

An
INDUSTRY-ORIENTED MINI PROJECT(IOMP) REPORT
on
TRUTH LENS : Facial Emotion Detection
Using Convolutional Neural Network
submitted to
JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, HYDERABAD
In partial fulfillment of the requirement for the award of the degree of
BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE AND INFORMATION TECHNOLOGY

Submitted by
(MIP-BATCH-18)

ABHISHEK 217Y1A3302

NAGIREDDY SRI CHARAN REDDY 217Y1A3352

PRASHANTH EERAKAR 227Y5A3303

Under the Guidance

of

Mr. A. SATCHIDANANDAM
Assoc. Professor



DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY
MARRI LAXMAN REDDY
INSTITUTE OF TECHNOLOGY AND MANAGEMENT
(AUTONOMOUS)

(Affiliated to JNTU-H, Approved by AICTE New Delhi and Accredited by NBA & NAAC With 'A' Grade)

JULY 2024



MARRI LAXMAN REDDY
INSTITUTE OF TECHNOLOGY AND MANAGEMENT

(AN AUTONOMOUS INSTITUTION)

(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)

Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section 2(f) & 12(B) of the UGC act, 1956

CERTIFICATE

This is to certify that the project report titled **“Truth Lens : Facial Emotion Detection using Convolutional Neural Network”** is being submitted by **ABHISHEK(217Y1A3302), NAGIREDDY SRI CHARAN REDDY(217Y1A3352) and PRASHANTH EERAKAR(227Y5A3303)** in **IV B. Tech I Semester Computer Science & Information Technology** is a record bonafide work carried out by them. The results embodied in this report have not been submitted to any other University for the award of any degree.

Internal Guide

HOD

Principal

External Examiner



MARRI LAXMAN REDDY
INSTITUTE OF TECHNOLOGY AND MANAGEMENT

(AN AUTONOMOUS INSTITUTION)

(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)

Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section 2(f) & 12(B) of the UGC act, 1956

DECLARATION

We hereby declare that the Mini Project Report entitled, **“Truth Lens : Facial Emotion Detection using Convolutional Neural Network”** submitted for the B. Tech degree is entirely our work and all ideas and references have been duly acknowledged. It does not contain any work for the award of any other degree.

Date:

ABHISHEK

NAGIREDDY SRI CHARAN REDDY

(217Y1A3302)

(217Y1A3352)

PRASHANTH EERAKAR

(227Y5A3303)



MARRI LAXMAN REDDY INSTITUTE OF TECHNOLOGY AND MANAGEMENT

(AN AUTONOMOUS INSTITUTION)

(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)

Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section 2(f) & 12(B) of the UGC act, 1956

ACKNOWLEDGEMENT

We are happy to express our deep sense of gratitude to the principal of the college **Dr. R. Murali Prasad**, Professor, Marri Laxman Reddy Institute of Technology & Management, for having provided us with adequate facilities to pursue my project.

We are happy to express our deep sense of gratitude to the Director of the college **Dr.P.Sridhar**, Director, Marri Laxman Reddy Institute of Technology & Management, for having provided us with adequate facilities to pursue our project.

We would like to thank **Dr A. Arun Kumar**, Professor and Head, Department of Computer Science and Information Technology, Marri Laxman Reddy Institute of Technology & Management, for having provided the freedom to use all the facilities available in the department, especially the laboratories and the library.

We are very grateful to our project guide **Mr. A. Satchidanandam**, Assoc. Prof., Department of Computer Science and Information Technology, Marri Laxman Reddy Institute of Technology & Management, for his extensive patience and guidance throughout our project work.

We sincerely thank our seniors and all the teaching and non-teaching staff of the Department of Computer Science and Information Technology for their timely suggestions, healthy criticism and motivation during this work.

We would also like to thank our classmates for always being there whenever We needed help or moral support. With great respect and obedience, We thank our parents and brother who were the backbone behind our deeds.

Finally, We express our immense gratitude with pleasure to the other individuals who have either directly or indirectly contributed to our need at right time for the development and success of this work.



MARRI LAXMAN REDDY **INSTITUTE OF TECHNOLOGY AND MANAGEMENT**

(AN AUTONOMOUS INSTITUTION)

(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)

Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section 2(f) & 12(B) of the UGC act, 1956

CONTENTS

| S NO. | TITLE | PAGE NO. |
|--------------|--------------------------------|-----------------|
| | CERTIFICATE | i |
| | DECLARATION | ii |
| | ACKNOWLEDGEMENT | iii |
| | CONTENTS | iv |
| | ABSTRACT | vii |
| | LIST OF FIGURES | viii |
| | LIST OF TABLES | ix |
| | SYMBOLS AND ABBREVIATIONS | x |
| 1 | INTRODUCTION | 1 |
| | 1.1 Scope of the project | 2 |
| | 1.2 Objective | 2 |
| | 1.3 Problem Definition | 3 |
| 2 | LITERATURE SURVEY | 5 |
| 3 | SYSTEM REQUIREMENTS | 7 |
| | 3.1 Software Requirement | 7 |
| | 3.2 Hardware Requirement | 7 |
| | 3.3 Non-Functional Requirement | 7 |
| 4 | SYSTEM ANALYSIS | 9 |
| | 4.1 Existing system | 9 |
| | 4.2 Proposed system | 10 |



MARRI LAXMAN REDDY
INSTITUTE OF TECHNOLOGY AND MANAGEMENT

(AN AUTONOMOUS INSTITUTION)

(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)

Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section 2(f) & 12(B) of the UGC act, 1956

| S NO. | TITLE | PAGE NO. |
|--------------|-----------------------------|-----------------|
| | 4.3 Project Modules | 11 |
| 5 | SYSTEM DESIGN | 15 |
| | 5.1 System architecture | 15 |
| | 5.2 Data flow Diagrams | 16 |
| | 5.3 UML Diagrams | 17 |
| 6 | IMPLEMENTATION | 22 |
| | 6.1 Development Environment | 22 |
| | 6.2 Dataset | 22 |
| | 6.3 Algorithm | 22 |
| | 6.4 Training and Validation | 24 |
| | 6.5 CNN Model | 26 |
| 7 | ANALYSIS AND TESTING | 31 |
| | 7.1 Analyzing the Result | 31 |
| | 7.2 System Testing | 33 |
| | 7.3 Functional Testing | 33 |
| | 7.4 Test Cases | 33 |
| 8 | RESULT | 34 |
| 9 | CONCLUSION | 37 |
| 10 | REFERENCES | 38 |

ABSTRACT

Human facial expressions convey a lot of information visually rather than articulately. Facial expression recognition plays a crucial role in the area of human machine interaction. Automatic facial expression recognition system has many applications including, but not limited to, human behavior understanding, detection of mental disorders, and synthetic human expressions. Recognition of facial expression by computer with high recognition rate is still a challenging task.

Through this project, we put forward a solution to recognize emotions by understanding different facial expressions by collecting live video through a Flask App created. I deploy a Flask App to video stream live feed captured through the local camera attached to the machine or computer system. The video captured is fed to various image extraction techniques.

The facial features are identified by different operations provided by OpenCV and the region consisting of parts of the face are made to surround or enclose by a contour. This region, enclosed by the contour is used as an input to Convolutional Neural Network (CNN). The CNN model created consists of six activation layers, of which four are convolution layers and two are fully controlled layers. Each layer is designed to undergo several training techniques.

The main objective of this project is to demonstrate the accuracy of Convolutional Neural Network model designed. The project is concluded by discussing the outcomes of our project and the ways to improve the efficiency of the model. The scope of this project is also analyzed to enhance technologies developed in the near future.

LIST OF FIGURES

| FIG. NO | FIG. NAME | PAGE NO. |
|---------|---|----------|
| 1.1 | Expressions | 2 |
| 1.2 | Flow of Problem formulation | 4 |
| 4.1 | Block diagram of facial emotion recognition and detection using CNN | 10 |
| 4.2 | System diagram of facial emotion detection | 12 |
| 4.3 | Flowchart of Training | 13 |
| 4.4 | Flowchart of Testing | 14 |
| 5.1 | System Architecture | 15 |
| 5.2 | Dataflow Diagram | 17 |
| 5.3 | Use case Diagram | 18 |
| 5.4 | Sequence Diagram | 19 |
| 5.5 | Activity Diagram | 20 |
| 5.6 | Class Diagram | 21 |
| 5.7 | State chart Diagram | 21 |
| 6.1 | Random forest Simplified | 23 |
| 6.2 | Sample Dataset | 23 |
| 6.3 | CNN Model | 27 |
| 7.1 | Acc vs v-Acc | 32 |
| 7.2 | Lost vs v-Lost | 33 |
| 8.1 | Folder of Project files | 38 |
| 8.2 | Compiling main file 1 | 33 |
| 8.4 | Output after Running | 34 |
| 8.5 | Neutral expression | 36 |

| FIG. NO | FIG. NAME | PAGE NO. |
|----------------|---------------------|-----------------|
| 8.6 | Happy Expression | 36 |
| 8.7 | Sad Expression | 36 |
| 8.8 | Surprise Expression | 36 |
| 8.9 | Fear Expression | 36 |
| 8.10 | Angry Expression | 36 |

LIST OF TABLES

| TABLE NO | TABLE TITLE | PAGE NO. |
|----------|----------------------|----------|
| 6.4 | Convolution operator | 27 |
| 6.5 | Single Depth Slice | 28 |
| 7.4 | Test Cases | 33 |

SYMBOLS AND ABBREVIATIONS

SYMBOLS

ABBREVIATIONS

| | | |
|------|---|------------------------------------|
| CNN | : | Convolutional Neural Network |
| AU | : | Action Units |
| PCA | : | Principal component analysis |
| LDA | : | Linear discriminant analysis |
| GPU | : | Graphical user Interface |
| FER | : | Facial Expression Recognition 2013 |
| HCI | : | Human Computer Interactions |
| YML | : | Yaml Ain't Markup Language |
| LBPH | : | local binary pattern histogram |
| DFD | : | data flow diagram |
| UML | : | Unified Modeling Language |
| OS | : | Operating System |
| NN | : | Neural Network |

CHAPTER 1

INTRODUCTION

Facial expression recognition is an evolving technology in the field of human- computer interaction. Facial expression recognition has its branches spread across various applications such as virtual reality, webinar technologies, online surveys, and many other fields. Even though high advancements have been witnessed in this field, there are several diplomacies that exist.

Image processing is the field of signal processing where both the input and output signals are images. One of the most important applications of Image processing is Facial expression recognition. Our emotion is revealed by the expressions in our face. Facial Expressions plays an important role in interpersonal communication. Facial expression is a nonverbal scientific gesture which gets expressed in our face as per our emotions.

Automatic recognition of facial expression plays an important role in artificial intelligence and robotics and thus it is a need of the generation. Some applications related to this include Personal identification and Access control, Videophone and Teleconferencing, Forensic application, Human-Computer Interaction, Automated Surveillance, Cosmetology and so on.

The traditional methods have high latency or delay in their response. Through these traditional methods, it is extremely difficult to extract the required features effectively and hence, is too hazardous to utilize for real time applications. In order to provide a panacea to such issues, a facial recognition method is proposed using CNN. This model thus, can be used to solve the above stated problems or difficulties.

The development of the facial expression recognition is done in Keras. A six layered CNN is developed to build and train the model. Each layer is defined with certain training techniques to enable faster and efficient feature extraction.

The trained model is deployed to a web interface using Flask app. The developed model is applied for real time videos and images and its accuracy is analysed.

The objective of this project is to develop Automatic Facial Expression Recognition System which can take human facial images containing some expression as input and recognize and classify it into seven different expression class such as:

1. Neutral
2. Angry
3. Disgust
4. Fear
5. Happy
6. Sadness
7. Surprise



Fig 1.1 Expressions

1.1 Scope of the Project

In this project facial expression recognition system is implemented using convolution neural network. Facial images are classified into seven facial expression categories namely Anger, Disgust, Fear, Happy, Sad, Surprise and 'Neutral. Kaggle dataset is used to train and test the classifier.

1.2 Objective

The objective of the project is to implement Convolutional Neural Networks for classification of facial expressions.

The trained model is deployed to a web interface using Flask app. The developed model is applied for real time videos and images and its accuracy is analysed.

1.3 Problem Definition

Human facial expressions can be easily classified into 7 basic emotions: happy, sad, surprise, fear, anger, disgust, and neutral.

Our facial emotions are expressed through activation of specific sets of facial muscles. These sometimes subtle, yet complex, signals in an expression often contain an abundant amount of information about our state of mind. Through facial emotion recognition, we are able to measure the effects that content and services have on the audience/users through an easy and low-cost procedure.

For example, retailers may use these metrics to evaluate customer interest. Healthcare providers can provide better service by using additional information about patients' emotional state during treatment. Entertainment producers can monitor audience engagement in events to consistently create desired content.

Humans are well-trained in reading the emotions of others, in fact, at just 14 months old, babies can already tell the difference between happy and sad. But can computers do a better job than us in accessing emotional states?

To answer the question, we designed a deep learning neural network that gives machines the ability to make inferences about our emotional states. In other words, we give them eyes to see what we can see.

Automatic recognition of facial expression plays an important role in artificial intelligence and robotics and thus it is a need of the generation. Some applications related to this include Personal identification and Access control, Videophone and Teleconferencing, Forensic application, Human-Computer Interaction, Automated Surveillance, Cosmetology and so on.

The traditional methods have high latency or delay in their response. Through these traditional methods, it is extremely difficult to extract the required features effectively and hence, is too hazardous to utilize for real time applications. In order to provide a panacea to such issues, a facial recognition method is proposed using CNN. This model thus, can be used to solve the above stated problems or difficulties.

Problem formulation of our project

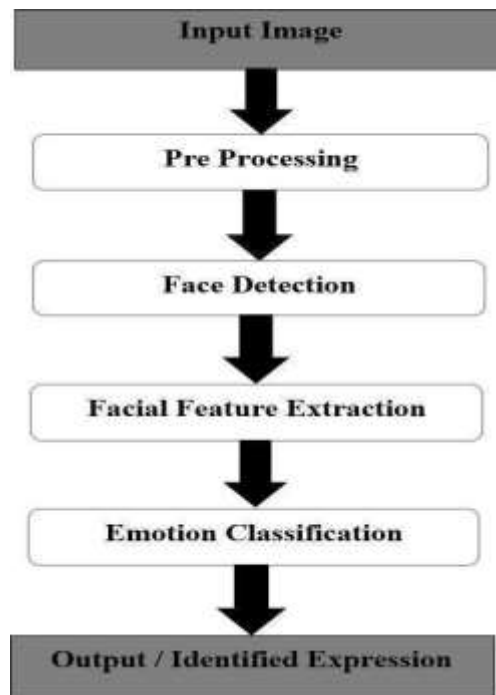


Fig 1.2 Flow of Problem formulation

Facial expression recognition is a process performed by humans or computers, which consist of:

1. Locating faces in the scene
2. Extracting facial features from the detected face region (e.g., detecting the shape of facial components or describing the texture of the skin in a facial area; this step is referred to as facial feature extraction),
3. Analysing the motion of facial features and/or the changes in the appearance of facial features and classifying this information into some facial-expression

Several projects have already been done in this field and our goal will not only be to develop an Automatic Facial Expression Recognition System but also improving the accuracy of this system compared to the other available systems.

CHAPTER 2

LITERATURE SURVEY

Two different approaches are used for facial expression recognition, both of which include two different methodologies, exist. Dividing the face into separate action units or keeping it for further processing appears to be the first and the primary distinction between the main approaches.

In both approaches, two different methodologies, namely the ‘Geometric based’ and the ‘Appearance-based’ parameterizations, can be used.

Making use of the whole frontal face image and processing it in order to end up with the classifications of 6 universal facial expression prototypes:

- Disgust
- Fear
- Joy
- Surprise
- Sadness
- anger

outlines the first approach. Here, it is assumed that each of the above-mentioned emotions have characteristic expressions on face and that’s why recognition of them is necessary and sufficient. Instead of using the face images as a whole, dividing them into some sub-sections for further processing forms up the main idea of the second approach for facial expression analysis.

As expression is more related with subtle changes of some discrete features such as eyes, eye brows and lip corners; these fine-grained changes are used for analysing automated recognition. There are two main methods that are used in both of the above explained approaches.

Geometric Based Parameterization is an old way which consists of tracking and processing the motions of some spots on image sequences, firstly presented by Suwa et al to recognize facial expressions.

Cohn and Kanade later tried geometrical modelling and tracking of facial features by claiming that each AU is presented with a specific set of facial muscles.

The disadvantages of this method are the contours of these features and components have to be adjusted manually in this frame, the problems of robustness and difficulties come out in cases of pose and illumination changes while the tracking is applied on images, as actions & expressions tend to change both in morphological and in dynamical senses, it becomes hard to estimate general parameters for movement and displacement. Therefore, ending up with robust decisions for facial actions under these varying conditions becomes to be difficult. Rather than tracking spatial points and using positioning and movement parameters that vary within time, color (pixel) information of related regions of face are processed in Appearance Based Parameterizations; in order to obtain the parameters that are going to form the feature 8 vectors.

Different features such as Gabor, Haar wavelet coefficients, together with feature extraction and selection methods such as PCA, LDA, and Adaboost are used within this framework.

For classification problem, algorithms like Machine learning, Neural Network, Support Vector Machine, Deep learning, Naive Bayes are used. Raghuvanshi A. et al have built a Facial expression recognition system upon recent research to classify images of human faces into discrete emotion categories using convolutional neural networks. Alizadeh, Shima, and Azar Fazel have developed Facial Expression Recognition system using Convolutional Neural Networks based on Torch model.

CHAPTER 3

SYSTEM REQUIREMENTS

Requirement Analysis

There are few requirements to implement this project they are as follows:

2.1 Software Requirements

Operating System: Any OS with clients to access the internet

Network: Wi-Fi Internet or cellular Network

Visual Studio: Create and design Data Flow and Context Diagram and to be able to use python.

Jupyter Notebook or Google Colab: Used to run our dataset, compute require results, and create models and execute them on the dataset and to obtain the accuracies.

Packages: All the Machine Learning files and libraries to perform the task.

2.2 Hardware Requirements

For Application development, the following Software Requirements are:

Processor : Intel or Ryzen

RAM : 8000 MB

Space on disk : minimum 1000 MB

2.3 Non-Functional Requirements

Performance Requirements

1. For this project the model requires a GPU for better performance. Since we don't have a GPU in our development machines, we opted to use Google Colab, a Cloud platform where you can run and deploy your machine learning and deep learning models.
2. Another requirement is a proper network connection. .

Google Colab

If you have used Jupyter notebook previously, you would quickly learn to use Google Colab. To be precise, Colab is a free Jupyter notebook environment that runs entirely in the cloud.

Most importantly, it does not require a setup and the notebooks that you create can be simultaneously edited by your team members - just the way you edit documents in Google Docs. Colab supports many popular machine learning libraries which can be easily loaded in your notebook

What Colab Offers You?

As a programmer, you can perform the following using Google Colab.

- Write and execute code in Python
- Document your code that supports mathematical equations
- Create/Upload/Share notebooks
- Import/Save notebooks from/to Google Drive
- Import/Publish notebooks from GitHub
- Import external datasets e.g. from Kaggle
- Integrate TensorFlow, Keras, OpenCV
- Free Cloud service with free GPU

CHAPTER 4

SYSTEM ANALYSIS

4.1 Existing System

Using the CNN model to classify test dataset as well as real-time images. The transfer learning theory can be used to recognize the emotion in images here in real-time.

The model developed during the training phase consists of the corresponding weights and values that can be used to detect new facial expressions.

CNN is a type of deep learning model for processing data that has a grid pattern, such as images, which is inspired by the organization of animal visual cortex and designed to learn spatial hierarchies of features automatically and adaptively, from low- to high-level patterns.

4.1.1 Advantages

- Facial expressions of emotion are probably the most important signal of the face because they tell us about people's personalities, emotions, motivations, or intent.
- They are not only signs of people's internal states; they are also signals to others to act in certain ways, providing messages for social coordination.
- Emotion recognition can be used to understand how candidates feel during interviews and to measure how they react to certain questions.
- This information can be used to optimize interview structure for future candidates and streamline the application process.
- Facial expressions help set the emotional tone for a speech, and it is important that your facial expressions stay consistent with your message.

4.1.2 Disadvantages

- Validation of emotion dataset is a challenge in order to have accurate emotion recognition system.
- Finding or detecting emotion is haystack.
- It is a challenge to make emotion available in different languages.
- There are limitations with different types and versions of the softwares such as dataset input is only textual data, image, pattern, video and audio inputs are invalid.

- Performance and results of the emotion sensing system depends on accuracy of the sensors such as cameras, thermal image sensors, facial recognition algorithm used and so on. Highly accurate system will be expensive due to use of costly components.

4.2 Proposed System

With the goal to improve the process of facial sentiment analysis systems, a classification mechanism is proposed using a CNN architecture. Due to the need of large data required for training of deep networks, FER2013 dataset which is available publicly is utilized here. In the subsequent section, the features of our chosen dataset are listed out, followed by the description of our network architecture and finally the performance measures used for evaluation.

4.2.1 Proposed system advantages

High picture quality improves the effectiveness of facial recognition; even low-resolution photographs are usable with the suggested method; and Higher accuracy while being more computationally efficient.

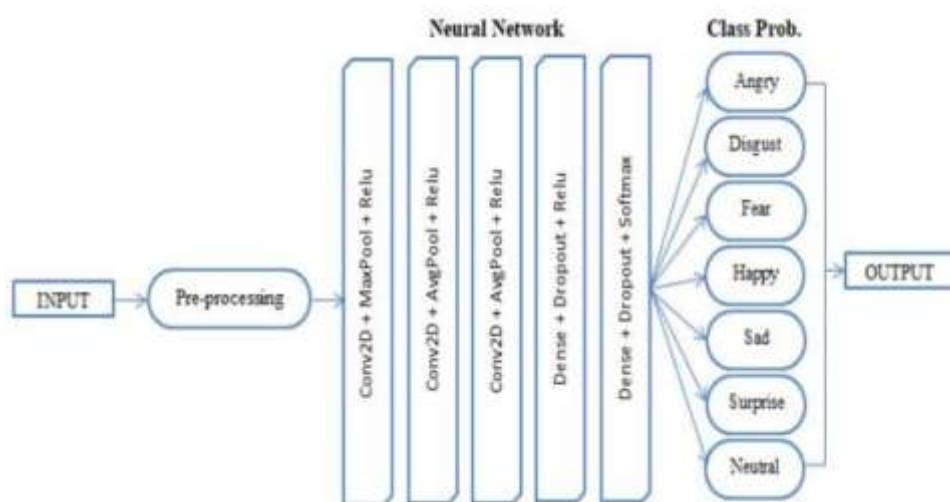


Fig 4.1 Block diagram of facial emotion recognition and detection using CNN

Figure 4.1 describes the proposed model for facial emotion detection using CNN machine learning model

4.2.2 Applications

- It is widely used in robotics, where Human Computer Interactions (HCI) are extremely crucial.
- The ability to predict emotions based on facial expressions has a number of scientific advantages.

4.3 Project Modules

Modules used in our model design are:

- i. Face capturing module
- ii. Pre-processing module
- iii. Training module
- iv. Face recognition module
- v. Expression recognition module

4.3.1 Face Capturing Module

During this phase, we are taking pictures of people's faces for further processing. We are utilising a webcam or an external web camera for this purpose [10]. There is no way to complete the procedure without first taking the image, and there is no way to identify the emotions without first capturing the image.

Pre-processing Module: Following the capture of photos, we will do image processing on the captured images. The grey scale photos will be created by converting the colour photographs to grey scale.

Training module: This step will involve the preparation of a dataset, which will consist of a binary array of all the photographs that have been taken. The collected photographs will be saved in a .YML file, which will contain all of the face data that was obtained. The .YML file allows us to process the collected photos more quickly because of its compressed nature.

Face Recognition Module: The first phase in the face recognition process is to train the host system on the facial data that has been collected. The face is photographed using the web camera on the computer system, which captures 60 different photos of the subject's face. In this session, we will learn how to detect people's faces using the LBP algorithm. The abbreviation LBPH stands for local binary pattern histogram. With the face ID and NAME that were previously stored, it will recognise the faces in the database.

Face Expression Recognition Module: Facial expression recognition software is a system that detects emotions in human faces by using biometric indicators. Because it collects and analyses information from images, it is possible to offer an unfiltered, unbiased emotional reaction or data that is unfiltered and impartial.

4.3.2 System Diagram

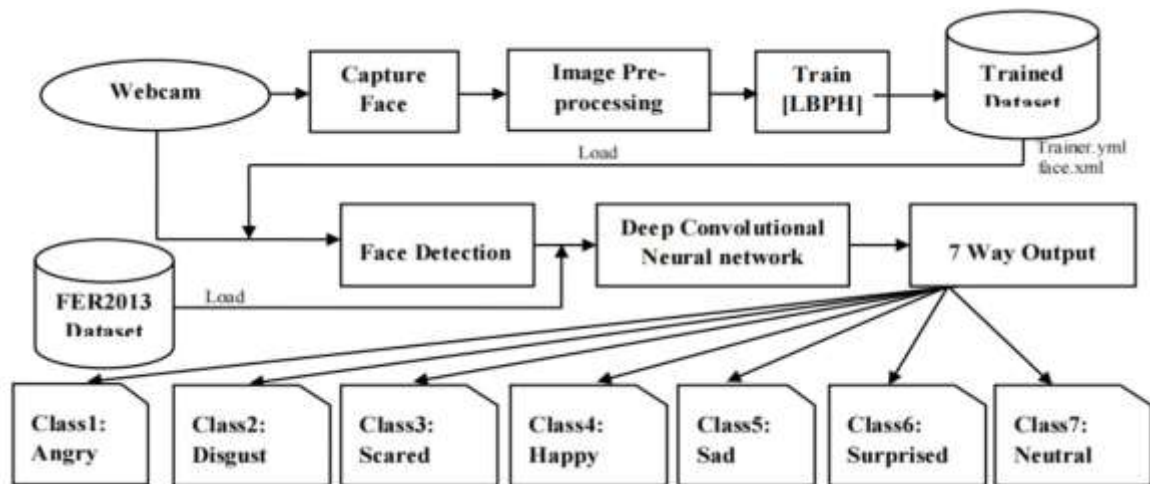


Fig 4.2 System diagram of facial emotion detection

A webcam is used to capture, identify, and recognise the facial expressions of a person, which is done using software. In the camera, a rectangular frame on the face area is obtained; this identification of the face region from a non-facial region is accomplished by the employment of the Viola Jones method, the LBPH Face Recognizer algorithm, and the Haarcascade frontal face dataset, among other techniques.

Captured person faces are pre-processed before being saved in a folder labelled with the subject's ID and name. These photos are trained using the LBPH method, and the resulting trained dataset is saved as Trainer. yml in the Trainer folder.

During the Face Detection process: A trained dataset is used to match the face in a video camera with the face in the dataset. If a person's face matches that in the trained dataset, his or her ID and name will be displayed on the screen. In order to classify the obtained face, convolutional neural networks are used in conjunction with the FER2013 database to do the classification.

The facial expression represents the chance of acquiring the maximum level of expression based on the characteristics of the individual. One of seven possible facial expressions is presented in conjunction with the recognised picture of the subject.

4.3.3 System Flowchat

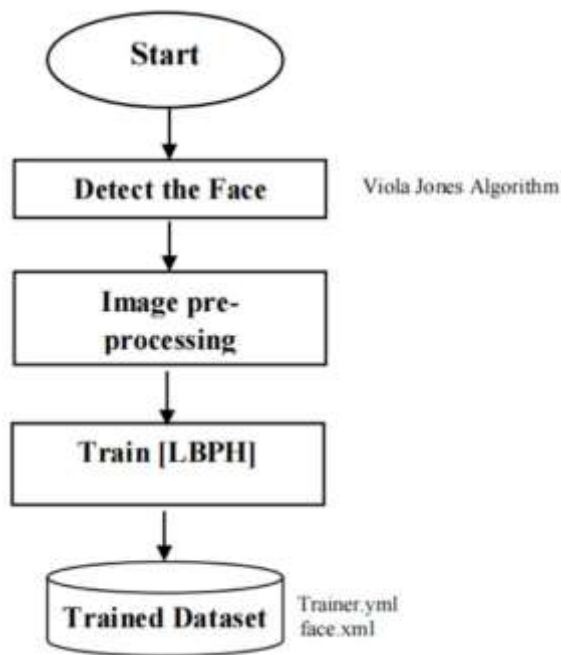


Fig 4.3 flowchart of training

During training Phase, the system received a training data comprising grayscale images of faces with their respective expression label and learns a set of weights for the network. The training step took as input an image with a face. Thereafter, an intensity normalization is applied to the image.

The normalized images are used to train the Convolutional Network. To ensure that the training performance is not affected by the order of presentation of the examples, validation dataset is used to choose the final best set of weights out of a set of trainings performed with samples presented in different orders.

The output of the training step is a set of weights that achieve the best result with the training data. During test, the system received a grayscale image of a face from test dataset, and output the predicted expression by using the final network weights learned during training. Its output is a single number that represents one of the seven basic expressions.

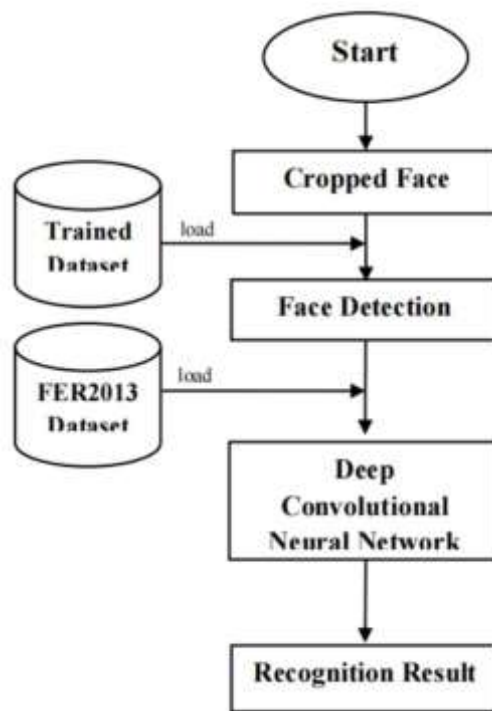


Fig 4.4 Flowchart of testing

CHAPTER 5

SYSTEM DESIGN

5.1 System Architecture

The user interface includes a login window as soon as the application starts with a text box for user-name and a password box for the password.

User can access the system through android and web application also. Then the user will be prompted to capture a selfie, and after the image is captured, it will be pre-processed and the features extracted will be stored into the image database.

These features obtained will then be sent to the trained neural network, which will predict features and use them to detect the emotion and obtain the results. Based on these results, the system will provide relevant recommendations to the user.

The user will then find some recommended tasks or some videos on his screen, as per the resultant mood, in order to improve their mood.

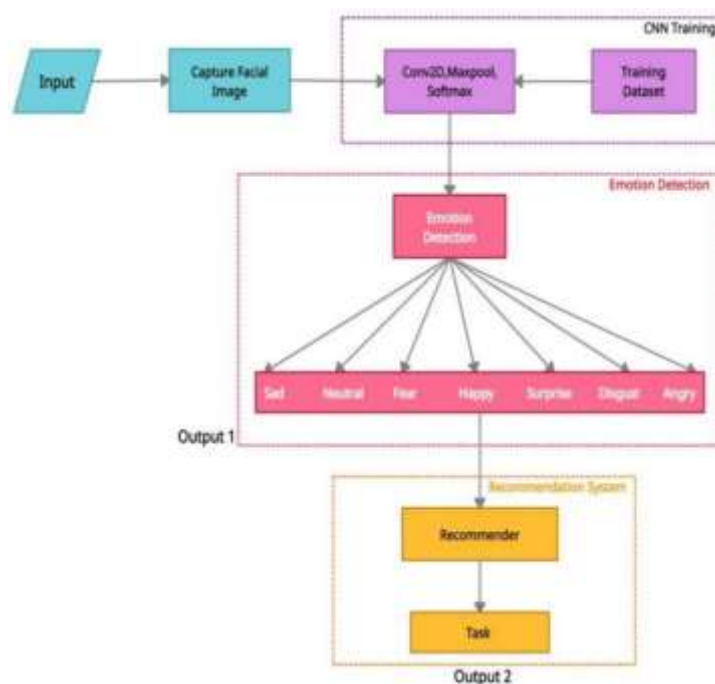


Fig 5.1 System Architecture

The general system design comprises of the following modules:

- Input Image
- Training using Convolutional Network

- c) Analysis and Classification
- d) Recommendation kernel
- e) Result

First of all, when the user registers into the application, it requests the user to capture a selfie. This image is then divided into different sections of the face, such as forehead, eyebrows, lower eye, right cheek, and left cheek. After all the pre-processing is done, then with Convolutional neural network it trains the given dataset and with every epoch accuracy increases. Then with the user's image it detects emotion and give accordingly suggest tasks to change mood of sad, depressed person.

5.2 Data Flow Diagram

A data flow diagram is a way of representing a flow of data through a process or a system (usually an information system). The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow — there are no decision rules and no loops. Specific operations based on the data can be represented by a flowchart.

There are several notations for displaying data-flow diagrams. The notation presented above was described in 1979 by Tom DeMarco as part of structured analysis.

For each data flow, at least one of the endpoints (source and / or destination) must exist in a process. The refined representation of a process can be done in another data-flow diagram, which subdivides this process into sub-processes.

The data-flow diagram is a tool that is part of structured analysis and data modelling. When using UML, the activity diagram typically takes over the role of the data-flow diagram. A special form of data-flow plan is a site-oriented data-flow plan.

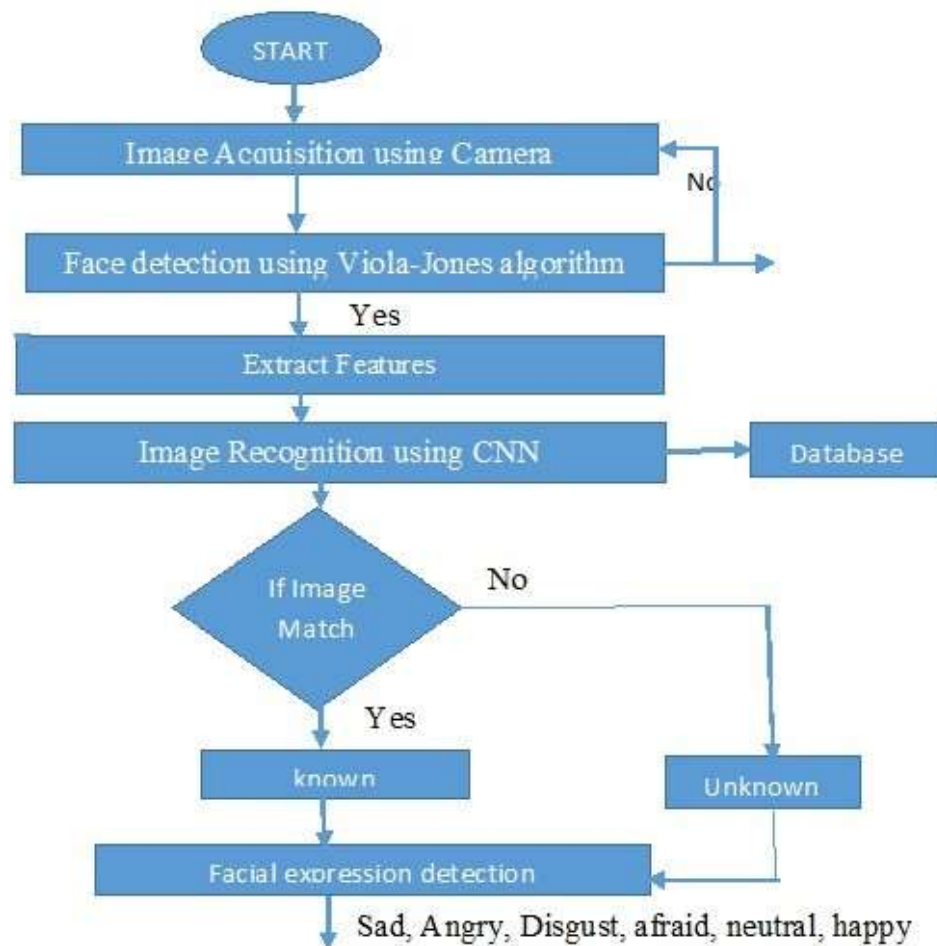


Fig 5.2 Data-flow diagram

5.3 UML Diagrams

5.3.1 Use Case Diagram

A use case diagram is a graphic representation of the communication among the element in the system [3]. It used in system analysis to identify, clarify, and organize system requirements. The use case is made up of a set of possible orders of interaction between application and user in a particular environment and related to a particular goal. It involves a group of elements for example, classes and interfaces that can be used together in a way that will have an impact greater than the sum of the separate elements combined. The use case should cover all application activities that have consequence to the user.

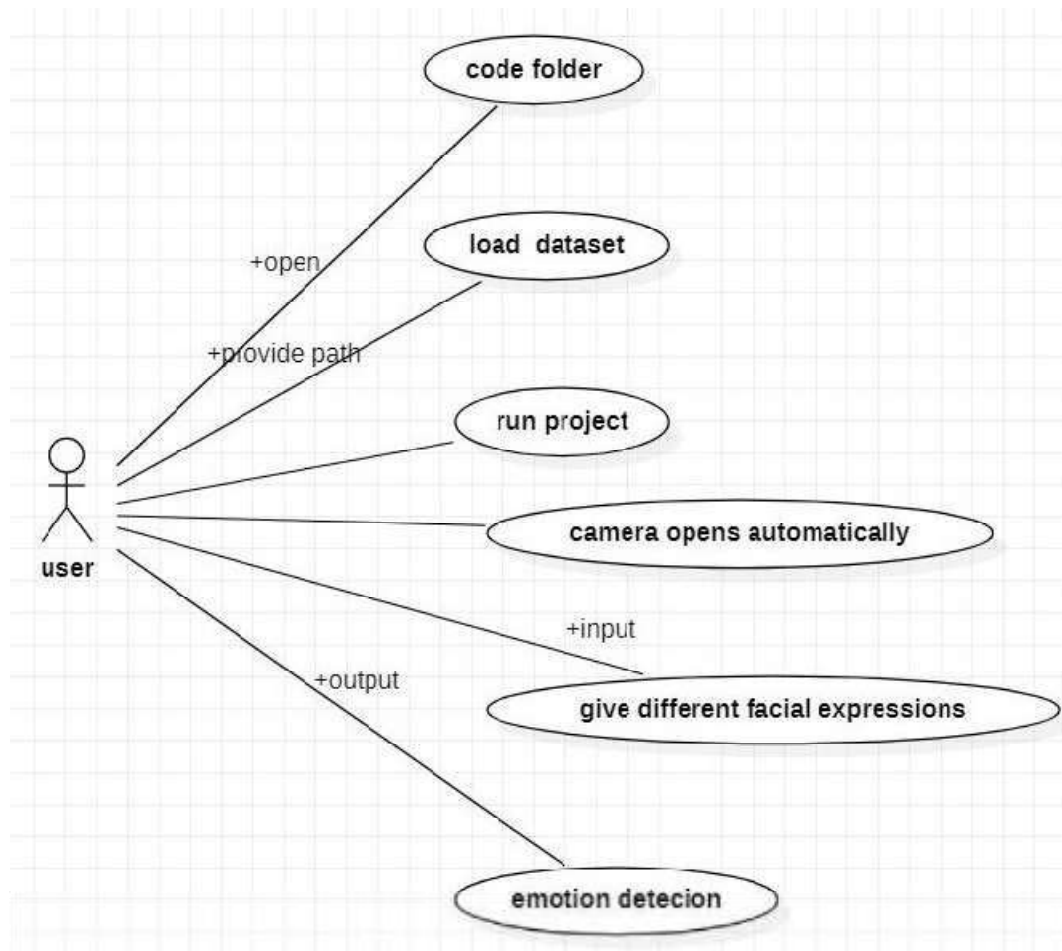


Fig 5.3 Use Case diagram

5.3.2 Sequence Diagram

A sequence diagram is an interaction diagram that shows how processes operate with one another and in what kind of order. A sequence diagram also shows object interaction arrange in the sequence. It depicts the object and classes involved in the scenario and the sequence of messages exchange between the object needed to carry out the functionality of the scenario. Sequence diagram are sometime called event diagram or event scenarios.

A sequence diagram shows parallel vertical lines (lifeline), different processes or object that lives simultaneously, the horizontal arrow, the message exchange between them, in the order which they occur in the system. This allows the specification of simple runtime scenarios in graphical manner.

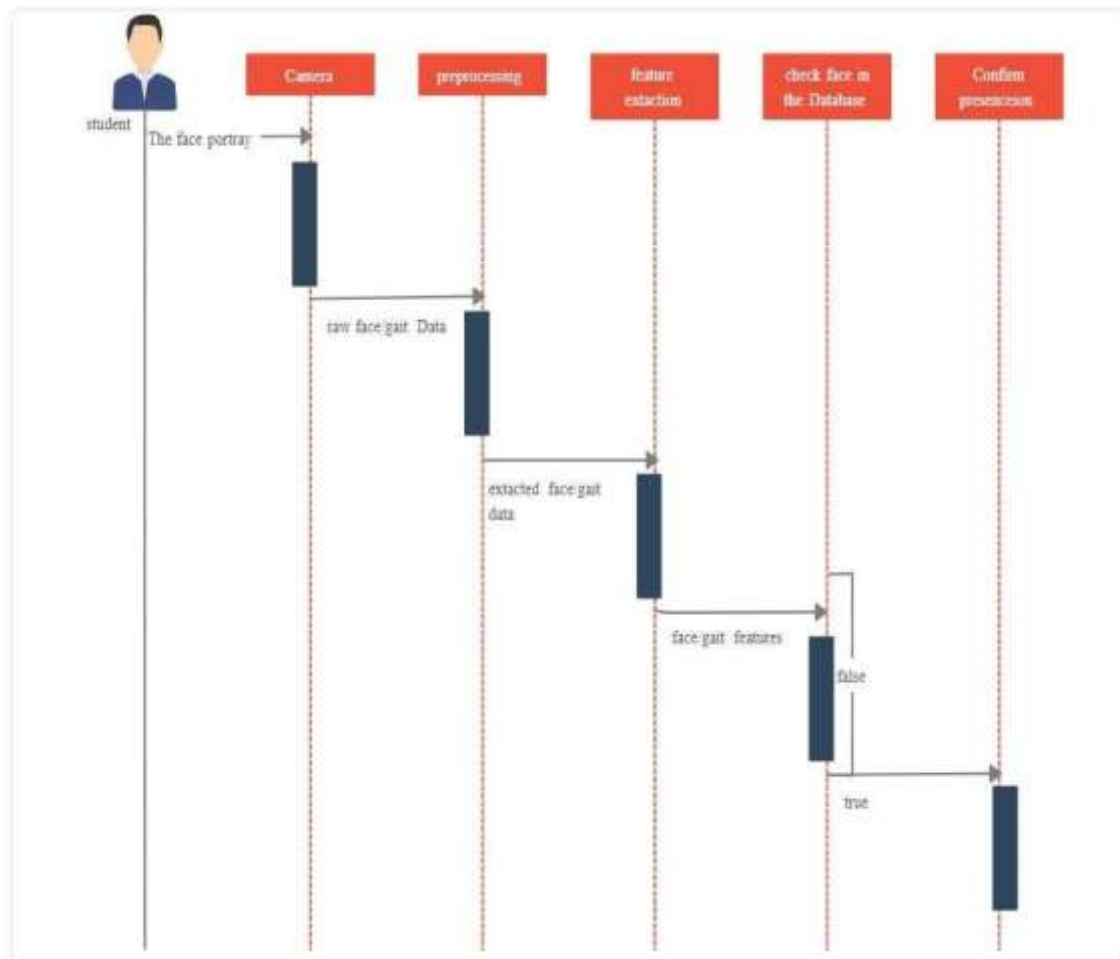


Fig 5.4 Sequence diagram

5.3.3 Activity Diagram

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency.

In the Unified Modelling Language, activity diagrams are intended to model both computational and organizational processes (i.e., workflows), as well as the data flows intersecting with the related activities.

Although activity diagrams primarily show the overall flow of control, they can also include elements showing the flow of data between activities through one or more data stores.

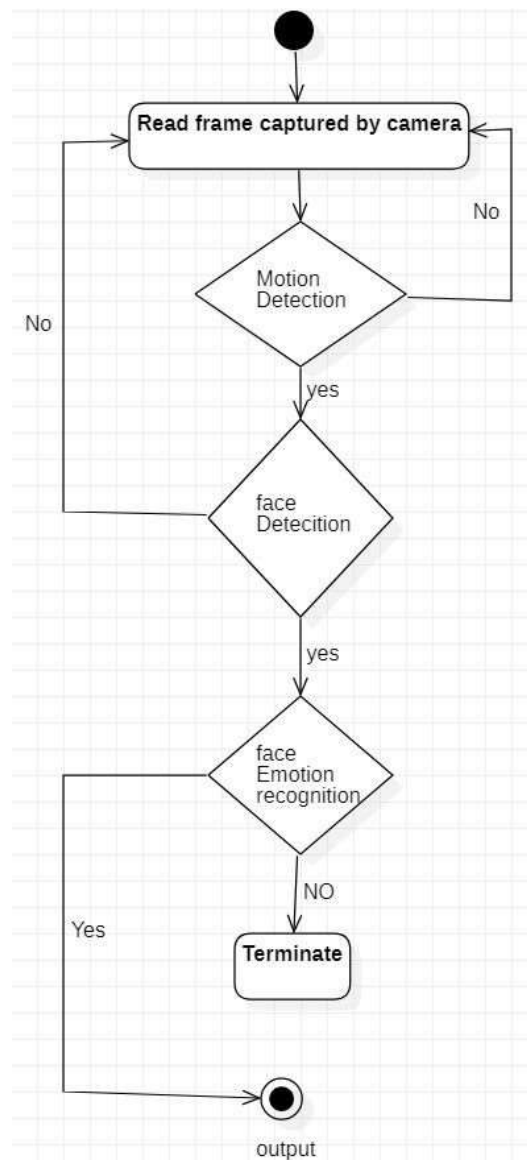


Fig 5.5 Activity diagram

5.3.4 Class Diagram

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of objectoriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

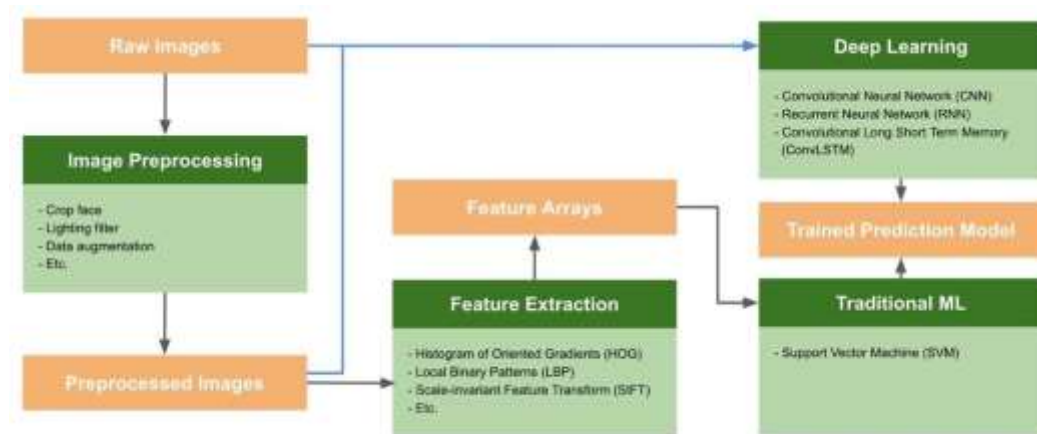


Fig 5.6 Class diagram

5.3.5 State Chart Diagram

The name of the diagram itself clarifies the purpose of the diagram and other details. It describes different states of a component in a system. The states are specific to a component/object of a system.

A Statechart diagram describes a state machine. State machine can be defined as a machine which defines different states of an object and these states are controlled by external or internal events. Activity diagram explained in the next chapter, is a special kind of a Statechart diagram. As Statechart diagram defines the states, it is used to model the lifetime of an object.

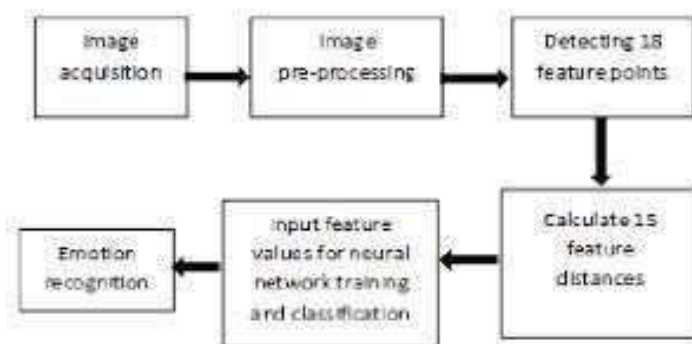


Fig 5.7 State chart diagram

CHAPTER 6

SYSTEM IMPLEMENTATION

6.1 Development Environment

1. The development environment that we have used is jupyter notebook through Google Colab.
2. The language that executes the cells in the jupyter notebook is python which is the base language.
3. To run the application we require dataset, python, libraries such as NumPy, Pandas, Matplotlib.

6.2 Dataset

In this project, the dataset used to train the models is FER-2013. The FER-2013 dataset consists of 35887 images, of which 28709 labelled images belong to the training set and the remaining 7178 images belong to the test set. The images in FER-2013 dataset is labeled as one of the seven universal emotions: Happy, Sad, Angry, Fear, Surprise, Disgust, and Neutral. Among these emotion classifications, the most images belong to 'happy' emotions account to 7215 images of the 28709 training images.

The number of images that belong to each emotion is given by returning the length of each directory using the OS module in Python. The images in FER-2013 dataset are in grayscale and is of dimensions 48x48 pixels. This dataset was created by gathering results from Google Image search of each emotion. The number of images of each emotion type is returned by the functions of 'OS' module in python. To explore the dataset further, and to understand what kind of images lie in the dataset, we plot few example images from the dataset using the 'utils' module in python.

6.3 Algorithm

A Random Forest is an ensemble technique capable of performing both regression and classification tasks with the use of multiple decision trees and a technique called Bootstrap and Aggregation, commonly known as bagging.

A random forest is a meta estimator that fits a number of classifying decision trees on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. We can see how random forest works using

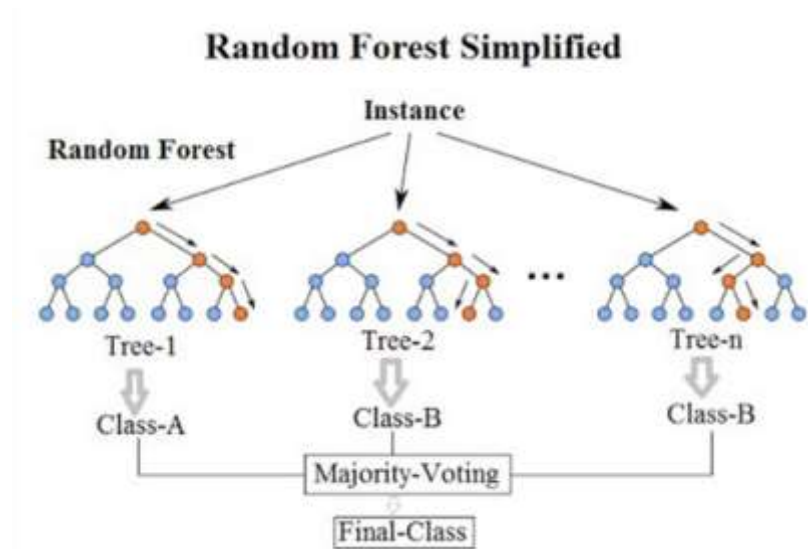


Fig 6.1 Random Forest Simplified

Sample Pictures of a Dataset:



Fig 6.2 Sample Dataset

To count number of train images for each expression:

The image expressions in our training dataset are pretty balanced, except for the 'disgust' category.

```
# count number of train images for each expression

for expression in os.listdir(base_path + "train"):
    print(str(len(os.listdir(base_path + "train/" + expression))) + " " + expression + " images")

4830 sad images
4965 neutral images
7215 happy images
436 disgust images
3171 surprise images
4097 fear images
3995 angry images
```

6.4 Training and Validation

During training, to minimize the losses of Neural Network, an algorithm called Mini-Batch Gradient Descent has been used. Minibatch Descent is a type of Gradient Descent algorithm used for finding the weights or co-efficient of artificial neural networks by splitting the training dataset into small batches. This algorithm provides more efficiency whilst training data.

Generating Training and Validation Batches

```
train_datagen = ImageDataGenerator(
    zoom_range = 0.2, shear_range = 0.2,
    horizontal_flip=True, rescale = 1./255
)

train_data = train_datagen.flow_from_directory(
    directory= "/content/train", target_size=(224,224),
    batch_size=32,
)

train_data.class_indices
```

Found 28709 images belonging to 7 classes.

```
{'angry': 0,  
'disgust': 1,  
'fear': 2,  
'happy': 3,  
'neutral': 4,  
'sad': 5,  
'surprise': 6}
```

```
val_datagen = ImageDataGenerator(rescale = 1./255 )
```

```
val_data = val_datagen.flow_from_directory(  
    directory= "/content/test",  
    target_size=(224,224), batch_size=32,  
)
```

Found 7178 images belonging to 7 classes.

Deep learning models are trained by being fed with batches of data. Keras has a very useful class to automatically feed data from a directory: ImageDataGenerator.

It can also perform data augmentation while getting the images (randomly rotating the image, zooming, etc.). This method is often used as a way to artificially get more data when the dataset has a small size.

The function flow_from_directory() specifies how the generator should import the images (path, image size, colors, etc.).

6.5 CNN Model

The CNN designed is based on sequential model and is designed to have six activation layers, of which 4 are convolutional layers and the remaining 2 are fully controlled layers.

The 4 convolutional layers consists of similar training techniques such as Batch Normalization, ReLu Activation function, Maxpooling and Dropout. Batch Normalization technique is used as it trains networks faster, allows higher learning rates, simplifies deeper networks, and also makes weights easier to initialize.

The ReLu activation function is used to increase non-linearity in the images and also makes evaluation quicker.

Maxpooling is done to reduce the dimensionality of an image, i.e., reduce its height and width. Dropout function is added to the layer to prevent overfitting of the training data.

The fourth convolutional layer has an additional training technique called Flatten. The Flatten function converts the image into a 1-Dimensional array. This then, outputs the 1-Dimensional array as input for the Fully Controlled layers.

The two fully controlled layers are designed with training techniques like Dense, Batch Normalization, ReLu activation function and Dropout.

The Dense function is used to connect each neuron of the previous layer to each neuron of the next layer. The remaining functions have similar applications.

The output layer has two training techniques, Dense and SoftMax. The SoftMax function outputs a vector that returns the probability distributions of a list of potential outcomes.

The volume of each successive layer becomes half of its previous layer and the number of channels doubles for each successive layer. The figure represents the structure of our CNN model.

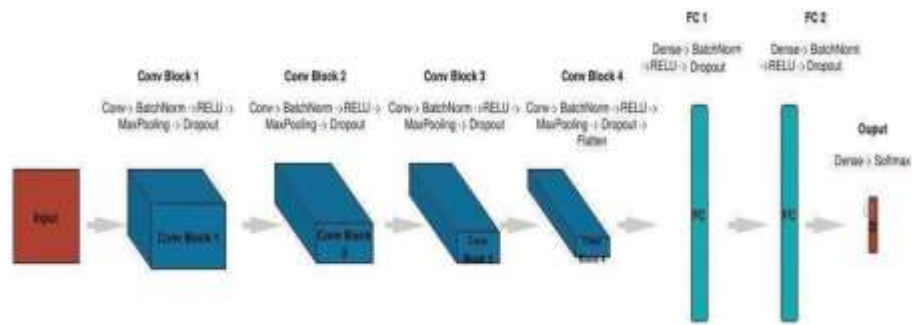


Fig 6.3 CNN Model

- A convolution operator: extracts feature from the input image using sliding matrices to preserve the spatial relations between the pixels. The following image summarizes how it works:

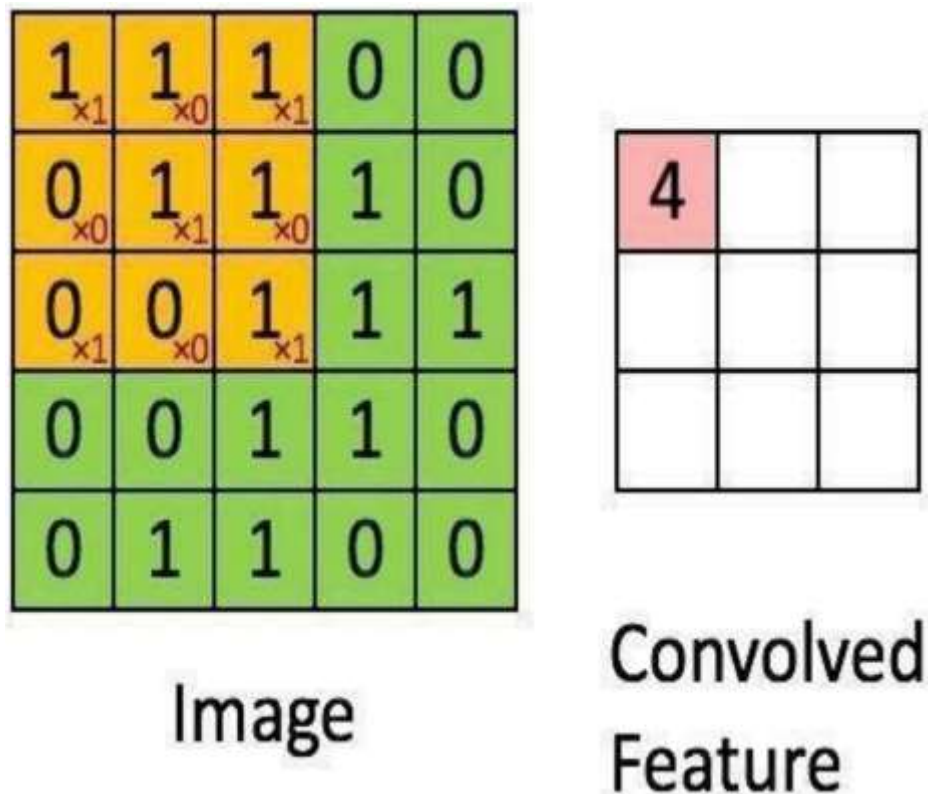


Fig 6.4 Convolution Operator

The green matrix corresponds to the raw image values. The orange sliding matrix is called A filter or kernel. this filter slides over the image by one pixel at each step(stride).

During Each step, we multiply the filter with the corresponding elements of the base matrix.

There are different types of filters and each one will be able to retrieve different image features

- We apply the ReLU function to introduce non linearity in our CNN. Other functions like tanh or sigmoid could also be used, but ReLU has been found to perform better in most situations.
- Pooling is used to reduce the dimensionality of each features while retaining the most important information. Like for the convolutional step, we apply a sliding function on our data. Different functions can be applied: max, sum, mean... The max function usually performs better.

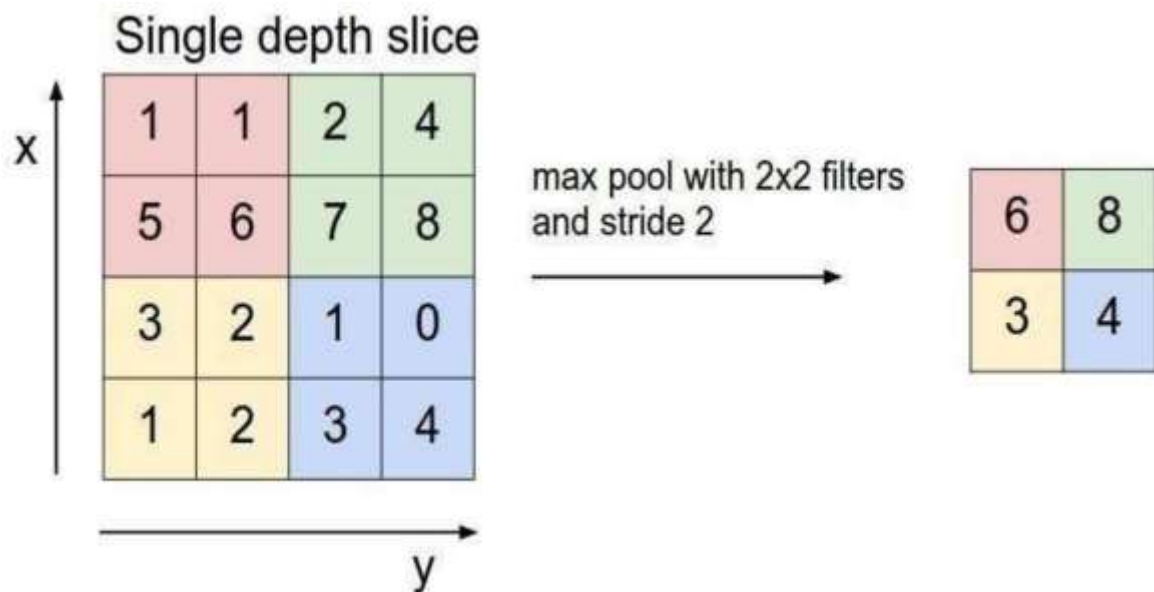


Fig 6.5 Single Depth Slice

We also use some common techniques for each layer:

- Batch normalization: improves the performance and stability of NNs by providing inputs with zero mean and unit variance.
- Dropout: reduces overfitting by randomly not updating the weights of some nodes. This helps prevent the NN from relying on one node in the layer too much.

We chose soft max as our last activation function as it is commonly used for multi-label classification.

Now that our CNN is defined, we can compile it with a few more parameters. We chose the Adam optimizer as it is one of the most computationally effective. We chose the categorical cross-entropy as our loss function as it is quite relevant for classification tasks. Our metric will be the accuracy, which is also quite informative for classification tasks on balanced datasets.

Input Layer

The input layer has pre-determined, fixed dimensions, so the image must be pre-processed before it can be fed into the layer. Normalized gray scale images of size 48 X 48 pixels from Kaggle dataset are used for training, validation and testing. For testing propose laptop webcam images are also used, in which face is detected and cropped using OpenCV Haar Cascade Classifier and normalized.

Convolution and Pooling (ConvPool) Layers:

Convolution and pooling is done based on batch processing. Each batch has N images and CNN filter weights are updated on those batches. Each convolution layer takes image batch input of four-dimension N x Color-Channel x width x height. Feature map or filter for convolution are also four dimensional (Number of feature maps in, number of feature maps out, filter width, filter height). In each convolution layer, four-dimensional convolution is calculated between image batch and feature maps. After convolution only parameter that change is image width and height.

New image width = old image width – filter width + 1

New image height = old image height – filter height + 1

After each convolution layer down sampling / subsampling is done for dimensionality reduction. This process is called Pooling. Max pooling and Average Pooling are two famous pooling method. In this project max pooling is done after convolution. Pool size of (2x2) is 12 taken, which splits the image into grid of blocks each of size 2x2 and takes maximum of 4 pixels. After pooling only height and width are affected.

Two convolution layer and pooling layer are used in the architecture. At first convolution layer size of input image batch is Nx1x48x48. Here, size of image batch is N, number of color channel is 1 and both image height and width are 48 pixel. Convolution with feature map of 1x20x5x5 results image batch is of size Nx20x44x44. After convolution pooling is done with pool size of 2x2, which results image batch of size Nx20x22x22. This is followed by second convolution layer with feature

map of 20x20x5x5, which results image batch of size $N \times 20 \times 18 \times 18$. This is followed by pooling layer with pool size 2x2, which results image batch of size $N \times 20 \times 9 \times 9$.

Fully Connected Layer

This layer is inspired by the way neurons transmit signals through the brain. It takes a large number of input features and transform features through layers connected with trainable weights.

Two hidden layers of size 500 and 300 unit are used in fully-connected layer.

The weights of these layers are trained by forward propagation of training data then backward propagation of its errors. Back propagation starts from evaluating the difference between prediction and true value, and back calculates the weight adjustment needed to every layer before. We can control the training speed and the complexity of the architecture by tuning the hyperparameters, such as learning rate and network density. Hyper-parameters for this layer include learning rate, momentum, regularization parameter, and decay.

The output from the second pooling layer is of size $N \times 20 \times 9 \times 9$ and input of first hidden layer of fully-connected layer is of size $N \times 500$. So, output of pooling layer is flattened to $N \times 1620$ size and fed to first hidden layer. Output from first hidden layer is fed to second hidden layer. Second hidden layer is of size $N \times 300$ and its output is fed to output layer of size equal to number of facial expression classes.

Output Layer

Output from the second hidden layer is connected to output layer having seven distinct classes. Using SoftMax activation function, output is obtained using the probabilities for each of the seven class. The class with the highest probability is the predicted class.

CHAPTER 7

ANALYSIS AND TESTING

7.1 Analysing the Result

We got outputs at each step of the training phase. All those outputs were saved into the 'history' variable. We can use it to plot the evolution of the loss and accuracy on both the train and validation datasets

```
plt.plot(h['accuracy'])  
plt.plot(h['val_accuracy'], c = "red")  
plt.title("acc vs v-acc")  
plt.show()
```

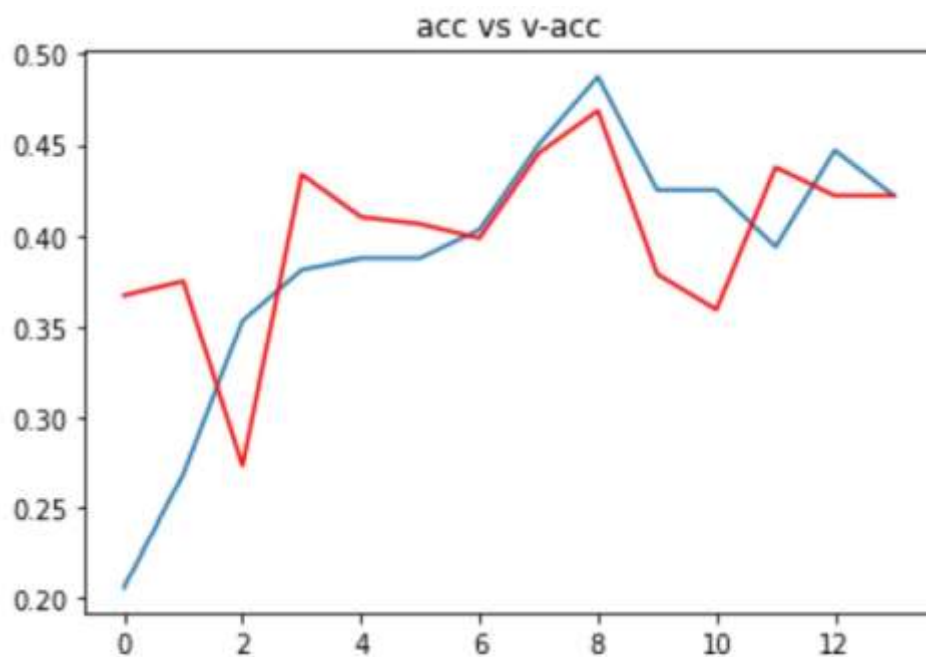


Fig 7.1: Acc vs v-Acc

```
plt.plot(h['loss'])
```

```
plt.plot(h['val_loss'], c = "red")
```

```
plt.title("loss vs v-loss")
```

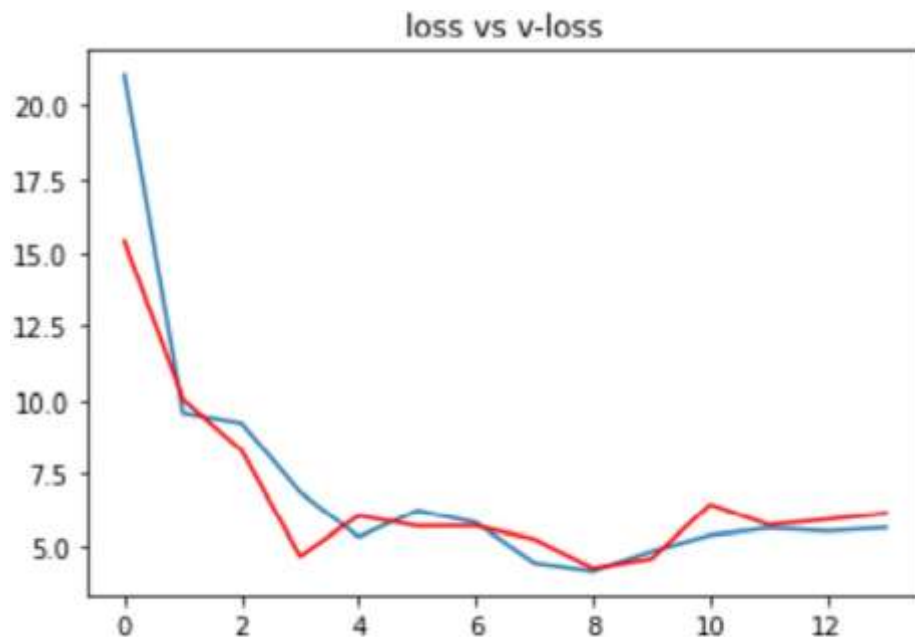


Fig 7.2 Loss vs v-Loss

The validation accuracy starts to stabilize at the end of the 50 epochs between 60% and 65% accuracy.

The training loss is slightly higher than the validation loss for the first epochs which can be surprising. Indeed, we are more used to see higher validation losses than training losses in machine learning. Here this is simply due to the presence of dropout, which is only applied during the training phase and not during the validation phase.

We can see that the training loss is becoming much smaller than the validation loss after the 20th epochs. This means that our model starts to overfit our training dataset after too much iterations. That is why the validation loss does not decrease a lot after. One solution consists in earlystopping the training of the model. We could also use some different dropout values and performing data augmentation. Those methods were tested on this dataset, but they did not significantly increase the validation accuracy although they reduced the overfitting effect. Using them slightly increased the training duration of the model.

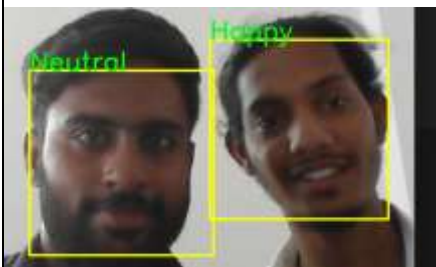
7.2 System Testing





System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

7.3 Functional Testing

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals. Functional testing is centered on the following Items: Valid Input: identified classes of valid input must be accepted. Invalid Input: identified classes of invalid input must be rejected. Functions: Identified functions must be exercised. Output: identified classes of application outputs must be exercised. Systems/Procedures: interfacing systems or procedures must.

7.4 Test Cases

| S. No | | Functional Testcases | Result | Negative/Positive Testcase | Reason |
|-------|--|------------------------------|--|----------------------------|---|
| 1 | | Verify with multiple Persons |  | Positive | It will take N no. of images for processing |

| | | | | | |
|---|--|--------------------------------|--|----------|--|
| 2 | | Verify with different Objects |  | Positive | It will detect static images also |
| 3 | | Verify with closing eyes |  | Positive | Even though it detects facial expressions |
| 4 | | Verify with animal expressions |  | Positive | It will detect the All-animal expressions |
| 5 | | Verify with no Faces |  | Positive | If there are no faces found It will shows a message "No faces" |

CONCLUSION AND FUTURE ENHANCEMENT

Using the FER-2013 dataset, a test accuracy of 62.7% is attained with this designed CNN model. The achieved results are satisfactory as the average accuracies on the FER-2013 dataset is 65% +/- 5% and therefore, this CNN model is nearly accurate. For an improvement in this project and its outcomes, it is recommended to add new parameters wherever useful in the CNN model and removing unwanted and not-so useful parameters.

Adjusting the learning rate and adapting with the location might help in improving the model. Accommodating the system to adapt to a low graded illumination setup and nullify noises in the image can also add onto the efforts to develop the CNN model. Increasing the layers in the CNN model might not deviate from the achieved accuracy, but the number of epochs can be set to higher number, to attain a higher accurate output. Though, increasing the number of epochs to a certain limit, will increase the accuracy, but increasing the number of epochs to a higher value will result in over- fitting. The similar CNN model can be trained and tested for other available datasets and checked for its accuracy.

CHAPTER 10

REFERENCES

- [1] <https://www.edureka.co/executive-programs/machine-learning-and-ai>
<https://github.com/dhruvpandey662/Emotion-detection>
- [2] B. Y. LeCun, Yann and G. Hinton, “Deep learning,” *Nature*, vol. 521, pp. 436–444.
- [3] D. Duncan, G. Shine, and C. English, “Facial emotion recognition in real time.” Available: http://cs231n.stanford.edu/reports/2016/pdfs/022_Report.pdf.
- [4] P. Lucey, J. F. Cohn, J. S. Takeo Kanade, Z. Ambadar, and I. Matthews, “A complete expression dataset for action unit and emotion-specified expression,” 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition-Workshopss, pp. 94–101, June 12, 2010.
- [5] M. Lyons, S. Akamatsu, M. Kamachi, and J. Gyoba, “A brief review of facial emotion recognition based on visual information,” *Automatic Face and Gesture Recognition*, 1998. Proceedings of Third IEEE International Conference on, pp. 200–205.
- [6] T. Kanade, J. F. Cohn, and Y. Tian, “Comprehensive database for facial expression analysis,” Proceedings of Fourth IEEE International Conference on Automatic Face and Gesture Recognition (Cat. No. PR00580), pp. 46–53.
- [7] B. C. Ko, “A brief review of facial emotion recognition based on visual information,” *Sensors*, vol. 18(2), p. 401.
- [8] <http://cs231n.github.io/>.

CHAPTER 8

RESULT

Project Organization

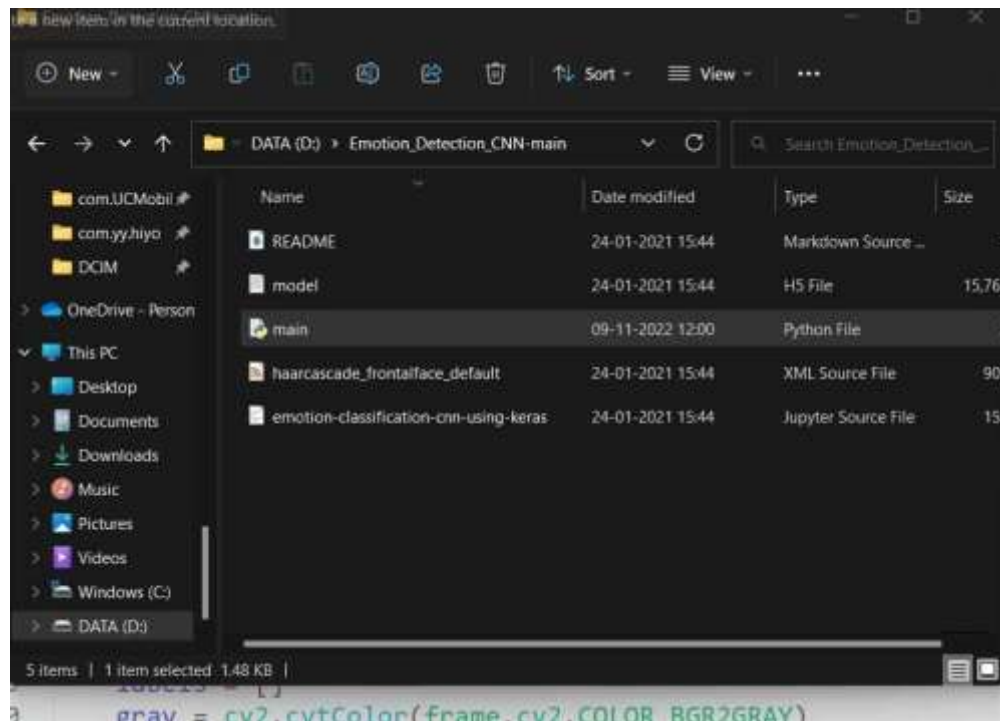


Fig 8.1 Folder of Project files

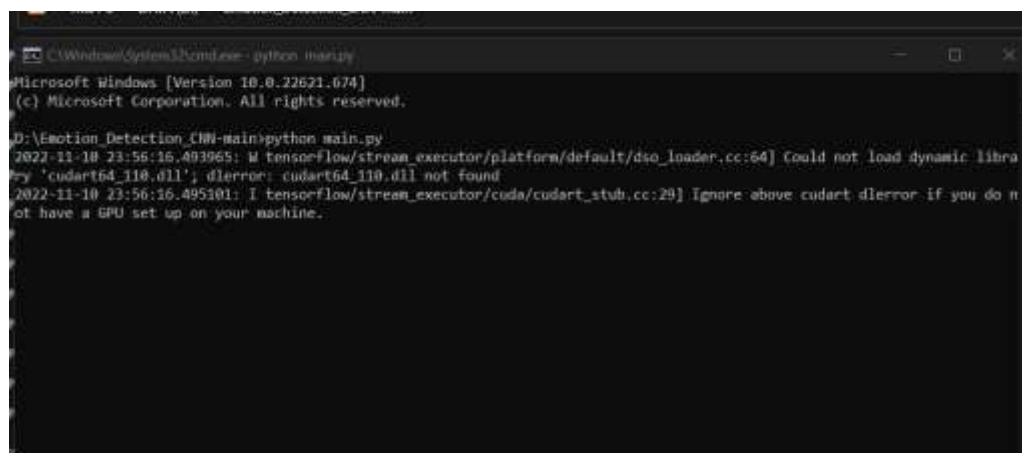


Fig 8.2 Compiling main file 1

Output Screenshots

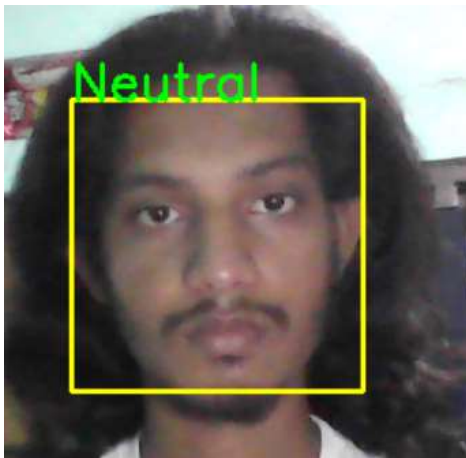


Fig 8.5 Neutral Expression



Fig 8.6 Happy Expression



Fig 8.7 Sad



Fig 8.8 Surprise Expression



Fig 8.9 Fear Expression



Fig 8.10 Angry Expression

