EMOTION RECOGNITION USING IMAGE ANALYSIS

A PROJECT REPORT

Submitted by

ARYAN OMKAR ASHAR-19BAI10094
SHREYA SHETYE-19BAI10028
SOUMYA RAJADHYAKSHA: -19BAI10120
PRANAV SHARMA: - 19BAI10154

in partial fulfillment for the award of the degree
of

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE AND ENGINEERING

Specialization in

Artificial intelligence and machine learning

*

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
VIT BHOPAL UNIVERSITY
KOTHRIKALAN, SEHORE
MADHYA PRADESH - 466114

MAY 2021.

VIT BHOPAL UNIVERSITY,KOTHRIKALAN, SEHORE
MADHYA PRADESH – 466114

BONAFIDE CERTIFICATE

Certified that this project report titled **“EMOTION RECOGNITION USING IMAGE ANALYSIS”** is the bonafide work of “**Aryan Omkar Ashar(19BAI10094), Shreya Shetye (19BAI10028), Soumya Rajadhyaksha(19BAI10120) and Pranav Sharma(19BAI10154)”** who carried out the project work under my supervision. Certified further that to the best of my knowledge the work reported here does not form part of any other project / research work on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

PROGRAM CHAIR
Dr. S. Sountharrajan
School of AI &ML division
VIT BHOPAL UNIVERSITY

PROJECT GUIDE
Dr. Nilamadhab Mishra
School of AI &ML division
VIT BHOPAL UNIVERSITY

The Project Exhibition I Examination is held on 05/05/2021

ACKNOWLEDGEMENT

First and foremost, I would like to thank the Lord Almighty for His presence and immense blessings throughout the project work.

I wish to express my heartfelt gratitude to Dr. Manas Kumar Mishra, Head of the Department, School of Computing Science and Engineering for much of his valuable support encouragement in carrying out this work.

I would like to thank my internal guide Dr. Nilamadhab Mishra, for continually guiding and actively participating in my project, giving valuable suggestions to complete the project work.

I would like to thank all the technical and teaching staff of the School of Computing Science and Engineering, who extended directly or indirectly all support.

Last, but not the least, I am deeply indebted to my parents who have been the greatest support while I worked day and night for the project to make it a success.

-

LIST OF ABBREVIATIONS-

AI – Artificial Intelligence
ML- Machine Learning
API- Application Programming Interface
OpenCV- Open-Source Computer Vision Library
cuDNN: - CUDA Deep Neural Network Library.

LIST OF FIGURES

LIST OF GRAPHS

ABSTRACT

Human emotion recognition plays an important and a vital role in interpersonal relationship. Human Emotions affect all the actions performed by the human body and help in coloring every thought an action. They are an essential ingredient of our life. But it is way easier for a human to hide his emotions and thus affects the environment around him. The automatic recognition of emotions has been an active research topic in the recent years. Emotion recognition is an important area of research to enable effective human-computer interaction. Today, Emotion recognition is used in many major areas involving business relations, recording the customer experience, official meetings, analyzing the behavior of students in universities and many other uses of emotion recognition.

In this project we work on real-time analysis of emotion of a person by the use of hardware components like web-camera and software programming involving different libraries of Python 3 programming. Also, the user can choose what type of input he will like to give. The user has a choice whether he/she wants to detect the emotion of person in real-time using web camera or he /she wants to feed an image/video file from the computer. This emotion recognition model uses various ML and Image analysis techniques. This project throws light on the feature integration and classification in machine learning and image analysis. This project goes to the very depth of emotion recognition in which we aim to identify 7 types of emotions namely angry, disgusted, fearful, happy, sad, surprised and neutral. The project contains a training data and a testing data used for model training and model testing which was developed in the project. The first module is based on processing of data, making of directories, segregating data by the use of machine learning algorithms and then saving the images in their appropriate folder. The second method contains of high-level data analytics and use of graphs and images in order to analyze the dataset.

In the next module we compile and run all the modules, open the webcam and perform recognition. The final module consists of importing the model and coding of the model used I order to recognize the emotion of person in real-time.

The project also contains an emotion called neutral if the model thinks that the mood of the person is neither of the emotions specified above. A total of 35,778 images are used for training the model. This project mainly focuses and throws light on real time implementation of an emotion recognition model trained and tested by the help of Python 3 programming. The model identifies and segregates the seven basic emotions that appear to be universal across humanity and can appear on each and every individual’s face.

TABLE OF CONTENTS

1. INTRODUCTION

1.1: Introduction

Human emotions can be used to determine what a person thinks as they are brought on by neurophysiological changes associated with thoughts, feelings, behavioral responses, and a degree of pleasure or displeasure. While there is currently no scientific consensus on a definition, emotions are often intertwined with mood, temperament, personality, disposition, creativity, and motivation. Human emotion detection is enforced in several areas in today's advanced world for requiring extra security or information regarding people. It can also help us provide additional security wherever needed by identifying the face and facial emotion, which can be used to guess his/her motive. Human emotion detection can also help by performing human verification. Another strong domain where emotion detection can be used is business promotions, especially in the advertising industry. As most companies usually thrive on customer response, using an artificially intelligent system, we can capture and identify real-time emotions based on user image or video, helping us decide whether the customer liked or disliked the product or an offer. Using the human emotion dataset is an excellent example of finding out the hardiness and nature of classification algorithms and how they perform various datasets. Additionally, we can also make our model recognize some common gestures and primary sign language, which will make it more advanced and compatible with all users.

1.2: Motivation for the work

We have very nicely observed that during human communication the non-verbal information prevails over spoken words. The personality and the emotions of the person speak a lot about his intentions and interest in the particular ongoing discussion. Computers have been a part of our lives since the past many years and we have always worked on the human-computer interaction since the entrance of AI and ML in the recent technological advancements in the recent years. Ubiquitous and universal usage of computers requires enhances human-computer interaction and thus Emotion recognition has been in great demand where different algorithms have been coming up each time increasing the accuracy of the model and making it more diverse. We have tried the same thing to make the model more accurate and more diverse in terms of the number of emotions detected.

Emotion recognition has been one of the most researched fields among today’s generation. It was a wide variety of use in all aspects of life including professional and personal both. The interaction between human and computers will be more natural if computers are able to perceive and respond to humans and are also able to read their emotions.

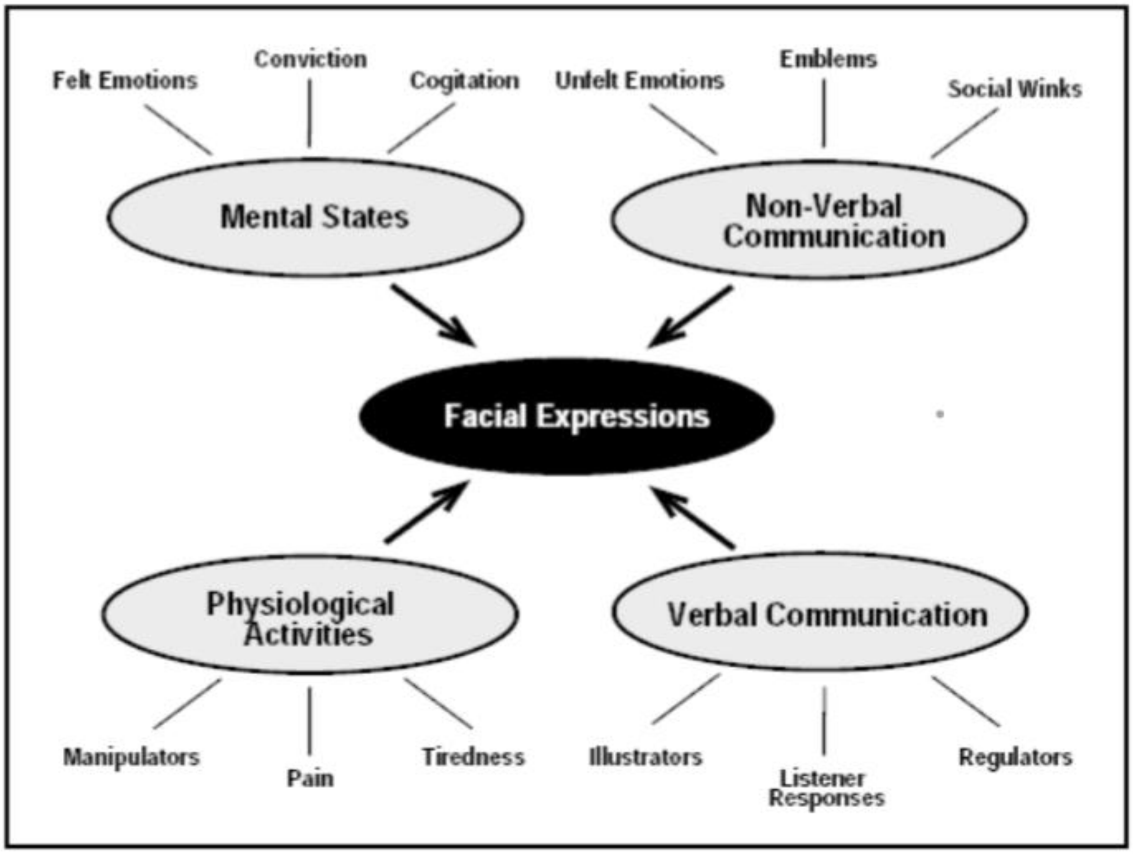


Fig 1.2.1: - Facial Expressions in communication.

1.3: About Introduction to the project including techniques

Facial expressions can express what a person is feeling or thinking about. Usually, we can interpret a person's emotions with the help of their facial expressions. We can use our model in real-time or on the image provide by the user to recognize the emotions of the people in the image through their facial expressions. Emotion detection can prove to be very useful in our daily lives by making certain tasks like self-evaluation easier. As per our current situation almost all the business interactions are taking place via an online platform. Due to this, emotion recognition can be used anytime to easily evaluate the feedback and get a statistical result.

It can mainly be used in the marketing industry to collect the feedback of the consumers so that we can evaluate the performance of our product or service. We can identify what emotion a person is feeling with the help of our model and then classify it as positive or negative feedback. This feedback can be used to evaluate our product/service and make the necessary changes or upgrades as required.

The basic techniques we have used in our project are as follows:

- We use the Fer2003 dataset to train our model. This dataset contains several images of people showing various emotions such as Angry, Disgust, Fear, Happy, Sad, Surprise and Neutral. We use this data set to categorize each face based on the emotion shown in the facial expression into one of the given seven categories.
- We train our model and analyse the dataset given and then we use ML classification techniques for training and image analysis for capturing the face in live-feed.

- We then identify the face of the person in the live feed or the given image provided by the user. Using our trained model, then we than classify the emotion of the recognized face based on our dataset.
- Our model gives us an output with a box around the face of the person with the recognised emotion written above it.
- In this way, we have successfully classified the emotion of the person using our emotion recognition model.
- We find that the value accuracy of our model is around 87-88% on finding 50 epochs. The maximum accuracy of the model comes out to be 98%. After that even on increasing the number of epochs, the accuracy remains the same.

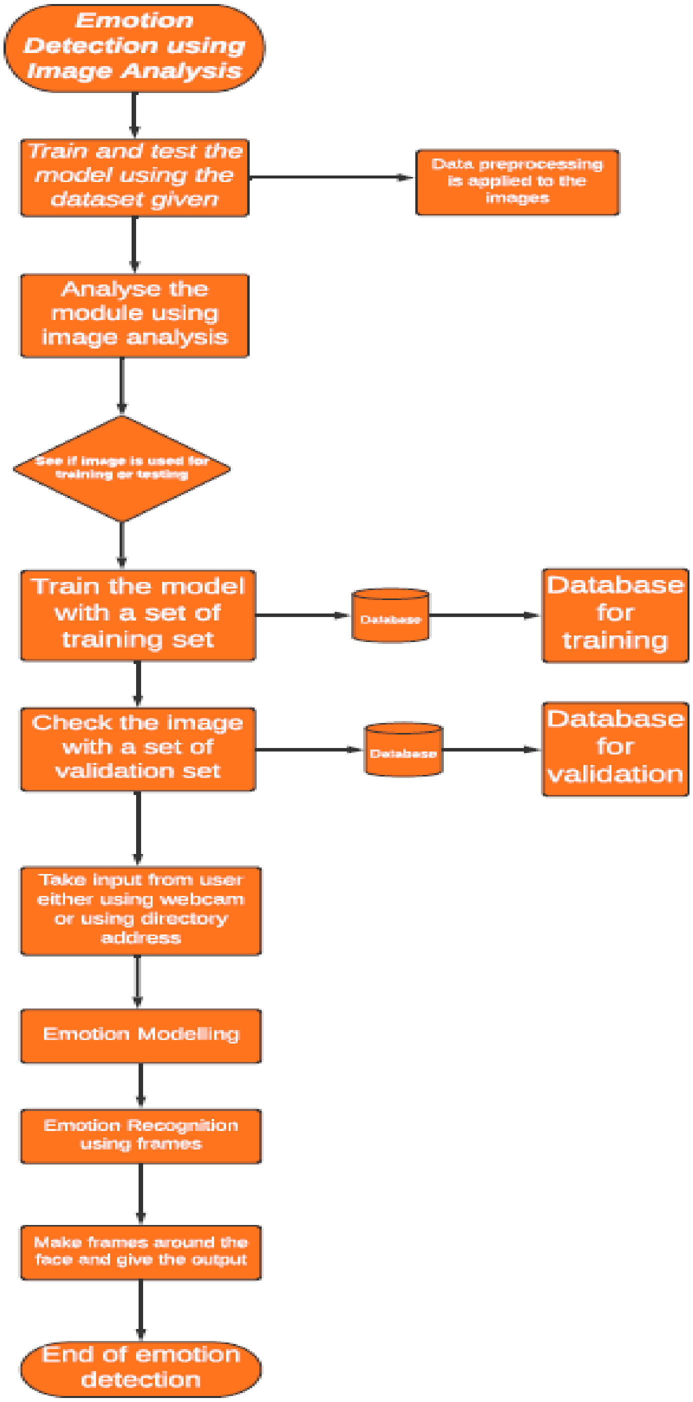

Emotion Detection using Image Analysis
Train and test the model using the dataset given
Data preprocessing is applied to the images
Analyse the module using image analysis
See if image is used for training or testing
Train the model with a set of training set
Database
Database for training
Check the image with a set of validation set
Database
Database for validation
Take input from user either using webcam or using directory address
Emotion Modelling
Emotion Recognition using frames
Make frames around the face and give the output
End of emotion detection

Fig. 1.3.1 METHODOLOGY

1.4: Problem Statement

- During the launch of a product by any company the human emotions are a prime factor that determine the success of the product, but there is no provision for detection of emotions at such a great scale.
- There is no full proof method for invigilation of online mode exams. This project can be of great help during invigilation to find out if the student is involved in any kind of malpractice.
- Emotion recognition can play a major role in detection of thefts. The model can detect any type of suspicious behaviour by any person and thus can help the cops to keep them as a suspect and track the thief.
- This model identifies a total of 7 types of emotions namely angry, fearful, sad, surprised, disgusted, happy and neutral.

1.5: Objective of the work

- In this project we have taken images with various facial expressions and gestures in our dataset which will be used in the detection of the accurate emotion of the user.
- The main idea behind this project is to make the person expressive enough for us to detect their emotion and help them accordingly if needed.
- We will involve live videos in this project, where the user will do a certain task and the detector will observe their expressions and gestures at the moment.
- Depending on the exhibited emotions at that point of time, the detector will display the appropriate emotion of the user.
- The project will be of great help in the outside world as emotion recognition has played a vital role in the recent era.

1.6: Organization of the thesis

1. In section 2, we discus about the research papers we referred to before starting with the project. There are some algorithms mentioned which already exist, the core area of the project and some of the new algorithms we used to make our project a success.
2. In section3, is about system analysis we have listed some of the disadvantages in the existing systems.
3. In section4, we have given proper description about the modules that we have created in the project. The implementation and working have been described individually.
4. In section5, we have mentioned about the project and its details using tables and graphs.
5. The next and the very last section of our project has all the advantages, limitations and the future uses of this project.

1.7: Summary

The topic that we have chosen for our project exhibition 2 is “Emotion detection using image analysis”. In this project we have built a model that detects emotions of people based on their facial expressions. The project comprises of five modules and the coding is done in python language. The first step is to download a dataset that will be trained and tested for the emotion detection. After downloading the dataset, we first divide it into two sections that is train and test and the images are equally divided among these two sections into seven different parts which is our seven emotions.

After we have obtained our training and testing dataset, we start to pre-process the images in the training dataset by applying several filters and algorithms to obtain the desired feature extraction location on the image. Now we train our model in which the machine detects the key features of a human face and the emotion related with those features. Now once our model is trained and aware of the human expressions, we have to test our model using the testing dataset that we have and determine the accuracy of the model that we have built and obtain the result. The final output of the model is to open camera of the device and detect real time emotion of the person. Also, the project takes photos as input and can detect the emotion of the person based only on images if web camera is not present. The model for this project is trained by the help of 35,778 images and the accuracy of this project is around 87%. The accuracy can be increased very easily by getting a faster GPU that will complete the epochs faster. The accuracy of this project can reach a whooping 98% and the project we have created is a self-learning model as well.

(P.T.O)

2. LITERATURE REVIEW

2.1: Introduction

1. Human emotions can be used to determine what a person feels about a situation or at that particular situation he/she is being put through. It says a lot about the character of the person.
2. Emotions are often intertwined with mood, temperament, personality, disposition, creativity, and motivation.
3. Emotion detection also helps us provide additional security wherever needed by identifying the face and facial emotion, which can be used to guess his/her motive.
4. Human emotion detection can also help by performing human verification. Below are the papers referred that helped in making of this project.
5. Many research papers, articles and blogs were referred in the successful completion of this project. We would like to express our gratitude to all the persons involved directly and indirectly in the making of this project.

2.2: Core area of the project

Our project revolves around the idea of making such a model that predicts the emotion of a person in real-time as well as on saved image and video. The project basically throws light on machine learning and image analytics algorithms used and implemented by the help of Python 3 programming. In the hardware section the requirement of the laptop is a NVIDIA graphic card and some updated modules inside the NVIDIA graphic card for smooth running of the trained model. The model uses the web camera of the device in order to depict the emotion of the person. A total of 4 modules is used in order to make this project. The core area of project focuses on libraries and algorithms involved in ML and Image analysis.

2.3: Existing Algorithms

2.3.1 Algorithm1

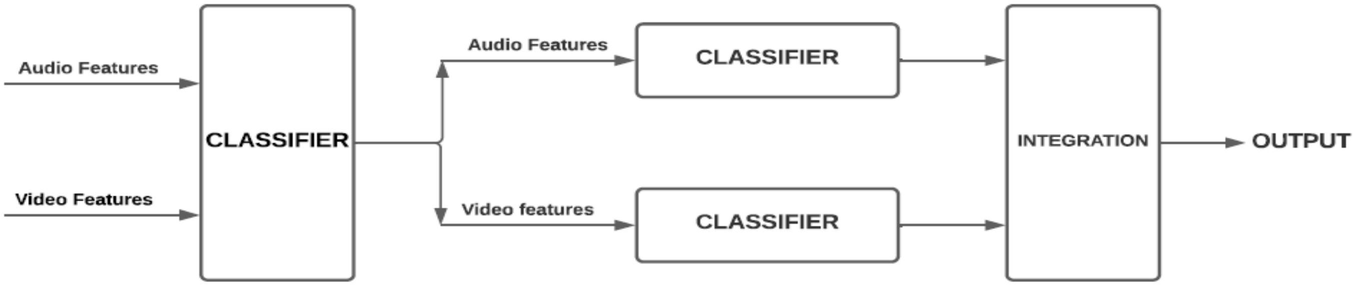


Fig. 2.3.1 ALGORITHM 1

2.3.2 Algorithm 2

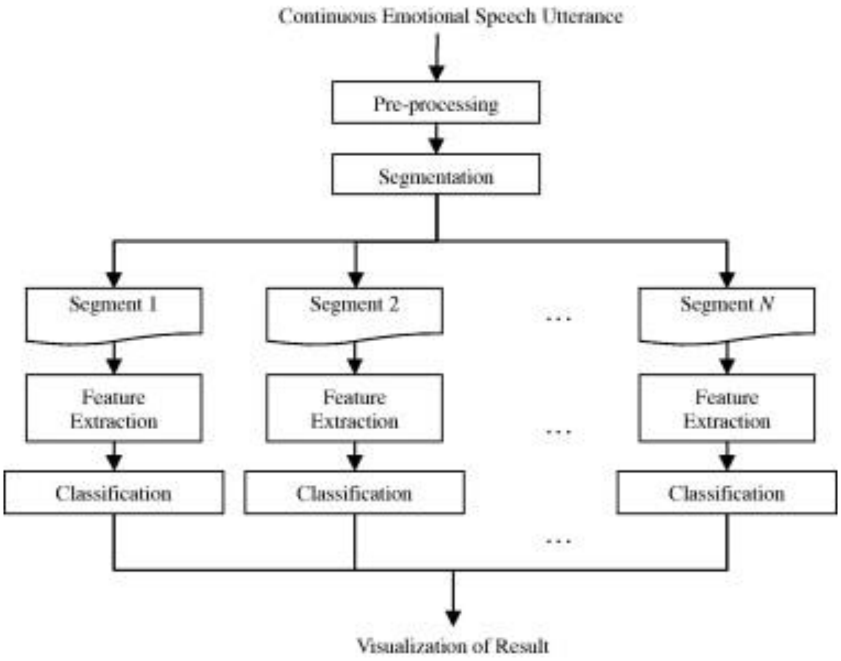


Fig. 2.3.2 ALGORITHM 2

2.3.3 Algorithm 3

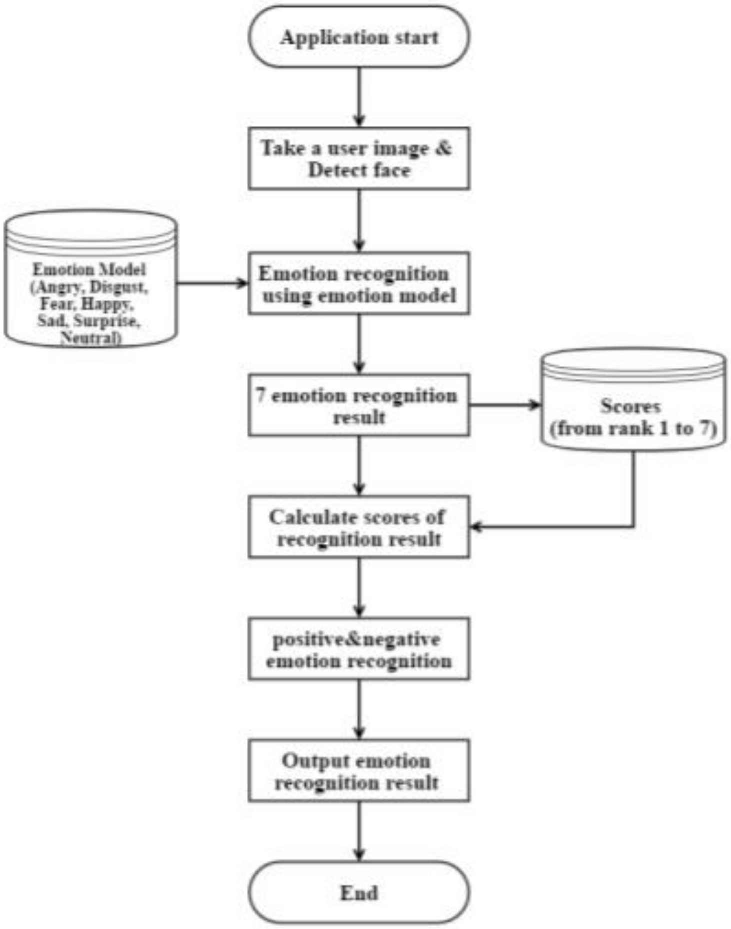

Application start
Take a user image & Detect face
Emotion Model (Angry, Disgust, Fear, Happy, Sad, Surprise, Neutral)
Emotion recognition using emotion model
7 emotion recognition result
Scores (from rank 1 to 7)
Calculate scores of recognition result
positive&negative emotion recognition
Output emotion recognition result
End

Fig. 2.3.3 ALGORITHM 3

2.3.4 Algorithm 4

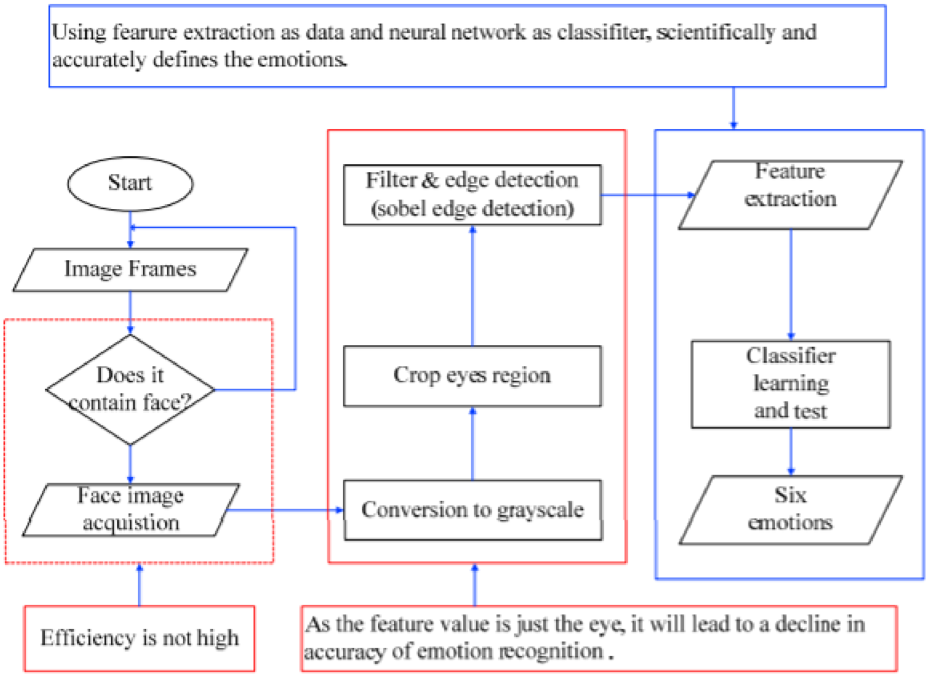

Using fearure extraction as data and neural network as classifiter, scientifically and accurately defines the emotions.
Start
Image Frames
Does it contain face?
Face image acquistion
Filter & edge detection (sobel edge detection)
Crop eyes region
Conversion to grayscale
Feature extraction
Classifier learning and test
Six emotions
Efficiency is not high
As the feature value is just the eye, it will lead to a decline in accuracy of emotion recognition .

Fig. 2.3.4 ALGORITHM 4

2.3.5 Algorithm 5

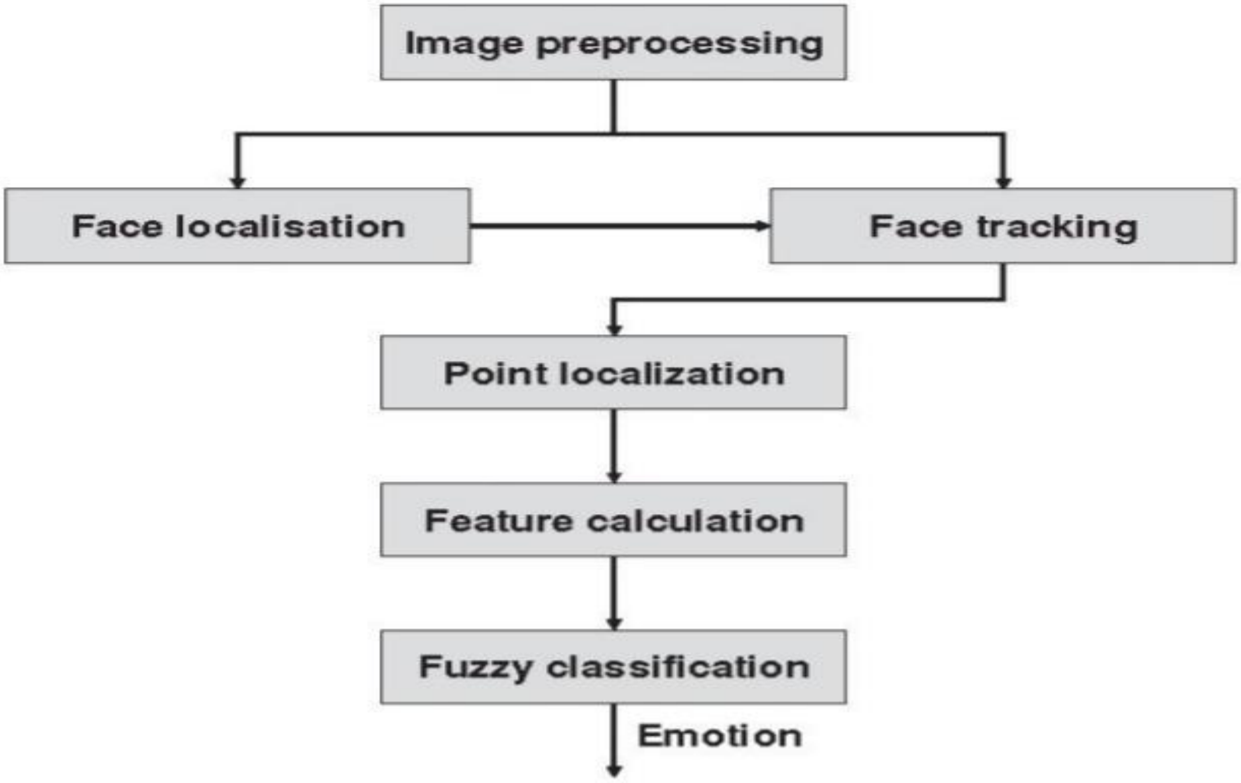


Fig. 2.3.5 ALGORITHM 5

2.5 Any other method used in the project:

Nvidia Developer toolkit and Nvidia Graphic card is required to make the project work efficiently. The project also uses different modules in python that aid us in analysing the data in order to analyse our dataset. Nvidia CUDA and Nvidia cuDNN developer toolkit is used in the making of this project. CUDA is a parallel computing platform and programming model developed by NVIDIA for general computing on graphical processing units (GPUs). With CUDA, developers are able to dramatically speed up computing applications by harnessing the power of GPUs. The NVIDIA CUDA Deep Neural Network library (cuDNN) is a GPU-accelerated library of primitives for deep neural networks. cuDNN provides highly tuned implementations for standard routines such as forward and backward convolution, pooling, normalization, and activation layers.

2.6: Observations from literature survey

- **“Goldman AI, Sripada CS. Simulationist models of face-based emotion recognition. Cognition. 2005 Jan;94(3):193-213. doi: 10.1016/j.cognition.2004.01.005. PMID: 15617671.”** In this paper the authors review a body of neuropsychological research that, we shall argue, supports simulation theory for a certain circumscribed mind reading task. The method for emotion detection used here is purely based on predefined emotions using simulations.
- **“R. Cowie et al., "Emotion recognition in human-computer interaction," in IEEE Signal Processing Magazine, vol. 18, no. 1, pp. 32-80, Jan 2001, doi: 10.1109/79.911197.”** This paper is based on the theoretical foundations of, and work carried out within, the collaborative EC project called ERMIS (for emotionally rich man–machine intelligent system), in which we have been involved recently. The authors of the paper intended to write a paper in order to make a system which could understand human emotions via voice, face or the choice of words made by them.
- **“Koolagudi, S.G., Rao, K.S. Emotion recognition from speech: a review. Int J Speech Technol 15, 99–117 (2012).”** The authors of the paper mention that, the recent literature on speech emotion recognition has been presented considering the issues related to emotional speech corpora, different types of speech features and models used for recognition of emotions from speech. The authors of the paper have listed down all the ways and means of identifying human emotion using only the voice of the user. They have distinguished and summarised all of the methods that could be used for emotion detection using voice.
- **“Hong-Wei Ng, Viet Dung Nguyen, Vassilios Vonikakis, and Stefan Winkler. 2015. Deep Learning for Emotion Recognition on Small Datasets using Transfer Learning. In Proceedings of the 2015 ACM on International Conference on Multimodal Interaction (ICMI '15).”** In this paper, the authors have used movie clips and cropped them to a full zoomed view of the face of the actor. The displayed emotions are then entitled and saved for the future emotion detection. This method of emotion detection may not be as accurate because the emotions shown in movies are usually an exaggeration of the actual emotions displayed by humans.

- **"Ioannou, S. V., Raouzaiou, A. T., Tzouvaras, V. A., Mailis, T. P., Karpouzis, K. C., & Kollias, S. D. (2005). Emotion recognition through facial expression analysis based on a neurofuzzy network.**" The authors of the paper have enlisted certain algorithms which identify the position of the eyebrows with respect to the rest of the face. Based on the position of the eyebrows, the emotion of the user will be determined. This method of identifying emotions will not be appropriate for those who have some facial dis-structure and for those who are not expressive as the system demands them to be.
- **"L. S. Chen, T. S. Huang, T. Miyasato and R. Nakatsu, "Multimodal human emotion/expression recognition,"** ***Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition*****, Nara, Japan, 1998.**" In this paper the authors have used all the methods- speech, images and video processing- for emotion detection. Pre-recorded or captured videos, audios and images will be saved as databases and used to match the ones which will be collected from the user at the point of examination. This method is more accurate as all the measures are taken into consideration while recognising the emotion of the person.
- **"Schmid, P.C., Schmid Mast, M. Mood effects on emotion recognition. Motiv Emot 34, 288–292 (2010)."** In this paper the authors have collected different types of happy, sad and angry expressions depending on the mood. The expressions of people based on the mood at that moment will determine the emotion being displayed by them. This method might be helpful for those having frequent mood swings or even to identify personality disorders of a person.
- **"N. Esau, E. Wetzel, L. Kleinjohann and B. Kleinjohann, "Real-Time Facial Expression Recognition Using a Fuzzy Emotion Model," 2007 IEEE International Fuzzy Systems Conference, London, UK."** This paper presents the fuzzy video- based emotion recognition system VISBER, that allows to analyse facial expressions in video sequences. The authors of the paper have basically made an algorithm which identifies the position of the facial elements such as eyes and lips, and accordingly will display the detected emotion. This algorithm will be limited to those who have an excellent ability to express themselves via facial expressions only.
- **"Young, Andrew W., et al. "Facial expression megamix: Tests of dimensional and category accounts of emotion recognition."** The authors of the papers have reported four experiments investigating the perception of photographic quality continua of interpolated ('morphed') facial expressions derived from prototypes of the 6 emotions. The expressions displayed by the user were captured and were tagged for further determinations. This method will be more accurate than the rest because the reactions shown by the will be instant and more natural unlike the others.

2.7 Summary of Literature review

After going through plenty of papers and analysing each algorithm used thoroughly, we, with the guidance of our internal guide, Dr. Nilamadhab Mishra, have successfully been able to develop this model which is efficient and has scope in the future too. Using many learning resources and research papers by many eminent scholars we have designed our algorithms and models and thus the emotion detecting system is efficient, accurate and easy to implement and affordable to use. A number of research papers were referred and are cited below in the report. This project has ensured to provide a maximum accuracy of 98% and this is possible by the aid of python libraries for ML and Image analysis as well as Nvidia Developer toolkit.

3. SYSTEM ANALYSIS

3. INTRODUCTION

This part of the report gives a brief idea about the analysis of the system. We analyze the currently existing systems in emotion detection and then propose our new system that will give a work on increasing the accuracy as well as have many new features in this. We check the disadvantages that are present in the existing system and we observe that the existing system can not give the user an option to chose between the type of input he/she would like to give. Thus, we propose a model that allows the user to choose between the type of input he/she would like to give and thus the user can choose from live a real-time input through web camera else a saved input in the form of an image/video format. Further the proposed system is explained in detail throwing light on the minute prospects that helped us in making our model better and more accurate as compared to the pre-existing models

3.1 DISADVANTAGES/LIMITATIONS IN EXISTING SYSTEM

The emotion recognition system that we have now-a-days is just a model which has been trained and tested using a couple of thousands or lakhs of images which is nowhere in comparison with a human brain and emotions. What a machine sees are just a couple of images but human emotions are more than that. One cannot completely rely on this system to detect the emotions even though the model has a lot of confidence and accuracy. Using these models can definitely help you in some ways and for some particular implementations. For example, using this model for detecting people's emotion during a product launch or for marketing to a specific set of customers can be useful because these people are not emotionally connected with the product and thus we can get analysis for accurate set of data but when it comes to implementations in other fields then it is not assured that we will get what we want. The AI Now institute conducted a research and concluded that there is no scientific consensus which can provide a base to the theories of these emotion recognition models.

Also, another problem that is faced is that our model detects facial expressions and it is not necessary in every case that your facial expression matches with the emotions that you express as said by few experts. This technology can prove to be very dangerous, many of the lie detectors use this technology and during the testing of these machines all the answers were not considered to be effective. Most of the emotions are context based like a baby cries during its birth that doesn't mean he is sad, a footballer puffing his chest and shouting doesn't mean he is angry and inviting you for a fight it can mean that he is celebrating for the victory. Thus, this means that detecting emotions in such case is almost very difficult. To conclude, not even the humans are completely capable of detecting emotions so it is a very long way for a machine to do that. Also, the models we have implements all the forms of input at the same time. We can give real-time input by webcam, image input and video format input for emotion recognition.

3.1.1 Disadvantages of Existing System: -

- The current emotion detection system is not completely capable of performing the task even though the model has high accuracy and confidence.
- These models can be used for specific implementations like product launch and for marketing to a specific crowd.
- There is no scientific consensus or proof for these emotion detection models as it detects facial expressions and it is not necessary that emotions are recognized using facial expressions.
- These types of models are not apt for context-based emotions like crying when happy, etc.
- Thus, it is a long way for these types of models to be completely reliable and accurate.
- The existing models can have only one form of input while we give 3 different forms of input and the user can decide the type of input he will like to feed.

3.2 PROPOSED WORK AND ITS ADVANTAGES

We have proposed a model that can identify the emotions of a person via live feed, an image or a video. We aim to create a model than will help users to identify what the reaction of a person is to a particular product or service. Such a model can be used in several industries in today's world. Especially now, due to the pandemic situation, most of the business interactions are taking place on an online platform. Our model can be integrated in these online platforms or a video or photos of these interactions can be fed into our model to produce an output that tells us about the emotions of the audience. Industries like the marketing industry, gaming industry, teaching industry, security industry, social media industry and many more can make use of our model to simplify certain tasks in their daily business interactions. They can be used to record the feedback of their consumers and use the data to improve their service or product according to the same. In the security industry we can use it to classify whether a person looks like he can pose a threat or not. In this way our model can prove to be really useful in several industries to improve their products and services. The model can take 3 different forms of input and gives the user an option to choose the type of input he will like to implement.

In the first module, we perform processing on the dataset we have. We use the fer2013 dataset from Kaggle and preprocess the data in order to split the images in training and testing folder. We split the images based on the emotions and then we put it in their respective files. In the analysis module, we analyze the given dataset and find that the dataset is balanced. We plot the graphs in this module. In the third module we train our model to recognize the emotion of the person. In the LiveRecognition.py module, we code such that webcam opens up and then the emotion is recognized by creating a frame around the face. In ImageRecognition.py, we give the user an option to select from Image and Video input and write a menu-driven code in order to allow the user to choose. The user chooses the type of input and enters the path and the model perform emotion recognition on it.

3.2.1 Advantages of Proposed System

- This model can be used by various businesses to process images and videos in real-time which can be used for monitoring video feeds or automating video analytics.
- With the help of this model the data obtained will be more accurate as it has been trained on a huge dataset. The model accuracy can reach a maximum of 98%.
- This will be useful when the audience to be examined is large and people manually can't take note of everyone's emotions.
- This will also help to save time as manual work is not required and feedback can be recorded instantly.
- With the help of this model the companies can frequently take feedbacks and consistently provide upgrades to their users.
- Also, our model runs on real-time input as well as on saved inputs. The user can choose whether he wants to input image/video or run a real-time webcam to perform emotion recognition.
- The model that we develop is a self-learning model and also it is very easy to achieve maximum accuracy in the model. The dataset provided consists of 35000+ images and thus gives very accurate results in terms of emotions of a person.

3.3 CONCLUSION

We can thus conclude from the above proposed system that we see a betterment in the actual existing system in many ways. We first develop a self-learning model in order to make the emotion detection model better than others. The dataset used in this is highly advanced containing many images of different types and we have trained our model using the best machine learning and Image analysis methods possible which allows us to achieve an accuracy of 98%.

4. SYSTEM IMPLEMENTATION

4.1 Preprocessing.py MODULE: -

Preprocessing is the basic and one of the most essential modules out of the many modules that is used in the making of a smart traffic control system. It is the module that has to be first executed for the whole project to run. Preprocessing is one of the basic and the most useful modules in the working of the project. This module ensures the pre-processing of the dataset which is used for training and testing of the data. It involves separating the testing as vell as the training data from the dataset and thus is of great help in model training. It separates data in different libraries. Preprocessing module involves a total of 5 libraries that are imported from the vast variety of libraries in python. We import numpy, pandas, Image, os and tqdm library for better result and accuracy of the emotion recognition model created in this project.

NumPy stands for Numerical Python. The main use of the numpy library in python programming is to work with arrays. It also has functions of linear algebra, fourier transform, and matrices. This library in python provides a high-performance multidimensional array and basic tools to compute with and manipulate these arrays. Pandas module is imported in this module for data analysis and interpretation of data. This helps us get a better understanding of the data that we are using. The dataset used for training and for testing is downloaded from Kaggle

PIL is the abbreviation of Python Imaging Library. Python Imaging Library is a free and a open source library available for all types of users on all types of operating systems that helps us perform operations on images such as reading and writing of the images. We can open, manipulate, save images in different forms. The OS library in python is imported so that the program can have a direct interaction with the operating system The OS module in python provides functions for interacting with the operating system. OS, comes under Python's standard utility modules. tqdm is a Python library that allows you to output a smart progress bar by wrapping around any iterable. A tqdm progress bar not only shows you how much time has elapsed, but also shows the estimated time remaining for the iterable. The dataset is divided into training and testing and based on the usage column we divide the dataset into training and testing by the help of for loop.

The module defines a function named atoi () and reads the fer2013.csv file which is the database used in this project. We have a total of 7 emotions that are to be detected by the model and those are the 7 basic emotions namely angry, sad, disgusted, fearful, happy, surprised and neutral.

WORKING/IMPLEMENTATION

At the start of the module, we define a function that returns n which is a variable storing the value of n*10 + ord(i) – ord (“0”). Next we read the dataset of fer2013 that we have downloaded from Kaggle.com using pandas. Next we let a for loop iterate over the emotions column in the database and list down all the emotions and store them in a list named labels. The same thing happens with the usage column and then we print both labels and usage in. Next we start segregating the images in the dataset. We iterate a for loop through the dataset having 3 variable i, j, k each in charge of “emotions”, “pixels” and “usage” column in the dataset. We first check if the usage of the image is training. If the usage is training we check the type of emotion that the image is of. For example, if i=0 which means emotion is angry and the same image is stored in the angry file in the training folder. The same happens for all types of emotion and at the end of this loop we have a folder named “training” consisting of 7 different files each having photos of its respective type of emotion.

The same thing is performed for testing images as well. We run a for loop and segregate the images having usage as testing and 7 files are created in the testing folder as well consisting of all the images in the respective type of emotion. Next we initialize the variable in order to count the number of images of each type of emotion in both training as well as in testing folder. We continue saving the images in their respective folder based on their usage specified in the usage column in the dataset.

4.2 Analysis.py MODULE

The analysis module is a module that is used to analyze the dataset that we have in different ways to get a better understanding of the data that is used to test and train the model. Also, from this we come to know that we have a balanced dataset and thus the features are applied in that way. It is an important step to analyze the data and involves many graphs and plots and emotion recognitions. A total of 4 libraries are required to be present and downloaded on the computer on which the code is executing so that the user can implement the code successfully. The modules that are imported are os, matplotlib. Warnings, seaborn. The OS library in python is imported so that the program can have a direct interaction with the operating system The OS module in python provides functions for interacting with the operating system. OS, comes under Python's standard utility modules. The matplotlib library in python is an amazing visualization library for 2D plots of arrays. It is a library that is used to work around images and graphs and is used to make different types of graphs and thus is of great help. In our code we have imported pyplot and image module from the matplotlib library.

The warnings module is imported in this module which is provided to warn the developer of situations. A warning is distinct from an error where we see that warning is not critical. The warning module is a subclass of Exception which is built in class in python. Next, we import the seaborn library which is a python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics. In this module we sketch only the scatter graph also known as the line graph of both the training images as well as the test images. We analyze the data that we convert into integers. The directories used in this module are highly related to the directories that were created in the processing module. Thus, for correct implementation of the code, we recommend to compile and run the processing module first and after that we can rub the analysis module.

WORKING/IMPLEMENTATION

Firstly, we import all the libraries that we need for this module to run smoothly. We import os, matplotlib, warnings and seaborn modules in our module. The filterwarnings function in the warning module is used to add an entry into the specifications of the warnings filter. The message which has warning as ignore is not displayed here in the output. Next, we assign the colour to the plots using matplotlib colour codes using the sns.set() function in the seaborn module. It assigns the colours to the plot that is to be created further in the class. We set the path for the training directory that we created in processing module in directory_training variable and dir_training variable contain a list of the names of entries in the directory given by the path using listdir() function of os module. We append the same folder names in the classes[] list using for loop. Next, we initialize a count variable to keep a track of the total number of emotions for which the plots are made. Path_of_class contains the path of the training directory. The incremented value of I is appended to the train_counts list that is made.

Next, we make a scatter plot of the count value in train_counts list by the help of plt.scatter() function. A variable named path_of_images is created as a list and next in a nested for loop we find the path_of_class by adding the directory_training and the folder path. We next check if the final_path which is path_of_class + file is an image or not. If it is an image, we append its path to the path_of_images list else we break it and display all path of images. Next, we read the image from the path_of_images using a for loop and we plot each and every image by the help of plt.figure(). We compute the pos1 and pos2 using the file.find() function by entering ‘/’ and ‘0’ as its arguments and then add the two values into the title variable, that are pos1 + 1:pos2 respectively, and plot the image in the title variable.

The same procedure is repeated again for the testing directory where we check the path_of_class again by the help of a for loop. The dir_test contains the list of all the folders present in the test directory created in the preprocessing module. We then plot the scatter graph for test_counts which is the same procedure as above which stores the count variable in the nested for loop and we print all the test_counts. Next, we check if the final_path ends with .png or not. If it ends with .png then we add the path of the image to path_of_images list that is created else we break the loop. At the end we set the title of the graph and the font size to 20 and then we proceed towards printing and computing the graph using matplotlib library.

This module shows that out training as well as test data is completely balanced and also we see that majority of the images are used for training and very less are used for testing. We recommend running the Preprocessing module first for proper and accurate results because the directory and the images added in the Preprocessing module is used in this module for analysis. We display all the graphs at the end of this module as the output. Also one of the image of all types of emotions in both training and testing us displayed in the output. We find out the=at we have a balanced dataset as well.

4.3 model.py MODULE

In this module we have trained and created a model “model1.h5” and we have also created a graph “plot.png” for accuracy and loss to get an idea about overfitting and underfitting of our model. The libraries of python that we have imported here are os, numpy, matplotlib and various instances from tensorflow. The os module is basically used for interacting with the operating system, numpy module is used for working and operating on arrays like for our extraction values of csv file which is treated as array and the matplotlib is used for plotting the graph of accuracy and loss. We have imported various methods from tensorflow module like Conv2D, Dense, Dropout, Flatten, MaxPooling2D from tensorflow.keras.layers, Sequential from tensorflow.keras.models, Adam from tensorflow.keras.optimizers and ImageDataGenerator from tensorflow.keras.preprocessing.image and ConfigProto and InteractiveSession from tensorflow.compat.v1. Tensorflow keras is a high-level interface which works on both CPU and GPU. It basically applies neural network calculations like convolution, Pooling, embedding and many other network interfaces. We have a function named plot_model_history which takes model_history as an argument and we apply all the methods imported from keras in this function. In this function we first define the plt.subplots method which takes row, colums and the index of the current plot as arguments. The next plot is to define the x and y axes using the plot() method. In the plot method we set the range for x and y axes using the range() method.

Now we set the labels for x and y axes using the set_xlabel() and set_ylabel() method and the title of the graph using set_title() method. Now we set the ticks or marks for our graphs using the set_ticks() method. Now after we have provided the data we want to rescale the images using ImageDataGenerator() method. Now we use the Sequential() model of Keras which divides our image into a plain stack of layers which has exactly one input and output tensor. The next method that we use is Conv2D which is used to convolve the stack layer and give the output tensor. Next we use the MaxPooling2D which takes pool_size, strides, padding and data_format as arguments.

We use the Dropout method of keras which is used to prevent overfitting and it takes rate, noise_shape and seed as arguments. The flatten method is used to flatten the input and it takes data_format as argument. Now we just train our model using the fit method and the model is saved in the specified folder.

WORKING/IMPLEMENTATION

The first step is to import all the libraries like numpy, os, matplotlib and tensorflow and then we set our environment variables to start with the execution. Now we define our main function which is the plot_model_history and provide it with the argument plot_history. Now we start setting our graphs by using the subplots method which has 1 row, 2 columns and the length of our graph is 15x5 cm.

We start plotting the graph with the plot() method which sets the range of axes from 1 to the length of the accuracy of model_history model that will be trained. Now we set the title of the graph as “Model Accuracy” using set_title() method and the label for x and y axes is set to “Accuracy” and “Epoch” using the set_xtitle() and set_ytitle() methods respectively. Now we create a variable named npdata which contains the value of range of the tick marks that have to be drawn at certain intervals on x and y axes using the arange() method. This variable is then passed as parameter to the set_ticks() method. We now set the area of the graph where our axes will be plotted using the legend() method and the value is set to best which is the upper left corner. We follow similar steps to plot the graph for Loss model and the figure is then saved at the specified location as .png file using the savefig() method and the image is shown using the plt.show() method.

We now set the training and validation directory to the variables training_directory and val_dir respectively. We also set the number of images that have to be used for training and testing purposes which is 28709 images for testing and 7178 images for validation. We also set the number of times that our model will be trained which is nothing but the epoch size which is 50 and each batch contains 64 images.

Now our images are resized to 48x48 size, converted to grayscale and the images are converted to categorical values for both testing and validation set. Now once we have filtered the images we have to extract features from it so as to train it. We use the Conv2D method to convolve the image and the parameters passed are 32, kernel_size=(3, 3), activation=**'relu'**, input_shape=(48, 48, 1). Next we use the MaxPool2D which takes a 2x2 array as a parameter and the model is saved from overfitting by using the Dropout() method and the rate is 0.25. Finally, we flatten our model using the Flatten() method. Similarly, we follow the same steps for validation set.

Now we have to compile our model using the model.compile() method and here we use our optimizer Adam which gives us the loss and accuracy percentage for each epoch.
Now it’s time to train our model by using the model.fit() method and the training values are stored in the variable model_info. This variable is then called by our function plot_model_history to plot the graph and the trained model is saved in “model1.h5” file at the specified location using save_weights() method. This model can be further implemented in the code using load_weights() method. This is the basic working of the model.py module.

4.4 LiveRecognition.py MODULE

The module is the utmost important module in the whole project which displays the outcome and the output of the project. In this module, we detect the emotion of the person in real-time, which means that the web camera of the person is turned on and the emotion of the person sitting in front of the web camera is detected by the help of all the modules described above and by the help of training given to the model. In this module we first import all the modules required for the functioning of the module. We import os, cv2, numpy and different libraries from tensorflow which will act as filter to our images.

We also initialize the epoch_num to 50. Remember that the more is the number of epochs, the more is the accuracy of the model. But each epoch takes about 5 min to get executed and thus the accuracy of our model comes out to be 87% which can be increased upto a maximum of 98% by just merely increasing the epoch_number and getting a faster GPU. Also, we apply all the filters to the images by the use of tensorflow module and add filters like Convo2D, MaxPooling2D, Flatten and Dropout along with Dense().

WORKING

In this module, we import the basic libraries of python, which are used for machine learning and image analysis. They are os, cv2, NumPy, and different modules from TensorFlow. A variable known as config is assigned to ConfigProto() function, and the environment is set to 2 using os.environ()

We have then provided the path to the training and test directories in our system, where the images extracted from the dataset are stored. The names' Train_num', 'val_num', ' batch_size', and 'epoch_num' are initialized as the variables training size, testing size, batch size, and the number of epochs, respectively, and the values are set according to the training of the model and accuracy required.

We then use the ImageDataGenerator function from TensorFlow.Keras.preprocessing.image module that generates batches of tensor image data with real-time data augmentation. Furthermore, the given data will be looped over in batches. A model is created using the sequential() function, and the image analysis techniques are to be applied to it. Sequential is an easy way to build a model in Keras as we can build a model layer by layer. We filter the image coming from the web camera using various filters. We first convert our images to grayscale and apply Convolution, MaxPooling, dropout, dense, and flatten in order on our 2D images to extract specific features for each type of emotion from every image. We then load the weights from the previous checkpoint in our model using the function model.load_weights().

Convolution is applied by the help of Convo2D() in which the kernel size is set to (3,3). In MaxPooling2D the pool size is set to (2,2). Next we flatten the picture and smoothen all the edges in the image that is taken by the help of webcam. This is done by the help of Flatten(), Dense() and Dropout() filters that have been imported from the tensorflow module.

A variable known as emotion_dict is initialized with a dictionary of all the emotions that the model can detect. Each label is assigned an emotion according to the dataset. We iterate through every emotion until the emotion is detected. This module is used to capture live feed frame by frame using the web camera of the device and thus detect the person's emotion in every frame in real-time. We start the video capture procedure and read the live feed frame by frame. The 'haarcascade_frontalface_default.xml' file is used to detect the face after converting the image to grayscale. The expand_dims function from the NumPy module is used to expand the shape of an array. In each frame, a rectangle is drawn around the detected face. The emotion detected is displayed in the text above the rectangle of the face recognized using the labels that we have assigned earlier. The output is shown in the video that shows the live webcam feed with the faces detected and the recognized emotion displayed above it. We press the 'q' key to exit the live feed window. We use the cv2 module to release and destroy all the window objects created in the making and working of this module, making space for the next module to work.

4.5 ImageRecognition.py MODULE

In this module ImageRecognition.py we are basically applying emotion detection on the input image and the input video. The libraries that we use in this module are os, numpy, cv2. Conv2D, MaxPooling2D, Flatten, Dense and Dropout from tensorflow.keras.layers and Sequential from tensorflow.keras.models. The os library in python is used for communicating with the operating system and for OS related functions and operations. The numpy is a mathematical library which is used for performing operations on mathematical arrays in python, in our module we use numpy operations on the array of emotions that we take and the extracted values. Cv2 is a library which is used for operations on images, we use this library for converting images to grayscale and for printing images. Tensorflow keras is a high level interface which works on both CPU and GPU. It basically applies neural network calculations like convolution, Pooling, embedding and many other network interfaces. The next method that we use is Conv2D which is used to convolve the stack layer and give the output tensor. First we use the MaxPooling2D which takes pool_size, strides, padding and data_format as arguments.

We use the Dropout method of keras which is used to prevent overfitting and it takes rate, noise_shape and seed as arguments. The flatten method is used to flatten the input and it takes data_format as argument. Now we have created a function named emotion which includes the load_weights() method to load the trained model. Now we use the haarcascade_frontalface_default.xml file in the method CascadeClassifier() and convert our images to grayscale first and then scale the images using detectMultiScale() method of cv2. Now we apply the rectangle() method to change the shape of the input image to a rectangle of given dimensions. Now we apply the predict() method to predict the emotion that the input image has and then we print the emotion on the top of image using putText() method with the dimensions as the parameter and return the final output. Now we ask the user whether he wants to process an image or a video and taking the input of the user we start the specific operation. Now if the user wants to process an image then we simply use our emotion function that we defined and give the output as discussed above. If the user inputs a video for processing then first we take the path of the video from the user using the VideoCapture() method and then process the video for each frame using the emotion function that we defined and finally we write the emotions on the video using the VideoWriter() method of cv2 and return the processed video with emotions detected in it.

WORKING

We start by importing all the necessary modules which include os, numpy, cv2, tensorflow.keras.models and tensorflow.keras.layers. then we set the os environment for our module and then we use the model as Sequential of Keras. Sequential divides the image into different layers and then performs operations on it. We use the Conv2D method to convolve the image and the parameters passed are 32, kernel_size=(3, 3), activation='**relu**', input_shape=(48, 48, 1). Next we use the MaxPool2D which takes a 2x2 array as a parameter and the model is saved from overfitting by using the Dropout() method and the rate is 0.25. Finally, we flatten our model using the Flatten() method. Now we define our function emotion which takes the frame as parameter. Frame is nothing but the image that we provide. In this function we first load our trained model “model1.h5” which we trained in the model.py module by using load_weights() method. Now we create a dictionary in which we associate each emotion with a number like 0 for angry, 1 for disgusted, 2 for fearful, 3 for happy, 4 for neutral, 5 for sad and 6 for surprised. We assign a variable facecasc for cv2 method CascadeClassifier and pass the haarcascade_frontalface_default.xml file as a parameter.

Now we convert the image to grayscale and scale the grayscale image with a scaling factor of 1.3 and minimum neighbours as 5. We apply a for loop now with x, y, w and h as parameters. We now convert our image shape to rectangle by using rectangle() method and it takes certain parameters which include image which is frame, start point which is (x, y-50), end point which is (x + w, y + h + 10), colour which is set to (255, 0, 255) and the thickness is 3. Now this cropped image is model for prediction purposes using predict() method and we put the emotion as a text on the top of the image using putText() method.

This function finally returns the output which is the processed image with emotion detected on top of it. Now we take input from the user, if user enters I then we will process for an image and if the user enters V then video is processed. Now if the user enters I then we read the image from imread() method of cv2 and this image is passed as a parameter to the emotion function and we receive the output. If the user inputs V then we read the input video of the user using VideoCapture() method of cv2. Now we take each frame of the video and then detect emotion for it using the emotion function where one frame is passed as a parameter of the video. Then we write the emotions on the video using the VideoWriter() method and this processed video is saved as output.avi in the given path location. Thus, we receive the final outputs as per the inputs we provide.

(P.T.O)

5. PERFORMANCE ANALYSIS

5.1 INTRODUCTION

Performance analysis is the technique of studying or comparing the performance of a specific situation in contrast to the aim and yet executed. We will discuss about the efficiency of our emotion recognition system in this section. Our project is capable of detecting a total of 7 different types of emotions on a human face. These emotions are: -

- Happy
- Sad
- Fearful
- Disgusted
- Surprised
- Neutral
- Angry

The project has a total accuracy of 87% when we run the project for a total of 50 epochs. This area of the project is further discussed below in the report.

5.2 PERFORMANCE MEASURES

The performance measure is the process of collecting and analyzing data, thus giving an appropriate measure or count of how efficient a system is. In our case, the system detects and differentiates between 7 different types of emotions as stated above. The model detects the emotions in both real-time as well as by the help of saved image/video format stored in the device on which the project is running. The performance of the whole model depends on the number of times we run the epoch. The more the number of times we run the epoch; more is the accuracy of the model that we get. We just face one drawback that each epoch takes a lot of time to run and thus the task can become very tedious. We have run a total of 50 epochs and calculated the accuracy of the model. Also, we plot a graph between model accuracy and epochs and the other graph between model loss and epoch. From these graphs we can say that model accuracy is directly proportional to number of epochs and model loss is inversely proportional to number of epochs.

5.3 PERFORMANCE ANALYSIS

There are a several factors that affect performance analysis. The factor that plays a major role in the performance of the model is the number of epochs the model.py module is allowed to run for. In our model we run the epochs till 50 which gives us an accuracy of 87%. The maximum accuracy possible in this project is 98%. Thus, we can conclude that 98% is the maximum accuracy to which the model can run and show as well as detect the emotion of the person both in real-time as well as in saved image/video format. Also, we can conquer that the model accuracy is directly proportional to number of epochs and the model loss is inversely proportional to number of epochs.

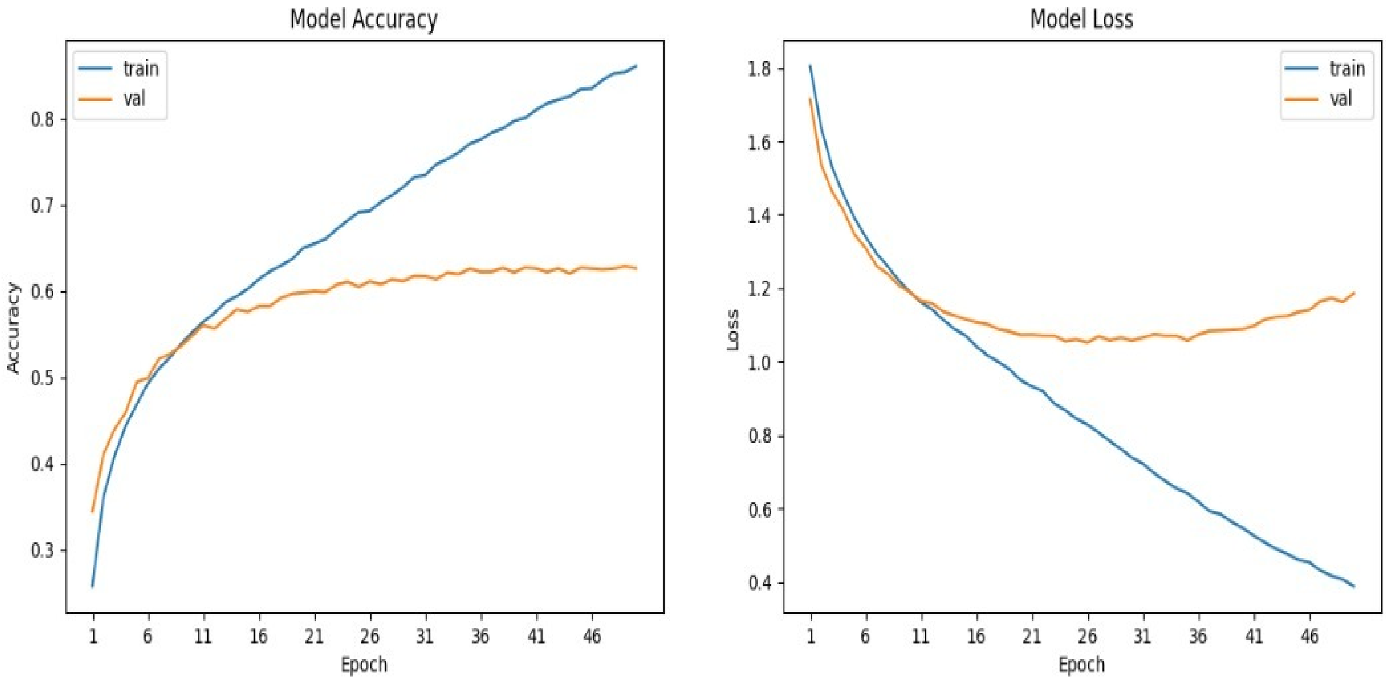
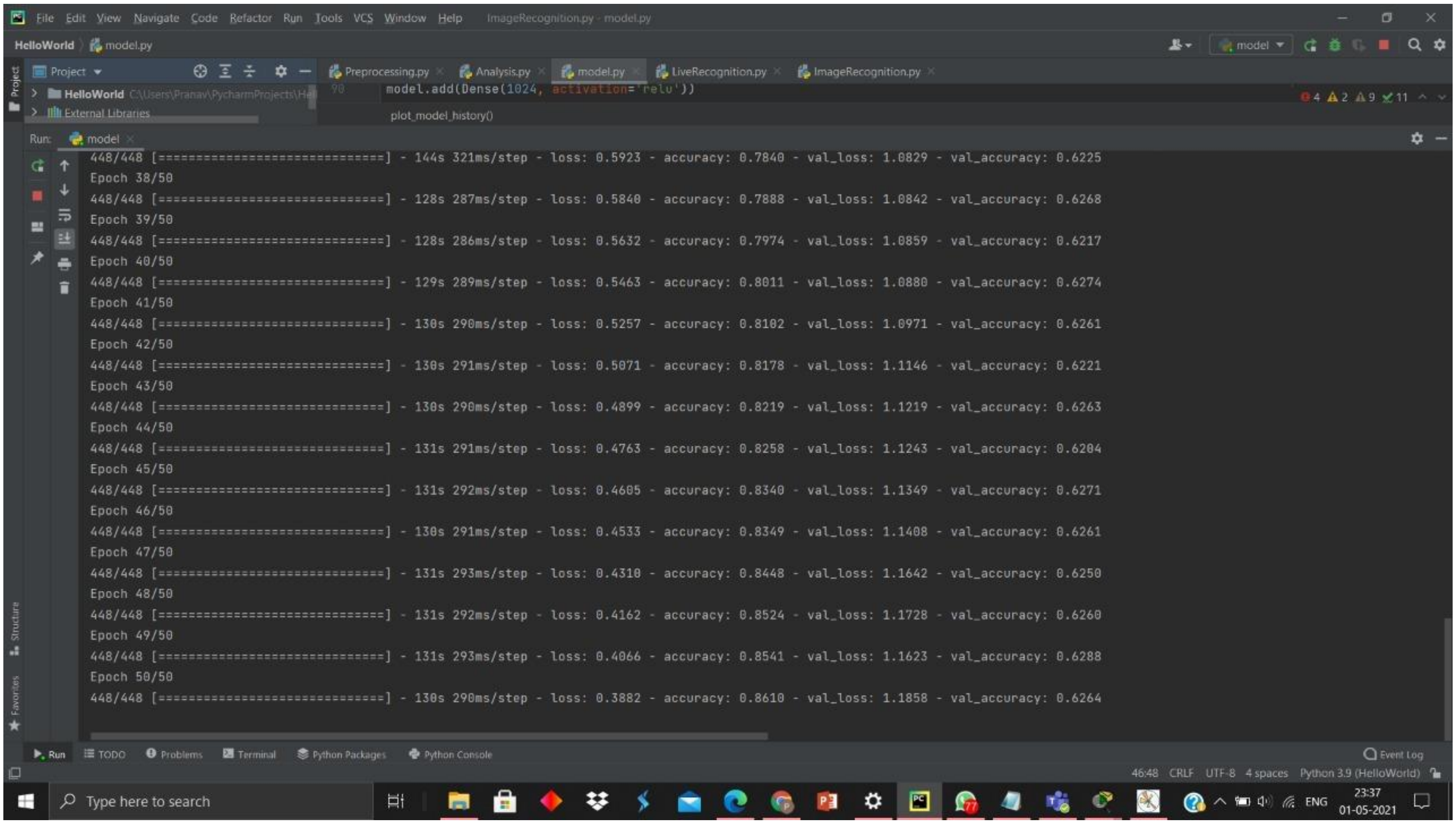


Fig. 5.3.1 GRAPH OF MODEL ACCURACY AND LOSS V/S NUMBER OF EPOCHS



```
model.add(Dense(1024, activation='relu'))
plot_model_history()
```

```
448/448 [==============================] - 144s 321ms/step - loss: 0.5923 - accuracy: 0.7840 - val_loss: 1.0829 - val_accuracy: 0.6225
Epoch 38/50
448/448 [==============================] - 128s 287ms/step - loss: 0.5840 - accuracy: 0.7888 - val_loss: 1.0842 - val_accuracy: 0.6268
Epoch 39/50
448/448 [==============================] - 128s 286ms/step - loss: 0.5632 - accuracy: 0.7974 - val_loss: 1.0859 - val_accuracy: 0.6217
Epoch 40/50
448/448 [==============================] - 129s 289ms/step - loss: 0.5463 - accuracy: 0.8011 - val_loss: 1.0880 - val_accuracy: 0.6274
Epoch 41/50
448/448 [==============================] - 130s 290ms/step - loss: 0.5257 - accuracy: 0.8102 - val_loss: 1.0971 - val_accuracy: 0.6261
Epoch 42/50
448/448 [==============================] - 130s 291ms/step - loss: 0.5071 - accuracy: 0.8178 - val_loss: 1.1146 - val_accuracy: 0.6221
Epoch 43/50
448/448 [==============================] - 130s 290ms/step - loss: 0.4899 - accuracy: 0.8219 - val_loss: 1.1219 - val_accuracy: 0.6263
Epoch 44/50
448/448 [==============================] - 131s 291ms/step - loss: 0.4763 - accuracy: 0.8258 - val_loss: 1.1243 - val_accuracy: 0.6204
Epoch 45/50
448/448 [==============================] - 131s 292ms/step - loss: 0.4605 - accuracy: 0.8340 - val_loss: 1.1349 - val_accuracy: 0.6271
Epoch 46/50
448/448 [==============================] - 130s 291ms/step - loss: 0.4533 - accuracy: 0.8349 - val_loss: 1.1408 - val_accuracy: 0.6261
Epoch 47/50
448/448 [==============================] - 131s 293ms/step - loss: 0.4310 - accuracy: 0.8448 - val_loss: 1.1642 - val_accuracy: 0.6250
Epoch 48/50
448/448 [==============================] - 131s 292ms/step - loss: 0.4162 - accuracy: 0.8524 - val_loss: 1.1728 - val_accuracy: 0.6260
Epoch 49/50
448/448 [==============================] - 131s 293ms/step - loss: 0.4066 - accuracy: 0.8541 - val_loss: 1.1623 - val_accuracy: 0.6288
Epoch 50/50
448/448 [==============================] - 130s 290ms/step - loss: 0.3882 - accuracy: 0.8610 - val_loss: 1.1858 - val_accuracy: 0.6264
```

Fig. 5.3.2 INCREASE IN ACCURACY WITH NUMBER OF EPOCHS

5.4 CONCLUSION

The overall performance analysis suggests that the above system is ready for implementation on many applications in business development, security, building customer relations and also detecting the emotion of the model in real-time as well as when saved in the file on the device on which the program runs. Also, the accuracy is 87% but is increased to 98% with the increase in the number of epochs. But epoch require lots of time to run and thus the model training is slow but the output is predicted very quickly in milliseconds. Also, the model that we have made is a self-learning model and can learn on its own based on the precious outputs.

6. FUTURE ENHANCEMENTS

6.1 INTRODUCTION

Human emotions can be used to determine what a person thinks as they are brought on by neurophysiological changes associated with thoughts, feelings, behavioral responses, and a degree of pleasure or displeasure. While there is currently no scientific consensus on a definition, emotions are often intertwined with mood, temperament, personality, disposition, creativity, and motivation. Human emotion detection is enforced in several areas in today's advanced world for requiring extra security or information regarding people. It can also help us provide additional security wherever needed by identifying the face and facial emotion, which can be used to guess his/her motive. Human emotion detection can also help by performing human verification.

Human emotion detection can be of great use for the future generation because the coming generation relies more on technology than any other person or professional. So, developing a technology that is assisted by a professional who monitors the usage of the user, we can keep a check on the user and note down some significant observations if needed. The person using the device can also detect itself of having problems or disorders like multiple personality disorder. This device in the future will be of great help to all the psychologists to detect and verify if a person really needs medical help or not.

6.2 LIMITATIONS

The model we have created has an accuracy of approximately 87% which can be easily increased to around 98% just by increasing the number of epochs for which the model is allowed to be trained. Another one limitation is that the GPU used to train the model is slow and thus it takes time to train the model. We can also use a laptop with better configuration is order to speed up the training process of the model.

The pre-existing emotion detectors work purely on the texts or conversation of the person during the analysis. These emotion detectors use texts as their dataset and determine the emotions of people using predetermined terms. If these terms are displayed or found in the text of the user, the associate emotion is being stated. Users nowadays find this system difficult because people now not only tend to express themselves through texts but also through images. So having text, images and live videos as our dataset is more of help in today’s generation. Like all the other devices our device too will not show accurate results as the outputs will vary from person to person. Having all the previously mentioned algorithms combined with ours, the level of accuracy has definitely increased to a certain amount.

6.3 FUTURE ENHANCEMENTS

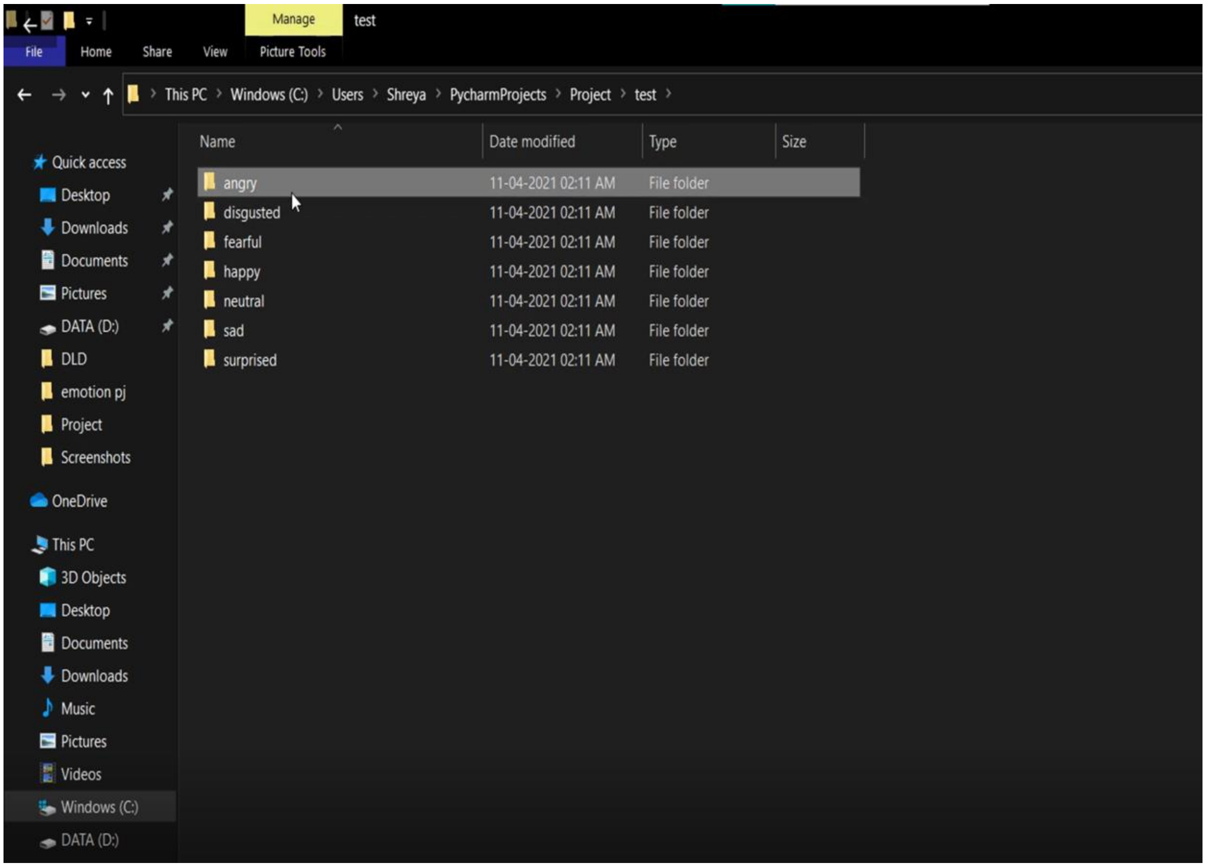
- The model we have created has an accuracy of approximately 87%. In this project, we have developed a model that takes live images and videos of people and describes their emotion accordingly. We have trained our model with multiple images and datasets expressing different emotions.
- In the future the project can be enhanced by clubbing the emotions together and tagging them under a certain disorder which will make the functioning of the device smooth and also increase the understanding of the user.
- Also, we can aim to detect the gestures of the human body with the emotions as well. Gestures also play a major role in shaping the personality of the person and a detection model of gestures along with the emotions of the person can speak a lot about the character of the person.
- We can also increase the accuracy of the model to 98% by just merely increasing the number of epochs but this takes huge amount of time. Thus, we can use a faster processor to speed this task up and thus increase its accuracy to whooping 98%

(P.T.O)

-

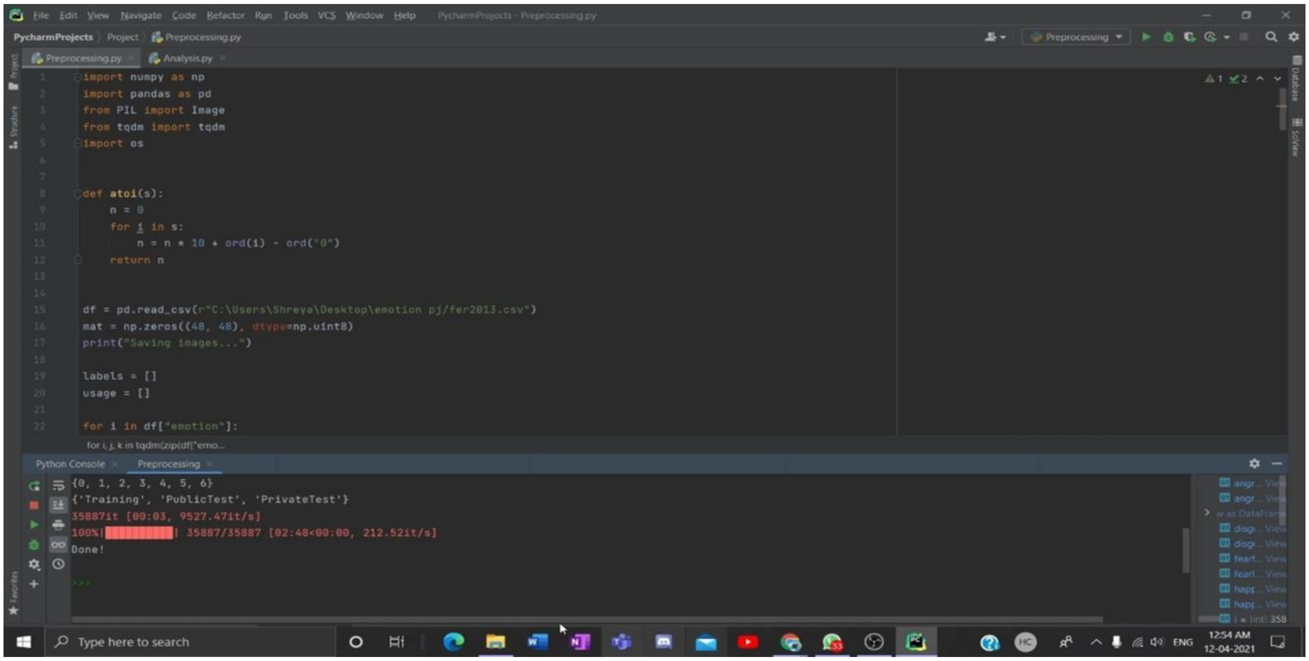
7. OUTPUT

i) Output of Preprocessing.py Module



Name	Date modified	Type	Size
angry	11-04-2021 02:11 AM	File folder	
disgusted	11-04-2021 02:11 AM	File folder	
fearful	11-04-2021 02:11 AM	File folder	
happy	11-04-2021 02:11 AM	File folder	
neutral	11-04-2021 02:11 AM	File folder	
sad	11-04-2021 02:11 AM	File folder	
surprised	11-04-2021 02:11 AM	File folder	

Fig 7.1.1: - Folders created in Preprocessing.py for testing (A similar folder is created for training images as well)



```
import numpy as np
import pandas as pd
from PIL import Image
from tqdm import tqdm
import os


def atoi(s):
    n = 0
    for i in s:
        n = n * 10 + ord(i) - ord("0")
    return n


df = pd.read_csv(r"C:\Users\Shreya\Desktop\emotion pj/fer2013.csv")
mat = np.zeros((48, 48), dtype=np.uint8)
print("Saving images...")

labels = []
usage = []

for i in df["emotion"]:
```

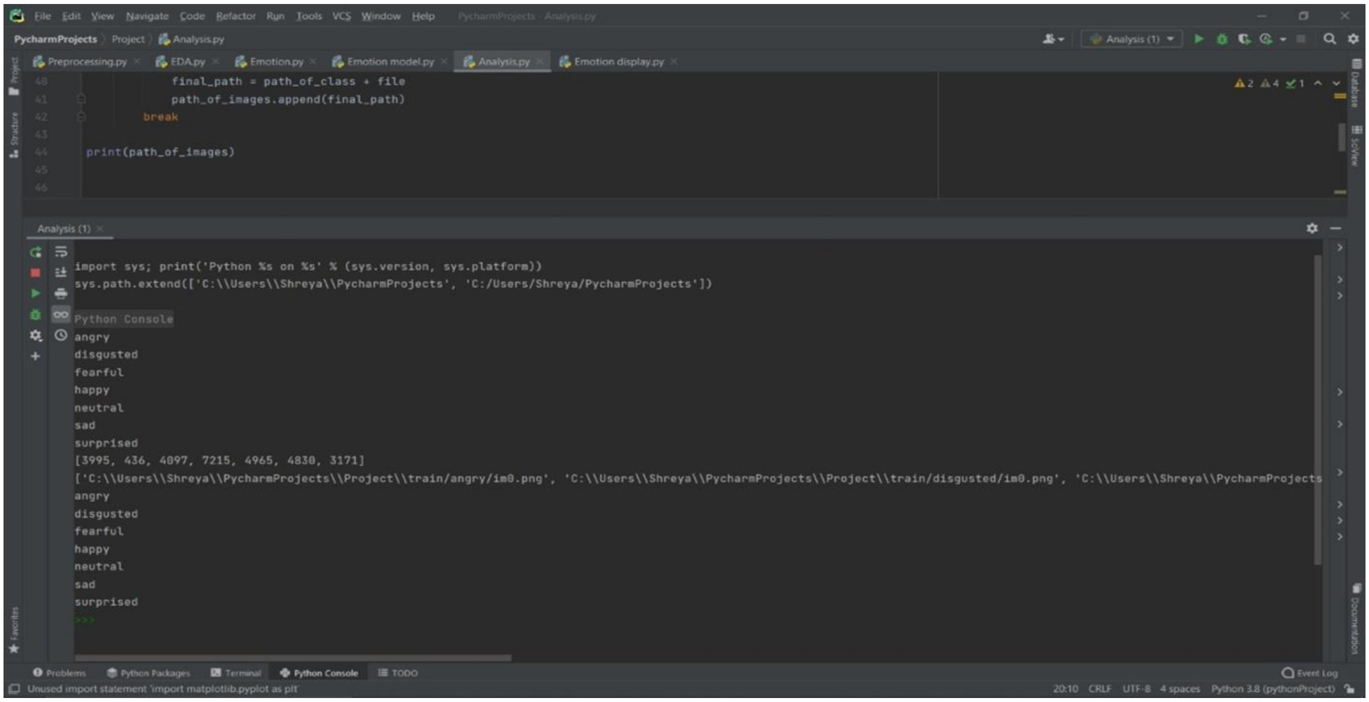
for i, j, k in tqdm(zip(df["emo...

```
{0, 1, 2, 3, 4, 5, 6}
{'Training', 'PublicTest', 'PrivateTest'}
35887it [00:03, 9527.47it/s]
100%|██████████| 35887/35887 [02:48<00:00, 212.52it/s]
Done!

>>>
```

Fig 7.1.2: - Terminal Window of Preprocessing.py

ii) Output of Analysis.py



```
                final_path = path_of_class + file
                path_of_images.append(final_path)
            break

print(path_of_images)
```

```
import sys; print('Python %s on %s' % (sys.version, sys.platform))
sys.path.extend(['C:\\Users\\Shreya\\PycharmProjects', 'C:/Users/Shreya/PycharmProjects'])

Python Console
angry
disgusted
fearful
happy
neutral
sad
surprised
[3995, 436, 4097, 7215, 4965, 4830, 3171]
['C:\\Users\\Shreya\\PycharmProjects\\Project\\train/angry/im0.png', 'C:\\Users\\Shreya\\PycharmProjects\\Project\\train/disgusted/im0.png', 'C:\\Users\\Shreya\\PycharmProjects
angry
disgusted
fearful
happy
neutral
sad
surprised
>>>
```

Unused import statement 'import matplotlib.pyplot as plt'

20:10 CRLF UTF-8 4 spaces Python 3.8 (pythonProject)

Fig 7.2.1: - Terminal Window of Analysis.py

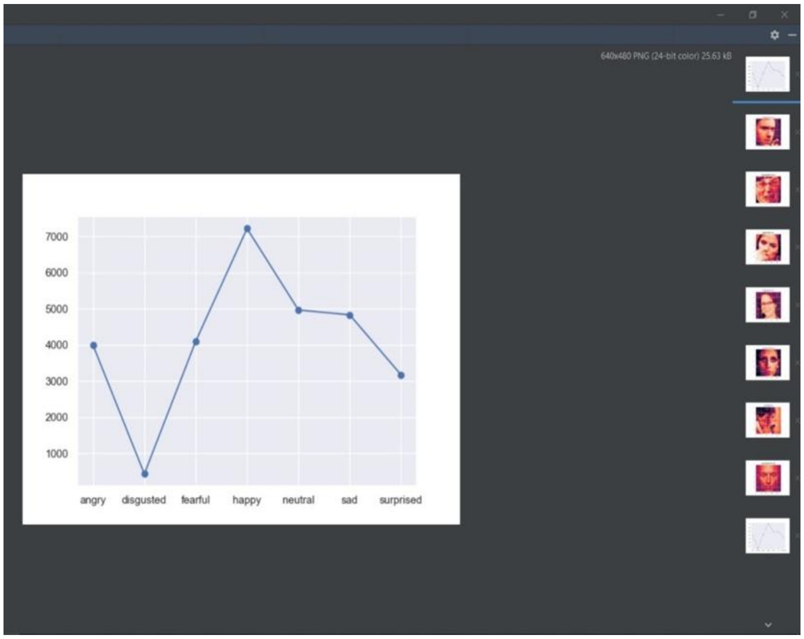


Fig 7.2.2: - Graph of training dataset

The graph alongside shows the plot between number of training images and the emotions. We can conquer that we have the maximum number of training images for happy (7215) and least for disgusted (436).

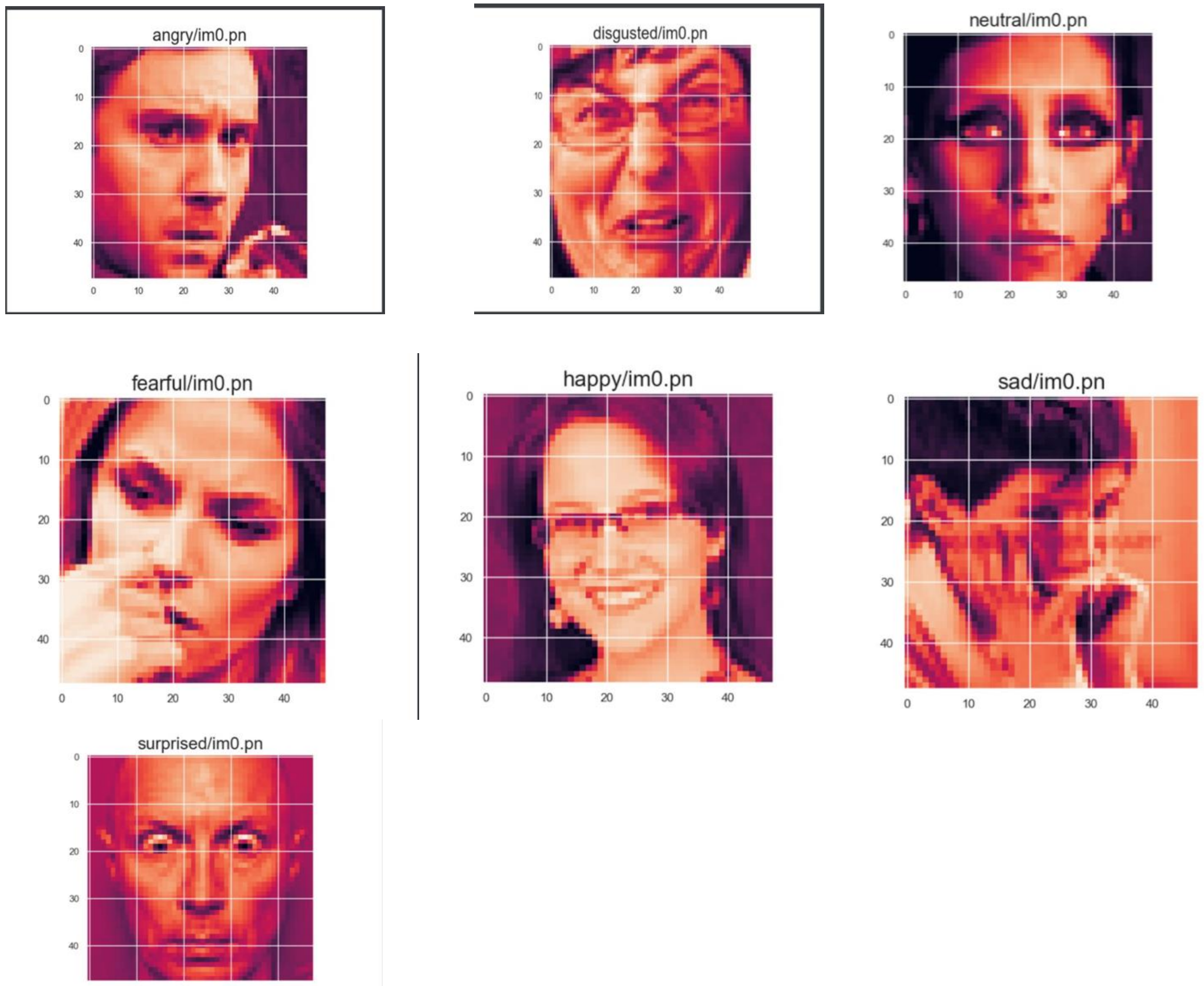
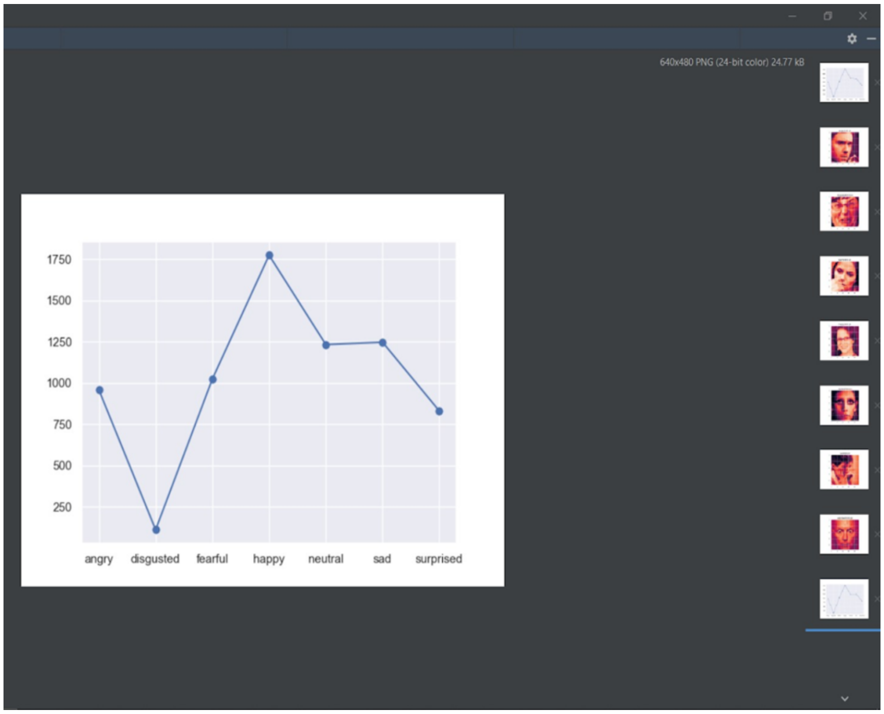


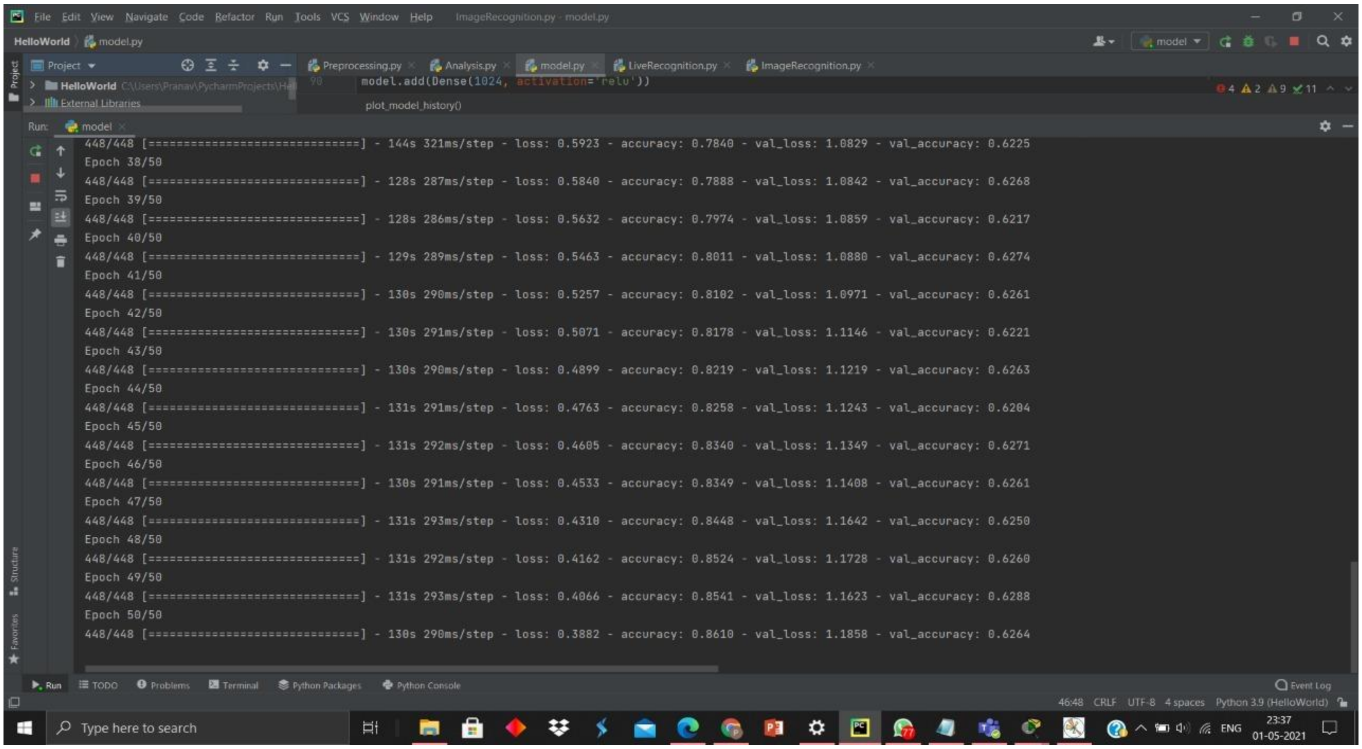
Fig 7.2.3: - OUTPUT IMAGES OF TRAINING AND TESTING FOLDER



The graph alongside shows the plot between number of testing images and the emotions. We can conquer that we have the maximum number of training images for happy (1750) and least for disgusted (111).

Fig 7.2.4: - Graph of testing dataset

iii) Output of model.py



```
model.add(Dense(1024, activation='relu'))

plot_model_history()
```

```
448/448 [==============================] - 144s 321ms/step - loss: 0.5923 - accuracy: 0.7840 - val_loss: 1.0829 - val_accuracy: 0.6225
Epoch 38/50
448/448 [==============================] - 128s 287ms/step - loss: 0.5840 - accuracy: 0.7888 - val_loss: 1.0842 - val_accuracy: 0.6268
Epoch 39/50
448/448 [==============================] - 128s 286ms/step - loss: 0.5632 - accuracy: 0.7974 - val_loss: 1.0859 - val_accuracy: 0.6217
Epoch 40/50
448/448 [==============================] - 129s 289ms/step - loss: 0.5463 - accuracy: 0.8011 - val_loss: 1.0880 - val_accuracy: 0.6274
Epoch 41/50
448/448 [==============================] - 130s 290ms/step - loss: 0.5257 - accuracy: 0.8102 - val_loss: 1.0971 - val_accuracy: 0.6261
Epoch 42/50
448/448 [==============================] - 130s 291ms/step - loss: 0.5071 - accuracy: 0.8178 - val_loss: 1.1146 - val_accuracy: 0.6221
Epoch 43/50
448/448 [==============================] - 130s 290ms/step - loss: 0.4899 - accuracy: 0.8219 - val_loss: 1.1219 - val_accuracy: 0.6263
Epoch 44/50
448/448 [==============================] - 131s 291ms/step - loss: 0.4763 - accuracy: 0.8258 - val_loss: 1.1243 - val_accuracy: 0.6204
Epoch 45/50
448/448 [==============================] - 131s 292ms/step - loss: 0.4605 - accuracy: 0.8340 - val_loss: 1.1349 - val_accuracy: 0.6271
Epoch 46/50
448/448 [==============================] - 130s 291ms/step - loss: 0.4533 - accuracy: 0.8349 - val_loss: 1.1408 - val_accuracy: 0.6261
Epoch 47/50
448/448 [==============================] - 131s 293ms/step - loss: 0.4310 - accuracy: 0.8448 - val_loss: 1.1642 - val_accuracy: 0.6250
Epoch 48/50
448/448 [==============================] - 131s 292ms/step - loss: 0.4162 - accuracy: 0.8524 - val_loss: 1.1728 - val_accuracy: 0.6260
Epoch 49/50
448/448 [==============================] - 131s 293ms/step - loss: 0.4066 - accuracy: 0.8541 - val_loss: 1.1623 - val_accuracy: 0.6288
Epoch 50/50
448/448 [==============================] - 130s 290ms/step - loss: 0.3882 - accuracy: 0.8610 - val_loss: 1.1858 - val_accuracy: 0.6264
```

File Edit View Navigate Code Refactor Run Tools VCS Window Help ImageRecognition.py - model.py
HelloWorld › model.py
Project
HelloWorld C:\Users\Pranav\PycharmProjects\Hel
External Libraries
Preprocessing.py
Analysis.py
model.py
LiveRecognition.py
ImageRecognition.py
Run: model
Run TODO Problems Terminal Python Packages Python Console
Event Log
46:48 CRLF UTF-8 4 spaces Python 3.9 (HelloWorld)
Type here to search
ENG
23:37
01-05-2021

Fig 7.3.1: - Number of epochs and Accuracy

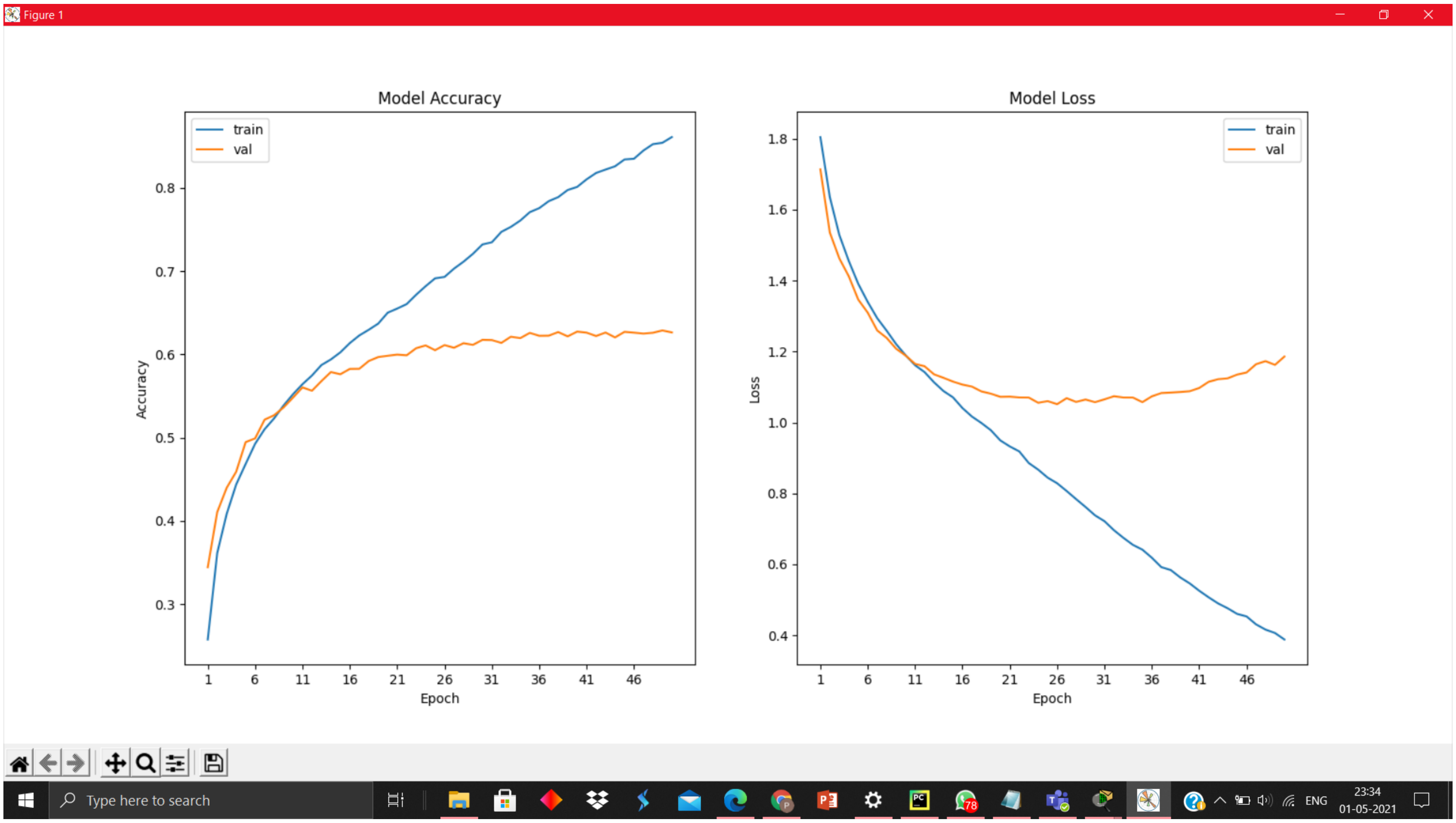


Fig 7.3.2: - Graph of Model Accuracy and Model loss

iv) Output of LiveRecognition.py

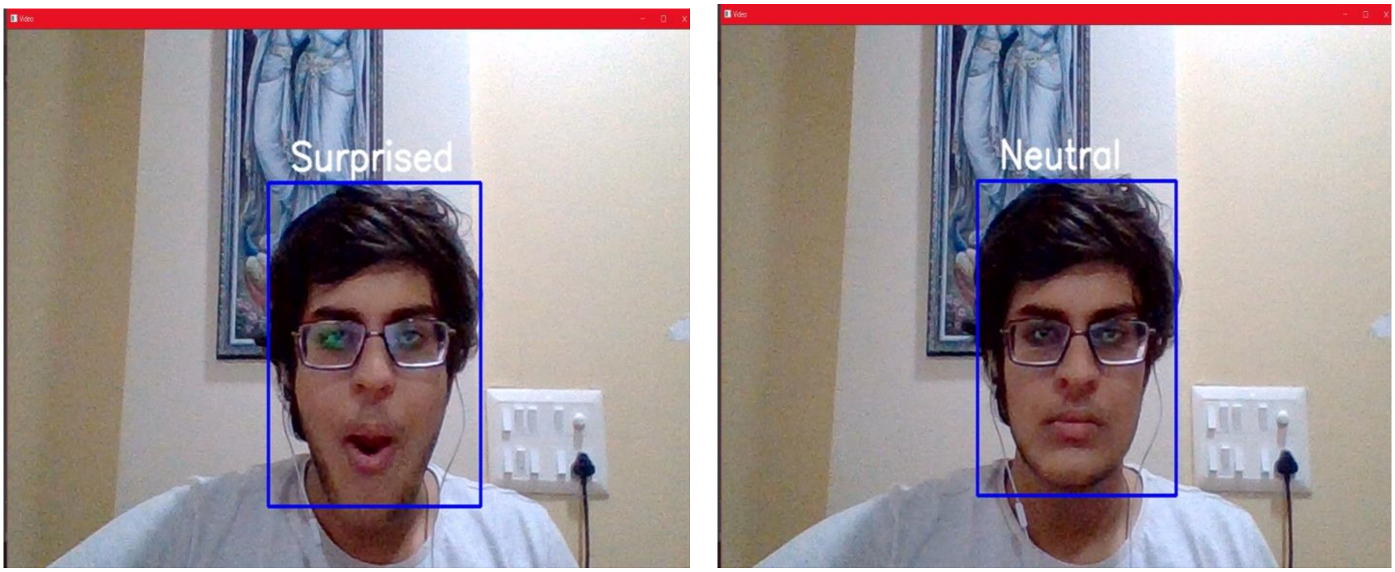


Fig 7.4.1: - Some emotions detected in real-time

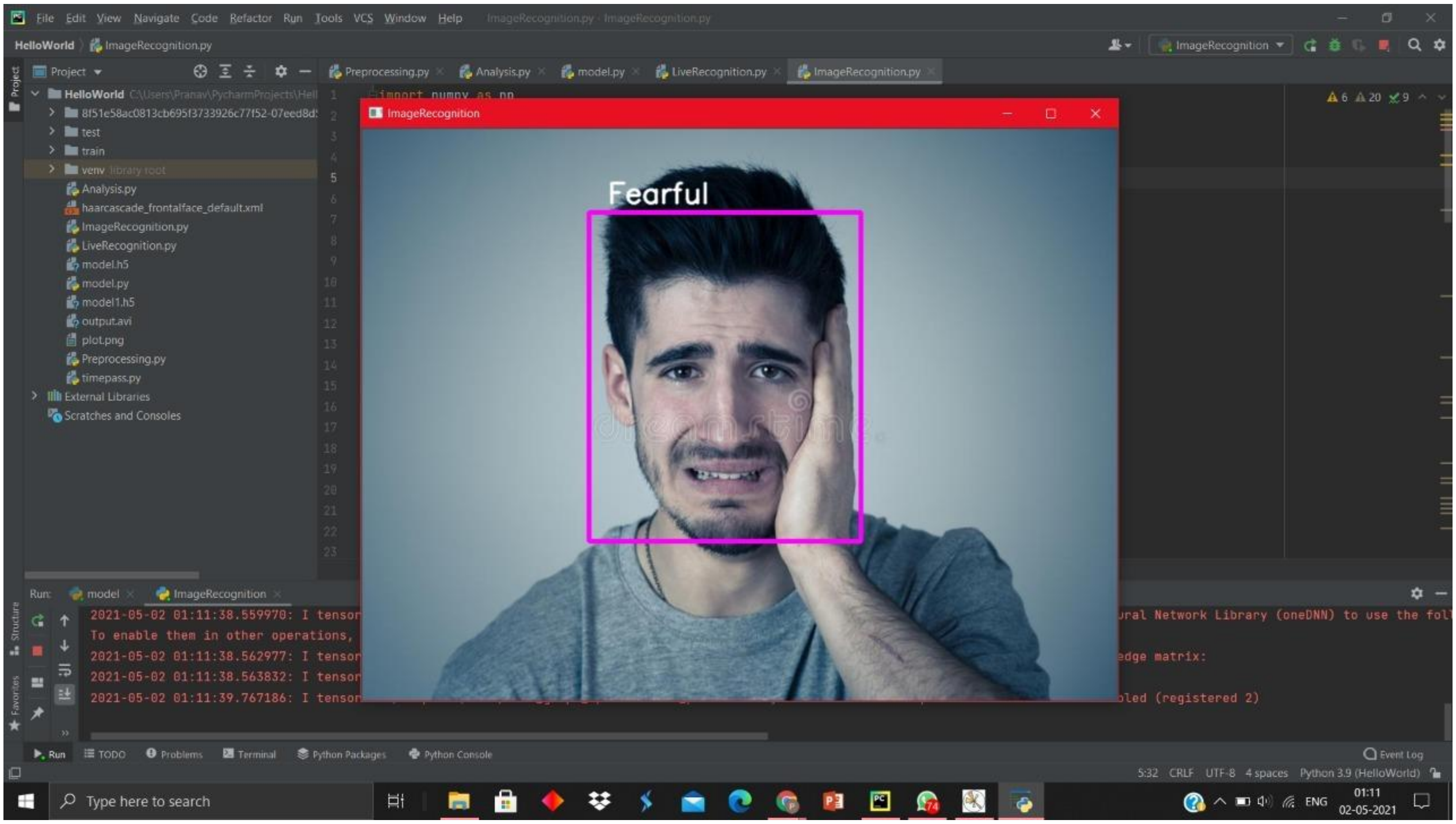
(NOTE: - We have shown only 2 outputs but our model
can detect 7 emotions as shown in the video)

v) Output of ImageRecognition.py

NOTE: -
The given module of ImageRecognition.py module gives the user an option whether he/she would like to give a video input or image input.

*

Input Image



```
import numpy as np
```

```
2021-05-02 01:11:38.559970: I tensor                                   ural Network Library (oneDNN) to use the foll
To enable them in other operations,
2021-05-02 01:11:38.562977: I tensor                                   edge matrix:
2021-05-02 01:11:38.563832: I tensor
2021-05-02 01:11:39.767186: I tensor                                   bled (registered 2)
```

Output Image

Fig. 7.5.1: - Output of ImageRecgonition.py

(P.T.O)

APPENDIX – A: - ABOUT NVIDIA DEVELOPER TOOLKIT

The NVIDIA Developer Toolkit is a group of various applications, spanning desktop and mobile targets, that can be proved helpful for developers to create, debug, profile, and develop class-leading and up-to-date code that helps to utilize the most recent visual computing hardware from NVIDIA. With the help of the NVIDIA Developer Toolkit, we can access the power of the NVIDIA hardware into our codes and software. We can install the required tools for the NVIDIA Developer Toolkit or install the entire toolkit according to our needs. Using the various elements in this toolkit can help us run our software efficiently and smoothly.

CUDA® is known as a parallel computing platform and programming model which was developed by NVIDIA for the purpose of general computing on graphical processing units (GPUs). With the help of CUDA, developers can be ready to dramatically speed up computing applications by harnessing the facility of their GPUs.

In GPU-accelerated applications, the sequential part of the work runs on the electronic equipment optimized for single-threaded performance. In contrast, an intensive portion of the appliance runs on thousands of GPU cores in parallel. We use CUDA, developers' programs in renowned programming languages such as C, C++, Fortran, Python, and MATLAB, and express similarity through extensions in the form of a few essential keywords.

The CUDA Toolkit from NVIDIA provides everything that would be required to develop GPU-accelerated applications. The CUDA Toolkit includes GPU-accelerated libraries, a compiler, development tools, and also the CUDA runtime.

The NVIDIA CUDA® Deep Neural Network library (cuDNN) is said to be a GPU-accelerated library that consists of all the basics required for deep neural networks. cuDNN can offer highly tuned implementations for normal routines such as forward and backward convolution, pooling, normalization, and activation layers.

Deep learning researchers and framework developers worldwide believe in cuDNN for superior GPU acceleration. It permits them to concentrate on training neural networks and developing software applications instead of using low-level GPU performance calibration. cuDNN can be used to accelerate widely used deep learning frameworks, including Caffe2, Chainer, Keras, MATLAB, MxNet, PaddlePaddle, PyTorch, and TensorFlow.

REFERENCES

- Goldman AI, Sripada CS. Simulationist models of face-based emotion recognition. Cognition. 2005 Jan;94(3):193-213.
- Fer2013 dataset from https://www.kaggle.com/
- R. Cowie et al., "Emotion recognition in human-computer interaction," in IEEE Signal Processing Magazine, vol. 18, no. 1, pp. 32-80, Jan 2001.
- Koolagudi, S.G., Rao, K.S. Emotion recognition from speech: a review. Int J Speech Technol 15, 99–117 (2012). https://doi.org/10.1007/s10772-011-9125-1
- Hong-Wei Ng, Viet Dung Nguyen, Vassilios Vonikakis, and Stefan Winkler. 2015. Deep Learning for Emotion Recognition on Small Datasets using Transfer Learning. In Proceedings of the 2015 ACM on International Conference on Multimodal Interaction (ICMI '15).
- Ioannou, S. V., Raouzaiou, A. T., Tzouvaras, V. A., Mailis, T. P., Karpouzis, K. C., & Kollias, S. D. (2005). Emotion recognition through facial expression analysis based on a neurofuzzy network. Neural Networks, 18(4), 423-435.
- L. S. Chen, T. S. Huang, T. Miyasato and R. Nakatsu, "Multimodal human emotion/expression recognition," Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition, Nara, Japan, 1998.
- Schmid, P.C., Schmid Mast, M. Mood effects on emotion recognition. Motiv Emot 34, 288–292 (2010).
- N. Esau, E. Wetzel, L. Kleinjohann and B. Kleinjohann, "Real-Time Facial Expression Recognition Using a Fuzzy Emotion Model," 2007 IEEE International Fuzzy Systems Conference, London, UK, 2007, pp. 1-6
- Young, Andrew W., et al. "Facial expression megamix: Tests of dimensional and category accounts of emotion recognition." Cognition 63.3 (1997): 271-313.