ML project, with additional fine-tuning procedures to make the model

function better. The flowchart represents a pictorial overview of the

process. The first step is data collection, which is one of our prime

importance. The model being developed will learn from the data provided

to it and all the decisions and results that a developed model will produce

is guided by the data. The second step, called feature engineering, is a

collection of several machine learning tasks that are executed over the

collected data. These procedures addressthe several data representation

and data quality issues. The third step is often considered the core of an

ML project where an algorithmic based model is developed. This model

uses an ML algorithm to learn about the data and train itself to respond to

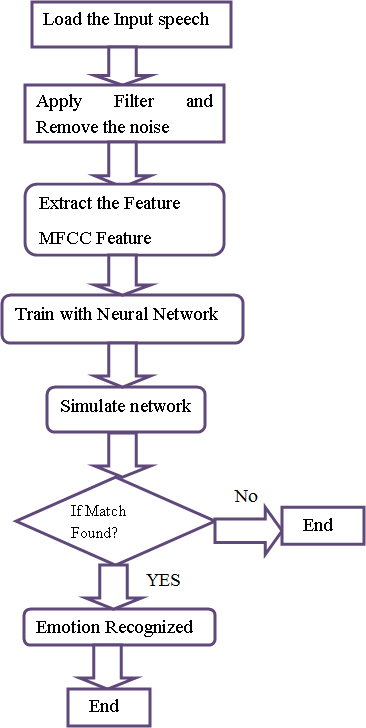
any new data it is exposed to. The final step is to evaluate the functioning

of the built model. Very often, developers repeat the steps of developing

a model and evaluating it to compare the performance of different

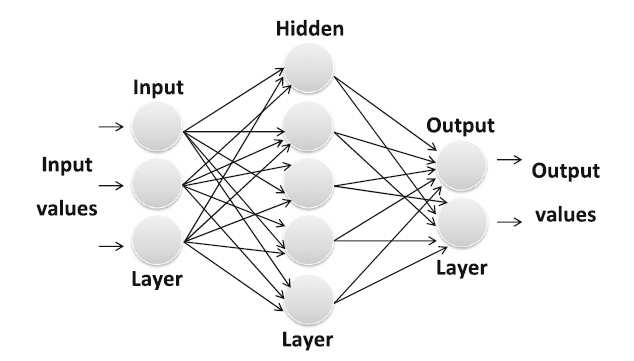
algorithms. Comparison results help to choose the appropriate ML

algorithm most relevant to the problem.



**Multilayer perceptron**

Multilayer perceptron (MLP) classifier is a supervised classification technique that uses backpropagation for training. It is one of the feed-forward artificial neural networks (ANN) classes. It consists of more than one perceptron. It consists of one output layer, one input layer, and in between these input and output layers, there may be an arbitrary number of hidden layers based on the user’s choice. That means it should contain at least three layers input layer, hidden layer, output layer. Expect the input layer; every layer is a neuron that uses a nonlinear activation function. Its nonlinear activation function, multiple layers distinguish this from a single layer feed forward neural network. Since it has nonlinear activation, it can be able to distinguish the data that is not linearly separable. We have performed train\_test\_split and we have given 75% as training data and 25% as testing data for training and testing the model using MLP Classifier, where all these features come under independent data and emotions comes under dependent data.



**MFCC — Mel-Frequency Cepstral Coefficients**

mfcc is used to calculate mfccs of a signal. By printing the shape of

mfccs you get how many mfccs are calculated on how many frames. The

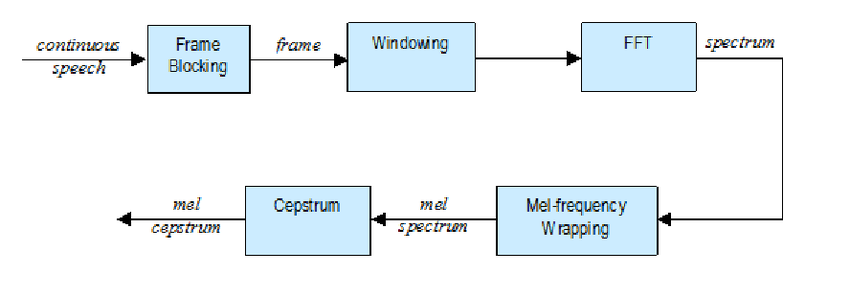
first value represents the number of mfccs calculated and another value

represents a number of frames available.

A small set of features (usually about 10-20) which concisely describe

the overall shape of a spectral envelope. In MIR, it is often used to

describe timbre.





From each segment, we will extract 39 features. Moreover, while breaking

the signal, if we directly chop it off at the edges of the signal, the sudden fall

in amplitude at the edges will produce noise in the high-frequency domain.

So instead of a rectangular window, we will use Hamming/Hanning windows

to chop the signal which won’t produce the noise in the high-frequency

region.

# CHROMA

# Chroma is a Python module for handling colors with ease.

# Manipulating colors can quickly escalate into a tedious and complicated

# task, particularly when you become concerned with color systems

# beyond RGB. Chroma is here to provide a simple API to do the heavy

# lifting.

# Powerful representation for music audio in which the entire spectrum is

# projected onto 12 bins representing the 12 distinct semitones.

# Enhanced chroma and chroma variants — librosa 0.9.1 documentation

# MEL

# One popular audio feature extraction method is the Mel-frequency

# cepstral coefficients (MFCC) which have 39 features. The feature count is

# small enough to force us to learn the information of the audio. 12

# parameters are related to the amplitude of frequencies.

The key objectives are:

* Remove vocal fold excitation (F0) — the pitch information.
* Make the extracted features independent.
* Adjust to how humans perceive loudness and frequency of sound.
* Capture the dynamics of phones (the context).

# MFCC Feature extraction for Sound Classification | Kaggle

# 

# CONTRAST

# Each frame of a spectrogram S is divided into sub-bands. For each sub-

# band, the energy contrast is estimated by comparing the mean energy in

# the top quantile (peak energy) to that of the bottom quantile (valley

# energy). High contrast values generally correspond to clear, narrow-

# band signals, while low contrast values correspond to broad-band noise.

# 

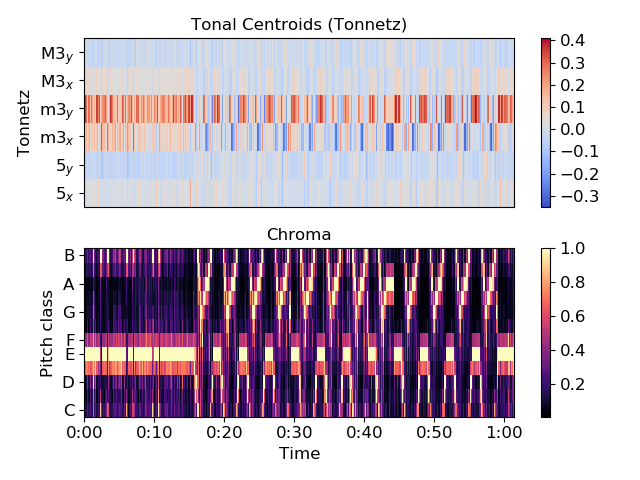
# TONNETZ

Computes the tonal centroid features (tonnetz)

This representation uses the method of [1](https://librosa.org/doc/main/generated/librosa.feature.tonnetz.html#id2) to project chroma features onto

a 6-dimensional basis representing the perfect fifth, minor third, and major

third each as two-dimensional coordinates.

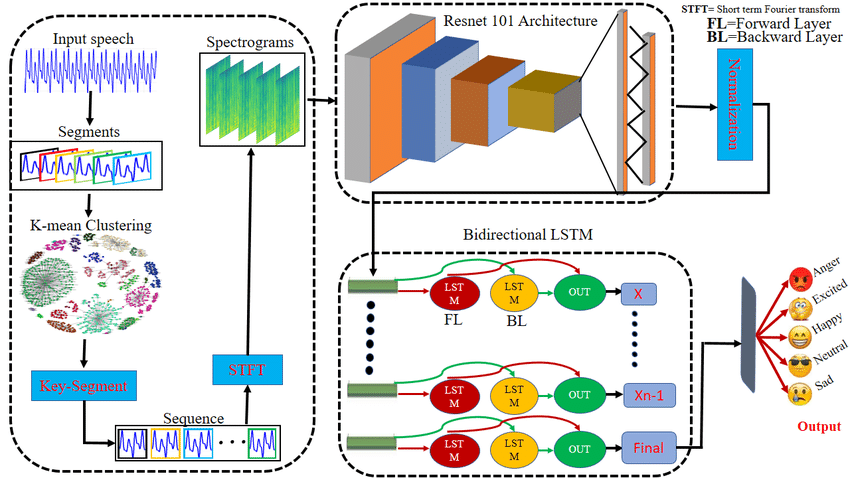


**LSTM**

Long Short-Term Memory (LSTM) networks are a type of recurrent neural network capable of learning order dependence in sequence prediction problems.

This is a behavior required in complex problem domains like machine translation, speech recognition, and more.

LSTMs are a complex area of deep learning. It can be hard to get your hands around what LSTMs are, and how terms like bidirectional and sequence-to-sequence relate to the field.



**FLATTEN**

In Python, for some cases, we need a one-dimensional array rather than a 2-D or multi-dimensional array. For this purpose, the numpy module provides a function called numpy. Ndarray. Flatten(), which returns a copy of the array in one dimensional rather than in 2-D or a multi-dimensional array.

# NumPy: ndarray.flatten()function - w3resource

# FEATURE EXTRACTION

# 

# For feature extraction we use chroma, mel, mfcc.

# normalization - How to normalize audio data for feature extraction - Signal Processing Stack Exchange

# 

# CLEAN SPEECH AND MASKING

# The clean speech data were also suppressed by generated noise signal.

# Gaussian white noise and sinusoid noise were both added to the speech

# database at several signal-to-noise ratio (SNR) which is determined in (4).

# Gaussian white noise and sinusoid noise appear frequently in both actual

# and research environments.

# 

# where xi is a sample from speech signal and yi from noise. Due to the

# variations of speech signals’ energy in different emotions, average SNR

# was measured among an individual’s utterances in all emotions. The

# SNRs of tested noisy speech were 21Db, 18Db, 15Db, 11Db, 7Db,

# approximately. Noisy speech with lower SNR wasn’t included, due to the

# difficulty of pitch extraction from them.

# 

# After cleaning the dataset save the audio file in a folder for evaluating

# the model and predict result.

# TRAINING AND TESTING

After defining the feature extraction function, we need to load the dataset path and pass them to the extract feature function. Once the feature is

# extracted split the dependent and independent value. Take random

# state as 9.

# Train and test the shape of code in the form of arrays as a output.

# Assume X and Y as training and testing the data and classify the data by

# Predicting.

# MODEL CLASSIFIER

# Apply the mlp multi layer perceptron to the dataset. Model selection

# depends on the data set by acknowledging that it needs complex model

# or simple model to predict. MLP is always gives an appropriate output for

# speech kind of problems because the neural networks perform very well

# at learning phoneme probability from highly parallel audio input.

# 

# PREDICTION ANALYSIS

# ACCURACY SCORE

# Accuracy classification score. In multilabel classification, this function

# computes subset accuracy: the set of labels predicted for a sample must

# exactly match the corresponding set of labels in y\_true.

# Accuracy represents the number of correctly classified data instances

# over the total number of data instances.

# Here it is calculated after exploratory analysis.

# F1 SCORE

# F1 Score. The F1 Score is the 2\*((precision\*recall)/(precision+recall)). It

# is also called the F Score or the F Measure. Put another way, the F1 score

# conveys the balance between the precision and the recall.

# F1 score is defined as the harmonic mean between precision and recall.

# PRECISION

# Precision is one indicator of a machine learning model’s performance

# – the quality of a positive prediction made by the model. Precision refers

# to the number of true positives divided by the total number of positive predictions (i.e., the number of true positives plus the number of false

# positives).

# 

# RESULT AND DISCUSSION

# Here we got 82.1% accuracy of Y\_true=Y\_test.

# 

# Now save the model in a pkl file using pickle and glob for any further

# new data to test and predict the emotion.

# Taking live audio as input:

# Record the audio of certain person to predict emotion with the help of in

# built microphone or through an external device and save it in afile

# Apply Mlp Classifier to the loaded live file and by using the extract

# feature to it. Not to forget load predictions from pkl file and predict the

# emotion for that certain data.

# 

# CONCLUSION

# Emotion detection based on speech is successful and emotions are predicted.

# Predicted emotions are

# Angry

# Fear

# Happy

# Sad

# Calm/Neutral

# Surprised.

# Program code:

# Since the program is very large to paste, I provide GitHub link for it.

# https://github.com/Srikanth-1234/Emotion-Recognition

# UNIT 2

# DATASET

# There is no specified data set for image detection I used some images

# from google and predicted those.

# We may take any image that should be in the form of jpg, img, jpeg.

# Make sure not to include animals or multi facial detection.

# LIBRARIES REQUIRED

# Matplotlib, FFMPEG, deepface, open cv

# NOTE: -Make sure to install deepface library as we use mostly regarding

# this.

# METHODOLOGY

# Recheck that you can import al the required libraries.

# Use open cv to for plotting the live emotion

# DEEPFACE

# Here I used a simple program for the image emotion detection.

# First take a image in the form of jpg, img or jpeg format and make sure it

# contains human photo in it.

# Now by using deep face library we get the external face recognition an

# dits emotion. Deep Face is a deep learning facial recognition system created by a research group at Facebook.

# Here we can also predict the age of that person and which country belongs

# to etc.

If you run face recognition with Deep Face, you get access to a set of features:

**Face Verification**: The task of face verification refers to comparing a face with another to verify if it is a match or not. Hence, face verification is commonly used to compare a candidate’s face to another. This can be used to confirm that a physical face matches the one in an ID document.

**Face Recognition**: The task refers to finding a face in an image database. Performing face recognition requires running face verification many times.

**Facial Attribute Analysis:** The task of facial attribute analysis refers to describing the visual properties of face images. Accordingly, facial attributes analysis is used to extract attributes such as age, gender classification, emotion analysis, or race/ethnicity prediction.

**Real-Time Face Analysis:** This feature includes testing face recognition and facial attribute analysis with the real-time video feed of your webcam.

##### **Import the library:**

From deep face import DeepFace

**PREDICTIONS**

For this image detection the predictions are:

# 

# 

# So the dominant prediction is angry.

# RESULT

# So here by loading the image we can predict the emotion.

# This code works only for image processing emotion so I made this in sa

# open cv to predict the live emotion but as a video which will be

# implemented in upcoming unit

# I also made some special gui screen regarding this.

# CONCLUSION

# I conclude that this image emotion recognition model is a complete simple one preferred to speech emotion recognition.

# It may be very useful if we have the same prediction model in SER.

# PROGRAM CODE

**import** cv2

**import** matplotlib.pyplot **as** plt

**import** os

**import** warnings

warnings**.**filterwarnings('ignore')

img **=** cv2**.**imread('angry2.jpg')

print(img)

plt**.**imshow(img)

plt**.**imshow(cv2**.**cvtColor(img,cv2**.**COLOR\_BGR2RGB))

**from** deepface **import** DeepFace

predictions **=** DeepFace**.**analyze(img)

predictions

result **=** DeepFace**.**analyze(img, actions **=** ['emotion'])

print("Predicted emotion is:",result['dominant\_emotion'])

**import** cv2

**from** deepface **import** DeepFace

faceCascade **=** cv2**.**CascadeClassifier(cv2**.**data**.**haarcascades **+** 'haarcascade\_frontalface\_default.xml')

gray **=** cv2**.**cvtColor(img, cv2**.**COLOR\_BGR2GRAY)

*#print(faceCascade.empty())*

faces **=** faceCascade**.**detectMultiScale(gray,1.1,4)

*# Draw a rectangle around the faces*

**for** (x, y, w, h) **in** faces:

cv2**.**rectangle(img, (x, y), (x**+**w, y**+**h), (0, 255, 0), 2)

plt**.**imshow(cv2**.**cvtColor(img,cv2**.**COLOR\_BGR2RGB))

font **=**cv2**.**FONT\_HERSHEY\_SIMPLEX

*# Use putText() method for*

*# inserting text on video*

cv2**.**putText(img,

predictions['dominant\_emotion'],

(0, 50),

font, 1,

(0, 0, 255),

2,

cv2**.**LINE\_4);

plt**.**imshow(cv2**.**cvtColor(img,cv2**.**COLOR\_BGR2RGB))

cv2**.**imshow('emotion test',img)

cv2**.**waitKey(5000)*#CV2 opens for 5 seconds*

print("Successfully captured emotion with image")

cv2**.**destroyAllWindows()

# UNIT 3

# INTRODUCTION

# Video emotion recognition based on facial expression.

# Several data sets have been established to build emotion recognition

# models and evaluate performances. Same methods can receive

# dramatically different results on different data sets. This is due to the

# variance in data sets. Facial emotion databases can be divided into two

# categories: lab data sets like CohnKanade CK+ database and wild data

# sets like Acted Facial Expressions in the Wild (AFEW).

# These dataset are certainly old to predict compare to the new ones.

# I have used FER2013 dataset from Kaggle open source system

# Geometric-feature-based methods are methods that extract information

# about facial components and their movements which imitate how humans

# understand facial emotions. One example of geometric-feature-based

# methods is the Facial Action Coding System (FACS). In order to describe

# facial expressions precisely, FACS was developed in which each facial

# expression broke down into several Action Units (AU) each representing

# a facial muscular movement.

# In this project I have used haarcascade models which is the best model

# for facial recognition and mainly this is used to predict emotions.

# In haarcascades there are lot of xml files; using those we can predict

# emotions, scan eyes, where abouts of a location in a face like ears nose

# hair, mouth etc.

# LIBRARIES REQUIRED

# Tensorflow,keras,cv2,warnings,model\_from\_json

# METHODOLOGY

# 

# Using data generator we validate the data into training the model.

# Here by importing ImageDataGenerator we can bring down the images in a model and train it by using train\_generator, validate it by using validation\_generator.

# CNN

# Convolutional neural network (CNN) is the most popular way of analyzing

# images. CNN is different from a multi-layer perceptron (MLP) as they have

# hidden layers, called convolutional layers. The proposed method is based on a two-level CNN framework. The first level recommended is the

# background removal and is used to extract emotions from an image.

# Here, the conventional CNN network module is used to extract primary

# expressional vector (EV). The expressional vector (EV) is generated by

# tracking down relevant facial points of importance. EV is directly related

# to changes in expression. The EV is obtained using a basic perceptron

# unit applied on a background-removed face image.

# figure 1

# To maintain uniformity irrespective of the type of input image, Hough

# transform (Fig. [3](https://link.springer.com/article/10.1007/s42452-020-2234-1#Fig3)c) was always used as the second input feature to

# background removal CNN. The formula used for Hough transform is

# 

# Image data augmentation is used to improve the performance and ability

# of the model to generalize. It’s always a good practice to apply some

# data augmentation before passing it to the model, which can be done

# using ImageDataGenetrator provided by Keras.

# Designing the CNN model for emotion detection using functional API.

# We are creating blocks using Conv2D layer, Batch-Normalization, Max-

# Pooling2D, Dropout, Flatten, and then stacking them together and at the

# end-use Dense Layer for output.

# PREDICTIONS

# Train the model. FER\_model takes input size and returns model for training. This

# make take a lot of time depending upon your gpu meter.

steps\_per\_epoch = TotalTrainingSamples / TrainingBatchSize

validation\_steps = TotalvalidationSamples / ValidationBatchSize

Training may take minutes ;don’t get surprised if it takes hours also to get the output

HERE I TOOK NEARLY 2HR 16 MIN.

If u want to reduce the time tur on gpu meter / use the best performance

laptops to predict the result smoothly and rapidly.

# One of the best thing is to change epochs to 50 or rather decrease to 10

# you can get within 10 -30 minutes.

# Evaluate the model summary.

# Plot the val accuracy vs acc

# 

# Plot the val loss vs loss

# 

# RESULT

# Here at last we get the trained accuracy with respect to accuracy and save the model in a json file so that there can be any availability for read and execute a new dataset.

# Save the trained weights in an h5 file for newest data prediction.

# PROGRAM CODE

# import cv2

# import warnings

# warnings.filterwarnings('ignore')

# from keras.models import Sequential

# from keras.layers import Conv2D, MaxPooling2D, Dense, Dropout, Flatten

# from tensorflow.keras.optimizers import Adam

# from keras.preprocessing.image import ImageDataGenerator

# # Initialize image data generator with rescaling

# train\_data\_gen = ImageDataGenerator(rescale=1./255)

# validation\_data\_gen = ImageDataGenerator(rescale=1./255)

# # Preprocess all test images

# train\_generator = train\_data\_gen.flow\_from\_directory(

# 'train',

# target\_size=(48, 48),

# batch\_size=64,

# color\_mode="grayscale",

# class\_mode='categorical')

# # Preprocess all train images

# validation\_generator = validation\_data\_gen.flow\_from\_directory(

# 'test',

# target\_size=(48, 48),

# batch\_size=64,

# color\_mode="grayscale",

# class\_mode='categorical')

# # create model structure

# emotion\_model = Sequential()

# emotion\_model.add(Conv2D(32, kernel\_size=(3, 3), activation='relu', input\_shape=(48, 48, 1)))

# emotion\_model.add(Conv2D(64, kernel\_size=(3, 3), activation='relu'))

# emotion\_model.add(MaxPooling2D(pool\_size=(2, 2)))

# emotion\_model.add(Dropout(0.25))

# emotion\_model.add(Conv2D(128, kernel\_size=(3, 3), activation='relu'))

# emotion\_model.add(MaxPooling2D(pool\_size=(2, 2)))

# emotion\_model.add(Conv2D(128, kernel\_size=(3, 3), activation='relu'))

# emotion\_model.add(MaxPooling2D(pool\_size=(2, 2)))

# emotion\_model.add(Dropout(0.25))

# emotion\_model.add(Flatten())

# emotion\_model.add(Dense(1024, activation='relu'))

# emotion\_model.add(Dropout(0.5))

# emotion\_model.add(Dense(7, activation='softmax'))

# cv2.ocl.setUseOpenCL(False)

# emotion\_model.compile(loss='categorical\_crossentropy', optimizer=Adam(lr=0.0001, decay=1e-6), metrics=['accuracy'])

# # Train the neural network/model

# emotion\_model\_info = emotion\_model.fit\_generator(

# train\_generator,

# steps\_per\_epoch=28709 // 64,

# epochs=100,

# validation\_data=validation\_generator,

# validation\_steps=7178 // 64)

# # save model structure in jason file

# model\_json = emotion\_model.to\_json()

# with open("emotion\_model.json", "w") as json\_file:

# 

# json\_file.write(model\_json)

# # save trained model weight in .h5 file

# emotion\_model.save\_weights('emotion\_model.h5')

# emotion\_model.summary()

# emotion = emotion\_model\_info.history

# emotion.keys()

# import matplotlib.pyplot as plt

# plt.plot(emotion['accuracy'])

# plt.plot(emotion['val\_accuracy'] , c = "red")

# plt.title("acc vs v\_acc")

# plt.show()

# plt.plot(emotion['loss'])

# plt.plot(emotion['val\_loss'] , c = "red")

# plt.title("loss vs v\_loss")

# plt.show()

# UNIT 4

# FINAL CONCLUSION AND USE CASE

# CONCLUSION

# I conclude that through this anybody can find their emotion either in the

# form of speech /image/video.

# We can predict emotion for a certain person even if he is blind, deaf,

# can’t speak etc.

# This model can predict all kinds of emotion that are majorly required

# USE CASE

# In police sectors and in military areas; One of the common examples about this is, In “SPYDER movie film” 🡪Mahesh babu actor (Tollywood hero). Telugu Industry. Ewe~~~

# He predicts the emotion by using these types of algorithms in real life. In that movie he recognizes the victim’s emotion as fear.

# The above one is example of (SER)

# Image detection we use this model mainly in uncontrollable, restricted so that they can watch over the enemy or trespassers so that to eradicate their moves.

# Another example is in some shopping centers we may not noticed but they place emotion detection whether the customer is satisfied or happy with needs.

# Another great use case is capturing thieves we can capture not only their face identity but also the emotion they possess at that time, how he traps a victim.

# The above examples comes under (FER) i.e. image emotion detection(IER) & video emotion recognition(VER).

# GITHUBLINK REPO

# <https://github.com/Srikanth-1234/Emotion-Recognition>

# All the source code is available in this link. This is a reference.

# THANKYOU