A Mini Project Report on

Mental Disorder Detection using FER System

Submitted to

Jawaharlal Nehru Technological University, Hyderabad

in partial fulfillment of requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

By

Mithilesh Kumar D (18BD1A052C)

Under the guidance of

Chinnam Madhuri
Assistant Professor
Department of CSE

Department of Computer Science and Engineering
KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY
Approved by AICTE, Affiliated to JNTUH
3-5-1206, Narayanaguda, Hyderabad – 500029

2021-2022

KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY

(Accredited by NBA & NAAC, Approved By A.I.C.T.E., Reg by Govt of Telangana State & Affiliated to JNTU, Hyderabad)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project entitled **Mental Disorder Detection using FER System**

being submitted by

Mithilesh Kumar D (18BD1A052C)

In partial fulfilment for the award of **Bachelor of Technology** in **Computer Science and Engineering** affiliated to the **Jawaharlal Nehru Technological University, Hyderabad** during the year 2021-22**.**

Internal Guide **Head of the Department**

(Mrs. Chinnam Madhuri) **(Dr. S. Padmaja)**

Submitted for Viva Voce Examination held on _____

External Examiner

Unit of Keshav Memorial Educational Society

#: 3-5-1026 Narayanguda Hyderabad 500029.

040-3261407 www.kmit.in e-mail: principal@kmit.in

Vision of KMIT

Producing quality graduates trained in the latest technologies and related tools and striving to make India a world leader in software and hardware products and services. To achieve academic excellence by imparting in depth knowledge to the students, facilitating research activities and catering to the fast growing and ever- changing industrial demands and societal needs.

Mission of KMIT

- To provide a learning environment that inculcates problem solving skills, professional, ethical responsibilities, lifelong learning through multi modal platforms and prepare students to become successful professionals.
- To establish industry institute Interaction to make students ready for the industry.
- To provide exposure to students on latest hardware and software tools.
- To promote research based projects/activities in the emerging areas of technology convergence.
- To encourage and enable students to not merely seek jobs from the industry but also to create new enterprises.
- To induce a spirit of nationalism which will enable the student to develop, understand lndia's challenges and to encourage them to develop effective solutions.
- To support the faculty to accelerate their learning curve to deliver excellent service to students.

Vision & Mission of CSE

Vision of the CSE

To be among the region's premier teaching and research Computer Science and Engineering departments producing globally competent and socially responsible graduates in the most conducive academic environment.

Mission of the CSE

- To provide faculty with state of the art facilities for continuous professional development and research, both in foundational aspects and of relevance to emerging computing trends.
- To impart skills that transform students to develop technical solutions for societal needs and inculcate entrepreneurial talents.
- To inculcate an ability in students to pursue the advancement of knowledge in various specializations of Computer Science and Engineering and make them industry-ready.
- To engage in collaborative research with academia and industry and generate adequate resources for research activities for seamless transfer of knowledge resulting in sponsored projects and consultancy.
- To cultivate responsibility through sharing of knowledge and innovative computing solutions that benefit the society-at-large.
- To collaborate with academia, industry and community to set high standards in academic excellence and in fulfilling societal responsibilities.

PROGRAM OUTCOMES (POs)

1. Engineering Knowledge: Apply the knowledge of mathematics, science, engineering fundamentals and an engineering specialization to the solution of complex engineering problems.

2. Problem analysis: Identify formulate, review research literature, and analyse complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences

3. Design/development of solutions: Design solutions for complex engineering problem and design system component or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural societal, and environmental considerations.

4. Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5. Modern tool usage: Create select, and, apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.

6. The engineer and society: Apply reasoning informed by the contextual knowledge to societal, health, safety. legal und cultural issues and the consequent responsibilities relevant to professional engineering practice.

7. Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts and demonstrate the knowledge of, and need for sustainable development.

8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams and in multidisciplinary settings.

10. Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write

effective reports and design documentation make effective presentations, and give and receive clear instructions.

11. Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO1: An ability to analyse the common business functions to design and develop appropriate Computer Science solutions for social upliftment's.

PSO2: Shall have expertise on the evolving technologies like Python, Machine Learning, Deep Learning, Internet of Things (IOT), Data Science, Full stack development, Social Networks, Cyber Security, Big Data, Mobile Apps, CRM, ERP etc.

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

PEO1: Graduates will have successful careers in computer related engineering fields or will be able to successfully pursue advanced higher education degrees.

PEO2: Graduates will try and provide solutions to challenging problems in their profession by applying computer engineering principles.

PEO3: Graduates will engage in life-long learning and professional development by rapidly adapting changing work environment.

PEO4: Graduates will communicate effectively, work collaboratively and exhibit high levels of professionalism and ethical responsibility.

PROJECT OUTCOMES

P1: Learnt how to implement a neural network and train efficiently

P2: Worked on the model which classifies 7 classes of expressions

P3: Used OpenCV for capturing real-time images

P4: Able to connect and store the expressions into the database for doctors' reference

LOW - 1

MEDIUM - 2

HIGH - 3

PROJECT OUTCOMES MAPPING PROGRAM OUTCOMES

PO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
P1	3		2	3	2				1		2	1
P2		1		2					2	2	2	
P3	2				3					1		
P4	3		1		1					1		1

PROJECT OUTCOMES MAPPING PROGRAM SPECIFIC OUTCOMES

PSO	PSO1	PSO2
P1	3	3
P2	2	3
P3	2	3
P4	1	2

PROJECT OUTCOMES MAPPING PROGRAM EDUCATIONAL OBJECTIVES

PEO	PEO1	PEO2	PEO3	PEO4
P1	2	2	1	2
P2	1			2
P3	2			
P4	1			

DECLARATION

We hereby declare that the project report entitled "**MENTAL DISORDER DETECTION USING FER SYSTEM"** is done in the partial fulfillment for the award of the Degree in Bachelor of Technology in Computer Science and Engineering affiliated to Jawaharlal Nehru Technological University, Hyderabad. This project has not been submitted anywhere else.

B. ABHISHEK (18BD1A0528)

A. MANUDEEP (18BD1A0521)

M. SHARATH CHANDRA REDDY (18BD1A053E)

MITHILESH KUMAR D (18BD1A052C)

ACKNOWLEDGMENT

We take this opportunity to thank all the people who have rendered their full support to our project work.

We render our thanks to **Dr. Maheshwar Dutta**, B.E., M Tech., Ph.D., Principal who encouraged us to do the Project.

We are grateful to **Mr. Neil Gogte**, Director for facilitating all the amenities required for carrying out this project.

We express our sincere gratitude to **Mr. S. Nitin**, Director and **Mrs. Anuradha**, Dean Academics for providing an excellent environment in the college.

We are also thankful to **Dr. S. Padmaja**, Head of the Department for providing us with both time and amenities to make this project a success within the given schedule.

We are also thankful to our guide **Mrs. Chinnam Madhuri**, for his/her valuable guidance and encouragement given to us throughout the project work.

We would like to thank the entire CSE Department faculty, who helped us directly and indirectly in the completion of the project. We sincerely thank our friends and family for their constant motivation during the project work.

B. ABHISHEK (18BD1A0528)

A. MANUDEEP (18BD1A0521)

M. SHARATH CHANDRA REDDY (18BD1A053E)

MITHILESH KUMAR D (18BD1A052C)

CONTENT

DESCRIPTION	PAGE NO.
ABSTRACT	**i**
LIST OF FIGURES	**ii**
LIST OF TABLES	**iii**
CHAPTERS	
1. INTRODUCTION	**1-5**
1.1. Purpose of the Project	1
1.2. Problem with the Existing System	1
1.3. Proposed System	2
1.4. Scope of the Project	2
1.5 Modules of the Project	2
1.6 CNN and it's architecture	2
2. SOFTWARE REQUIREMENTS SPECIFICATIONS	**7-9**
2.1. What is SRS?	7
2.2. Role of SRS	7
2.3. Requirements specification document	7
2.4. Functional requirements	8
2.5. Non-functional Requirements	9
3. LITERATURE SURVEY	**11-12**
4. SYSTEM DESIGN	**14-17**
4.1. Introduction to UML	14
4.2. UML Diagrams	15
4.2.1 Use Case diagram	16
4.2.3 Activity diagram	17
5. IMPLEMENTATION	**19-25**

5.1. Code Snippets 19

6. TESTING 27-32

6.1. Introduction to Testing 27

6.2. Test Cases 29

7. SCREENSHOTS 34-38

8.FURTHER ENHANCEMENTS 40

9.CONCLUSION 42

10.REFERENCES 44

ABSTRACT

Now-a-days, facial expression recognition is a significant area in computer vision. It is computerized software that helps to identify human feelings such as Happiness, Anger, Sadness, Fear, Disgust, and Surprise. Moreover, facial expression recognition has its foothold on public places. Examples of human emotions on public place like severe argument between two persons, driving vehicle with anger and so on. By understanding emotion of the person, able to prevent any gruesome act or danger. This project describes the survey of Face Expression Recognition (FER) techniques which include the two major stages such as feature extraction and classification. Our system takes the subtle advantage by using the ML paradigms and creating a sophisticated model which helps understand the human behaviour and psychology through their expressions and being an entry point for medical practitioners to study their patterns using our ML model which stores all the required data in a backend service which can be referred anytime by the doctors to consolidate their reviews.

LIST OF FIGURES

S. No	LIST OF FIGURES	PAGE NO
3.1	CNN Architecture	12
4.2.1	UML Diagram	12
4.2.2	Sequence diagram	14
4.2.3	Activity Diagram	15

LIST OF TABLES

S. No	LIST OF TABLES	PAGE NO
6.2.1	Test Case 1	24
6.2.2	Test Case 2	25
6.2.3	Test Case 3	25
6.2.4	Test Case 4	26
6.2.5	Test Case 5	27

CHAPTER -1

INTRODUCTION

1.1 Purpose of Project

Now-a-days, emotion recognition is a significant area in computer vision. It is computerized software that helps to identify human feelings such as Happiness, Anger, Sadness, Fear, Disgust, and Surprise. Moreover, emotion Recognition has its foothold on public places. Examples of human emotions on public place like severe argument between two persons, driving vehicle with anger and so on. By understanding emotion of the person, able to prevent any gruesome act or danger. This project describes the survey of Face Expression Recognition (FER) techniques which include the three major stages such as pre-processing, feature extraction and classification. Our system takes the subtle advantage by using the ML paradigms and creating a sophisticated model which helps understand the human behaviour and psychology through their expressions and being an entry point for medical practitioners to study their patterns using our ML model which stores all the required data in a backend service which can be referred anytime by the doctors to consolidate their reviews.

1.2 Problems with Existing System

Older FER systems relied mostly on computation done by CPU. CPUs are best at handling single, more complex calculations sequentially.
There is no individual feature extraction, the input image is compared to model as a whole and classification is done. There is a lot of irrelevant information such as colour which has no impact on facial expression. In terms of emotion recognition, there is no indefinite way or method which is the univocal solution. A lot of solution have come and many to comes in near future with significant improvement in terms of efficiency, accuracy, and usability. In past and the current research shows that multimodalities dominated the area of emotion recognition than unimodality.

1.3 Proposed System

This task of performing FER was done in five steps:

1. Collection of data
2. Preprocessing and Feature extraction of the data
3. Implementing Logistic Regression and training and testing the model
4. Evaluating the performance of the model through accuracy
5. Predicting the result of user's input review

1.4 Scope of the Project

Facial Expression Recognition System aims to determine the emotions of a person which are classified into 7 defined classes. This can be achieved by CNN and DN models. So, when an image is given as an input it is classified into one of the 7 emotions.

1.5 Modules of the Project

- The architecture of the neural network is implemented.
- The implemented architecture is trained with *fer2013* dataset from Kaggle.
- The real-time faces are extracted and sent to the neural network for classifying the expressions.
- The predicted expression of a person at every particular timestamp are stored in the database.

1.6 CNN and it's architecture

Convolutional neural network (CNN) is the most popular way of analysing images. CNN is different from a multi-layer perceptron (MLP) as they have hidden layers, called convolutional layers. The proposed method is based on a two-level CNN framework. The first level recommended is background removal, used to extract emotions from an image. Here, the conventional CNN network module is used to extract primary expressional vector (EV). The expressional vector (EV) is generated by tracking down relevant facial points of importance. EV is directly related to changes in expression. The EV is obtained using a basic perceptron unit applied on a background-removed face image. In the proposed FERC model,

we also have a non-convolutional perceptron layer as the last stage. Each of the convolutional layers receives the input data (or image), transforms it, and then outputs it to the next level. This transformation is convolution operation. All the convolutional layers used are capable of pattern detection. Within each convolutional layer, four filters were used. The input image fed to the first-part CNN (used for background removal) generally consists of shapes, edges, textures, and objects along with the face. The edge detector, circle detector, and corner detector filters are used at the start of the convolutional layer 1. Once the face has been detected, the second-part CNN filter catches facial features, such as eyes, ears, lips, nose, and cheeks. The edge detection filters used in this layer. The second-part CNN consists of layers with 3×33×3 kernel matrix, e.g., [0.25, 0.17, 0.9; 0.89, 0.36, 0.63; 0.7, 0.24, 0.82]. These numbers are selected between 0 and 1 initially. These numbers are optimized for EV detection, based on the ground truth we had, in the supervisory training dataset. Here, we used minimum error decoding to optimize filter values. Once the filter is tuned by supervisory learning, it is then applied to the background-removed face (i.e., on the output image of the first-part CNN), for detection of different facial parts (e.g., eye, lips. nose, ears, etc.)

Max Pooling:

Max Pooling is a pooling operation that calculates the maximum value for patches of a feature map, and uses it to create a sample (pooled) feature map. It is usually used after a convolutional layer. It adds a small amount of translation invariance - meaning translating the image by a small amount does not significantly affect the values of most pooled outputs.

Pooling layers provide an approach to down sampling feature maps by summarizing the presence of features in patches of the feature map. Two common pooling methods are average pooling and max pooling that summarize the average presence of a feature and the most activated presence of a feature respectively.

- Pooling is required to down sample the detection of features in feature maps.
- How to calculate and implement average and maximum pooling in a convolutional neural network.
- How to use global pooling in a convolutional neural network

- Pooling involves selecting a pooling operation, much like a filter to be applied to feature maps. The size of the pooling operation or filter is smaller than the size of the feature map; specifically, it is almost always 2×2 pixels applied with a stride of 2 pixels.

- This means that the pooling layer will always reduce the size of each feature map by a factor of 2, e.g. each dimension is halved, reducing the number of pixels or values in each feature map to one quarter the size. For example, a pooling layer applied to a feature map of 6×6 (36 pixels) will result in an output pooled feature map of 3×3 (9 pixels).

Normalization:

The purpose of normalization is to transform data in a way that they are either dimensionless and/or have similar distributions. This process of normalization is known by other names such as standardization, feature scaling etc. Normalization is an essential step in data pre-processing in any machine learning application and model fitting.

Several methods are applied for normalization, three popular and widely used techniques are as follows:

- **Rescaling:** also known as "min-max normalization", it is the simplest of all methods and calculated as:

- **Mean normalization:** This method uses the mean of the observations in the transformation process:

- **Z-score normalization:** Also known as standardization, this technic uses Z-score or "standard score". It is widely used in machine learning algorithms such as SVM and logistic regression:

Here, z is the standard score, μ is the population mean and σ is the population standard deviation.

Normalization compresses data within a certain range, reduces the variance and applies equal weights to all features. You lose a lot of important information in the process.

Normalization is a technique applied during data preparation so as to change the values of numeric columns in the dataset to use a common scale. This is generally applied when the attributes of the dataset have different ranges.

- In cases when the ranges are different, the model may misunderstand the data and give more weightage to the attribute having a higher range. That's why to create a non-biased model, techniques like normalization and standardization are applied.
- Normalization also helps the gradient descent algorithm to converge to the global minima quickly.

The formula for normalization:

***x* normalized = (*x* – *x* minimum) / (*x* maximum – *x* minimum)**

CHAPTER -2

SYSTEM REQUIREMENT SPECIFICATIONS

2.1 What is SRS?

Software Requirement Specification (SRS) is the starting point of the software developing activity. As system grew more complex it became evident that the goal of the entire system cannot be easily comprehended. Hence the need for the requirement phase arose. The software project is initiated by the client needs. The SRS is the means of translating the ideas of the minds of clients (the input) into a formal document (the output of the requirement phase.)

The SRS phase consists of two basic activities:

Problem/Requirement Analysis:

The process is order and more nebulous of the two, deals with understand the problem, the goal and constraints.

Requirement Specification:

Here, the focus is on specifying what has been found giving analysis such as representation, specification languages and tools, and checking the specifications are addressed during this activity.

The Requirement phase terminates with the production of the validate SRS document. Producing the SRS document is the basic goal of this phase.

2.2 Role of SRS

The purpose of the Software Requirement Specification is to reduce the communication gap between the clients and the developers. Software Requirement Specification is the medium though which the client and user needs are accurately specified. It forms the basis of software development. A good SRS should satisfy all the parties involved in the system.

2.3 Requirements Specification Document

Software requirement specification (SRS) is a document that describes the nature of project, software or application. In simple words, SRS document is a manual of a project provided it is prepared before you kick-start a project/application. This document is also known by the names SRS report, software document. A software document is primarily prepared for a project, software or any kind of application.

There are a set of guidelines to be followed while preparing the software requirement specification document. This includes the purpose, scope, functional and nonfunctional requirements, software and hardware requirements of the project. In addition to this, it also contains the information about environmental conditions required, safety and security requirements, software quality attributes of the project etc.

The purpose of SRS (Software Requirement Specification) document is to describe the external behavior of the application developed or software. It defines the operations, performance and interfaces and quality assurance requirement of the application or software. The complete software requirements for the system are captured by the SRS.
This section introduces the requirement specification document for Word Building Game using Alexa which enlists functional as well as non-functional requirements.

2.4 Functional Requirements

For documenting the functional requirements, the set of functionalities supported by the system are to be specified. A function can be specified by identifying the state at which data is to be input to the system, its input data domain, the output domain, and the type of processing to be carried on the input data to obtain the output data. Functional requirements define specific behavior or function of the application. Following are the functional requirements:

1. The system can display the expression of the person accurately.
2. The system can store the expressions in MongoDB.

2.5 Non-Functional Requirements

A non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behavior. Especially these are the constraints the system must work within. Following are the non-functional requirements:

1. Portable – The system can be easily transferred from one environment to another.
2. Scalable – The system can accommodate larger volumes of data over time.
3. Accurate – The supervised model used in the project is more accurate compared to unsupervised model.

2.6 Software Requirements

Operating System	:	Windows 10/ MacOS/ Linux.

2.7 Hardware Requirements

Processor	:	Intel core i3 and above.
Hard Disk	:	2 GB or above.
RAM	:	2 GB or above.
Web camera	:	VGA or above.
GPU	:	Nvidia GeForce GTX 1060ti or higher

CHAPTER -3

LITERATURE SURVEY

Recent Related Work in the Relevant Field :

Previous works are focused on eliciting results from unimodal systems. Machines used to predict emotion by only facial expressions or only vocal sounds. After a while, multimodal systems that use more than one features to predict emotion has more effective and gives more accurate results. So that, the combination of features such as audio-visual expressions, EEG, body gestures have been used since

Research has demonstrated that deep neural networks can effectively generate discriminative features that approximate the complex non-linear dependencies between features in the original set. These deep generative models have been applied to speech and language processing, as well as emotion recognition tasks.

A. Yao, D. Cai, P. Hu, S. Wang, L. Shan, and Y. Chen used a well-designed Convolutional Neural Network (CNN) architecture regarding the video based emotion recognition. They proposed the method named as HOLONET has three critical considerations in network design.

(1) To reduce redundant filters and enhance the non-saturated non-linearity in the lower convolutional layers, they used modified Concatenated Rectified Linear Unit (CReLU) instead of ReLU.

(2) To enjoy the accuracy gain from considerably increased network depth and maintain efficiency, they combine residual structure and CReLU to construct the middle layers.

(3) To broaden network width and introduce multi-scale feature extraction property, the topper layers are designed as a variant of the inception-residual structure. This method more realistic than other methods.

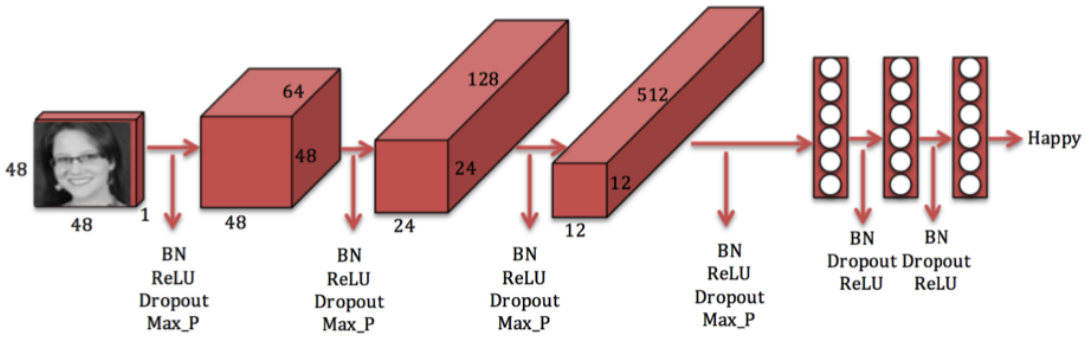


Fig 3.1 CNN Architecture

Reference and year	Approach and Method	Performance
Wei-Long Zheng and Bao-Liang Lu (2016)	EEG-based affective models without labeled target data using transfer learning techniques (TCA-based Subject Transfer)	Positive (85.01%) emotion recognition rate is higher than other approaches but neutral (25.76%) and negative (10.24%) emotions are often confused with each other.
Zixing Zhang, Fabien Ringeval, Fabien Ringeval, Eduardo Coutinho, Erik Marchi and Björn Schüller (2016)	Semi-Supervised Learning (SSL) technique	Delivers a strong performance in the classification of high/low emotional arousal (UAR = 76.5%), and significantly outperforms traditional SSL methods by at least 5.0% (absolute gain).
Y. Fan, X. Lu, D. Li, and Y. Liu. (2016)	Video-based Emotion Recognition Using CNN-RNN and C3D Hybrid Networks	Achieved accuracy 59.02% (without using any additional Emotion labeled video clips in training set) which is the best till now.
A. Yao, D. Cai, P. Hu, S. Wang, L. Shan and Y. Chen (2016)	HoloNet: towards robust emotion recognition in the wild	Achieved mean recognition rate of 57.84%.
Yelin Kim and Emily Mower Provos (2016)	Data driven framework to explore patterns (timings and durations) of emotion evidence, specific to individual emotion classes	Achieved 65.60% UW accuracy, 1.90% higher than the baseline.

CHAPTER -4

SYSTEM DESIGN

4.1 Introduction to UML

The Unified Modeling Language allows the software engineer to express an analysis model using the modeling notation that is governed by a set of syntactic, semantic and pragmatic rules. A UML system is represented using five different views that describe the system from distinctly different perspective. Each view is defined by a set of diagrams, which is as follows:

1. User Model View

   This view represents the system from the users' perspective. The analysis representation describes a usage scenario from the end-users' perspective.

2. Structural Model View

   In this model, the data and functionality are arrived from inside the system. This model view models the static structures.

3. Behavioral Model View

   It represents the dynamic of behavioral as parts of the system, depicting he interactions of collection between various structural elements described in the user model and structural model view.

4. Implementation Model View

   In this view, the structural and behavioral as parts of the system are represented as they are to be built.

5. Environmental Model View

   In this view, the structural and behavioral aspects of the environment in which the system is to be implemented are represented.

4.2 UML Diagrams

4.2.1 Use Case Diagram

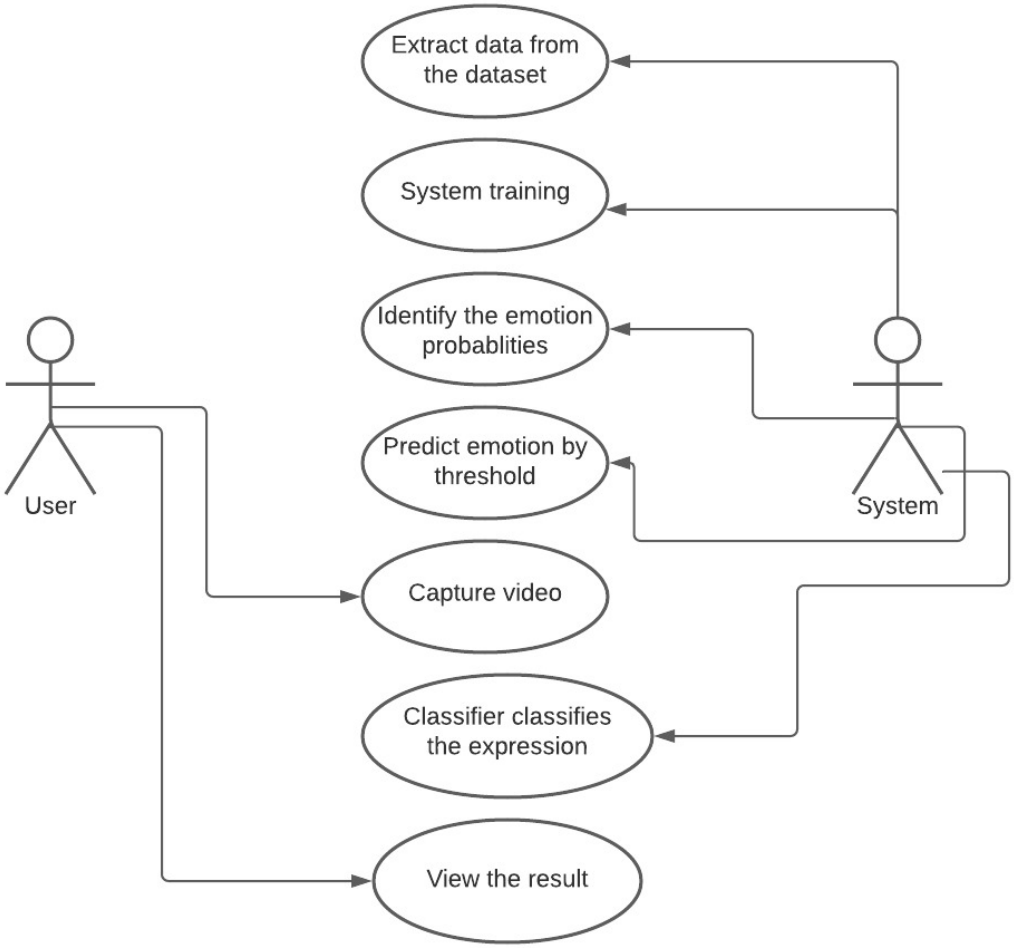


Fig 4.2.1 Use case diagram

4.2.2 Sequence Diagram

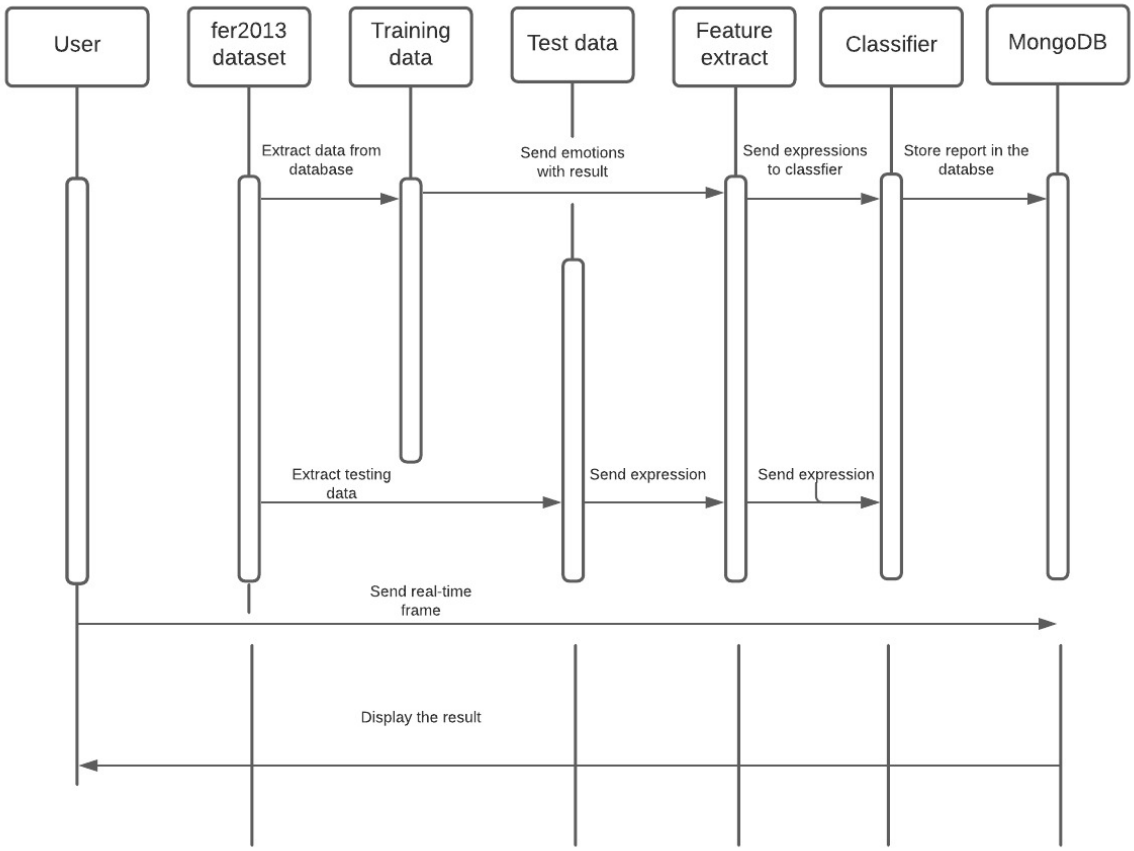


Fig 4.2.2 Sequence diagram

4.2.3 Activity Diagram

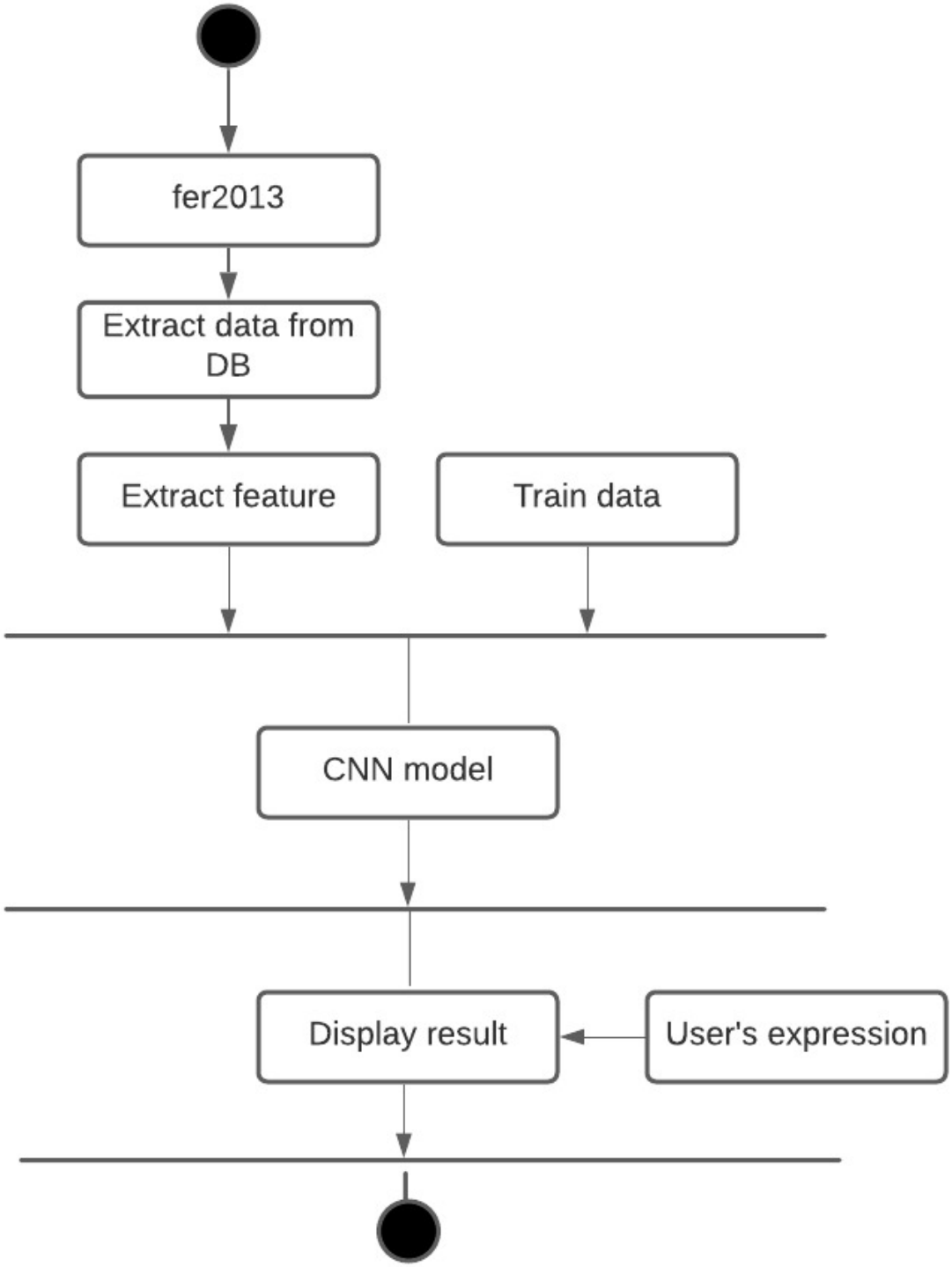


Fig 4.2.3 Activity diagram

CHAPTER -5

IMPLEMENTATION

5.1 Code Snippets

model.py

```
from keras.models import Sequential
from keras.layers import Dense,Flatten,Dropout
from keras.layers.convolutional import Conv2D,MaxPooling2D
from tensorflow.keras.layers import BatchNormalization

def create_convolutional_model(classes):
    model = Sequential()

model.add(Conv2D(32,kernel_size=(2,2),strides=(1,1),activation='relu',input_shape=(48,48,1
)))
    model.add(BatchNormalization())
    model.add(MaxPooling2D(pool_size=(2,2),strides=(2,2)))
    model.add(Dropout(0.25))

    model.add(Conv2D(filters=64,kernel_size=(2,2),strides=(1,1),activation='relu'))
    model.add(BatchNormalization())
    model.add(MaxPooling2D(pool_size=(2,2),strides=(1,1)))
    model.add(Dropout(0.25)) #to prevent neural network from overfitting

    model.add(Conv2D(filters=128,kernel_size=(2,2),strides=(1,1),activation='relu'))
    model.add(BatchNormalization())
    model.add(MaxPooling2D(pool_size=(2,2),strides=(1,1)))
    model.add(Dropout(0.25))

    model.add(Conv2D(filters=256,kernel_size=(2,2),strides=(1,1),activation='relu'))
    model.add(BatchNormalization())
```

```
model.add(MaxPooling2D(pool_size=(2,2),strides=(1,1)))
model.add(Dropout(0.25))

model.add(Flatten())

model.add(Dense(256,activation='relu'))
model.add(BatchNormalization())
model.add(Dropout(0.25))

model.add(Dense(512,activation='relu'))
model.add(BatchNormalization())
model.add(Dropout(0.25))

model.add(Dense(classes,activation='softmax'))

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

return model
```

train.py

```
import pandas as pd
import numpy as np
from model import create_convolutional_model
from sklearn.model_selection import train_test_split
from tensorflow.keras.utils import to_categorical

def main():
    # Read data from csv file
    data = pd.read_csv("fer2013.csv")

    labels = data.iloc[:,[0]].values
    pixels = data['pixels']

    # Facial Expressions
    Expressions = {
        0:"Angry",
        1:"Disgust",
        2:"Fear",
        3:"Happy",
        4:"Sad",
        5:"Surprise",
        6:"Neutral"
    }
    labels = to_categorical(labels,len(Expressions))

    #converting pixels to Gray Scale images of 48X48
    images = np.array([np.fromstring(pixel, dtype=int, sep=" ")for pixel in pixels])
    images = images/255.0
    images = images.reshape(images.shape[0],48,48,1).astype('float32')

    classes = 7
```

```
    model = create_convolutional_model(classes)
    print(model.summary())

    # Splitting data into training and test data
    train_images,test_images,train_labels,test_labels =
train_test_split(images,labels,test_size=0.2,random_state=0)

    # Train the CNN
    model.fit(train_images,train_labels,batch_size=105,epochs=30,verbose=2)

if __name__ == "__main__":
    main()
```

predict.py

```
import cv2
import numpy as np
from model import create_convolutional_model
from mongo_integrate import add_data_to_mongo, connect_mongo

def make_prediction(unknown):
    unknown=cv2.resize(unknown,(48,48))
    unknown=unknown/255.0
    unknown=np.array(unknown).reshape(-1,48,48,1)
    predict=np.argmax(model.predict(unknown),axis = 1)
    return predict[0]

def face_in_video():
    face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades +
'haarcascade_frontalface_default.xml')
    cap = cv2.VideoCapture(0)

    while True:
        ret, img=cap.read()
        gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
        faces = face_cascade.detectMultiScale(gray,1.3,5)
        ans = "unknown expression"
        for (x,y,w,h) in faces:
            sub_face = gray[y:y+h, x:x+w]
            cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)
            res = make_prediction(sub_face)
            ans = str(Expressions[res])
            font = cv2.FONT_HERSHEY_SIMPLEX
            cv2.putText(
                img,
                ans,
```

```
                (50, 50),
                font,
                3,
                (0, 255, 255),
                2,
                cv2.LINE_4
            )
        cv2.imshow('img',img)
        if ans != "unknown expression":
            add_data_to_mongo(db, ans)
        if cv2.waitKey(1) & 0xFF ==ord('q'):
            break
    cap.release()
    cv2.destroyAllWindows()

if __name__=='__main__':
    Expressions =
{0:"Angry",1:"Disgust",2:"Fear",3:"Happy",4:"Sad",5:"Surprise",6:"Neutral"}
    filename = 'model_weights.hdf5'
    model = create_convolutional_model(7)
    model.load_weights(filename)
    db = connect_mongo()
    face_in_video()
```

mongo_integrate.py

```
from keras.engine.sequential import clear_previously_created_nodes
import motor.motor_tornado
from datetime import datetime

MONGODB_URL = "mongodb://localhost:27017/admin?retryWrites=true&w=majority"

def connect_mongo():
    try:
        client = motor.motor_tornado.MotorClient(MONGODB_URL)
        db = client.FER
        print("Connected to MongoDB")
        return db
    except Exception as err:
        print(Exception, err)

def add_data_to_mongo(db, expression):
    try:
        json = {"time": datetime.now(), "expression": expression}
        db.Expressions.insert_one(json)
    except Exception as err:
        print(Exception, err)

def main():
    db = connect_mongo()
    table = db.Expressions
    x = table.delete_many({})
    print("All documents deleted.")

main()
```

CHAPTER -6

TESTING

6.1 Introduction to Testing

Testing is the process of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not. Testing is executing a system in order to identify any gaps, errors, or missing requirements in contrary to the actual requirements.

According to ANSI/IEEE 1059 standard, Testing can be defined as - A process of analyzing a software item to detect the differences between existing and required conditions (that is defects/errors/bugs) and to evaluate the features of the software item.

Who does Testing?

It depends on the process and the associated stakeholders of the project(s). In the IT industry, large companies have a team with responsibilities to evaluate the developed software in context of the given requirements. Moreover, developers also conduct testing which is called Unit Testing. In most cases, the following professionals are involved in testing a system within their respective capacities:

- Software Tester
- Software Developer
- Project Lead/Manager
- End User

Levels of testing include different methodologies that can be used while conducting software testing. The main levels of software testing are:

- Functional Testing
- Non-functional Testing

Functional Testing

This type of black box testing that is based on the specifications of the software that is to be tested. The application is tested by providing input and then the results are examined that need to confirm to the functionality it was intend for. Functional testing of a software is conducted

on a complete, integrated system to evaluate the system's compliance with its specified requirements.

Software Testing Life Cycle

The process of testing a software in a well-planned and systematic way is known as software testing lifecycle (STLC).

Different organizations have different phases in STLC however generic Software Test Life Cycle (STLC) for waterfall development model consists of the following phases.

1. Requirements Analysis
2. Test Planning
3. Test Analysis
4. Test Design

Requirements Analysis

In this phase testers analyze the customer requirements and work with developers during the design phase to see which requirements are testable and how they are going to test those requirements.

It is very important to start testing activities from the requirements phase itself because the cost of fixing defect is very less if it is found in requirements phase rather than in future phases.

Test Planning

In this phase all the planning about testing is done like what needs to be tested, how the testing will be done, test strategy to be followed, what will be the test environment, what test methodologies will be followed, hardware and software availability, resources, risks etc. A highlevel test plan document is created which includes all the planning inputs mentioned above and circulated to the stakeholders.

Test Analysis

After test planning phase is over test analysis phase starts, in this phase we need to dig deeper into project and figure out what testing needs to be carried out in each SDLC phase. Automation activities are also decided in this phase, if automation needs to be done for

software product, how will the automation be done, how much time will it take to automate and which features need to be automated. Non-functional testing areas (Stress and performance testing) are also analysed and defined in this phase.

Test Design

In this phase various black-box and white-box test design techniques are used to design the test cases for testing, testers start writing test cases by following those design techniques, if automation testing needs to be done then automation scripts also needs to written in this phase.

6.2 Test Cases

6.2.1 Test Case 1

Test Scenario ID	Check web camera	**Test Case ID**	Check web camera
Test Case Description	Command line Interface	**Test Priority**	High
Pre-Requisite	NA	**Post Requisite**	NA

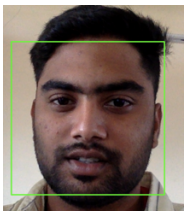
Test Execution Steps:

S. No	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**
1	Run predict.py	NA	Connected	Connected	Pass	NA

6.2.2 Test Case 2

Test Scenario ID	Predict expression	**Test Case ID**	Predict expression
Test Case Description	Command line Interface	**Test Priority**	High
Pre-Requisite	NA	**Post Requisite**	NA

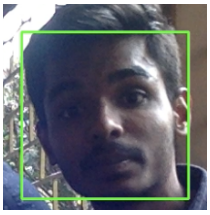
Test Execution Steps:

S. No	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**
1	Image to be predicted is sent to the CNN		Neutral	Neutral	Pass	NA

6.2.3 Test Case 3

Test Scenario ID	Predict expression	**Test Case ID**	Predict expression
Test Case Description	Command line Interface	**Test Priority**	High
Pre-Requisite	NA	**Post Requisite**	NA

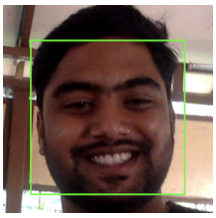
Test Execution Steps:

S. No	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**
1	Image to be predicted is sent to the CNN		Neutral	Surprise	Fail	NA

6.2.4 Test Case 4

Test Scenario ID	Predict expression	**Test Case ID**	Predict expression
Test Case Description	Command line Interface	**Test Priority**	High
Pre-Requisite	NA	**Post Requisite**	NA

Test Execution Steps:

S. No	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**
1	Image to be predicted is sent to the CNN		Happy	Happy	Pass	NA

6.2.5 Test Case 5

Test Scenario ID		Mongo Integration		**Test Case ID**	Mongo Integration	
Test Case Description		Command line Interface		**Test Priority**	High	
Pre-Requisite		NA		**Post Requisite**	NA	
Test Execution Steps:						
S. No	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**
1	Predicted expression is stored in the database	API call	Success	Success	Pass	NA

CHAPTER -7

SCREENSHOTS

Importing libraries

```
from keras.models import Sequential
from keras.layers import Dense,Flatten,Dropout
from keras.layers.convolutional import Conv2D,MaxPooling2D
from tensorflow.keras.layers import BatchNormalization
```

```
from keras.engine.sequential import clear_previously_created_nodes
import motor.motor_tornado
from datetime import datetime
```

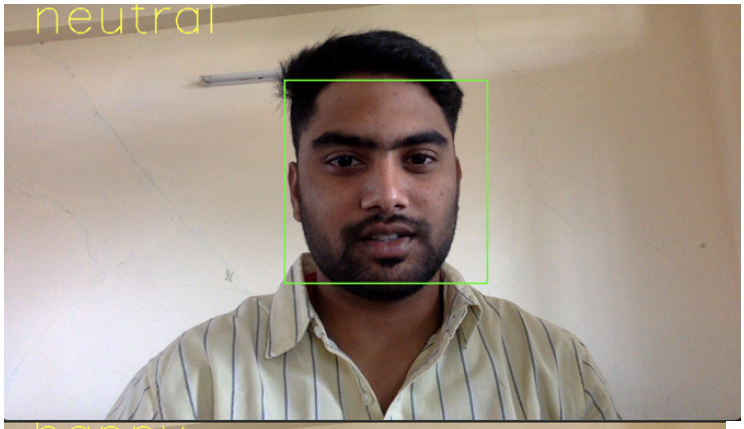
```
import cv2
import numpy as np
from model import create_convolutional_model
from mongo_integrate import add_data_to_mongo, connect_mongo
```

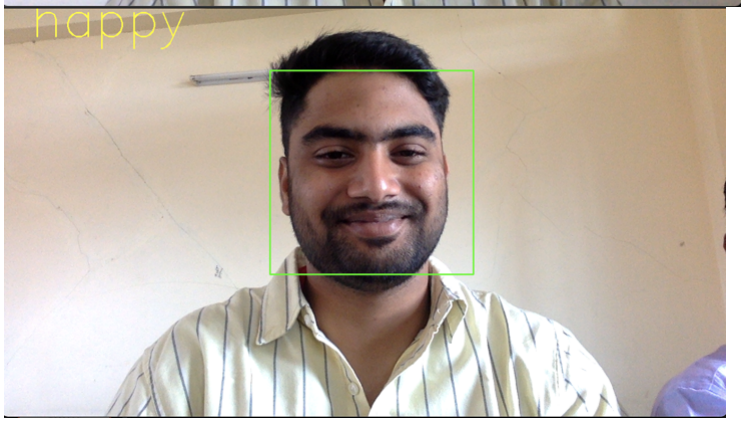
```
import pandas as pd
import numpy as np
from model import create_convolutional_model
from sklearn.model_selection import train_test_split
from tensorflow.keras.utils import to_categorical
```

CNN Architecture

```
def create_convolutional_model(classes):
    model = Sequential()
    model.add(Conv2D(32,kernel_size=(2,2),strides=(1,1),activation='relu',input_shape=(48,48,1)))
    model.add(BatchNormalization())
    model.add(MaxPooling2D(pool_size=(2,2),strides=(2,2)))
    model.add(Dropout(0.25))

    model.add(Conv2D(filters=64,kernel_size=(2,2),strides=(1,1),activation='relu'))
    model.add(BatchNormalization())
    model.add(MaxPooling2D(pool_size=(2,2),strides=(1,1)))
    model.add(Dropout(0.25)) #to prevent neural network from overfitting

    model.add(Conv2D(filters=128,kernel_size=(2,2),strides=(1,1),activation='relu'))
    model.add(BatchNormalization())
    model.add(MaxPooling2D(pool_size=(2,2),strides=(1,1)))
    model.add(Dropout(0.25))

    model.add(Conv2D(filters=256,kernel_size=(2,2),strides=(1,1),activation='relu'))
    model.add(BatchNormalization())
    model.add(MaxPooling2D(pool_size=(2,2),strides=(1,1)))
    model.add(Dropout(0.25))

    model.add(Flatten())

    model.add(Dense(256,activation='relu'))
    model.add(BatchNormalization())
    model.add(Dropout(0.25))

    model.add(Dense(512,activation='relu'))
    model.add(BatchNormalization())
    model.add(Dropout(0.25))

    model.add(Dense(classes,activation='softmax'))

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

    return model
```

Outputs

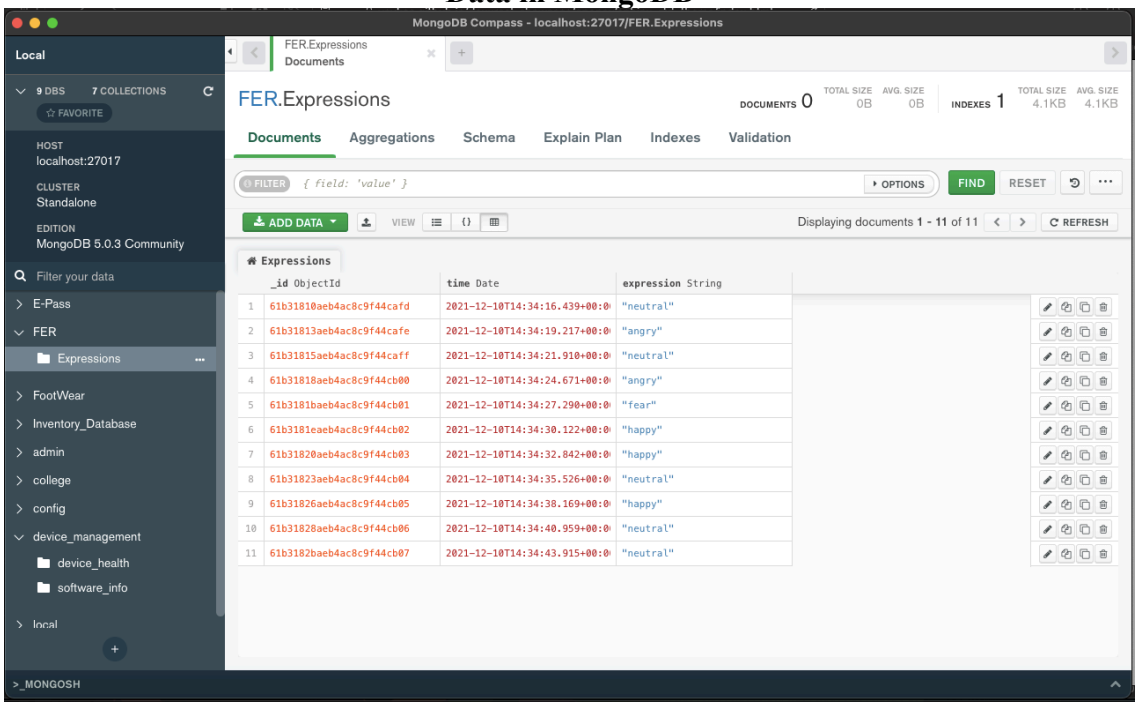




MongoDB integration

```
MONGODB_URL = "mongodb://localhost:27017/admin?retryWrites=true&w=majority"

def connect_mongo():
    try:
        client = motor.motor_tornado.MotorClient(MONGODB_URL)
        db = client.FER
        print("Connected to MongoDB")
        return db
    except Exception as err:
        print(Exception, err)

def add_data_to_mongo(db, expression):
    try:
        json = {"time": datetime.now(), "expression": expression}
        db.Expressions.insert_one(json)
    except Exception as err:
        print(Exception, err)

def main():
    db = connect_mongo()
    table = db.Expressions
    x = table.delete_many({})
    print("All documents deleted.")
```

Data in MongoDB



CHAPTER -8

FUTURE ENHANCEMENTS

There are few changes that can be made. For example -

- The number or classes in which the expressions are classified can be increased by adding new expressions and train them accordingly.
- Only one expression is captured from the frame i.e only one face is detected and his/her expressions are stored in the database, this can be increased to detect multiple faces and thus multiple expressions can be stored in the database.
- Computation time for extracting features makes the analysis a slow process, this happens with most feature extracting methods
- Data storage can be done by using a cloud service which is more secure and maintenance free.

CHAPTER -9

CONCLUSION

The network with the best overall performances, in terms of accuracy and loss for FER detection was ResNet and the results were confirmed by the confusion matrix. This model obtained a training accuracy of 90.14%, while VGG had 87% accuracy and Inception V3 reached 81%. By adding 5–6% of laboratory controlled images from other databases the accuracy of FER2013 increased with more than 10%, of the highest score so far, using the ResNet model together with other optimizations. In order to develop a model that better generalizes FER it is important to mix controlled and uncontrolled images.

CHAPTER -10

REFERENCES

1. Lopez-Rincon, A. Emotion recognition using facial expressions in children using the NAO Robot. In Proceedings of the International Conference on Electronics, Communications and Computers (CONIELECOMP), Cholula, Mexico, 27 February–1 March 2019; IEEE: Piscataway, NJ, USA; pp. 146–153. [Google Scholar]
2. Faria, D.R.; Vieira, M.; Faria, F.C. Towards the development of affective facial expression recognition for human-robot interaction. In Proceedings of the 10th International Conference on PErvasive Technologies Related to Assistive Environments, Island of Rhodes, Greece, 21–23 June 2017; pp. 300–304. [Google Scholar]
3. Zhang, Z.; Luo, P.; Loy, C.C.; Tang, X. From facial expression recognition to interpersonal relation prediction. *Int. J. Comput. Vis.* **2018**, *126*, 550–569. [Google Scholar] [CrossRef]
4. Zhao, X.; Liang, X.; Liu, L.; Li, T.; Han, Y.; Vasconcelos, N.; Yan, S. Peak-piloted deep network for facial expression recognition. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; Springer: Cham, Switzerland, 2016; pp. 425–442. [Google Scholar]
5. Ding, H.; Zhou, S.K.; Chellappa, R. Facenet2expnet: Regularizing a deep face recognition net for expression recognition. In Proceedings of the 12th IEEE International Conference on Automatic Face & Gesture Recognition, Washington, DC, USA, 30 May–3 June 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 118–126. [Google Scholar]
6. Ng, H.W.; Nguyen, V.D.; Vonikakis, V.; Winkler, S. Deep learning for emotion recognition on small datasets using transfer learning. In Proceedings of the 2015 ACM on International Conference on Multimodal Interaction, Seattle, WA, USA, 9–13 November 2015; pp. 443–449. [Google Scholar]
7. Lu, G.; He, J.; Yan, J.; Li, H. Convolutional neural network for facial expression recognition. *J. Nanjing Univ. Posts Telecommun.* **2016**, *36*, 16–22. [Google Scholar]
8. http://api.mongodb.org/python/current
9. https://docs.opencv.org/4.x/d4/db1/tutorial_documentation.html