Annexure

Department of Data Science and Business Systems
SRM Institute of Science & Technology
Own Work* Declaration Form

This sheet must be filled in (each box ticked to show that the condition has been met). It must be signed and dated along with your student registration number and included with all assignments you submit – work will not be marked unless this is done.

To be completed by the student for all assessments

Degree/ Course : **B.Tech/Computer Science Engineering with Specialization in BD**

Student Name : **Vadlamudi Sai Lokesh, Kotha Sai Narasimha Rao**

Registration Number : **RA1811027010069, RA1811027010070**

Title of Work : **Facial Expression Analysis with Smart Song Player**

I / We hereby certify that this assessment compiles with the University's Rules and Regulations relating to Academic misconduct and plagiarism**, as listed in the University Website, Regulations, and the Education Committee guidelines.

I / We confirm that all the work contained in this assessment is my / our own except where indicated, and that I / We have met the following conditions:

- Clearly references / listed all sources as appropriate
- Referenced and put in inverted commas all quoted text (from books, web, etc)
- Given the sources of all pictures, data etc. that are not my own
- Not made any use of the report(s) or essay(s) of any other student(s) either past or present
- Acknowledged in appropriate places any help that I have received from others (e.g. fellow students, technicians, statisticians, external sources)
- Compiled with any other plagiarism criteria specified in the Course handbook / University website

I understand that any false claim for this work will be penalized in accordance with the University policies and regulations.

DECLARATION:

I am aware of and understand the University's policy on Academic misconduct and plagiarism and I certify that this assessment is my / our own work, except where indicated by referring, and that I have followed the good academic practices noted above.

If you are working in a group, please write your registration numbers and sign with the date for every student in your group.

Vadlamudi Sai Lokesh

Kotha Sai Narasimha Rao

ABSTRACT

Human face recognition is one of the important tasks in social communications. face expressions directly reflect human emotion. Face expressions are helpful to identify the real feeling of the person during a certain situation. Face expressions are key for indirect communication. The proposed system is focused on creating a robust Architecture using VGG Net. The system considers a face expression dataset (FER 2013) for analyzing the present model. The proposed architecture extracts and learns various unique expressions from the facial images available in FER 2013. The feature mapped values are further used to train and model the VGG net-enabled deep convolution networks. The proposed model also considers a music recommendation framework. appropriate music recommendations are done to the various expressions made by the person. The system achieved a higher accuracy of 92% and compared with existing state of art approaches to performance metrics. Keywords— Face expressions, Machine learning, Face recognition, Viola Jones algorithm, Music recommendation

ACKNOWLEDGEMENTS

We express our humble gratitude to **Dr C. Muthamizhchelvan**, Vice-Chancellor, SRM Institute of Science and Technology, for the facilities extended for the project work and his continued support.

We extend our sincere thanks to Dean-CET, SRM Institute of Science and Technology, **Dr T.V.Gopal**, for his invaluable support.

We wish to thank **Dr Revathi Venkataraman**, Professor & Chairperson, School of Computing, SRM Institute of Science and Technology, for her support throughout the project work.

We are incredibly grateful to our Head of the Department**, Dr M. Lakshmi** Professor, Department of Data Science and Business Systems, SRM Institute of Science and Technology, for her suggestions and encouragement at all the stages of the project work.

We want to convey our thanks to our program coordinators **Dr.Murgan,** professor, and Panel Head, **Dr.Rajasekar P,** Assistant professor, Department of Data Science and Business Systems, SRM Institute of Science and Technology, for their inputs during the project reviews and support.

We register our immeasurable thanks to our Faculty Advisor**, Dr.L.ANAND,** Assistant professor, Department of Networking and Communications, SRM Institute of Science and Technology, for leading and helping us to complete our course.

Our inexpressible respect and thanks to my guide, **Dr.L.ANAND,** Assistant professor SRM Institute of Science and Technology, for providing me with an opportunity to pursue my project under his/her/their mentorship. He/She/They provided me with the freedom and support to explore the research topics of my interest. Her/His/Their passion for solving problems and making a difference in the world has always been inspiring.

We sincerely thank the Data Science and Business Systems staff and students, SRM Institute of Science and Technology, for their help during our project. Finally, we would like to thank parents, family members, and friends for their unconditional love, constant support, and encouragement.

Vadlamudi Sai Lokesh

Kotha Sai Narasimha Rao

TABLE OF CONTENTS

CHAPTER NO PAGE NO.

LIST OF FIGURES

ABBREVIATIONS

CNN	Convolutional Neural Network
VGG	Visual Geometry Group
LBP	Local Binary Patterns
RESNET	Residual Neural Network
HOG	Histogram of Oriented Gradients

CHAPTER 1
INTRODUCTION

Face recognition models are developed in recent years rapidly since most of the deep learning applications engaged in today's world are formulated with face recognition frameworks. Despite many face recognition datasets available publicly, Live face recognition also created more interest in research. In a similar context, obtaining facial landmark detection is one of the methods to achieve a face recognition system[1]. Face reconstruction is an art of image processing technology and deep learning technology. Creating accurate geometry is mandatory in facial reconstruction. To formulate the face expressions accurately, face landmarks mapping is an impacted task. Identification of face geometry is important to conclude the facial expressions[2]. For the synthesis of exact emotions of the face, significant models are developed in the existing systems. A self-organized automated mapping model is used in facial expression identification. For realistic emotion extraction such as happy, sad, anger, disgust, confusion, hurt, etc, a combination of facial expressions is needed. Some researchers focused on creating basic emotions to be sequentially considered to form the real expression depicted in the face. Expressions reflect unique landmark changes in the face, such as Eyebrows changes, smiles, closed eyes, etc. [3]. In today's technological world, the interaction between Human-computers has increased. facial expressions act as the means of an intelligent human-computer interface. A face expression recognition system is an additional tool for making unique and accurate recommendations to the users based on demand. In certain cases, facial expressions are also utilized in the medical domain for acquiring the result of antidepressants given to the patients[4]. Face expression can be detected from images captured as well as video streams. attributes are important to model face expressions. pipelined facial expressions are collectively used to find the emotions of the person. Augmented reality systems are developed using human-computer interfacing systems[5]. Face expressions show unique signs like, head poses, and lighting poses[6]

1.1 OBJECTIVES

Facial Expression is the affluent origin of the human being. We understand the other person's thoughts just from seeing their Expressions. Here in this project, we are predicting a total of 6 emotions Happy, Sad, Angry, Surprise, Neutral, Scared, and playing songs based on the predicted emotion

1.2 INNOVATION IDEA OF THE PROJECT

The proposed model is created using VGG 16 architecture. the detection of facial expressions, with appropriate music recommendations, are made based on the detected expressions. The (FER2013) Facial expressions recognition dataset is used here for analysis. The dataset contains publicly collected 37000 face expression images of various classes. After predicting the emotion of the user the model will start playing songs based on the emotion

1.3 PURPOSE OF THE PROJECT

This Proposed system is able to detect the face and it has the ability to identify the emotion through facial expressions. detected emotions can fall into one of these categories angry, sad, fear, surprise, happy, and neutral. This proposed project can be used in real-time. Thus to our knowledge, this model has achieved the highest accuracy than the existing models.

1.4 SCOPE

This System has been trained with optimized deep learning algorithms so it has a huge scope in many fields such as Gaming, Human-computer interaction, the Medical sector, Analytics field.

1. This system is helpful for the companies who need feedback from the people for their advertisements in public places to enhance the customers.

2. This system is helpful to educational institutions like tracking the student's emotions during the class.

3. This model can be used for lie detection amongst criminals during interrogation.

1.5 PROBLEM STATEMENT

Face Detection and facial expression recognition is a complicated task. The main goal of this project is to identify the emotion of the user and classify them into the one of the categories happy,fear,sad,surprise,angry and neutral in the real time.we have collected the data from the kaggle website.The preprocessing steps are done.By using vgg16 architecture it is able to classify the facial expression into the one of these categories angry,sad,happy,surprise,fear.

1.6 EXISTING WORK RELATED TO YOUR PROJECT

Existing systems are confined to a particular expressions.performance of the existing models are reasonably good.These models need some further exploration.

1.7 PROPOSED SYSTEM

The proposed system preprocesses the image dataset, and normalizes the data before making the analysis. The face expression recognition system is developed using convolution neural network architecture. Many existing frameworks have been developed in recent years. In that deep learning convolutional neural network system acts as the foremost method that gives better results. Here VGG-16 architecture is developed. The VGG-16 with modified layers, suitable only for the connected dataset is focused here. The Model is customized to form a clear attribute extraction from various face feature learnings. A customized music recommendation system also developed here based on the detected expressions such as happy, sad, disgust, excited etc.

CHAPTER 2

LITERATURE STUDY

2.1)Facial Expression Recognition based on Arousal-valence Emotion model and Deep Learning Method

The author presented an additive feature extraction based facial expression detection system. Adaptive weighted estimation basedCognitive model of feature in balance is discussed here.Percentage system utilised FCR + data set and achieved the accuracy of 88%using transfer learning approaches and compared with existing state of art approaches. If the number of images used for the training process increases then the performance of the prediction system increases. The presented system act as a basic model for deriving a deep learning based face expression detection system with top.

2.2)Multi-Modal Emotion Recognition from Speech and Facial Expression based on Deep Learning

The percentage system utilized the FER2013 dataset of facial expression detection and implemented a convolutional neural network architecture.The normalized face feature fusions are extracted.The present system achieved the accuracy of 88 % and utilised for image based computing applications. Spontaneous expression detection Framework is modelled here to stop the data set consists of various emotions of individuals comprised together is useful to detect the real emotions of the individuals pitstopDatabase consists of 428 Facial expressions collectively gathered from spontaneous detection of camera.Through the process of video clipping based facial expression detection and self assessment method is used here to make the validation process.

2.3)Deep Learning Based Facial Recognition System Mehmet

The author presented a system where the correspondence between two different scenarios of face expressions is detected.The presented approach tests the non professional users by making different facial expressions and the validation is processed with the existing frameworks. Using geometric and conceptual frameworks the presented model is validated

2.4)Exploiting multi-CNN features in CNN-RNN based Dimensional Emotion Recognition on the OMG in the wild Dataset

The author presented a neural network architecture enabled facial expression detection system using a massive data set collection of facial images.Impact of environment age, physical factors, the images of various individuals are collected for the training purpose.The test data are collected in real time camera for the process to with the deep neural network architecture to detect the exact emotion present in the face to stop. H. Yang, et al., (2021) Using the approach of generalized adversal network(GAN), Facial changes Obtained in a progressive way or detected by the proposed algorithm.The proposed system focused mainly on specific face expression changes framework, using MORPH and CACD databases, the static work achieved the accuracy of 99% accuracy. The quantitative measures are compared with the existing state-of-the-art approaches.

CHAPTER 3

HARDWARE AND SOFTWARE REQUIREMENTS

3.1 HARDWARE REQUIREMENTS

1. The processor is intel processor 13 or 15
2. RAM should be a minimum of 4 GB
3. Hard Disk of minimum 160 GB
4. GRAPHIC CARD for faster execution of the model

3.2 SOFTWARE REQUIREMENTS

1. Operating system- Windows 10
2. server-side script- python, anaconda
3. Tool- Google colab, visual studio, Jupyter

CHAPTER 4

INITIAL ANALYSIS AND MODULES

1. IMAGE ANALYSIS:

   1.1 Pre-processing

   1.2 Segmentation
2. FACE DETECTION
3. EMOTION RECOGNITION
4. PLAYING SONGS

4.1 IMAGE ANALYSIS:

DATASET:

FER2013

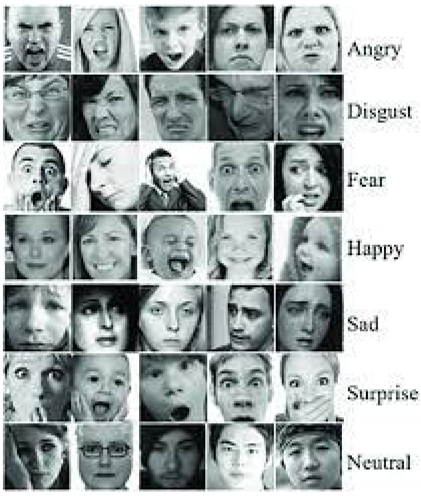


Fig 4.1 Dataset Images

There are total of 37,000 images in this Dataset. In this Dataset we excluded disgust because it contains only 100 images due to that less number model may tend to give less accuracy and false predictions

PRE-PROCESSING:

pre Processing of image is the actual mere and ideal step that needs to be done significantly to the input data set taken. In pre processing the image taken as input from the data set is amplified by magnifying and removing the original (present) distortions, noise also improves its features to promote for next stage of processing. Here details of input image are preserved, at the same time redundancy which cause noise is removed. In overall this stage reads the input image and then re scales it through the processes of filtration and normalization to remove noise. Then the rotated and uniform sized image it is further sent to segmentation stage. The image obtained from the pre processing stage is taken an input to the Segmentation stage

SEGMENTATION:

Segmentation partitions the digital input image taken as input from pre processing stage separates image into multiple fragments. The input image is divided into compatible, analogous, coherent regions corresponding to different entities in the input image on the framework of disposition, boundary, and potency. The image obtained from the segmentation stage is takenan input to the feature extraction stage.

4.2 FACE DETECTION:

For Detecting the face we used four different approaches and made a comparitive analysis we used

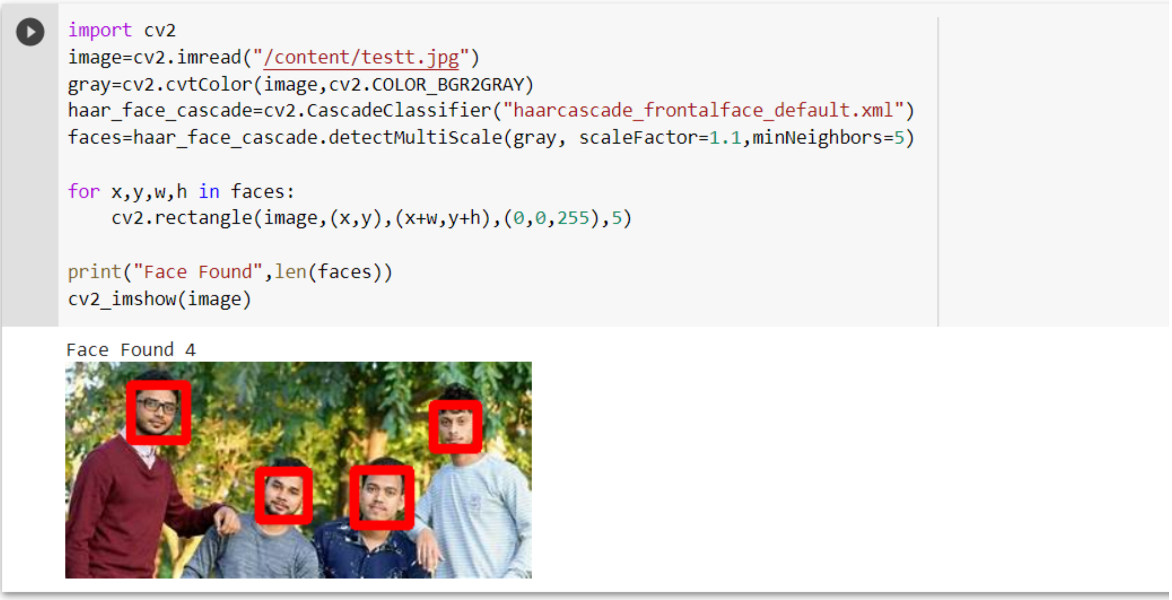
1.Haarcascade

2.lbpcasacde

3.Dlib with HOG Detector

4.DNN

After examining these four we finalized Haarcascade is best for our model as it detect the frontal face well even in bad lighting conditions

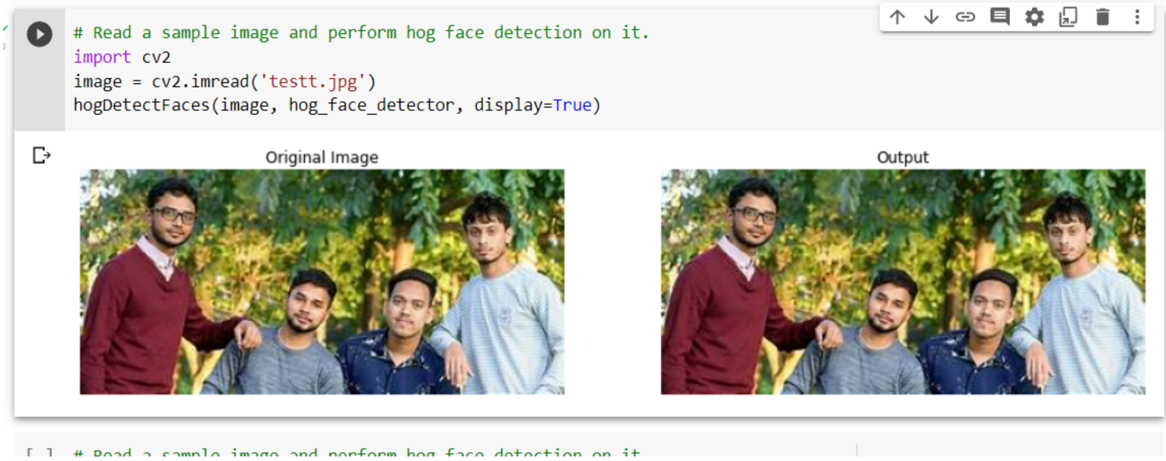


```
import cv2
image=cv2.imread("/content/testt.jpg")
gray=cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)
haar_face_cascade=cv2.CascadeClassifier("haarcascade_frontalface_default.xml")
faces=haar_face_cascade.detectMultiScale(gray, scaleFactor=1.1,minNeighbors=5)

for x,y,w,h in faces:
    cv2.rectangle(image,(x,y),(x+w,y+h),(0,0,255),5)

print("Face Found",len(faces))
cv2_imshow(image)
```

```
Face Found 4
```

fig 4.2:Face detection using Haarcascade

From fig 4.2 we can see that all faces are detected

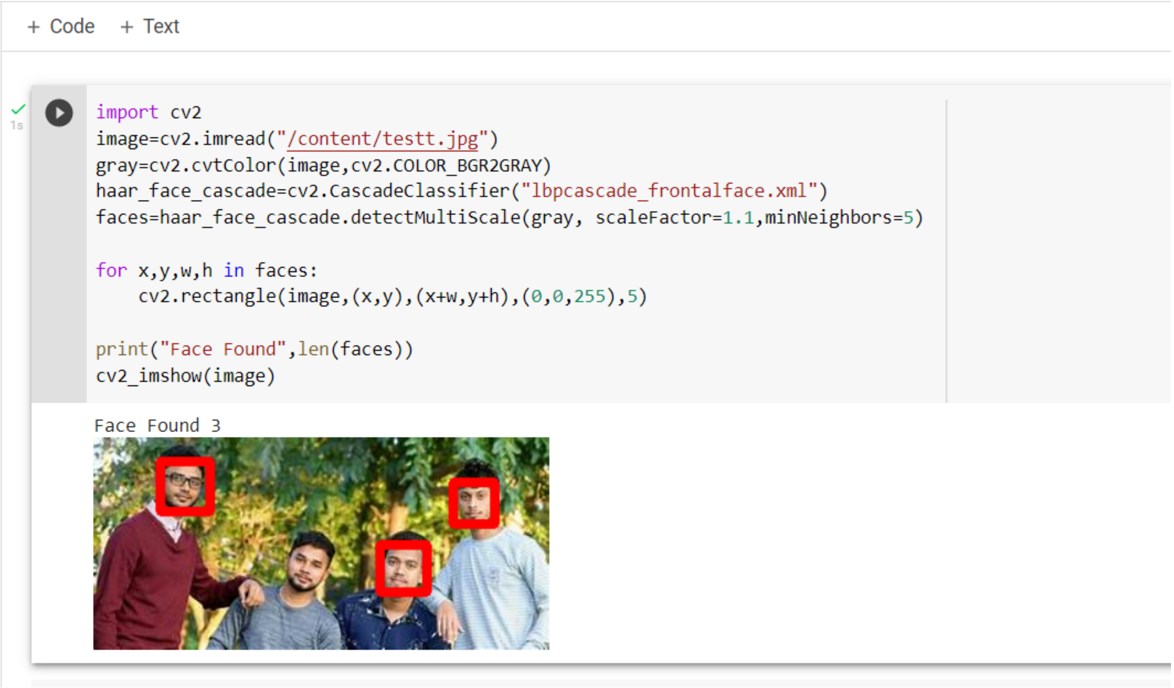


```
# Read a sample image and perform hog face detection on it.
import cv2
image = cv2.imread('testt.jpg')
hogDetectFaces(image, hog_face_detector, display=True)
```

```
[ ] # Read a sample image and perform hog face detection on it
```

fig 4.3:FaceDetection using Dlib with HOG\

In Fig 4.3 none of the faces are detected

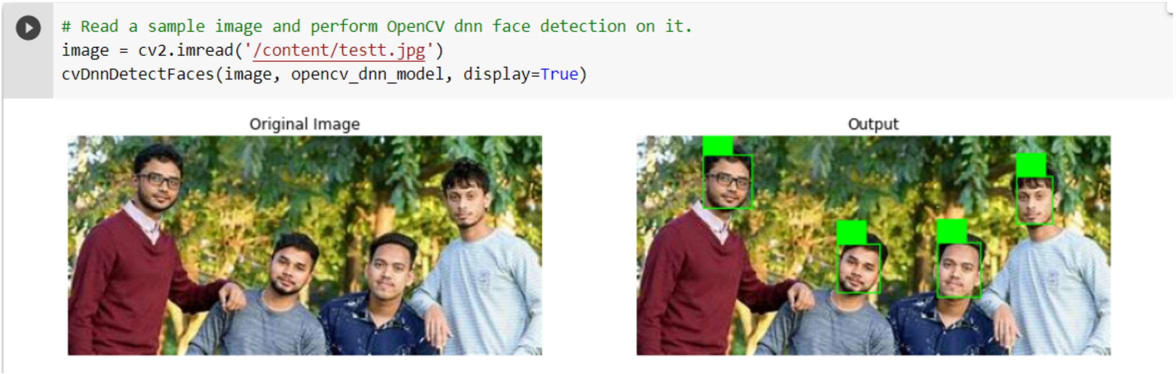


```
import cv2
image=cv2.imread("/content/testt.jpg")
gray=cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)
haar_face_cascade=cv2.CascadeClassifier("lbpcascade_frontalface.xml")
faces=haar_face_cascade.detectMultiScale(gray, scaleFactor=1.1,minNeighbors=5)

for x,y,w,h in faces:
    cv2.rectangle(image,(x,y),(x+w,y+h),(0,0,255),5)

print("Face Found",len(faces))
cv2_imshow(image)
```

```
Face Found 3
```

Fig 4.4:Face Detection using Lbpcascade

In fig 4.4 only 3 out of 4 faces are detected



```
# Read a sample image and perform OpenCV dnn face detection on it.
image = cv2.imread('/content/testt.jpg')
cvDnnDetectFaces(image, opencv_dnn_model, display=True)
```

fig 4.5: Face Detection using DNN

In fig 4.5 all four faces are detected

4.3 EMOTION RECOGNITION:

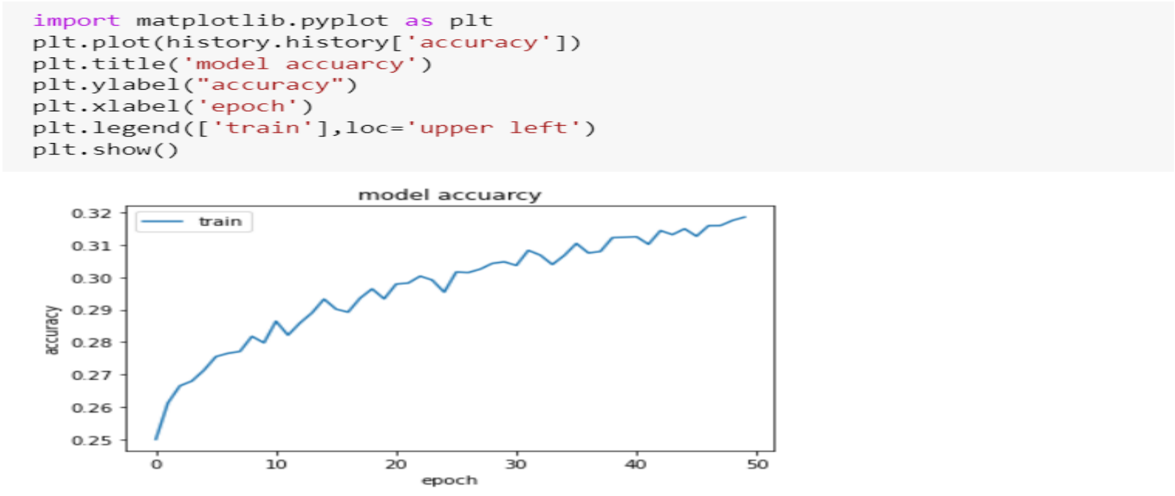
For emotion Recognition, we used four different algorithms and made a comparative analysis

the four algorithms are

1. RESNET50
2. EMOTIONNETB0
3. BASE MODEL
4. VGG16

RESET50:

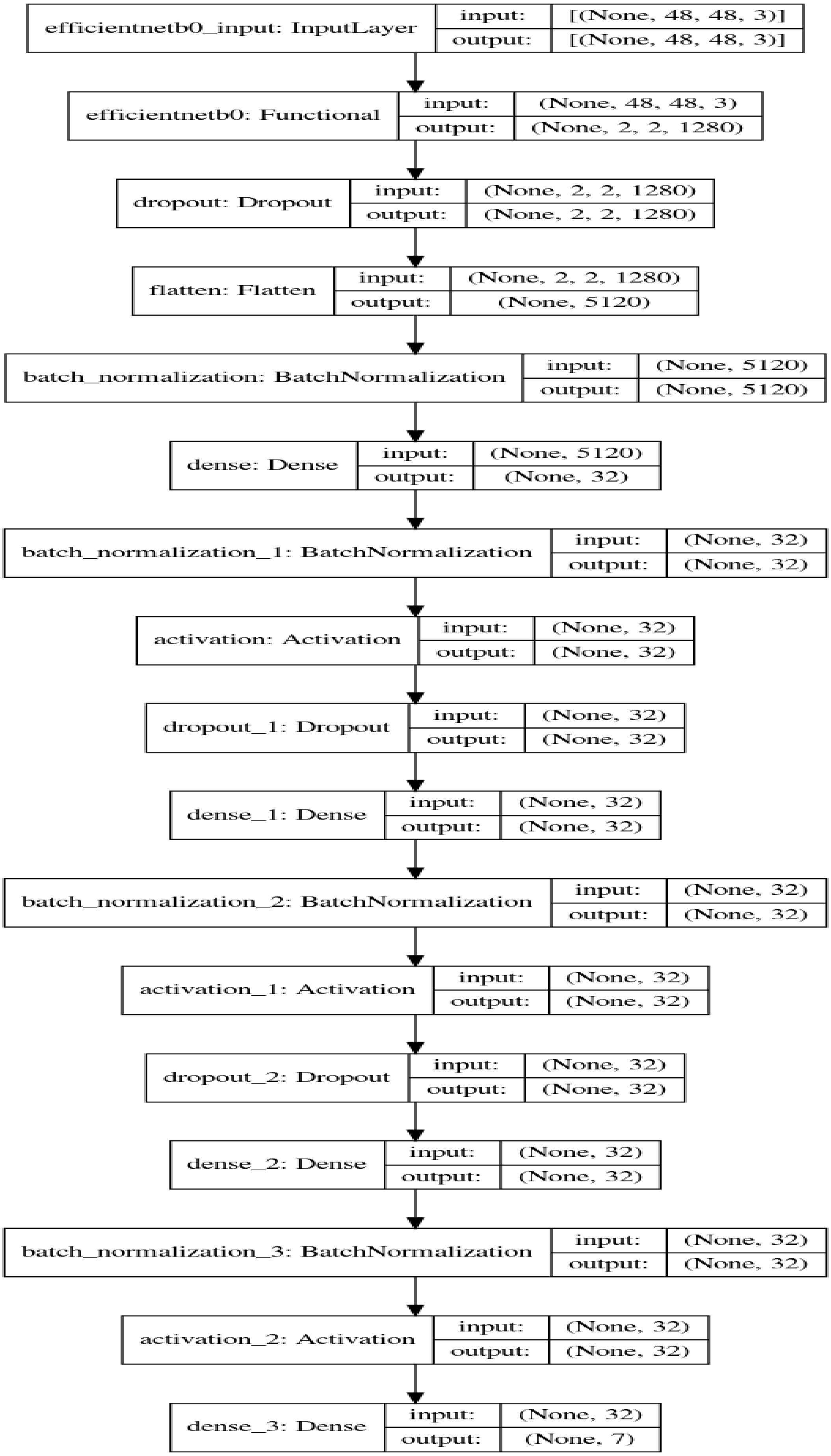
RESNET50 consists of 50 layers with 48 convolution layers, 1 max pool layer and 1 avg pool layer. The best feature of this architecture is that it contains identify function which ensures that the accuracy is always increasing or equal to that of previous layers while passing through deep layers. Here while building this model first we need to mount our google drive to collab such that access of data will be faster compared to the direct uploading of dataset in google collab. This model has been trained for 50 epochs and yielded the best accuracy of 35 %



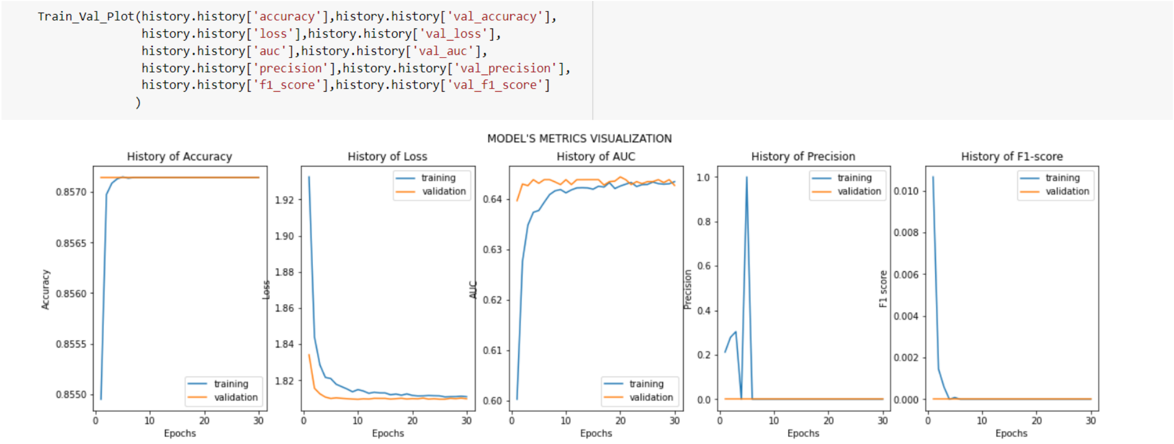
```
import matplotlib.pyplot as plt
plt.plot(history.history['accuracy'])
plt.title('model accuarcy')
plt.ylabel("accuracy")
plt.xlabel('epoch')
plt.legend(['train'],loc='upper left')
plt.show()
```

fig 4.6 : accuracy graph for RESNET50 model

2)EFFICIENTNETB0:



It contains 237 layers. It has compound scaling which is used to scale all the dimensions of the image uniformly. The benefit of compound scaling is that instead of directly preprocessing the large image which may result in feature loss, it will help preprocess the image uniformly. It has been trained for 50 epochs and yielded the best accuracy of 87 %

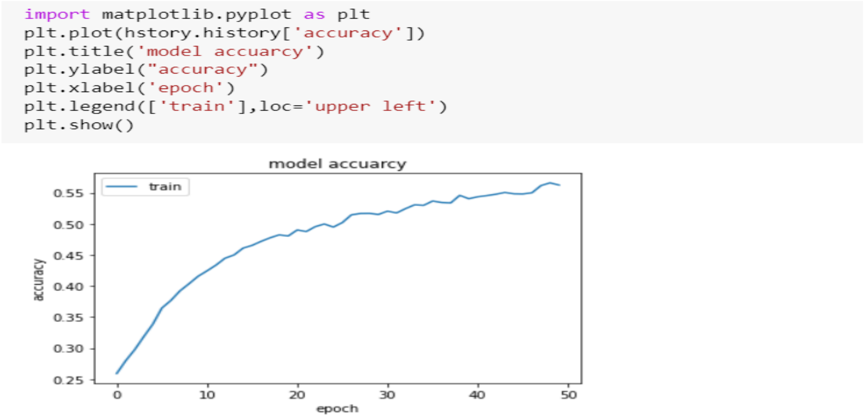


```
Train_Val_Plot(history.history['accuracy'],history.history['val_accuracy'],
               history.history['loss'],history.history['val_loss'],
               history.history['auc'],history.history['val_auc'],
               history.history['precision'],history.history['val_precision'],
               history.history['f1_score'],history.history['val_f1_score']
              )
```

fig 4.7 : accuracy graph for EfficientnetB0 model

BASE MODEL:

It is a sequential model in which we manually build the layers



```
import matplotlib.pyplot as plt
plt.plot(hstory.history['accuracy'])
plt.title('model accuarcy')
plt.ylabel("accuracy")
plt.xlabel('epoch')
plt.legend(['train'],loc='upper left')
plt.show()
```

fig 4.8: accuracy graph for BaseModel

VGG16:

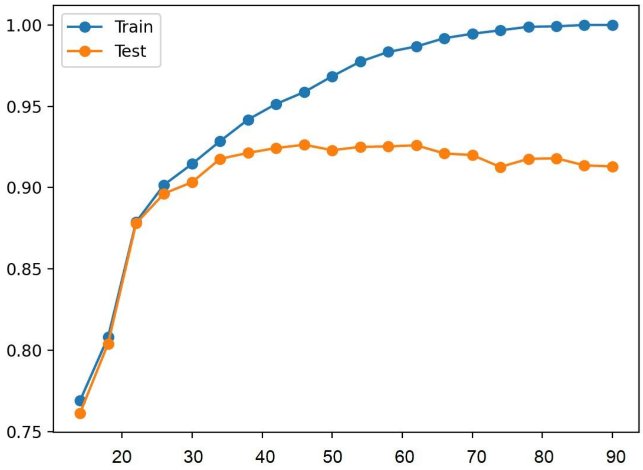


fig 4.9: accuracy graph of VGG16 Model

4.4 PLAYING SONGS:

For song playing “playsound” library is used

Playsound:

The playsound library is python based library which is used to play the songs. It contains only one function which is also called playsound and there is also an optional parameter that is set to True by default. If we set that to false then the module runs asynchronously

playsound(“audio_file_name.mp3”_)

CHAPTER 5

MODEL DESIGN

5.1 System architecture

Many existing frameworks are implemented for facial expressions detections. Accurate localization of facial expressions are mandatory to make the recommendations related to the expressions. Existing works with the issue of false positive rate, that depicts the expression classification that provides false results. Considering the Problem identified, it is highly important to make a robust framework for facial expression detection. The proposed model is developed using Python IDE, with VGG net architecture. The presented system considers the FER2013 dataset for training the model. Python is the High level computing tool used for Machine learning, Deep learning and Image Processing frameworks

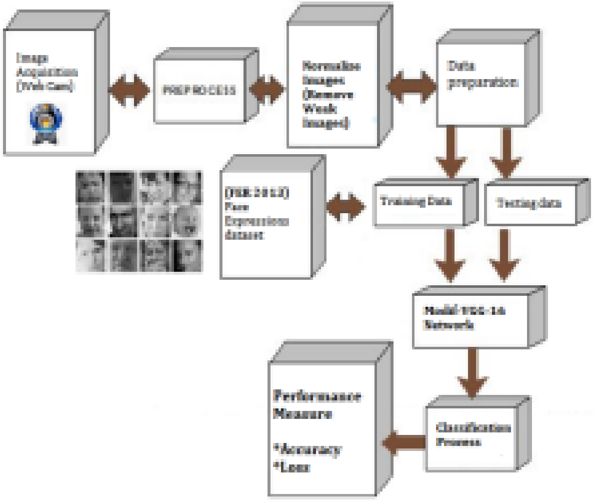


fig 5.1 : system architecture

5.2 Image acquisition

Image capturing is done with the help of a webcam. The Live acquisition of video streams is given as input. Further, the face is detected from the streamed video. Once a face pattern is recognized by the system, image capturing is performed and saved into the backend

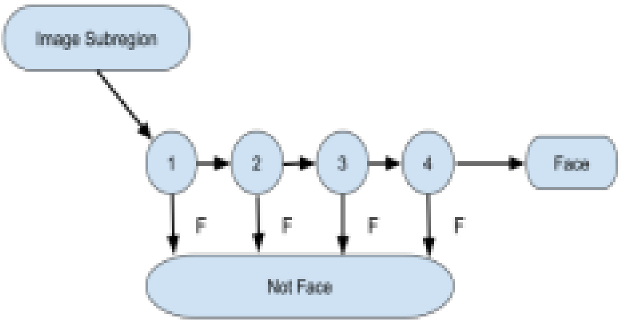


Fig.5.2. Face feature Cascading process

Fig.5.2. Shows the Face feature cascade process. Through Viola Jones algorithm, unique features of the face are extracted one by one and formulated into common robust features. The Cascade prediction is implemented using the OpenCV library.

5.3 Haar cascade technique

The Haar cascade algorithm is considered as one of the fastest feature extraction techniques in the image processing field. The edge detection, line detection and shape detection is the basic model extraction technique. The region of interest is first selected. based on the adjacent pixels present with the selected region, The features of the pixels associated are detected. The differences between the pixel boxes are determined to make the training decision. Cascade Filter

5.4 Cascade Filter

The Haar Cascade technique or the Viola Jones algorithm is based on Cascaded values of unique features. The features of relativity are determined by the weights obtained. These weights of the pixels selected in the region of interest are updated at every iterative analysis. The general form of face features cascading is depicted in the Fig 5.2.

5.5 VGG16 Architecture

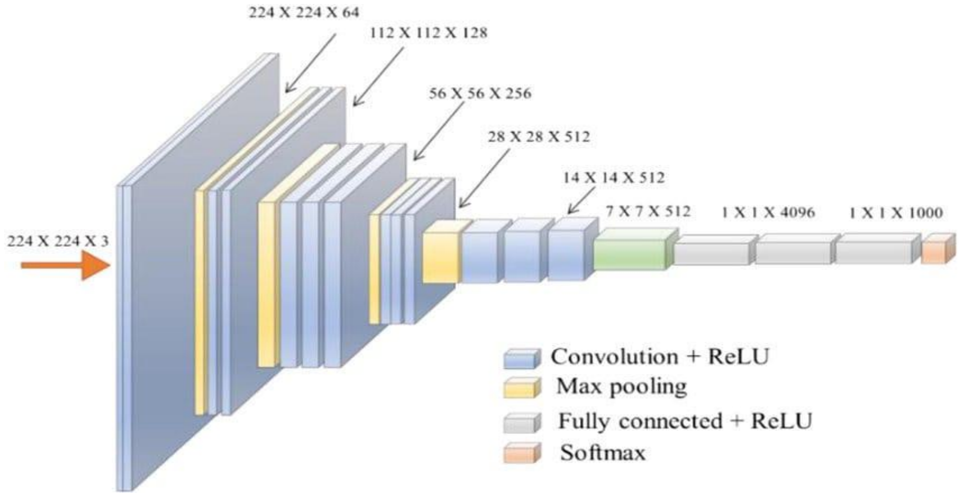


Fig 5.3. VGG-16 Architecture

Fig. 5.3. shows the VGG-16 architecture with deep extraction of feature maps. The presented model utilized an input layer of 224x224x64, followed by the Convolution layer of 112x112x128 unique features mapped. a further number of convolution layers are mapped together in order to extract deep features. The extension of the features depends on the complexity of the input images. Convolution deep layer of 28x28x512 is formulated, finally with the very unique convolution filter of size 14x14x512 is modelled. The output layer holds the very robust feature of the input test image, with the size of 1x1x1000 at the output layer. the Max pooling layer is connected between the output layer and ReLu layer.

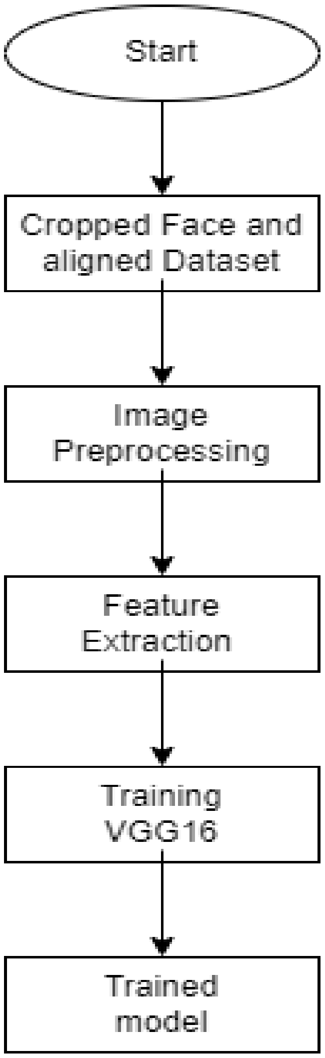
E. Implementation Summary

The proposed real time face expression detection framework is developed here.FER2013 face expression dataset is considered for analysis. The present approach is divided into two phases of operations. Such as Training process and Testing process. In the training process, the FER2013 dataset images are independently fetched into the VGG-16 Deep convolutional neural network architecture. The feature extraction process through Viola-Jones algorithm with Haar Cascade model is implemented for all the database images. Further, Learned features are saved for making the validation process with Test image collected from live webcam, The robust model is created using 80% of training images randomly selected from the FER2013 database, with 20% of testing image from the same folder. Based on the accuracy obtained from the training data and testing data, the Model performance is evaluated. The Live test image captured is fetched into the created model and further the interpretation is expanded. If the face is recognized with Haar Cascade algorithm, expressions are detected using VGG-16 network. A small Validating routine with saddle point estimation framework is modelled here. it considers the accuracy of prediction obtained and further based on the classified expressions in the face image, it recommends a Music to be played in the backend Speaker device. The Music list is a predefined pattern of songs with mixed emotions. Based on the expression, for example if a Happy expression is detected then, a song for excitement and happiness is played.

CHAPTER 6

TABLES AND FIGURES

Fig 6.1. Flow chart diagram of Training



The flow chart of the training model starts with preprocessing the dataset and doing the feature Extraction. Here the preprocessing is done with the help of keras by using image datagen function and later fitting this defined preprocessor to our dataset. Later
we have to feed this dataset to VGG16 Model. it is a slow model, it took lot of time to train
Here for face detection Haar cascade is used.

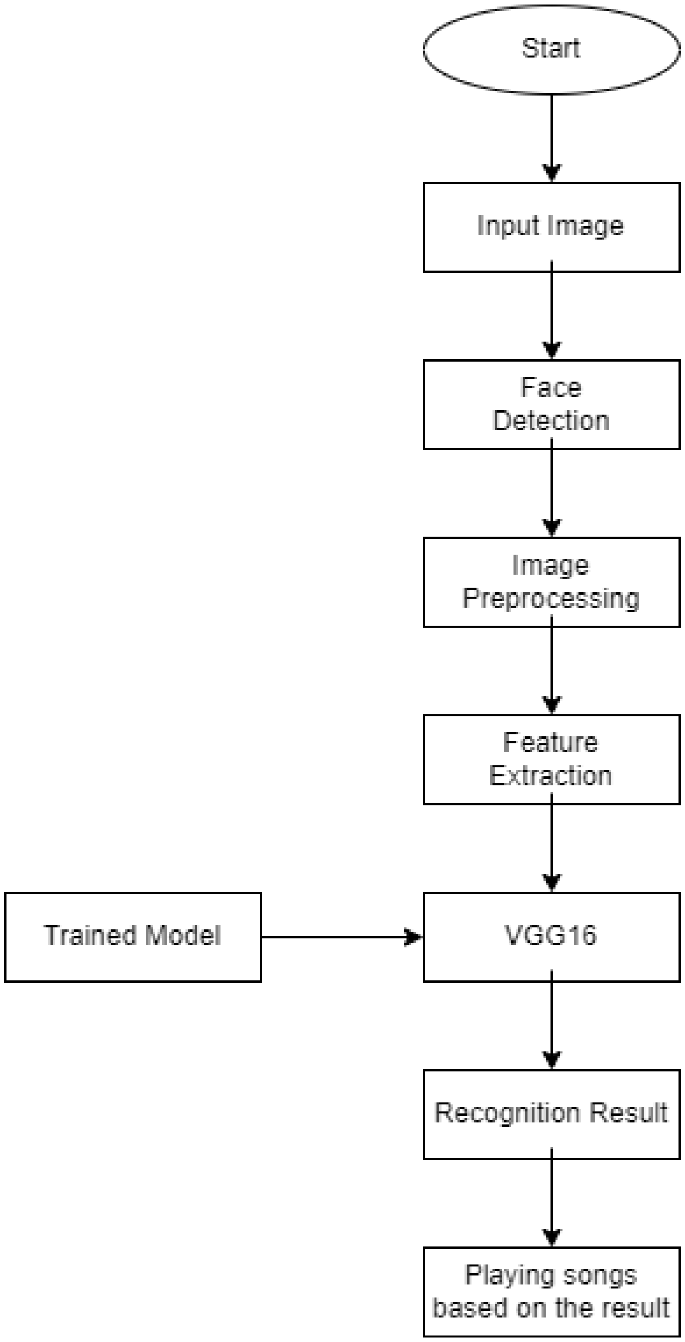
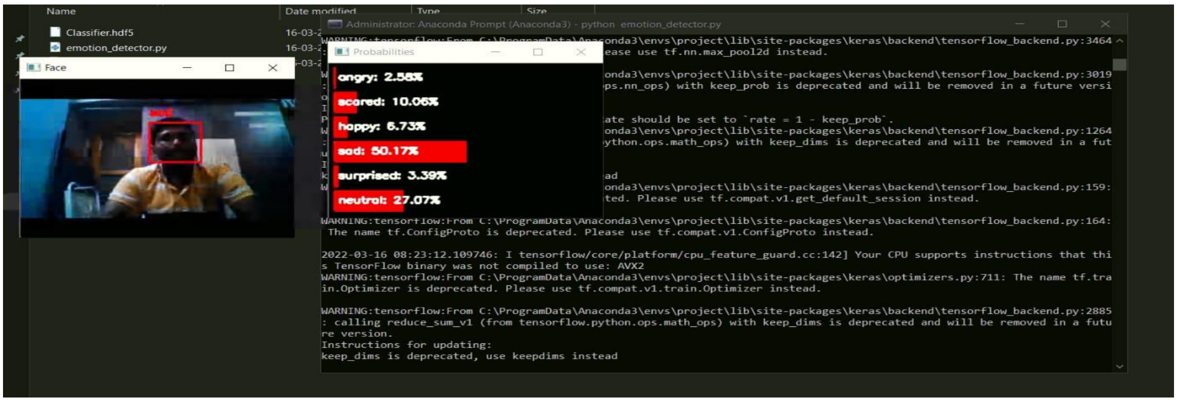


Fig 6.2. Flow chart diagram for testing

OUTPUT:

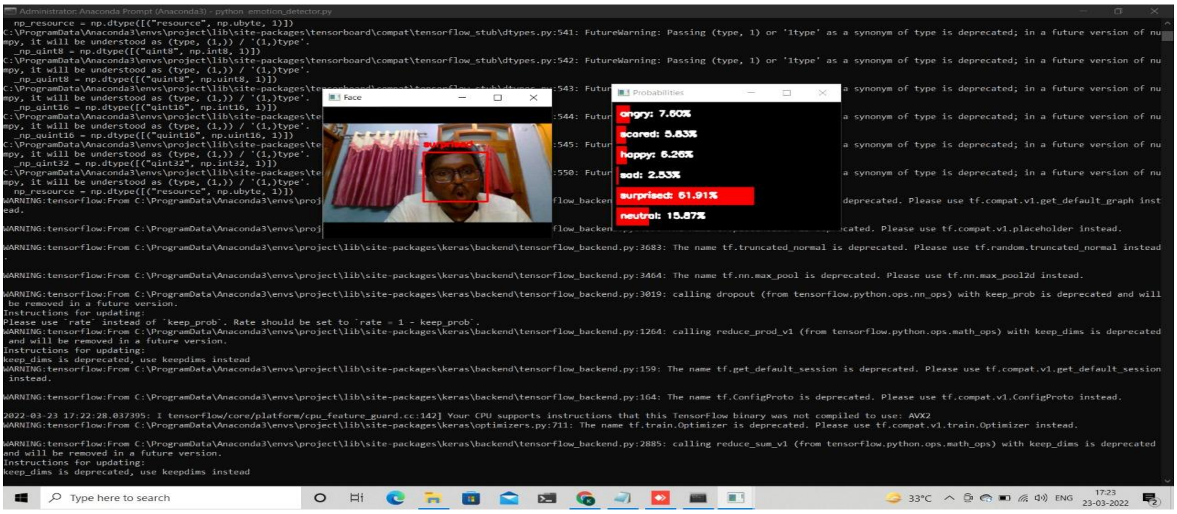


```
angry: 2.58%
scared: 10.06%
happy: 6.73%
sad: 50.17%
surprised: 3.39%
neutral: 27.07%
```

```
WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\project\lib\site-packages\keras\backend\tensorflow_backend.py:3464
:                              ease use tf.nn.max_pool2d instead.

W                              onda3\envs\project\lib\site-packages\keras\backend\tensorflow_backend.py:3019
:                              ps.nn_ops) with keep_prob is deprecated and will be removed in a future versi
o
I
P                              ate should be set to `rate = 1 - keep_prob`.
W                              onda3\envs\project\lib\site-packages\keras\backend\tensorflow_backend.py:1264
:                              ython.ops.math_ops) with keep_dims is deprecated and will be removed in a fut
u
I
k                              ad
W                              onda3\envs\project\lib\site-packages\keras\backend\tensorflow_backend.py:159:
                               ted. Please use tf.compat.v1.get_default_session instead.

WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\project\lib\site-packages\keras\backend\tensorflow_backend.py:164:
 The name tf.ConfigProto is deprecated. Please use tf.compat.v1.ConfigProto instead.

2022-03-16 08:23:12.109746: I tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU supports instructions that thi
s TensorFlow binary was not compiled to use: AVX2
WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\project\lib\site-packages\keras\optimizers.py:711: The name tf.tra
in.Optimizer is deprecated. Please use tf.compat.v1.train.Optimizer instead.

WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\project\lib\site-packages\keras\backend\tensorflow_backend.py:2885
: calling reduce_sum_v1 (from tensorflow.python.ops.math_ops) with keep_dims is deprecated and will be removed in a futu
re version.
Instructions for updating:
keep_dims is deprecated, use keepdims instead
```

Fig 6.3

Fig 6.3 Shows the Simulation result of Live video streaming, where face is detected and shown with a bounding box. Further the face expression with obtained accuracy value is simultaneously shown. At an accuracy of 50.17% as Sad is detected and 27.07% neutral is detected.

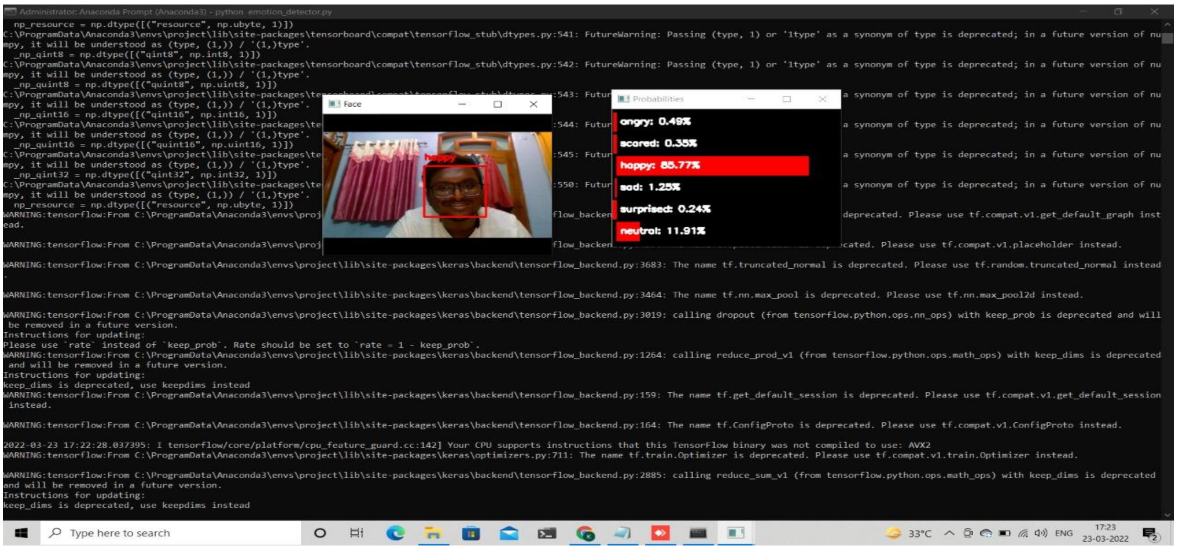

Administrator: Anaconda Prompt (Anaconda3) - python emotion_detector.py
Face
Probabilities
surprised
Type here to search
33°C
ENG
17:23
23-03-2022

```
angry: 7.60%
scared: 5.83%
happy: 6.26%
sad: 2.53%
surprised: 51.91%
neutral: 15.87%
```

```
  np_resource = np.dtype([("resource", np.ubyte, 1)])
C:\ProgramData\Anaconda3\envs\project\lib\site-packages\tensorboard\compat\tensorflow_stub\dtypes.py:541: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of nu
mpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint8 = np.dtype([("qint8", np.int8, 1)])
C:\ProgramData\Anaconda3\envs\project\lib\site-packages\tensorboard\compat\tensorflow_stub\dtypes.py:542: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of nu
mpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
C:\ProgramData\Anaconda3\envs\project\lib\site-packages\te                     :543: Futur                                  a synonym of type is deprecated; in a future version of nu
mpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
C:\ProgramData\Anaconda3\envs\project\lib\site-packages\te                     :544: Futur                                  a synonym of type is deprecated; in a future version of nu
mpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
C:\ProgramData\Anaconda3\envs\project\lib\site-packages\te                     :545: Futur                                  a synonym of type is deprecated; in a future version of nu
mpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
C:\ProgramData\Anaconda3\envs\project\lib\site-packages\te                     :550: Futur                                  a synonym of type is deprecated; in a future version of nu
mpy, it will be understood as (type, (1,)) / '(1,)type'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])
WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\proj                     flow_backen                                  deprecated. Please use tf.compat.v1.get_default_graph inst
ead.

WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\proj                     flow_backen                                cated. Please use tf.compat.v1.placeholder instead.

WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\project\lib\site-packages\keras\backend\tensorflow_backend.py:3683: The name tf.truncated_normal is deprecated. Please use tf.random.truncated_normal instead
.

WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\project\lib\site-packages\keras\backend\tensorflow_backend.py:3464: The name tf.nn.max_pool is deprecated. Please use tf.nn.max_pool2d instead.

WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\project\lib\site-packages\keras\backend\tensorflow_backend.py:3019: calling dropout (from tensorflow.python.ops.nn_ops) with keep_prob is deprecated and will
 be removed in a future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_prob`.
WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\project\lib\site-packages\keras\backend\tensorflow_backend.py:1264: calling reduce_prod_v1 (from tensorflow.python.ops.math_ops) with keep_dims is deprecated
 and will be removed in a future version.
Instructions for updating:
keep_dims is deprecated, use keepdims instead
WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\project\lib\site-packages\keras\backend\tensorflow_backend.py:159: The name tf.get_default_session is deprecated. Please use tf.compat.v1.get_default_session
 instead.

WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\project\lib\site-packages\keras\backend\tensorflow_backend.py:164: The name tf.ConfigProto is deprecated. Please use tf.compat.v1.ConfigProto instead.

2022-03-23 17:22:28.037395: I tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX2
WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\project\lib\site-packages\keras\optimizers.py:711: The name tf.train.Optimizer is deprecated. Please use tf.compat.v1.train.Optimizer instead.

WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\project\lib\site-packages\keras\backend\tensorflow_backend.py:2885: calling reduce_sum_v1 (from tensorflow.python.ops.math_ops) with keep_dims is deprecated
and will be removed in a future version.
Instructions for updating:
keep_dims is deprecated, use keepdims instead
```

Fig 6.4

Fig 6.4. Shows the Simulation result face expression with obtained accuracy value is simultaneously shown. At an accuracy of 51.91% as expression Surprised is detected.



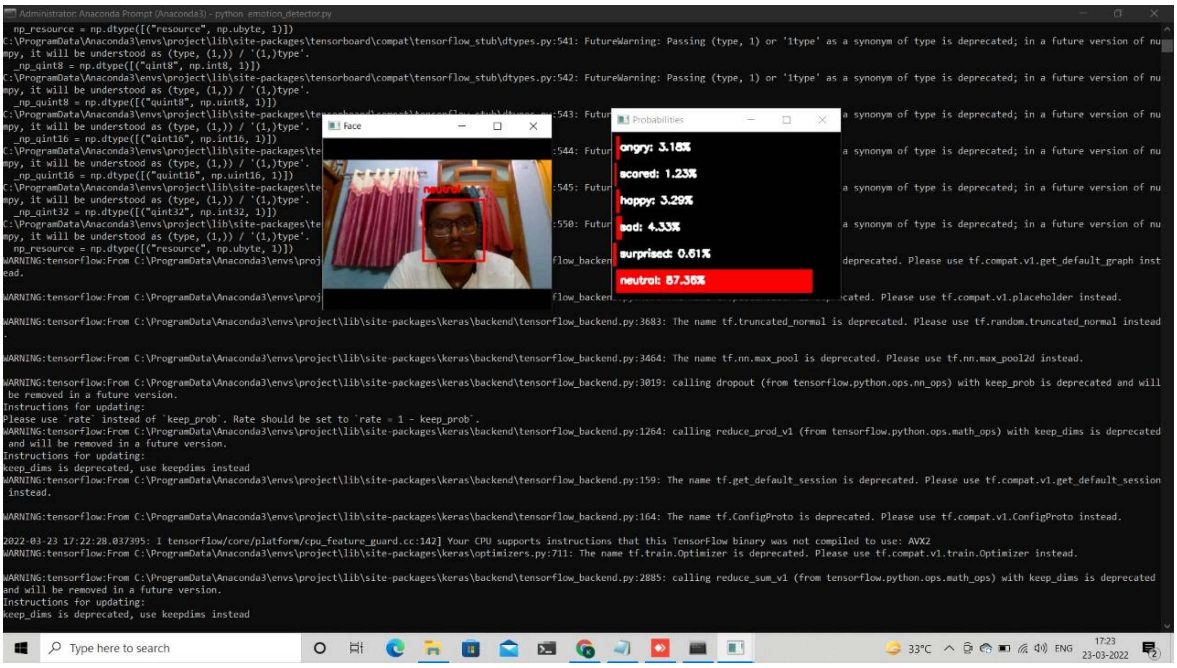
Administrator: Anaconda Prompt (Anaconda3) - python emotion_detector.py
Face
happy
Probabilities
Type here to search
33°C
ENG
17:23
23-03-2022

```
  np_resource = np.dtype([("resource", np.ubyte, 1)])
C:\ProgramData\Anaconda3\envs\project\lib\site-packages\tensorboard\compat\tensorflow_stub\dtypes.py:541: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of nu
mpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint8 = np.dtype([("qint8", np.int8, 1)])
C:\ProgramData\Anaconda3\envs\project\lib\site-packages\tensorboard\compat\tensorflow_stub\dtypes.py:542: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of nu
mpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
C:\ProgramData\Anaconda3\envs\project\lib\site-packages\te                             :543: Futur                                   a synonym of type is deprecated; in a future version of nu
mpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
C:\ProgramData\Anaconda3\envs\project\lib\site-packages\te                             :544: Futur                                   a synonym of type is deprecated; in a future version of nu
mpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
C:\ProgramData\Anaconda3\envs\project\lib\site-packages\te                             :545: Futur                                   a synonym of type is deprecated; in a future version of nu
mpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
C:\ProgramData\Anaconda3\envs\project\lib\site-packages\te                             :550: Futur                                   a synonym of type is deprecated; in a future version of nu
mpy, it will be understood as (type, (1,)) / '(1,)type'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])
WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\proj                             flow_backen                                   deprecated. Please use tf.compat.v1.get_default_graph inst
ead.

WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\proj                             flow_backen                               cated. Please use tf.compat.v1.placeholder instead.

WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\project\lib\site-packages\keras\backend\tensorflow_backend.py:3683: The name tf.truncated_normal is deprecated. Please use tf.random.truncated_normal instead
.

WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\project\lib\site-packages\keras\backend\tensorflow_backend.py:3464: The name tf.nn.max_pool is deprecated. Please use tf.nn.max_pool2d instead.

WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\project\lib\site-packages\keras\backend\tensorflow_backend.py:3019: calling dropout (from tensorflow.python.ops.nn_ops) with keep_prob is deprecated and will
 be removed in a future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_prob`.
WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\project\lib\site-packages\keras\backend\tensorflow_backend.py:1264: calling reduce_prod_v1 (from tensorflow.python.ops.math_ops) with keep_dims is deprecated
 and will be removed in a future version.
Instructions for updating:
keep_dims is deprecated, use keepdims instead
WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\project\lib\site-packages\keras\backend\tensorflow_backend.py:159: The name tf.get_default_session is deprecated. Please use tf.compat.v1.get_default_session
 instead.

WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\project\lib\site-packages\keras\backend\tensorflow_backend.py:164: The name tf.ConfigProto is deprecated. Please use tf.compat.v1.ConfigProto instead.

2022-03-23 17:22:28.037395: I tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX2
WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\project\lib\site-packages\keras\optimizers.py:711: The name tf.train.Optimizer is deprecated. Please use tf.compat.v1.train.Optimizer instead.

WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\project\lib\site-packages\keras\backend\tensorflow_backend.py:2885: calling reduce_sum_v1 (from tensorflow.python.ops.math_ops) with keep_dims is deprecated
and will be removed in a future version.
Instructions for updating:
keep_dims is deprecated, use keepdims instead
```

```
angry: 0.49%
scared: 0.35%
happy: 85.77%
sad: 1.25%
surprised: 0.24%
neutral: 11.91%
```

Fig 6.5

Fig 6.5 Shows the Simulation result face expression with obtained accuracy value is simultaneously shown. At an accuracy of 85.77% as expression Happy is detected.



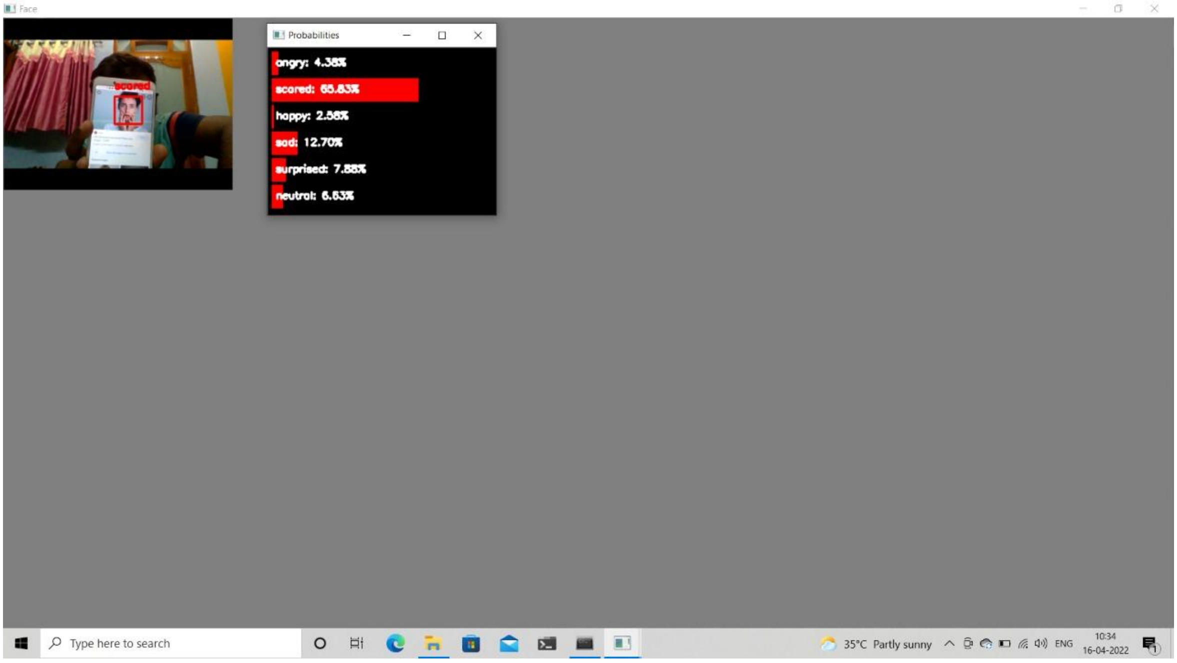
Administrator: Anaconda Prompt (Anaconda3) - python emotion_detector.py
Face
neutral
Probabilities
Type here to search
33°C
ENG
17:23
23-03-2022

```
  np_resource = np.dtype([("resource", np.ubyte, 1)])
C:\ProgramData\Anaconda3\envs\project\lib\site-packages\tensorboard\compat\tensorflow_stub\dtypes.py:541: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of nu
mpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint8 = np.dtype([("qint8", np.int8, 1)])
C:\ProgramData\Anaconda3\envs\project\lib\site-packages\tensorboard\compat\tensorflow_stub\dtypes.py:542: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of nu
mpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
C:\ProgramData\Anaconda3\envs\project\lib\site-packages\te                             :543: Futur                                   a synonym of type is deprecated; in a future version of nu
mpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
C:\ProgramData\Anaconda3\envs\project\lib\site-packages\te                             :544: Futur                                   a synonym of type is deprecated; in a future version of nu
mpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
C:\ProgramData\Anaconda3\envs\project\lib\site-packages\te                             :545: Futur                                   a synonym of type is deprecated; in a future version of nu
mpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
C:\ProgramData\Anaconda3\envs\project\lib\site-packages\te                             :550: Futur                                   a synonym of type is deprecated; in a future version of nu
mpy, it will be understood as (type, (1,)) / '(1,)type'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])
WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\proj                             flow_backen                                   deprecated. Please use tf.compat.v1.get_default_graph inst
ead.

WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\proj                             flow_backen                               cated. Please use tf.compat.v1.placeholder instead.

WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\project\lib\site-packages\keras\backend\tensorflow_backend.py:3683: The name tf.truncated_normal is deprecated. Please use tf.random.truncated_normal instead
.

WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\project\lib\site-packages\keras\backend\tensorflow_backend.py:3464: The name tf.nn.max_pool is deprecated. Please use tf.nn.max_pool2d instead.

WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\project\lib\site-packages\keras\backend\tensorflow_backend.py:3019: calling dropout (from tensorflow.python.ops.nn_ops) with keep_prob is deprecated and will
 be removed in a future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_prob`.
WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\project\lib\site-packages\keras\backend\tensorflow_backend.py:1264: calling reduce_prod_v1 (from tensorflow.python.ops.math_ops) with keep_dims is deprecated
 and will be removed in a future version.
Instructions for updating:
keep_dims is deprecated, use keepdims instead
WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\project\lib\site-packages\keras\backend\tensorflow_backend.py:159: The name tf.get_default_session is deprecated. Please use tf.compat.v1.get_default_session
 instead.

WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\project\lib\site-packages\keras\backend\tensorflow_backend.py:164: The name tf.ConfigProto is deprecated. Please use tf.compat.v1.ConfigProto instead.

2022-03-23 17:22:28.037395: I tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX2
WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\project\lib\site-packages\keras\optimizers.py:711: The name tf.train.Optimizer is deprecated. Please use tf.compat.v1.train.Optimizer instead.

WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\project\lib\site-packages\keras\backend\tensorflow_backend.py:2885: calling reduce_sum_v1 (from tensorflow.python.ops.math_ops) with keep_dims is deprecated
and will be removed in a future version.
Instructions for updating:
keep_dims is deprecated, use keepdims instead
```

```
angry: 3.18%
scared: 1.23%
happy: 3.29%
sad: 4.33%
surprised: 0.61%
neutral: 87.36%
```

Fig 6.6

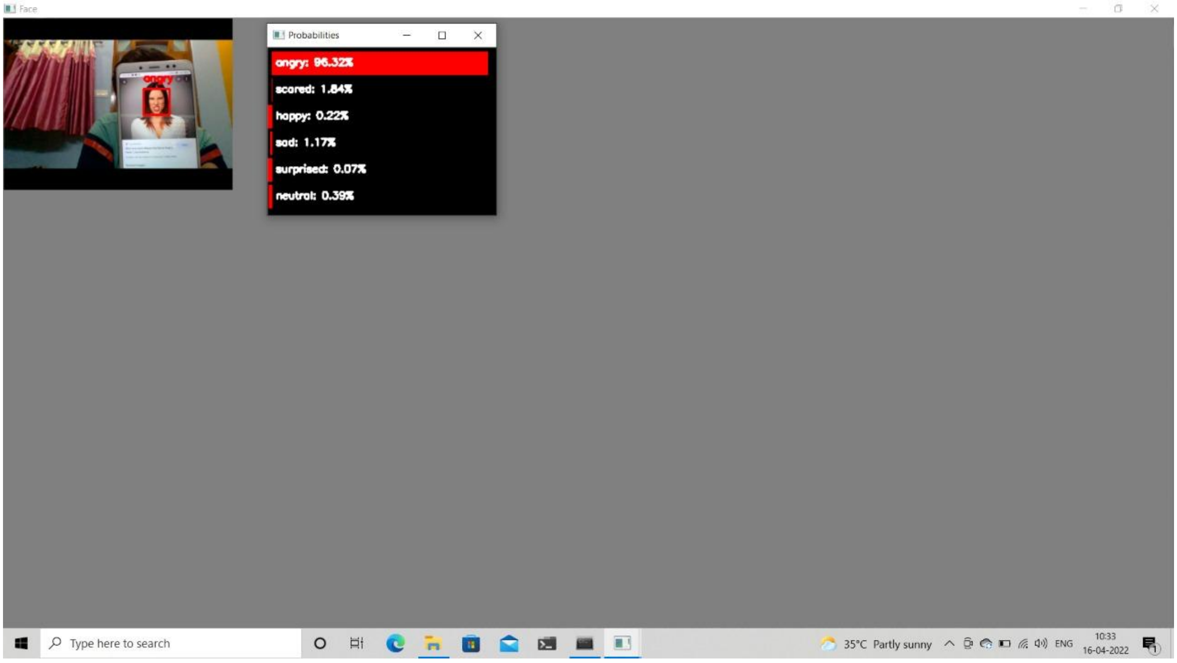
Fig 6.6 Shows the Simulation result face expression with obtained accuracy value is simultaneously shown. At an accuracy of 87.36% as expression Neutral is detected.



```
angry: 4.38%
scared: 65.83%
happy: 2.58%
sad: 12.70%
surprised: 7.88%
neutral: 6.63%
```

Fig 6.7

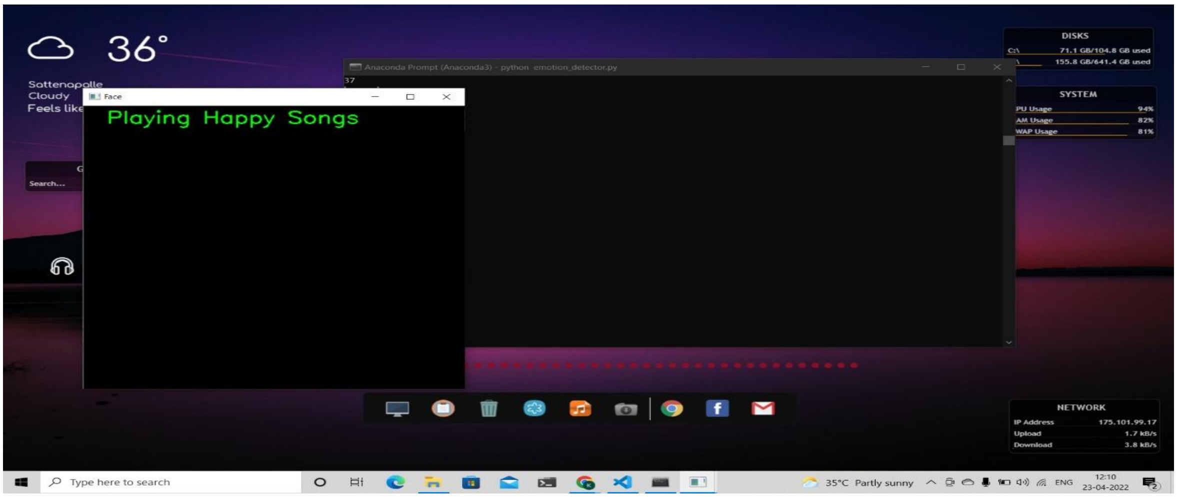
Fig 6.7 Shows the Simulation result face expression with obtained accuracy value is simultaneously shown. At an accuracy of 65.83% as expression scared is detected



```
angry: 96.32%
scared: 1.84%
happy: 0.22%
sad: 1.17%
surprised: 0.07%
neutral: 0.39%
```

Fig 6.8

Fig 6.8 Shows the Simulation result face expression with obtained accuracy value is simultaneously shown. At an accuracy of 96.32% as expression angry is detected



```
37
```

```
Playing Happy Songs
```

Fig 6.9

Fig 6.9 Shows the Simulation result of playing songs when happy emotion is detected

ALGORITHM	**ACCURACY**
RESNET50	37%
BASE MODEL	59%
EFFICIENTNET B0	87%
VGG16	92%

Table 6.1 The table denotes accuracy of all four models

CHAPTER 7
CONCLUSION

Emerging technologies utilized artificial intelligence enabled systems for many products. Various core products depend on face recognition for security and sourcing. Face recognition systems act as one of the foremost needs in many real time emerging products. Facial expressions are helpful for making automated reviews of online courses, products, also to detect the emotional effect of the individuals. expressions are a direct impact of emotions. The proposed framework is modeled using Modif-VGG16 network for facial expression detection as well as the Saddle point recommendation framework for music suggestions. The proposed structure is validated with the FER-2013 dataset. The System achieved the accuracy of 92% on an average and further the system can be improved with more real time dataset with the help of transfer learning approaches.

CHAPTER 8
REFERENCES

[1] F. Liu, Q. Zhao, X. Liu and D. Zeng, "Joint Face Alignment and 3D Face Reconstruction with Application to Face Recognition," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 42, no. 3, pp. 664-678, 1 March 2020, doi: 10.1109/TPAMI.2018.2885995.

[2] M. Emambakhsh and A. Evans, "Nasal Patches and Curves for Expression- Robust 3D Face Recognition," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 5, pp. 995-1007, 1 May 2017, doi: 10.1109/TPAMI.2016.2565473.

[3] F. Qu, S. Wang, W. Yan, H. Li, S. Wu and X. Fu, "CAS(ME)$^2$ : A Database for Spontaneous Macro-Expression and Micro-Expression Spotting and Recognition," in IEEE Transactions on Affective Computing, vol. 9, no. 4, pp. 424-436, 1 Oct.-Dec. 2018, doi: 10.1109/TAFFC.2017.2654440

[4] M. Wairagkar et al., "Emotive Response to a Hybrid-Face Robot and Translation to Consumer Social Robots," in IEEE Internet of Things Journal, vol. 9, no. 5, pp. 3174-3188, 1 March1, 2022, doi: 10.1109/JIOT.2021.3097592.

[5] S. Tulyakov, L. A. Jeni, J. F. Cohn and N. Sebe, "Viewpoint-Consistent 3D Face Alignment," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 40, no. 9, pp. 2250-2264, 1 Sept. 2018, doi: 10.1109/TPAMI.2017.2750687.

[6] R. Zatarain-Cabada, M. L. Barrón-Estrada, F. González-Hernández and H. Rodriguez-Rangel, "Building a Face Expression Recognizer and a Face Expression Database for an Intelligent Tutoring System," 2017 IEEE 17th International Conference on Advanced Learning Technologies (ICALT), 2017, pp. 391-393, doi: 10.1109/ICALT.2017.141.

[7] C. Petpairote, S. Madarasmi and K. Chamnongthai, "A pose and expression face recognition method using transformation based on single face neutral reference," 2017 Global Wireless Summit (GWS), 2017, pp. 123-126, doi: 10.1109/GWS.2017.8300485.

[8] H. Li, N. Wang, X. Ding, X. Yang and X. Gao, "Adaptively Learning Facial Expression Representation via C-F Labels and Distillation," in IEEE Transactions on Image Processing, vol. 30, pp. 2016-2028, 2021, doi: 10.1109/TIP.2021.3049955.

[9] H. Zhang, A. Jolfaei and M. Alazab, "A Face Emotion Recognition Method Using Convolutional Neural Network and Image Edge Computing," in IEEE Access, vol. 7, pp. 159081-159089, 2019, doi: 10.1109/ACCESS.2019.2949741.

[10] H. Mo, L. Liu, W. Zhu, S. Yin and S. Wei, "Face Alignment With Expression- and Pose-Based Adaptive Initialization," in IEEE Transactions on Multimedia, vol. 21, no. 4, pp. 943-956, April 2019, doi: 10.1109/TMM.2018.2867262.

[11] S. L. Happy, P. Patnaik, A. Routray and R. Guha, "The Indian Spontaneous Expression Database for Emotion Recognition," in IEEE Transactions on Affective Computing, vol. 8, no. 1, pp. 131-142, 1 Jan.-March 2017, doi: 10.1109/TAFFC.2015.2498174.

[12] J. Zhang, K. Chen and J. Zheng, "Facial Expression Retargeting From Human to Avatar Made Easy," in IEEE Transactions on Visualization and Computer Graphics, vol. 28, no. 2, pp. 1274-1287, 1 Feb. 2022, doi: 10.1109/TVCG.2020.3013876.

[13] M. Garcia Villanueva and S. Ramirez Zavala, "Deep Neural Network Architecture: Application for Facial Expression Recognition," in IEEE Latin America Transactions, vol. 18, no. 07, pp. 1311-1319, July 2020, doi: 10.1109/TLA.2020.9099774.

[14] H. Yang, D. Huang, Y. Wang and A. K. Jain, "Learning Continuous Face Age Progression: A Pyramid of GANs," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 43, no. 2, pp. 499-515, 1 Feb. 2021, doi: 10.1109/TPAMI.2019.2930985.

[15] V. Wati, K. Kusrini and H. A. Fatta, "Real Time Face Expression Classification Using Convolutional Neural Network Algorithm," 2019 International Conference on Information and Communications Technology (ICOIACT), 2019, pp. 497-501, doi: 10.1109/ICOIACT46704.2019.8938521.

[16] M. Phankokkruad and P. Jaturawat, "Influence of facial expression and viewpoint variations on face recognition accuracy by different face recognition algorithms," 2017 18th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), 2017, pp. 231-237, doi: 10.1109/SNPD.2017.8022727.

[17] X. Chai, J. Chen, C. Liang, D. Xu and C. -W. Lin, "Expression-Aware Face Reconstruction via a Dual-Stream Network," in IEEE Transactions on Multimedia, vol. 23, pp. 2998-3012, 2021, doi: 10.1109/TMM.2021.3068567.

[18] B. Vishnudharan and K. Anusudha, "A discriminative model for facial expression recognition using local directional number pattern," 2016 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT), 2016, pp. 349-352, doi: 10.1109/ICCICCT.2016.7987972

APPENDIX

CODE

FACE DETECTION:

```
# import the necessary packages
import cv2
import imutils

# load the face detector
detector_face = cv2.CascadeClassifier("cascades\\haarcascade_frontalface_default.xml")

# Get the Camera

camera = cv2.VideoCapture(0)

# We go on a continous capture from webcam
while True:
        # grab the current frame
        (grabbed, frame) = camera.read()

        #  if no frame is grabbed then break the loop
        if not grabbed:
                break

        # Resize the frame for better speed, convert it to grayscale, then detect faces and
eyes
        frame = imutils.resize(frame, width=500)
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

        # Detect Faces and get the Bounding Boxes
        faceRects = detector_face.detectMultiScale(gray, scaleFactor=1.05,
minNeighbors=5,minSize=(30, 30), flags=cv2.CASCADE_SCALE_IMAGE)
```

```
		# loop over the faces and draw a rectangle around each
		for (x, y, w, h) in faceRects:

			cv2.rectangle(frame, (x, y), (x + w, y + h), (255,0, 0), 5)

		# show the frame to our screen
		cv2.imshow("Frame", frame)
		key = cv2.waitKey(1) & 0xFF

		# if the 'q' key is pressed, stop the loop
		if key == ord("q"):
			break

# cleanup the camera and close any open windows
camera.release()
cv2.destroyAllWindows()
```

DATSET GENERATION:

```
# USAGE
# python build_dataset.py

# import the necessary packages
from config import emotion_config as config
from tool.io import HDF5DatasetWriter
import numpy as np

# open the input file for reading (skipping the header), then
# initialize the list of data and labels for the training,
# validation, and testing sets
print("[INFO] loading input data...")
f = open(config.INPUT_PATH)
f.__next__() # f.next() for Python 2.7
(trainImages, trainLabels) = ([], [])
```

```
(valImages, valLabels) = ([], [])
(testImages, testLabels) = ([], [])

# loop over the rows in the input file
for row in f:
    # extract the label, image, and usage from the row
    (label, image, usage) = row.strip().split(",")
    label = int(label)

    # if we are ignoring the "disgust" class there will be 6 total
    # class labels instead of 7
    if config.NUM_CLASSES == 6:
        # merge together the "anger" and "disgust classes
        if label == 1:
            label = 0

        # if label has a value greater than zero, subtract one from
        # it to make all labels sequential (not required, but helps
        # when interpreting results)
        if label > 0:
            label -= 1

    # reshape the flattened pixel list into a 48x48 (grayscale)
    # image
    image = np.array(image.split(" "), dtype="uint8")
    image = image.reshape((48, 48))

    # check if we are examining a training image
    if usage == "Training":
        trainImages.append(image)
        trainLabels.append(label)

    # check if this is a validation image
    elif usage == "PrivateTest":
```

```
                valImages.append(image)
                valLabels.append(label)

        # otherwise, this must be a testing image
        else:
                testImages.append(image)
                testLabels.append(label)

# construct a list pairing the training, validation, and testing
# images along with their corresponding labels and output HDF5
# files
datasets = [
        (trainImages, trainLabels, config.TRAIN_HDF5),
        (valImages, valLabels, config.VAL_HDF5),
        (testImages, testLabels, config.TEST_HDF5)]

# loop over the dataset tuples
for (images, labels, outputPath) in datasets:
        # create HDF5 writer
        print("[INFO] building {}...".format(outputPath))
        writer = HDF5DatasetWriter((len(images), 48, 48), outputPath)

        # loop over the image and add them to the dataset
        for (image, label) in zip(images, labels):
                writer.add([image], [label])

        # close the HDF5 writer
        writer.close()

# close the input file
f.close()
```

EMOTION CONFIGURATION:

```
# import the necessary packages
from os import path

# define the base path to the emotion dataset
BASE_PATH = "/raid/datasets/fer2013"

# use the base path to define the path to the input emotions file
INPUT_PATH = path.sep.join([BASE_PATH, "fer2013/fer2013.csv"])

# define the number of classes (set to 6 if you are ignoring the
# "disgust" class)
# NUM_CLASSES = 7
NUM_CLASSES = 6

# define the path to the output training, validation, and testing
# HDF5 files
TRAIN_HDF5 = path.sep.join([BASE_PATH, "hdf5/train.hdf5"])
VAL_HDF5 = path.sep.join([BASE_PATH, "hdf5/val.hdf5"])
TEST_HDF5 = path.sep.join([BASE_PATH, "hdf5/test.hdf5"])

# define the batch size
BATCH_SIZE = 128

# define the path to where output logs will be stored
OUTPUT_PATH = path.sep.join([BASE_PATH, "output"])
```

EFFICIENTNETB0:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sn
import skimage.io
import keras.backend as K
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications import EfficientNetB0
from tensorflow.keras.layers import Dense, Flatten, Dropout,BatchNormalization
,Activation
from tensorflow.keras.models import Model, Sequential
from keras.applications.nasnet import NASNetLarge
from tensorflow.keras.callbacks import ReduceLROnPlateau, ModelCheckpoint,
EarlyStopping
from tensorflow.keras.optimizers import Adam

train_datagen = ImageDataGenerator(rescale = 1./255,
                    validation_split = 0.2,


    rotation_range=5,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    #zoom_range=0.2,
    horizontal_flip=True,
    vertical_flip=True,
    fill_mode='nearest')

valid_datagen = ImageDataGenerator(rescale = 1./255,
                    validation_split = 0.2)

test_datagen  = ImageDataGenerator(rescale = 1./255
                    )

train_dataset  = train_datagen.flow_from_directory(directory = '../input/fer2013/train',
                                 target_size = (48,48),
                                 class_mode = 'categorical',
                                 subset = 'training',
                                 batch_size = 64)

valid_dataset = valid_datagen.flow_from_directory(directory = '../input/fer2013/train',
                                target_size = (48,48),
                                class_mode = 'categorical',
                                subset = 'validation',
                                batch_size = 64)

test_dataset = test_datagen.flow_from_directory(directory = '../input/fer2013/test',
                                target_size = (48,48),
                                class_mode = 'categorical',
                                batch_size = 64)

from keras.preprocessing import image
img = image.load_img("../input/fer2013/test/angry/PrivateTest_10131363.jpg",target_size=(48,48))
img = np.array(img)
plt.imshow(img)
print(img.shape)

img = np.expand_dims(img, axis=0)
from keras.models import load_model
print(img.shape)

base_model = tf.keras.applications.EfficientNetB0(input_shape=(48,48,3),include_top=False,weights="imagenet")
```

```
# Freezing Layers

for layer in base_model.layers[:-4]:
    layer.trainable=False

# Building Model

model=Sequential()
model.add(base_model)
model.add(Dropout(0.5))
model.add(Flatten())
model.add(BatchNormalization())
model.add(Dense(32,kernel_initializer='he_uniform'))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(32,kernel_initializer='he_uniform'))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(32,kernel_initializer='he_uniform'))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(Dense(7,activation='softmax'))

# Model Summary

model.summary()

from tensorflow.keras.utils import plot_model
from IPython.display import Image
plot_model(model, to_file='convnet.png', show_shapes=True,show_layer_names=True)
Image(filename='convnet.png')
```

```
def f1_score(y_true, y_pred): #taken from old keras source code
  true_positives = K.sum(K.round(K.clip(y_true * y_pred, 0, 1)))
  possible_positives = K.sum(K.round(K.clip(y_true, 0, 1)))
  predicted_positives = K.sum(K.round(K.clip(y_pred, 0, 1)))
  precision = true_positives / (predicted_positives + K.epsilon())
  recall = true_positives / (possible_positives + K.epsilon())
  f1_val = 2*(precision*recall)/(precision+recall+K.epsilon())
  return f1_val

METRICS = [
   tf.keras.metrics.BinaryAccuracy(name='accuracy'),
   tf.keras.metrics.Precision(name='precision'),
   tf.keras.metrics.Recall(name='recall'),
   tf.keras.metrics.AUC(name='auc'),
    f1_score,
]

lrd = ReduceLROnPlateau(monitor = 'val_loss',patience = 20,verbose = 1,factor = 0.50,
min_lr = 1e-10)

mcp = ModelCheckpoint('model.h5')

es = EarlyStopping(verbose=1, patience=20)

model.compile(optimizer='Adam', loss='categorical_crossentropy',metrics=METRICS)

history=model.fit(train_dataset,validation_data=valid_dataset,epochs = 50,verbose =
1,callbacks=[lrd,mcp,es])

#%% PLOTTING RESULTS (Train vs Validation FOLDER 1)

def
Train_Val_Plot(acc,val_acc,loss,val_loss,auc,val_auc,precision,val_precision,f1,val_f1):
```

```
fig, (ax1, ax2,ax3,ax4,ax5) = plt.subplots(1,5, figsize= (20,5))
fig.suptitle(" MODEL'S METRICS VISUALIZATION ")

ax1.plot(range(1, len(acc) + 1), acc)
ax1.plot(range(1, len(val_acc) + 1), val_acc)
ax1.set_title('History of Accuracy')
ax1.set_xlabel('Epochs')
ax1.set_ylabel('Accuracy')
ax1.legend(['training', 'validation'])


ax2.plot(range(1, len(loss) + 1), loss)
ax2.plot(range(1, len(val_loss) + 1), val_loss)
ax2.set_title('History of Loss')
ax2.set_xlabel('Epochs')
ax2.set_ylabel('Loss')
ax2.legend(['training', 'validation'])

ax3.plot(range(1, len(auc) + 1), auc)
ax3.plot(range(1, len(val_auc) + 1), val_auc)
ax3.set_title('History of AUC')
ax3.set_xlabel('Epochs')
ax3.set_ylabel('AUC')
ax3.legend(['training', 'validation'])

ax4.plot(range(1, len(precision) + 1), precision)
ax4.plot(range(1, len(val_precision) + 1), val_precision)
ax4.set_title('History of Precision')
ax4.set_xlabel('Epochs')
ax4.set_ylabel('Precision')
ax4.legend(['training', 'validation'])

ax5.plot(range(1, len(f1) + 1), f1)
```

```
    ax5.plot(range(1, len(val_f1) + 1), val_f1)
    ax5.set_title('History of F1-score')
    ax5.set_xlabel('Epochs')
    ax5.set_ylabel('F1 score')
    ax5.legend(['training', 'validation'])


    plt.show()


Train_Val_Plot(history.history['accuracy'],history.history['val_accuracy'],
               history.history['loss'],history.history['val_loss'],
               history.history['auc'],history.history['val_auc'],
               history.history['precision'],history.history['val_precision'],
               history.history['f1_score'],history.history['val_f1_score']
              )
```

VGG16:

```
# USAGE
# python train_recognizer.py --checkpoints fer2013/checkpoints
# python train_recognizer.py --checkpoints fer2013/checkpoints --model
fer2013/checkpoints/epoch_20.hdf5 \
#         --start-epoch 20

# set the matplotlib backend so figures can be saved in the background
import matplotlib
matplotlib.use("Agg")

# import the necessary packages
from config import emotion_config as config
from tool.preprocessing import ImageToArrayPreprocessor
from tool.callbacks import EpochCheckpoint
from tool.callbacks import TrainingMonitor
from tool.io import HDF5DatasetGenerator
from tool.nn.conv import EmotionVGGNet
from keras.preprocessing.image import ImageDataGenerator
from keras.optimizers import Adam
from keras.models import load_model
import keras.backend as K
import argparse
import os

# construct the argument parse and parse the arguments
ap = argparse.ArgumentParser()
ap.add_argument("-c", "--checkpoints", required=True,
	help="path to output checkpoint directory")
ap.add_argument("-m", "--model", type=str,
	help="path to *specific* model checkpoint to load")
ap.add_argument("-s", "--start-epoch", type=int, default=0,
	help="epoch to restart training at")
args = vars(ap.parse_args())

# construct the training and testing image generators for data
# augmentation, then initialize the image preprocessor
trainAug = ImageDataGenerator(rotation_range=10, zoom_range=0.1,
	horizontal_flip=True, rescale=1 / 255.0, fill_mode="nearest")
valAug = ImageDataGenerator(rescale=1 / 255.0)
iap = ImageToArrayPreprocessor()

# initialize the training and validation dataset generators
trainGen = HDF5DatasetGenerator(config.TRAIN_HDF5, config.BATCH_SIZE,
	aug=trainAug, preprocessors=[iap], classes=config.NUM_CLASSES)
valGen = HDF5DatasetGenerator(config.VAL_HDF5, config.BATCH_SIZE,
	aug=valAug, preprocessors=[iap], classes=config.NUM_CLASSES)

# if there is no specific model checkpoint supplied, then initialize
# the network and compile the model
```

```
if args["model"] is None:
	print("[INFO] compiling model...")
	model = EmotionVGGNet.build(width=48, height=48, depth=1,
		classes=config.NUM_CLASSES)
	opt = Adam(lr=1e-3)
	model.compile(loss="categorical_crossentropy", optimizer=opt,
		metrics=["accuracy"])

# otherwise, load the checkpoint from disk
else:
	print("[INFO] loading {}...".format(args["model"]))
	model = load_model(args["model"])

	# update the learning rate
	print("[INFO] old learning rate: {}".format(
		K.get_value(model.optimizer.lr)))
	K.set_value(model.optimizer.lr, 1e-5)
	print("[INFO] new learning rate: {}".format(
		K.get_value(model.optimizer.lr)))

# construct the set of callbacks
figPath = os.path.sep.join([config.OUTPUT_PATH,
	"vggnet_emotion.png"])
jsonPath = os.path.sep.join([config.OUTPUT_PATH,
	"vggnet_emotion.json"])
callbacks = [
	EpochCheckpoint(args["checkpoints"], every=5,
		startAt=args["start_epoch"]),
	TrainingMonitor(figPath, jsonPath=jsonPath,
		startAt=args["start_epoch"])]

# train the network
model.fit_generator(
	trainGen.generator(),
          steps_per_epoch=trainGen.numImages // config.BATCH_SIZE,
          validation_data=valGen.generator(),
          validation_steps=valGen.numImages // config.BATCH_SIZE,
          epochs=40,
          max_queue_size=10,
          callbacks=callbacks, verbose=1)

# close the databases
trainGen.close()
valGen.close()
```

EMOTION DETECTION:

```
 USAGE
# python emotion_detector.py --video (path to the video) # optional video argument

# import the necessary packages
from keras.preprocessing.image import img_to_array
from keras.models import load_model
import numpy as np
import argparse
import imutils
import cv2
from playsound import playsound
import os
import time

label = "0"
# construct the argument parse and parse the arguments
ap = argparse.ArgumentParser()
ap.add_argument("-v", "--video",
          help="path to the (optional) video file")
args = vars(ap.parse_args())
```

```
# load the face detector cascade, emotion detection CNN, then define
# the list of emotion labels
detector = cv2.CascadeClassifier("haarcascade_frontalface_default.xml")
model = load_model("Classifier.hdf5")
EMOTIONS = ["angry", "scared", "happy", "sad", "surprised",
        "neutral"]

# if a video path was not supplied, grab the reference to the webcam
if not args.get("video", False):
        camera = cv2.VideoCapture(0)

# otherwise, load the video
else:
        camera = cv2.VideoCapture(args["video"])
prev_label = "neutral"
count = 0
# keep looping
while True:



        canvas_1 = np.zeros((500, 500, 3), dtype="uint8")*255
        # grab the current frame
        (grabbed, frame) = camera.read()

        # if we are viewing a video and we did not grab a
        # frame, then we have reached the end of the video
        if args.get("video") and not grabbed:
                break

        # resize the frame and convert it to grayscale
        frame = imutils.resize(frame, width=300)
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
```

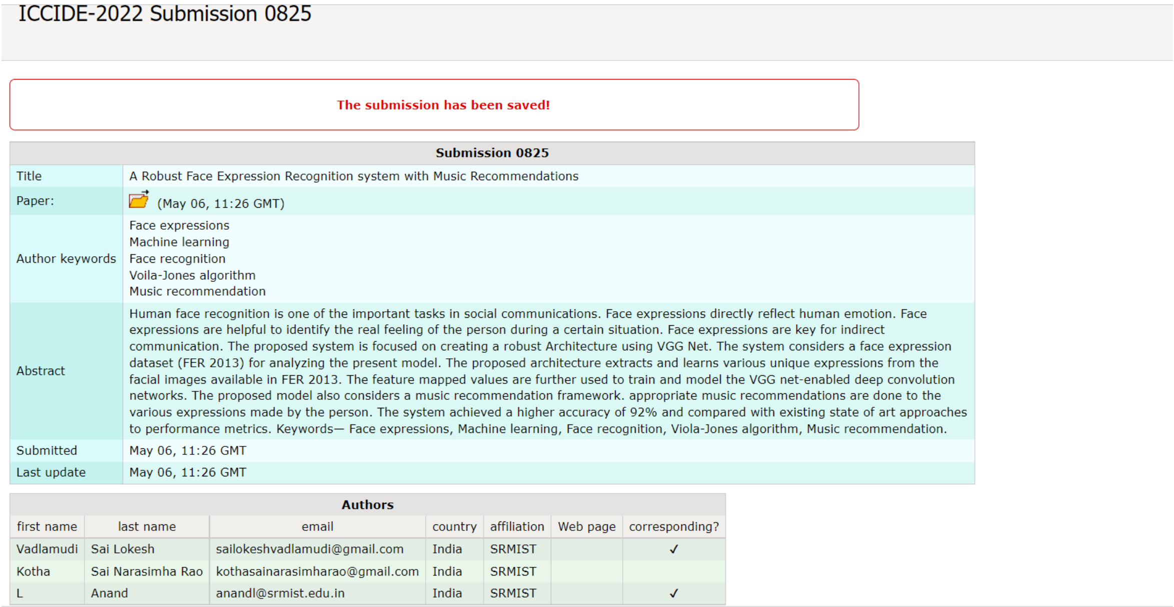
```
# initialize the canvas for the visualization, then clone
# the frame so we can draw on it
canvas = np.zeros((220, 300, 3), dtype="uint8")
frameClone = frame.copy()

# detect faces in the input frame, then clone the frame so that
# we can draw on it
rects = detector.detectMultiScale(gray, scaleFactor=1.1,
        minNeighbors=5, minSize=(30, 30),
        flags=cv2.CASCADE_SCALE_IMAGE)

# ensure at least one face was found before continuing
if len(rects) > 0:
        # determine the largest face area
        rect = sorted(rects, reverse=True,
                key=lambda x: (x[2] - x[0]) * (x[3] - x[1]))[0]
        (fX, fY, fW, fH) = rect

        # extract the face ROI from the image, then pre-process
        # it for the network
        roi = gray[fY:fY + fH, fX:fX + fW]
        roi = cv2.resize(roi, (48, 48))
        roi = roi.astype("float") / 255.0
        roi = img_to_array(roi)
        roi = np.expand_dims(roi, axis=0)

        # make a prediction on the ROI, then lookup the class
        # label
        preds = model.predict(roi)[0]
        label = EMOTIONS[preds.argmax()]

        # loop over the labels + probabilities and draw them
        for (i, (emotion, prob)) in enumerate(zip(EMOTIONS, preds)):
                # construct the label text
```

```
            text = "{}: {:.2f}%".format(emotion, prob * 100)

            # draw the label + probability bar on the canvas
            w = int(prob * 300)
            cv2.rectangle(canvas, (5, (i * 35) + 5),
                (w, (i * 35) + 35), (0, 0, 255), -1)
            cv2.putText(canvas, text, (10, (i * 35) + 23),
                cv2.FONT_HERSHEY_SIMPLEX, 0.45,
                (255, 255, 255), 2)

        # draw the label on the frame
        cv2.putText(frameClone, label, (fX, fY - 10),
            cv2.FONT_HERSHEY_SIMPLEX, 0.45, (0, 0, 255), 2)
        cv2.rectangle(frameClone, (fX, fY), (fX + fW, fY + fH),
            (0, 0, 255), 2)

    # show our classifications + probabilities
    cv2.imshow("Face", frameClone)
    cv2.imshow("Probabilities", canvas)

    print (label,prev_label)
    if (str(prev_label) == str(label)):
        count += 1
        print (count)
        #prev_label = label
    else :
        count = 0
    prev_label = label
    #print ("test")
    if count > 50:
        if (label == "happy"):
            cv2.putText(canvas_1, 'Playing Happy Songs', (30, 30),
cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2)
            cv2.imshow("Face", canvas_1)
            cv2.waitKey(5000)
            cv2.destroyAllWindows()
            listOfFiles = os.listdir("happy")
            count = 0
            print ("playing Happy songs")
            for i in range (0, len(listOfFiles)):
                playsound(("happy\\"+str(listOfFiles[i])))
                time.sleep(2)
        elif (label == "angry"):
            cv2.putText(canvas_1, 'Playing Angry Songs', (30, 30),
cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2)
            cv2.imshow("Face", canvas_1)
            cv2.waitKey(5000)
            cv2.destroyAllWindows()
            listOfFiles = os.listdir("angry")
            count = 0
            print ("Playing angry songs")
            for i in range (0, len(listOfFiles)):
                playsound(("happy\\"+str(listOfFiles[i])))
                time.sleep(2)
        elif (label == "sad"):
            cv2.putText(canvas_1, 'Playing Sad Songs', (30, 30),
cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2)
            cv2.imshow("Face", canvas_1)
            cv2.waitKey(5000)
            cv2.destroyAllWindows()
            listOfFiles = os.listdir("sad")
            count = 0
            print ("playing sad songs")
            for i in range (0, len(listOfFiles)):
                playsound(("sad\\"+str(listOfFiles[i])))
                time.sleep(2)
        elif (label == "scared"):
```

```
                        cv2.putText(canvas_1, 'Playing scared Songs', (30, 30),
cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2)
                        cv2.imshow("Face", canvas_1)
                        cv2.waitKey(5000)
                        cv2.destroyAllWindows()
                        listOfFiles = os.listdir("scared")
                        count = 0
                        print ("playing scared songs")
                        for i in range (0, len(listOfFiles)):
                                playsound(("scared\\"+str(listOfFiles[i])))
                                time.sleep(2)
                elif (label == "surprised"):
                        cv2.putText(canvas_1, 'Playing surprised Songs', (30, 30),
cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2)
                        cv2.imshow("Face", canvas_1)
                        cv2.waitKey(5000)
                        cv2.destroyAllWindows()
                        listOfFiles = os.listdir("surprised")
                        count = 0
                        print ("playing surprised songs")
                        for i in range (0, len(listOfFiles)):
                                playsound(("surprised\\"+str(listOfFiles[i])))
                                time.sleep(2)
                #break




        # if the 'q' key is pressed, stop the loop
        if cv2.waitKey(1) & 0xFF == ord("q"):
                break
# cleanup the camera and close any open windows
camera.release()
cv2.destroyAllWindows()
```

PAPER PUBLICATION STATUS

Submitted to a Conference(Scopus Indexed)



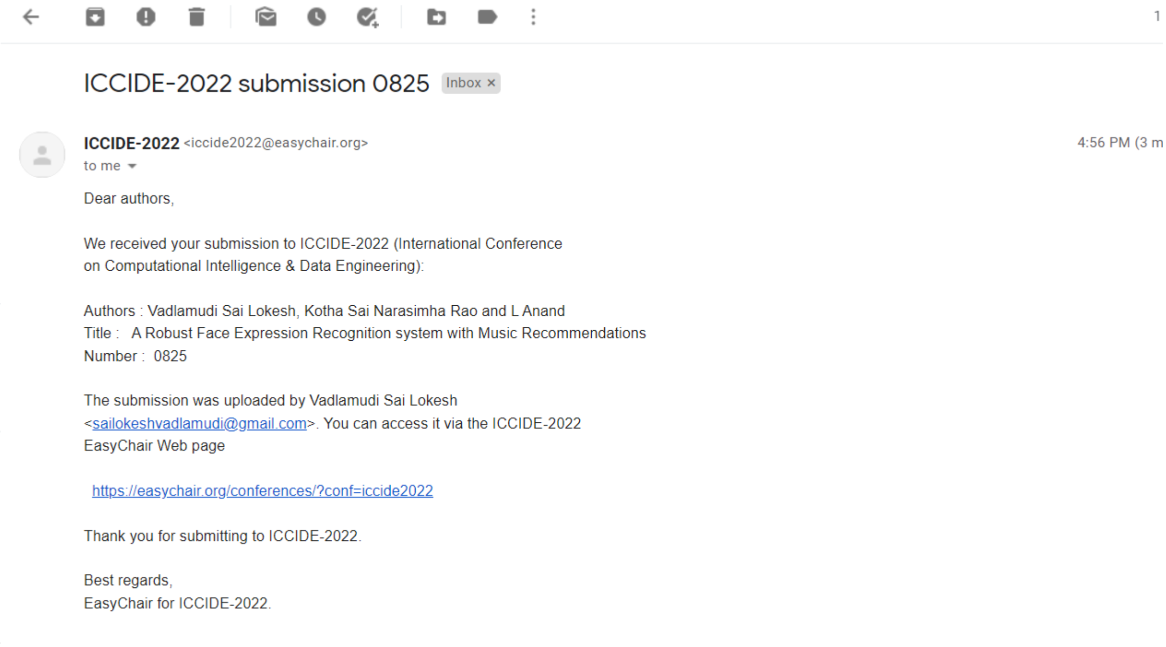
ICCIDE-2022 Submission 0825

The submission has been saved!

Submission 0825	
Title	A Robust Face Expression Recognition system with Music Recommendations
Paper:	(May 06, 11:26 GMT)
Author keywords	Face expressions Machine learning Face recognition Voila-Jones algorithm Music recommendation
Abstract	Human face recognition is one of the important tasks in social communications. Face expressions directly reflect human emotion. Face expressions are helpful to identify the real feeling of the person during a certain situation. Face expressions are key for indirect communication. The proposed system is focused on creating a robust Architecture using VGG Net. The system considers a face expression dataset (FER 2013) for analyzing the present model. The proposed architecture extracts and learns various unique expressions from the facial images available in FER 2013. The feature mapped values are further used to train and model the VGG net-enabled deep convolution networks. The proposed model also considers a music recommendation framework. appropriate music recommendations are done to the various expressions made by the person. The system achieved a higher accuracy of 92% and compared with existing state of art approaches to performance metrics. Keywords— Face expressions, Machine learning, Face recognition, Viola-Jones algorithm, Music recommendation.
Submitted	May 06, 11:26 GMT
Last update	May 06, 11:26 GMT

Authors

first name	last name	email	country	affiliation	Web page	corresponding?
Vadlamudi	Sai Lokesh	sailokeshvadlamudi@gmail.com	India	SRMIST		✓
Kotha	Sai Narasimha Rao	kothasainarasimharao@gmail.com	India	SRMIST		
L	Anand	anandl@srmist.edu.in	India	SRMIST		✓



ICCIDE-2022 submission 0825 Inbox ×

ICCIDE-2022 <iccide2022@easychair.org> 4:56 PM (3 m

to me

Dear authors,

We received your submission to ICCIDE-2022 (International Conference on Computational Intelligence & Data Engineering):

Authors : Vadlamudi Sai Lokesh, Kotha Sai Narasimha Rao and L Anand
Title : A Robust Face Expression Recognition system with Music Recommendations
Number : 0825

The submission was uploaded by Vadlamudi Sai Lokesh <sailokeshvadlamudi@gmail.com>. You can access it via the ICCIDE-2022 EasyChair Web page

https://easychair.org/conferences/?conf=iccide2022

Thank you for submitting to ICCIDE-2022.

Best regards,
EasyChair for ICCIDE-2022.