

DEPRESSION PREDICTION

Submitted in partial fulfillment of the required for the award of

Bachelor of Computer Science

by

HARINI R (41290021)

SHAJIYA KIRAN M(41290050)



DEPARTMENT OF COMPUTER SCIENCE

SCHOOL OF SCIENCE AND HUMANITIES

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade “A++” by NAAC | 12B Status by UGC | Approved by

AICTEJEPPPIAR NAGAR, RAJIV GANDHI SALAI, CHENNAI 600 119

APRIL 2024



SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited "A++" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE
BONAFIDE CERTIFICATE

This is to certify that this Project Report is the Bonafide work of
SHAJIYA KIRAN M (41290050), HARINI R (41290021) who carried
out the project entitled **"DEPRESSION PREDICTION"** under our supervision
from NOVEMBER 2023 TO APRIL 2024.

Internal Guide

MS. AISHWARYA.B MCA

DEAN AND HEAD OF THE DEPARTMENT

Dr. REKHA CHAKRAVARTHI M.E., Ph.D.

Submitted for Viva voice examination held on 26/04/2024

Internal Examiner

External Examiner

DECLARATION

We , **SHAJIYA KIRAN M (41290050), HARINI R (41290021)** here
by declare that the Project Report entitled “**DEPRESSION PREDICTION**” done by
us under the guidance of **MS AISHWARYA B M.CA.**, Assistant Professor
Department of Computer Science at **SATHYABAMA INSTITUTE SCIENCE
AND TECHNOLOGY**, Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai-600 119 is
submitted in partial fulfillment of the requirements for the award of Bachelor of
Science degree in Computer Science.

DATE: 26/04/2024
PLACE: CHENNAI

SHAJIYA KIRAN M
SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

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

We convey our thanks to **Dr. REKHA CHAKRAVARTHI M.E., Ph.D. Dean of Art and Science** for providing me necessary support and details at the right time during the progressive reviews.

We would like to express my sincere and deep sense of gratitude to my project guide **AISHWARYAB MCA.,** Assistant Professor Department of Computer Science for their valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

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

SHAJIYA KIRAN M

ABSTRACT

Emojis are ways to indicate verbal cues. These cues have become an essential part of online chatting, product review, brand emotion, and numerous further. It also leads to adding data wisdom exploration devoted to emoji driven liar. With advancements in computer vision and deep learning, it's now possible to identify expressions from images. Interpersonal relations are affected by facial expressions, which convey verbal cues. The automatic recognition of human facial expressions can be an important aspect of human and machine interactions and it may also be used in behavioral wisdom. The discovery of feelings in facial expressions has always been easy for humans but doing the same thing with a computer algorithm is relatively difficult. As a result of recent advancements in machine learning, it's now possible to recognise the emotions through the facial expressions of human. In this deep learning design, we will classify human facial expressions to match and map with the corresponding emojis. Our design use human 'facial emotional expressions as system input to map emojis by identifying the expression. Despite that emojis can represent conduct, objects, nature, and other symbols, the most generally used emojis are faces which express emotion

TABLE CONTENT

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	5
1	INTRODUCTION	8
	1.1 DOMAIN INTRODUCTION	8
	1.2 OBJECTIVE OF THE PROJECT	8
	1.3 PROJECT INTRODUCTION	9
2	LITERTURE SURVEY	10
	2.1 REVIEW ON LITERATURE WORK	10
3	METHODOLOGY	12
	3.1 PROJECT SPECIFICATION	12
	3.2 SOFTWARE SPECIFICATION	12
	3.3 REQUIREMENTS	13
	3.4 EXISTING SYSTEM	13
	3.5 PROPOSED SYSTEM	14
4	SYSTEM DESIGN	15
	4.1 ARCHITECTURE DIAGRAM	16
	4.2 TABLE DESIGN	17
	4.3 SEQUENCE FLOW DIAGRAM	17
	4.4 DATA FLOW DIAGRAM	18
4	SOFTWARE DESCRIPTION	18
	4.1 MATLAB SYSTEM	18

	4.2 HIGHLIGHTS OF MATLAB	19
5	TYPES OF TESTING AND PROCESSES	21
	5.1 WHITE BOX TESTING	21
	5.2 BLACK BOX TESTING	22
6	SYSTEM IMPLEMENTATION	23
	6.1 MODULE LIST	23
	6.2 MODULE DESCRIPTION	27
7	APPENDICES	25
	7.1 SAMPLE CODE	25
	7.2 SAMPLE OUTPUT	34
8	CONCLUSION	37
9	REFERENCES	38

LIST OF FIGURES

FIG NO	LIST OF THE FIGURES	PAGE NO
4.1	SYSTEM DESIGN	16
4.2	ARCHITECTURE DIAGRAM	17
4.3	TABLE DESIGN	18
4.4	SEQUENCE FLOW DIAGRAM	18
4.5	DATA FLOW DIAGRAM	19
4.6	TESTINGS AND PROCESSES	21
7.1	SAMPLE OUTPUT	35
7.2	EMOTION DETECTION: SLEEP	36
7.3	EMOTION DETECTION:STARE	36
7.4	EMOTION DETECTION:SAD	37
7.5	EMOTION DETECTION: Half Sleep	37

CHAPTER 1

INTRODUCTION

1.1 DOMAIN INTRODUCTION

The process of extracting information to identify patterns, trends, and useful data that would allow the business to take the data-driven decision from huge sets of data is called Data Mining. Data mining includes the utilization of refined data analysis tools to find previously unknown, valid patterns and relationships in huge data sets. These tools can incorporate statistical models, machine learning techniques, and mathematical algorithms, such as neural networks or decision trees. Thus, data mining incorporates analysis and prediction. Depending on various methods and technologies from the intersection of machine learning, database management, and statistics, professionals in data mining have devoted their careers to better understanding how to process and make conclusions from the huge amount of data, but what are the methods they use to make it happen? In recent data mining projects, various major data mining techniques have been developed and used, including association, classification, clustering, prediction, sequential patterns, and regression.

APPLICATIONS OF DATA MINING

Data is a set of discrete objective facts about an event or a process that have little use by themselves unless converted into information. We have been collecting numerous data, from simple numerical measurements and text documents to more complex information such as spatial data, multimedia channels, and hypertext documents. Nowadays, large quantities of data are being accumulated. The amount of data collected is said to be almost doubled every year. An extracting data or seeking knowledge from this massive data, data mining techniques is used.

1.2 OBJECTIVE OF THE PROJECT

The goal of this project is to recognize the emotions through the facial expressions of human. Our design use human 'facial emotional expressions as system input to map emojis by identifying the expression. Despite that emojis can represent conduct, objects, nature, and other symbols, the most generally used emojis are faces which express emotion

1.3 PROJECT INTRODUCTION

The automatic recognition of human facial expressions can be an important aspect of human and machine interactions and it may also be used in behavioral wisdom. The main purpose of this project is to predict 4 to 5 emotions like angry, happy, sad, fearful, surprised etc. Then we are going to map those emotions with the corresponding emojis based on the emotions. Hence the aim of the project to capture and recognize the facial expression and displaying the corresponding emoji. As a result of recent advancements in it's now possible to recognize the emotions through the facial expressions of human.

CHAPTER 2

LITERATURE SURVEY

2.1 REVIEW ON LITERATURE WORK

TITLE 1: CLASSIFICATION ALGORITHM BASED MENTAL HEALTH USING DATAMINING

YEAR: 2022

AUTHOR: Mrs. M. Kavitha, Dr. Madhavi Pingili, M. Spurthi, 4 SK. Fayaz, 5K. Kirthan, Santhosh

ABSTRACT:

The emotional, psychological and social welfare of a person is revealed by their mental health. It influences how an individual will Think, feel or handle a situation. Positive mental health helps an Individual to work productively and achieve their full potential. At each point in life, mental health is vital, from childhood to adulthood. Numerous factors contribute to mental health issues which lead to mental illness like stress, social anxiety, depression, obsessive compulsive disorder, drug addiction, workplace issues and personality disorders. The onset of mental illness should be determined without flaws for maintaining an appropriate life balance. We have collected data from online available datasets. The data has been label encoded for better prediction. The data is being subject to various machine learning techniques to obtain labels.

TITLE 2: Real-Time Emotion Recognition System using Facial Expressions

YEAR: 2003

AUTHOR: 1 S.Arun Inigo, 2 V.Rajesh Kumar, 3 P.Ashokram

ABSTRACT:

Facial Expression conveys non-verbal cues, which plays an important role in interpersonal relations. The Cognitive Emotion AI system is the process of identifying the emotional state of a person. The main aim of our study is to develop a robust system which can detect as well as recognize human emotion from live feed. There are some emotions which are universal to all human beings like angry, sad, happy, surprise, fear, disgust and neutral. The methodology of this system is based on two stages- facial detection is done by extraction of Haar Cascade features of a face using Viola Jones algorithm and then the emotion is verified and recognized using Artificial Intelligence Techniques. The system will take image or frame as an input and by providing the image to the model the model will perform the preprocessing and feature selection after that it will predict the emotional state.

TITLE 3: FACIAL EXPRESSION DETECTION AND RECOGNITION SYSTEM

YEAR: 2004

AUTHOR: W.K. Teo¹, Liyanage C De Silva² and Prahlad Vadakkepat¹

ABSTRACT:

In this paper, the integration of face feature detection and extraction, and facial expression recognition are discussed. In this paper, we propose an algorithm that utilizes multi- stage integral projection to extract facial features. Furthermore, in this project, we propose a statistical approach to process the optical flow data to obtain the overall value for the respective feature region in the face. This approach has eliminated the requirement of accurate identification of the feature boundary. Optical flow computations are utilized to identify the directions and the amount of motions in image sequences that are caused by human facial expressions. The optical flow computation results are processed using Kalman filtering. The filtered results are given to a neural network to realize a mapping into the facial expression space.

CHAPTER 3

METHODOLOGY

3.1 PROJECT SPECIFICATION

Due to the broad and undefined environment of this case study it is very important that we focal point on the number of exact tools and case study. The data mining tools in the region of which this study case will revolve are: the Neural-Network Toolbox, a proprietary tool presented from The Math-Works, distributors of MATLAB. The Fuzzy cluster and Data study Toolbox [Balasko et al. 2015] and the association Rule Miner and presumption study tool [Malone 2013], which are both open-platform; and lastly an execution of the C4.5 judgment tree method [Woelf, 2015].

3.2 SOFTWARE SPECIFICATION

MATLAB is an elite dialect for specialized figuring. It incorporates calculation, perception, and programming in a simple to-utilize condition where issues and arrangements are communicated in commonplace numerical documentation.

Common place uses incorporate

- ❖ Math and calculation
- ❖ Algorithm improvement
- ❖ Data obtaining
- ❖ Modeling, reenactment, and prototyping
- ❖ Data investigation, investigation, and representation
- ❖ Scientific and building illustrations
- ❖ Application improvement, including graphical UI building

MATLAB is an intelligent framework whose fundamental information component is an exhibit that does not require dimensioning. This enables you to explain numerous specialized figuring issues, particularly those with grid and vector definitions, in a small amount of the time it would take to compose a program in a scalar non interactive dialect, for example, C or Fortran.

The name MATLAB remains for lattice research facility. MATLAB was initially written to give simple access to lattice programming created by the LINPACK what's more, EISPACK ventures. Today, MATLAB motors join the LAPACK what's more, BLAS libraries, installing the best in class in programming for lattice calculation.

3.3 Requirements

HARDWARE CONFIGURATION

- **HDD:** >90GB
- **PROCESSOR:** >Pentium IV 2.4GHz
- **SYSTEM TYPE:** 32bit / 64 bit
- **RAM:** >2GB
- **OS:** WINDOWS 7/8/8.1/10

SOFTWARE CONFIGURATION

- **TOOL:** Python
- **TOOL BOX:** data tool box

3.4 EXISTING SYSTEM

Current approaches focus mainly on facial investigation, maintaining the background intact and thus constructs a large number of unnecessary and misleading characteristics that confuse the process of CNN formation. Predicting the human emotions through their facial expression. Most of the current approaches are not so accurate. Some approaches take lot of time to detect the facial expressions.

DISADVANTAGES:

- Not Accurate
- Take lot of time

2.3 PROPOSED SYSTEM

In this project, we wish to build a convolution neural network to recognize facial emotions. In this project, we mapped emojis to classify facial emotions over static facial images using deep learning techniques was developed. Emojis are non-verbal way to express the emotion. We are using Sequential Model. We like to build the model that would be able to predict 4 to 5 emotions like angry, happy, sad, fearful, surprised etc. We would like to train our model on the FER2013 dataset. It comprises 32,298 gray scale expressions, collected from the web using 184 emotion-related keywords. In total 32298 gray- level expressions, we have 28709 training expressions, 3589 cross-validation expressions and 3589 test images. Then we are going to map those emotions with the corresponding emojis based on the emotions. Hence the aim of the project to capture and recognize the facial expression and displaying the corresponding emoji.

ADVANTAGES:

- No internet connection required.
- High transfer rate.

CHAPTER 4

SYSTEM DESIGN

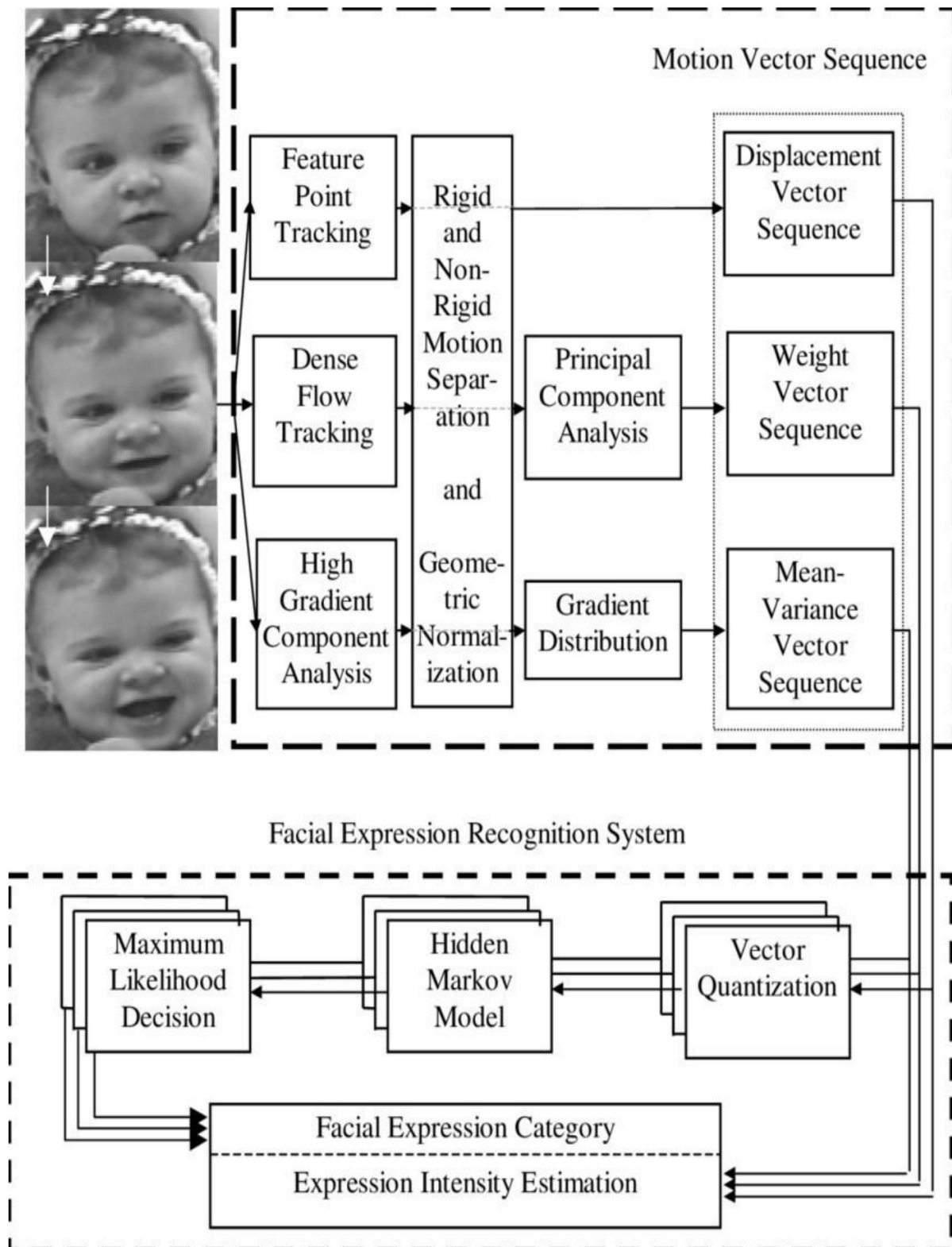


FIG 4.1

4.1 ARCHITECTURE DIAGRAM

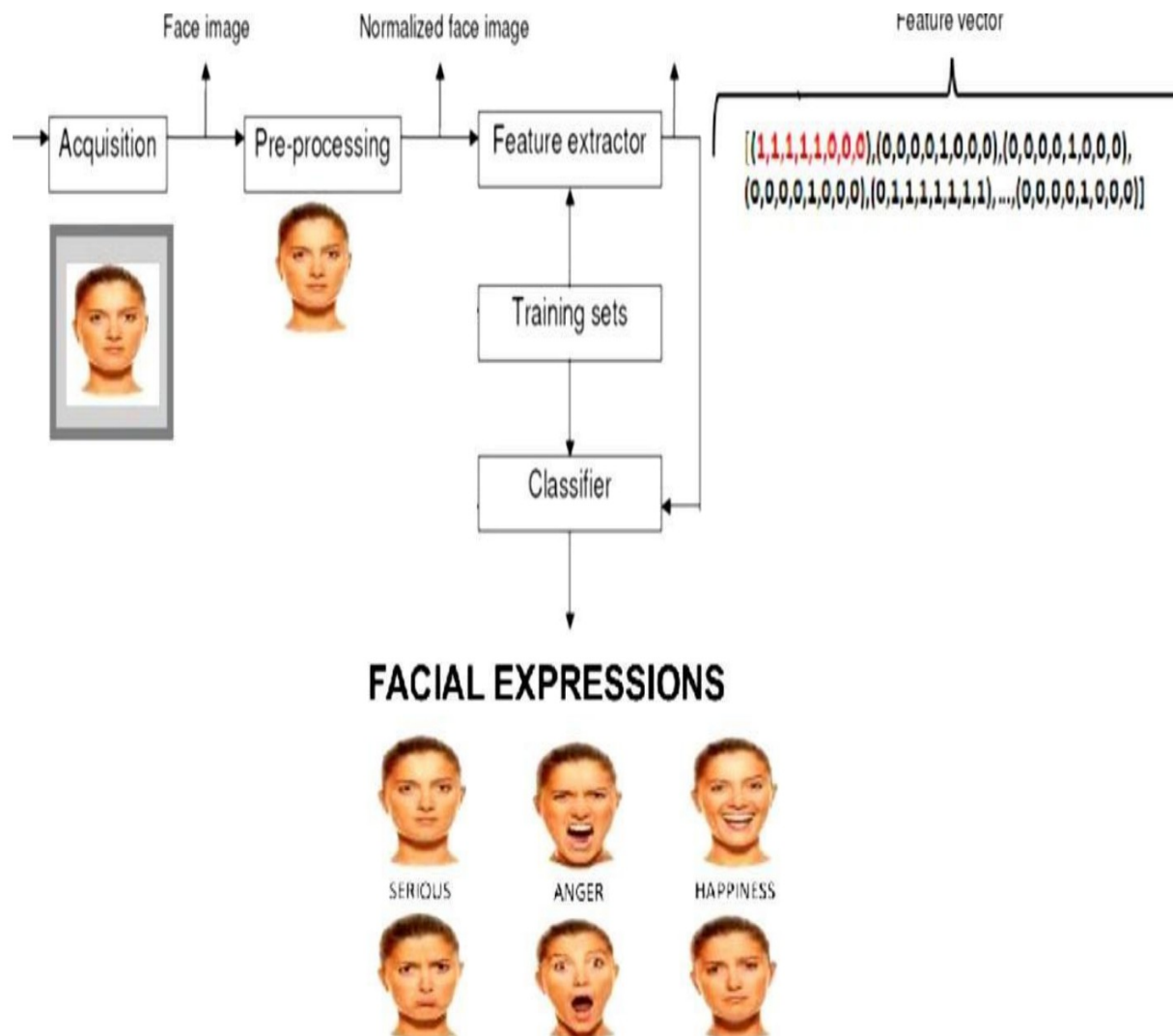


FIG 4.2

TABLE DESIGN

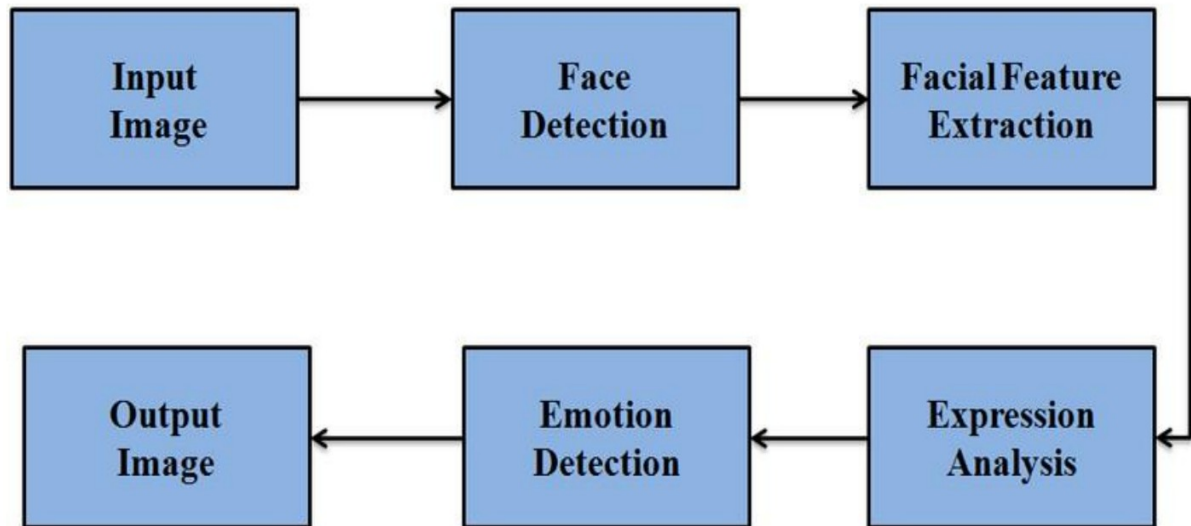


FIG 4.3

SEQUENCE FLOW DIAGRAM



FIG 4.4

DATA FLOW DIAGRAM

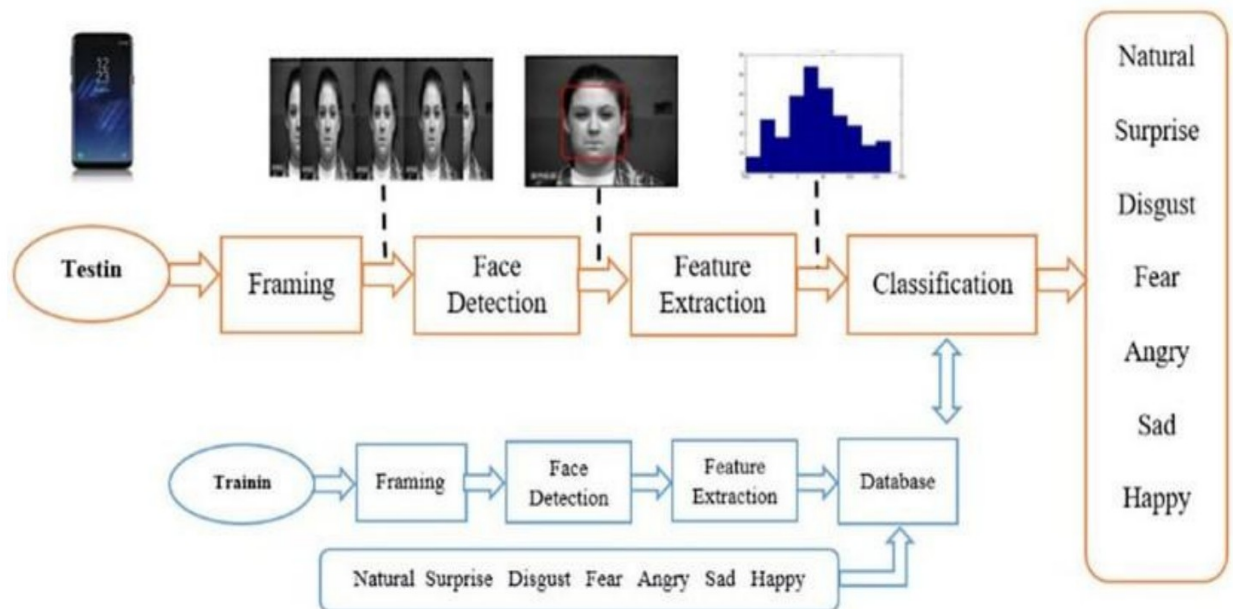


FIG 4.5

SOFTWARE DESCRIPTION:

4.1 THE MATLAB SYSTEM

The MATLAB framework comprises of five fundamental parts:

- ❖ **Improvement Environment.** This is the arrangement of apparatuses and offices that assistance you utilize MATLAB capacities and records. A considerable lot of these instruments are graphical UIs. It incorporates the MATLAB work area and Command Window, a charge history, an editorial manager and debugger, and programs for review help, the workspace, records, what's more, the inquiry way.
- ❖ **The MATLAB Mathematical Function Library.** This is a huge gathering of computational calculations going from basic capacities, similar to total, sine, cosine, and

complex number-crunching, to more advanced capacities like network backwards, framework eigen values, Bessel capacities, and quick Fourier changes

- ❖ **The MATLAB Language.** This is an abnormal state framework/exhibit dialect with control stream proclamations, capacities, information structures, input/yield, and protest situated programming highlights. It permits both "programming in the little" to quickly make snappy discard projects, and "programming in the huge" to make substantial and complex application programs.
- ❖ **Designs.** MATLAB has broad offices for showing vectors and lattices as diagrams, and additionally commenting on and printing these charts. It incorporates abnormal state capacities for two-dimensional and three-dimensional information perception, picture handling, activity, and introduction illustrations. It too incorporates low-level capacities that enable you to completely tweak the presence of illustrations and in addition to assemble finish graphical UIs on your MATLAB applications.
- ❖ **The MATLAB External Interfaces/API.** This is a library that enables you to compose C and Fortran programs that collaborate with MATLAB. It incorporates offices for calling schedules from MATLAB (dynamic connecting), calling MATLAB as a computational motor, and for perusing and composing MAT-records.

4.2 HIGHLIGHTS OF MATLAB

Following are the essential highlights of MATLAB:

- ❖ It is an abnormal state dialect for numerical calculation, representation and application advancement.
- ❖ It additionally gives an intelligent domain to iterative investigation, plan what's more, critical thinking.
- ❖ It gives immense library of numerical capacities for direct variable based math, measurements, Fourier examination, sifting, advancement, numerical coordination and comprehending standard differential conditions.
- ❖ It gives worked in illustrations to picturing information and instruments for making custom plots.
- ❖ MATLAB's customizing interface gives improvement devices for moving forward code quality, practicality, and augmenting execution.

- ❖ It gives devices to building applications with custom graphical interfaces.
- ❖ It gives capacities to coordinating MATLAB based calculations with outer applications and dialects, for example, C, Java, .NET and Microsoft Excel.

CHAPTER 5

TYPES OF TESTINGS AND PROCESSES

The figure indicates the areas of testing involved during different database testing methods, such as black-box testing and white-box testing.

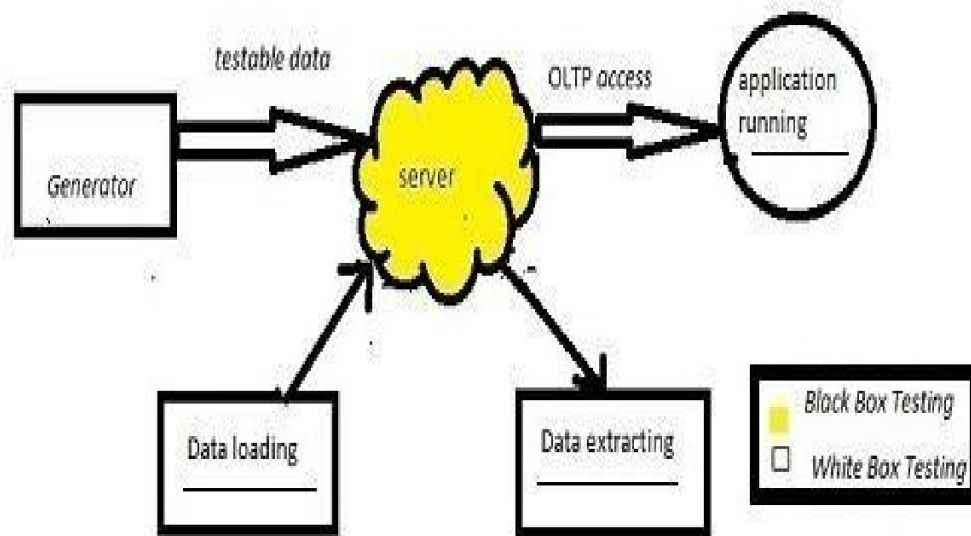


FIG 5.1

5.1 WHITE BOX TESTING :

White-box testing (also known as clear box testing, glass box testing, transparent box testing, and structural testing) is a method of testing software that tests internal structures or workings of an application, as opposed to its functionality (i.e. black-box testing). In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the expected outputs. This is analogous to testing nodes in a circuit, e.g. in-circuit testing (ICT). White-box testing can be applied at the unit, integration and system levels of the software testing process. Although traditional testers tended to think of white-box testing as being done at the unit level, it is used for integration and system testing more frequently today. It can test paths within a unit, paths between units during integration, and between subsystems during a system-level

test. Though this method of test design can uncover many errors or problems, it has the potential to miss unimplemented parts of the specification or missing requirements.

5.2 BLACK BOX TESTING:

Black-box testing is a method of software testing that examines the functionality of an application without peering into its internal structures or workings. This method of test can be applied virtually to every level of software testing: unit, integration, system and acceptance. It is sometimes referred to as specification-based testing. Test cases are built around specifications and requirements, i.e., what the application is supposed to do. Test cases are generally derived from external descriptions of the software, including specifications, requirements and design parameters. Although the tests used are primarily functional in nature, non-functional tests may also be used. The test designer selects both valid and invalid inputs and determines the correct output, often with the help of a test oracle or a previous result that is known to be good, without any knowledge of the test object's internal structure.

CHAPTER 6

Implementation Phase

6.1 MODULE LIST

- Input the dataset
- Data pre-processing and applying augmentation Strategies
- Neural Network architecture
- Accuracy and loss.

6.2 MODULE DESCRIPTION

Step 1: Input the dataset Below are sample images from the FER 2013 dataset that are used to classify emotions. These Images are categorized based on the emotion shown in the facial expressions such as happiness, neutral, sadness, anger, surprise, disgust, fear.

Step 2: Data pre-processing and applying augmentation Strategies. Image data augmentation is used to expand the training dataset in order to improve the performance and ability of the model to generalize. Images are rescaled from [0,255] to [0,1] using the ImageDataGenerator python module. Benefits of this are:

- Treat all images in the same manner: some images are high pixel range, some are low pixel range. The images are all sharing the same model, weights and learning rate. The high range image tends to create stronger loss while low range creates weak loss, the sum of them will all contribute to the back propagation update.
- Using typical learning rate: when we reference the learning rate from other's work, we can directly reference their learning rate if both works do the scaling preprocessing over images data set. Otherwise, higher pixel range image results in higher loss and should use a smaller learning rate, lower pixel range image will need a larger learning rate.

Step 3: Neural Network architecture. After pre-processing the dataset, the next step is to build a convolutional neural network. The convolution layer consists of the input layers, the hidden layers and the output layer. Depending on the architecture that is built in the neural network we add convolutional layers with filters. Below is the architecture of the neural network.

Step 4: Accuracy and loss. With the implementation of the above neural network, we got accuracy.

CHAPTER 7

APPENDICES

7.1 SAMPLE CODE

```
global flag;global count;
% Create the face detector object.
faceDetector = vision.CascadeObjectDetector();
% Create the point tracker object.
pointTracker = vision.PointTracker('MaxBidirectionalError', 2);
% Create the webcam object. if
exist('cam')==0
    cam = webcam(1);
end
% Capture one frame to get its size.
videoFrame = snapshot(cam); frameSize =
size(videoFrame);
% Create the video player object.
videoPlayer = vision.VideoPlayer('Position', [100 100 [frameSize(2), frameSize(1)]+30]);
% Detection & Tracking
runLoop = true; numPts
= 0;
frameCount = 0;
while runLoop && frameCount < 100
% Get the next frame. videoFrame =
snapshot(cam);
% Get frame to save data to database videoFrameGray =rgb2gray(videoFrame); frameCount = frameCount + 1;
if numPts < 10
    % Detection mode.
    bbox = faceDetector.step(videoFrameGray); if
    ~isempty(bbox)
    % Find corner points inside the detected region.
    points = detectMinEigenFeatures(videoFrameGray, 'ROI', bbox(1, :));
    % Re-initialize the point tracker.
```

```

xyPoints = points.Location;
numPts = size(xyPoints,1);
release(pointTracker);
initialize(pointTracker, xyPoints, videoFrameGray);
% Save a copy of the points. oldPoints =
xyPoints;
% Convert the rectangle represented as [x, y, w, h] into an
% M-by-2 matrix of [x,y] coordinates of the four corners. This
% is needed to be able to transform the bounding box to display
% the orientation of the face. bboxPoints =
bbox2points(bbox(1, :));
% Convert the box corners into the [x1 y1 x2 y2 x3 y3 x4 y4]
% format required by insertShape. bboxPolygon =
reshape(bboxPoints', 1, []);
% Display a bounding box around the detected face.
videoFrame = insertShape(videoFrame, 'Polygon', bboxPolygon, 'LineWidth', 3);
% Display detected corners.
videoFrame = insertMarker(videoFrame, xyPoints, '+', 'Color', 'white'); end

else % Tracking mode.
[xyPoints, isFound] = step(pointTracker, videoFrameGray); visiblePoints = xyPoints(isFound, :); oldInliers
= oldPoints(isFound, :);
numPts = size(visiblePoints, 1);

if numPts >= 10
    % Estimate the geometric transformation between the old points and the new points. [xform,
    oldInliers, visiblePoints] = estimateGeometricTransform(... oldInliers, visiblePoints,
'similarity', 'MaxDistance', 4);
    % Apply the transformation to the bounding box. bboxPoints =
    transformPointsForward(xform, bboxPoints);
    % Convert the box corners into the [x1 y1 x2 y2 x3 y3 x4 y4] format required by insertShape. bboxPolygon
    = reshape(bboxPoints', 1, []);
    % Display a bounding box around the face being tracked.
    videoFrame = insertShape(videoFrame, 'Polygon', bboxPolygon, 'LineWidth', 3);
    % Display tracked points.

```

```

        videoFrame = insertMarker(videoFrame, visiblePoints, '+', 'Color', 'white');
        % Reset the points. oldPoints = visiblePoints;
        setPoints(pointTracker, oldPoints);
    end
end
% Display the annotated video frame using the video player object. step(videoPlayer,
videoFrame);
% Check whether the video player window has been closed. runLoop
= isOpen(videoPlayer);
end
% Clean up. clear cam;
release(videoPlayer);
release(pointTracker);
release(faceDetector);
% Save your image to your database
% You have to repeat it from step 1 to here to save more face either same person or different person. position1 =
min(bboxPolygon(2),bboxPolygon(4));
position2 = max(bboxPolygon(6),bboxPolygon(8));
position3 = min(bboxPolygon(1),bboxPolygon(7));
position4 = max(bboxPolygon(3),bboxPolygon(5));
warning('off')
getimage = videoFrameGray(position1:position2,position3:position4,:);
% figure();
imshow(getimage)
%resize image
getimage = imresize(getimage, [300 300]);
% Modify here. In the folder database2, label the name of ppl and put their
% faces inside the folder. imwrite(getimage,'database2\Shobika\35.jpg');
imwrite(getimage,'database2\Shobika_G\35.jpg');
% Train your Image in the database folder
%% Load Image Information from ATT Face Database Directory
faceDatabase = imageSet('database2','recursive');
%% Split Database into Training & Test Sets
[training,test] = partition(faceDatabase,[0.8 0.2]);
%% Extract and display Histogram of Oriented Gradient Features for single face person = 2; [hogFeature,
visualization]= ... extractHOGFeatures(read(training(person),1));

```

```

% figure();
% subplot(2,1,1);
imshow(read(training(person),1));
title('Input Face');
% subplot(2,1,2);
plot(visualization);
title('HoG Feature');
% Extract HOG Features for training set
trainingFeatures = zeros(size(training,2)*training(1).Count,46656); featureCount = 1; for
i=1:size(training,2)
    for j = 1:training(i).Count
        points = detectSURFFeatures(read(training(i),j)); trainingFeatures(featureCount,:)=
        extractHOGFeatures(read(training(i),j)); trainingLabel{featureCount} =
        training(i).Description;
        featureCount = featureCount + 1;
    end
    personIndex{i} = training(i).Description;
end
% Create 40 class classifier using
faceClassifier = fitcecoc(trainingFeatures,trainingLabel);
% Predict Person in the database folder
% Create the face detector object.
faceDetector = vision.CascadeObjectDetector();
% Create the point tracker object.
pointTracker = vision.PointTracker('MaxBidirectionalError', 2);
% Create the webcam object. if
exist('cam')==0
    cam = webcam(1); end
% Capture one frame to get its size.
videoFrame = snapshot(cam); frameSize = size(videoFrame);
% Create the video player object.
videoPlayer = vision.VideoPlayer('Position', [100 100 [frameSize(2), frameSize(1)]+30]);
% Run Video for Prediction
runLoop = true;
numPts = 0;
frameCount = 0;

```

```

% cam = webcam('HP TrueVision HD Camera'); while
runLoop && frameCount < 300
    % Get the next frame. videoFrame
    = snapshot(cam);
    % Get frame to save data to database
    videoFrame2 = snapshot(cam); videoFrameGray
    = rgb2gray(videoFrame); frameCount =
    frameCount + 1;
if numPts < 10
    % Detection mode.
    bbox = faceDetector.step(videoFrameGray); if
    ~isempty(bbox)
    % Find corner points inside the detected region.
    points = detectMinEigenFeatures(videoFrameGray, 'ROI', bbox(1, :));
    % Re-initialize the point tracker.
    xyPoints = points.Location; numPts
    = size(xyPoints,1);
    release(pointTracker);
    initialize(pointTracker, xyPoints, videoFrameGray);
    % Save a copy of the points.
    oldPoints = xyPoints;
    % Convert the rectangle represented as [x, y, w, h] into an
    % M-by-2 matrix of [x,y] coordinates of the four corners. This
    % is needed to be able to transform the bounding box to display
    % the orientation of the face. bboxPoints =
    bbox2points(bbox(1, :));
    % Convert the box corners into the [x1 y1 x2 y2 x3 y3 x4 y4] format required by insertShape. bboxPolygon
    = reshape(bboxPoints', 1, []);
    % Display a bounding box around the detected face.
    videoFrame = insertShape(videoFrame, 'Polygon', bboxPolygon, 'LineWidth', 3);
    % Display detected corners.
    videoFrame = insertMarker(videoFrame, xyPoints, '+', 'Color', 'white');

```

% Tracking mode.

```
[xyPoints, isFound] = step(pointTracker, videoFrameGray)end
% Capture one frame to get its size. videoFrame = snapshot(cam); frameSize = size(videoFrame);
% Create the video player object.
videoPlayer = vision.VideoPlayer('Position', [100 100 [frameSize(2), frameSize(1)]+30]);
% Run Video for Prediction runLoop = true; numPts =
0;
frameCount = 0;
% cam = webcam('HP TrueVision HD Camera'); while runLoop && frameCount < 300
% Get the next frame. videoFrame = snapshot(cam);
% Get frame to save data to database videoFrame2 = snapshot(cam); videoFrameGray = rgb2gray(videoFrame);
frameCount = frameCount + 1;
if numPts < 10
% Detection mode.
bbox = faceDetector.step(videoFrameGray); if
~isempty(bbox)
% Find corner points inside the detected region.
points = detectMinEigenFeatures(videoFrameGray, 'ROI', bbox(1, :));
% Re-initialize the point tracker. xyPoints = points.Location; numPts = size(xyPoints,1); release(pointTracker);
initialize(pointTracker, xyPoints, videoFrameGray);
% Save a copy of the points. oldPoints = xyPoints;
% Convert the rectangle represented as [x, y, w, h] into anoldInliers, visiblePoints, 'similarity', 'MaxDistance', 4);

% Apply the transformation to the bounding box
. boxPoints = transformPointsForward(xform, boxPoints);
% Convert the box corners into the [x1 y1 x2 y2 x3 y3 x4 y4]
% format required by insertShape. bboxPolygon = reshape(bboxPoints', 1, []);% Display a
bounding box round the face being tracked.
videoFrame = insertShape(videoFrame, 'Polygon', bboxPolygon, 'LineWidth', 3);
% Display tracked points.
videoFrame = insertMarker(videoFrame, visiblePoints, '+', 'Color', 'white');
% Reset the points. oldPoints = visiblePoints; setPoints(pointTracker,
oldPoints); end
end
% Display the annotated video frame using the video player object. step(videoPlayer, videoFrame);
% Check whether the video player window has been closed. runLoop = isOpen(videoPlayer); end
% Clean up. clear cam;
release(videoPlayer); release(pointTracker); release(faceDetector);
% Save your image to your database
% You have to repeat it from step 1 to here to save more face either same person or different
person.
position1 = min(bboxPolygon(2),bboxPolygon(4));
position2 = max(bboxPolygon(6),bboxPolygon(8));
```

```

    position3 = min(bboxPolygon(1),bboxPolygon(7));
    position4 = max(bboxPolygon(3),bboxPolygon(5));
    warning('off')
    getimage = videoFrameGray(position1:position2,position3:position4,:);
    % figure();imshow(getimage)
    %resize image
    getimage = imresize(getimage, [300 300]);
    % Modify here. In the folder database2, label the name of ppl and put their
    % faces inside the folder. imwrite(getimage,'database2\Shobika\35.jpg');
    imwrite(getimage,'database2\Shobika_G\35.jpg');% Train your Image in the database folder
    %% Load Image Information from ATT Face Database Directory faceDatabase = imageSet('database2','recursive');
    %% Split Database into Training & Test Sets [training,test] = partition(faceDatabase,[0.8 0.2]);
    %% Extract and display Histogram of Oriented Gradient Features for single face person = 2; [hogFeature,
    visualization]= ... extractHOGFeatures(read(training(person),1));
    % figure();
    % subplot(2,1,1);imshow(read(training(person),1));title('Input Face');
    % subplot(2,1,2);plot(visualization);title('HoG Feature');
    % Extract HOG Features for training set
    trainingFeatures = zeros(size(training,2)*training(1).Count,46656); featureCount = 1; for
    i=1:size(training,2)
    for j = 1:training(i).Count
    points = detectSURFFeatures(read(training(i),j)); trainingFeatures(featureCount,:) =
    extractHOGFeatures(read(training(i),j)); trainingLabel{featureCount} = training(i).Description;
    featureCount = featureCount + 1; end
    personIndex{i} = training(i).Description;
    end
    % Create 40 class classifier using
    faceClassifier = fitcecoc(trainingFeatures,trainingLabel);
    % Predict Person in the database folder
    % Create the face detector object.
    FaceDetector = vision.CascadeObjectDetector();
    % Create the point tracker object.
    pointTracker = vision.PointTracker('MaxBidirectionalError', 2);
    % Create the webcam object. if exist('cam')==0 cam
    = webcam(1); end
    % Capture one frame to get its size. videoFrame = snapshot(cam); frameSize = size(videoFrame);
    % Create the video player object.
    videoPlayer = vision.VideoPlayer('Position', [100 100 [frameSize(2), frameSize(1)]+30]);
    % Run Video for Prediction runLoop = true;
    numPts = 0;
    frameCount = 0;
    % cam = webcam('HP TrueVision HD Camera'); while runLoop && frameCount < 300

```

```

% Get the next frame. videoFrame = snapshot(cam);
% Get frame to save data to database videoFrame2 = snapshot(cam); videoFrameGray = rgb2gray(videoFrame);
frameCount = frameCount + 1;
if numPts < 10
% Detection mode. box = faceDetector.step(videoFrameGray); f
~isempty(bbox)
% Find corner points inside the detected region.
points = detectMinEigenFeatures(videoFrameGray, 'ROI', bbox(1, :));
% Re-initialize the point tracker. xyPoints = points.Location; numPts = size(xyPoints,1); release(pointTracker);
initialize(pointTracker, xyPoints, videoFrameGray);
% Save a copy of the points. oldPoints = xyPoints;
% Convert the rectangle represented as [x, y, w, h] into an
% M-by-2 matrix of [x,y] coordinates of the four corners. This
% is needed to be able to transform the bounding box to display
% the orientation of the face. bboxPoints = bbox2points(bbox(1, :));
% Convert the box corners into the [x1 y1 x2 y2 x3 y3 x4 y4]
% format required by insertShape. bboxPolygon = reshape(bboxPoints', 1, []);
% Display a bounding box around the detected face.
videoFrame = insertShape(videoFrame, 'Polygon', bboxPolygon, 'LineWidth', 3);
% Display detected corners.
VideoFrame = insertMarker(videoFrame, xyPoints, '+', 'Color', 'white'); end else
% Tracking mode.
[xyPoints, isFound] = step(pointTracker, videoFrameGray); visiblePoints = xyPoints(isFound, :); oldInliers
= oldPoints(isFound, :); numPts = size(visiblePoints, 1); if numPts >= 10
% Estimate the geometric transformation between the old points
% and the new points.
[xform, oldInliers, visiblePoints] = estimateGeometricTransform(... oldInliers, visiblePoints, 'similarity',
'MaxDistance', 4);
% Apply the transformation to the bounding box. bboxPoints = transformPointsForward(xform, bboxPoints);
% Convert the box corners into the [x1 y1 x2 y2 x3 y3 x4 y4]
% format required by insertShape. bboxPolygon = reshape(bboxPoints', 1, []);
% Display a bounding box around the face being tracked.
videoFrame = insertShape(videoFrame, 'Polygon', bboxPolygon, 'LineWidth', 3);
% Display tracked points.
videoFrame = insertMarker(videoFrame, visiblePoints, '+', 'Color', 'white');
% Reset the points. oldPoints = visiblePoints; setPoints(pointTracker,
oldPoints);
position1 = min(bboxPolygon(2),bboxPolygon(4)); position2 = max(bboxPolygon(6),bboxPolygon(8));
position3 = min(bboxPolygon(1),bboxPolygon(7)); position4 = max(bboxPolygon(3),bboxPolygon(5));
warning('off')
f position2<640 && position4<640 && position1>0 && position2>0 getimage =
videoFrameGray(position1:position2,position3:position4,:);
%resize image
getimage = imresize(getimage, [300 300]);
queryFeatures = extractHOGFeatures(getimage); [personLabel,PostProbs] =
predict(faceClassifier,queryFeatures); maxpro = max(abs(PostProbs(1)),abs(PostProbs(2))); position =
[position3 position2];
box_color = {'yellow'};
% disp(maxpro)
string = strcat(personLabel,num2str(maxpro));
videoFrame = insertText(videoFrame,position,string,'FontSize',18,'BoxColor',...

```

```

box_color,'BoxOpacity',0.4,'TextColor','white');
if(maxpro>=0.62 && maxpro<=0.72) disp('Happy');tts('Happy'); imshow('EMOJI/Happy.jpg'); end
if(maxpro>=0.59 && maxpro<0.62) disp('Sad');tts('Sad');imshow('EMOJI/Sad.jpg'); flag=1;count=count
+1;break
end
if(maxpro>0.48 && maxpro<=0.58) disp('Sleep');tts('Sleep');imshow('EMOJI/Sleep.jpg'); flag=1;count=count
+1;break
end
if(maxpro>0.52 && maxpro<=0.54)
disp('Half sleep');tts('Half sleep');imshow('EMOJI/Half sleep.jpg'); flag=1;count=count +1;break end
if(maxpro>0.24 && maxpro<=0.28) disp('Stare');tts('Stare');imshow('EMOJI/stare.jpg');
flag=1;count=count +1;break
end end
end end
% Display the annotated video frame using the video player object. step(videoPlayer, videoFrame);
% Check whether the video player window has been closed. runLoop = isOpen(videoPlayer); end
clear cam; release(videoPlayer); release(pointTracker); release(faceDetector);

```

7.2 SAMPLE OUTPUT:

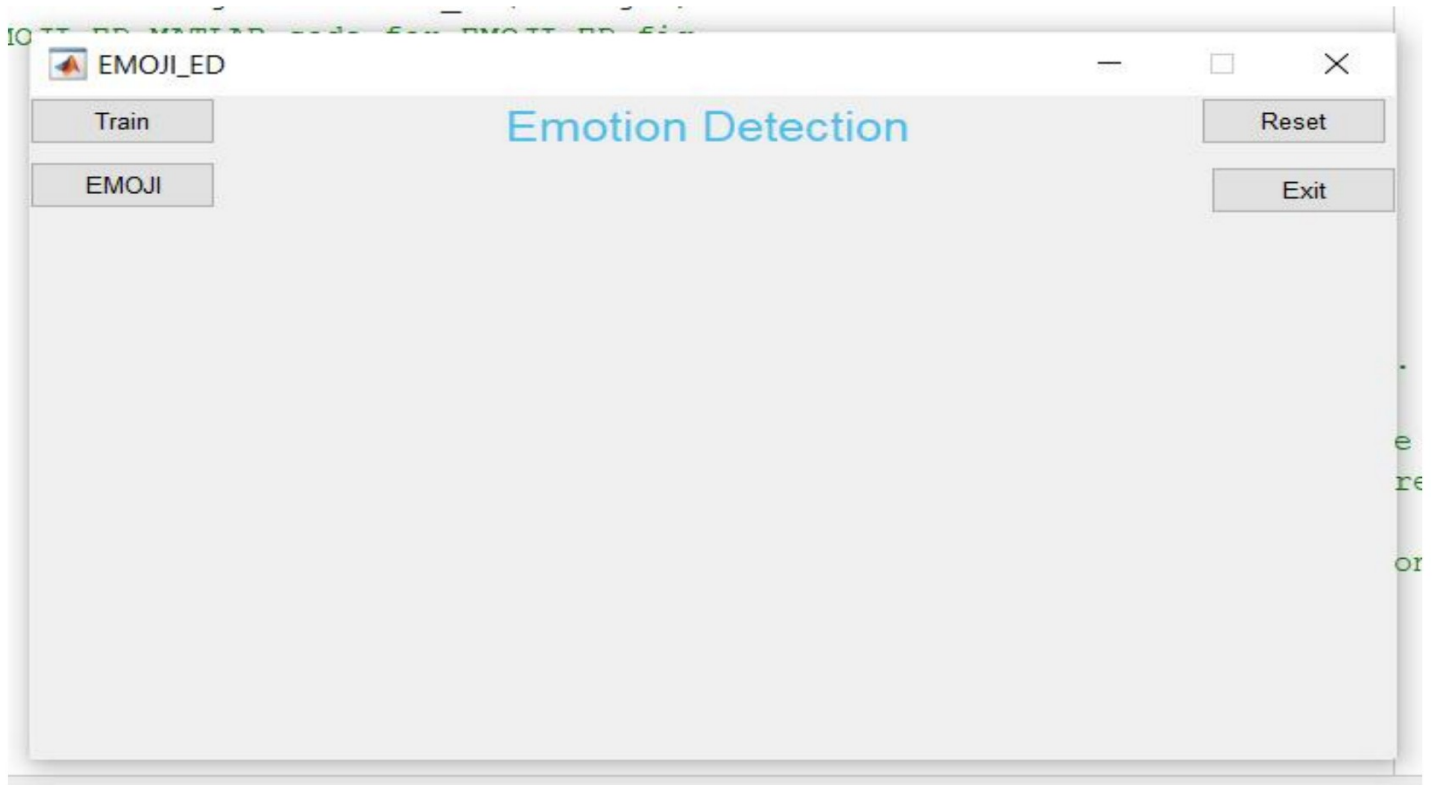


FIG 7.1



FIG 7.2

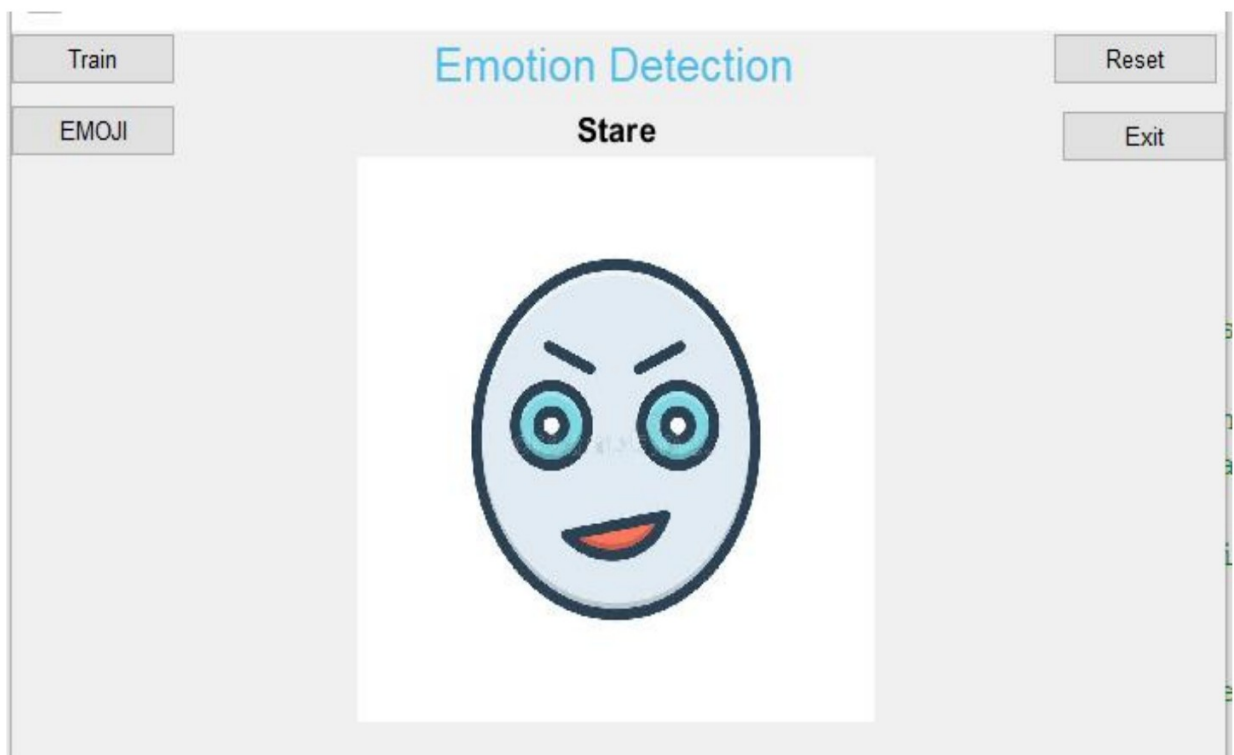


FIG 7.3

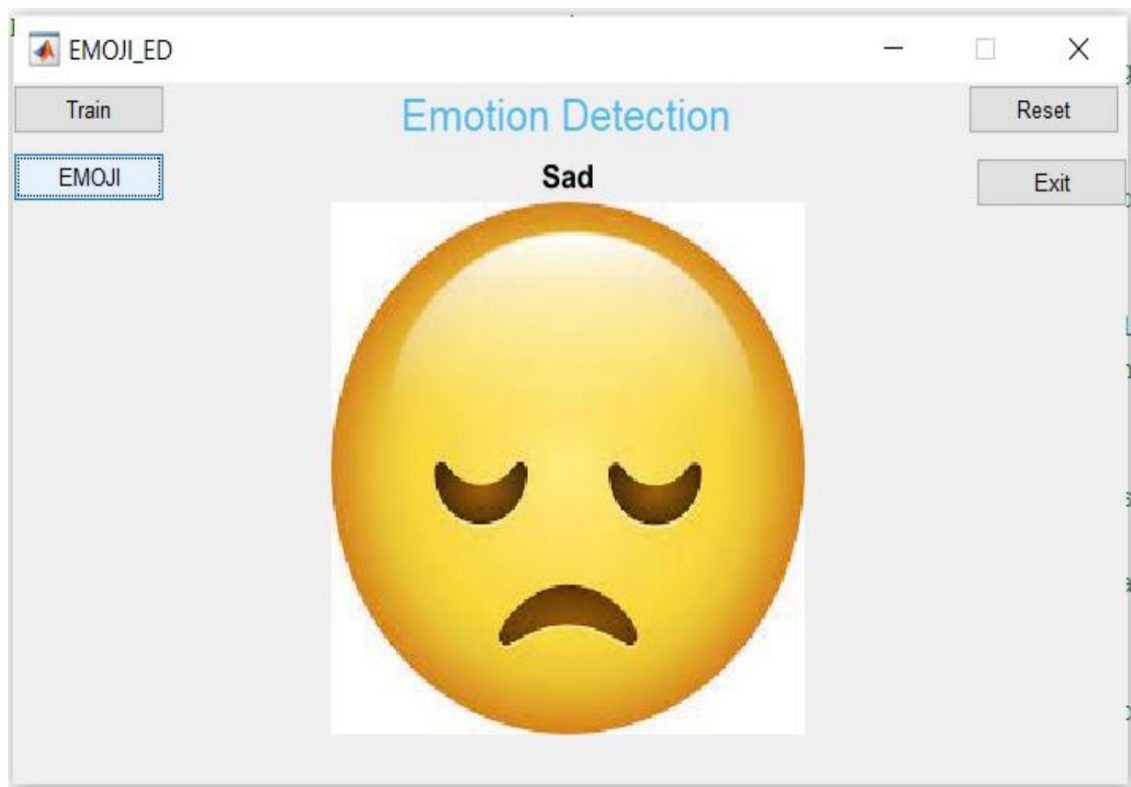


FIG 7.4

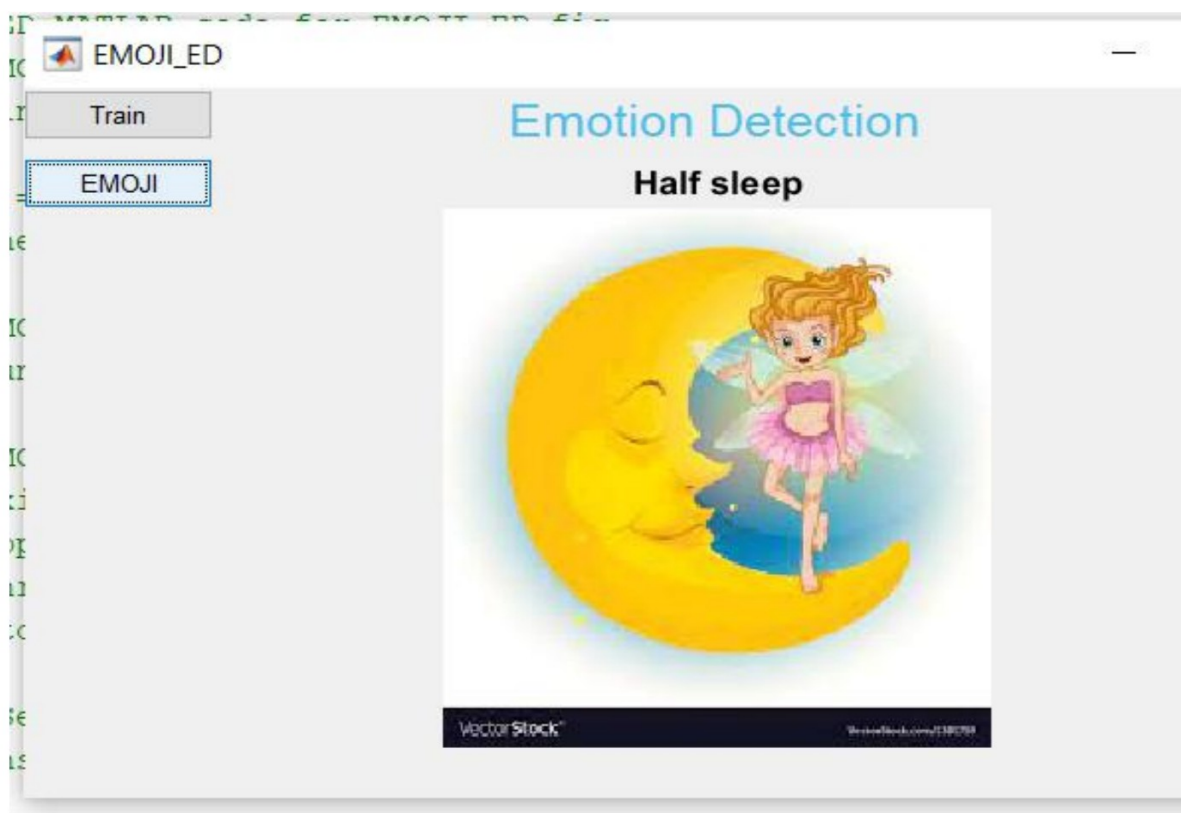


FIG 7.5

CHAPTER 8

CONCLUSION

8.1 CONCLUSION

Emojis are approaches to signify nonverbal cues. These cues have come to be a crucial part of on-line chatting, product review, logo emotion, and plenty of more. It additionally led to growing information technology studies devoted to emoji-pushed storytelling. We

construct a convolution neural community structure and educate the version on FER2013 dataset for emotion popularity from photographs additionally with improvements in computer –vision and deep learning, it's far viable to come across human feelings from photographs. In this project, we are able to classify human facial expressions to clear out and map corresponding emojis.

CHAPTER 9

REFERENCES

- [1]. Convolutional Neural Networks Architectures for Facial Expression Recognition Authors - George Porusniuc, Florin Leon, Radu Timofte, CasianMiron, Year Published - 2019 [8]Real Time Emotion Recognition from Facial Expressions Using CNN Architecture Authors - Mehmet Akif Ozdemir, BerkayElagoz, Reza Sadighzadeh, Aydin Akan, Year Published - 2019,IEEE
- [2]. Face Detection using Haar Cascades to Filter Selfie Face Image on Instagram, Authors - Adri Priadana, MuhammadHabibi, Year Published – 2019, IEEE [10]Facial Expression Recognition Using Neural Networks, Authors - Yoshitha Gali, Kevin Sanghoi and Mrinalini Chava, Year Published – 2019,IEEE
- [3]. Emotion and gender classification in realtime, Authors - Utkarsh Agrawal, Anjnesh Sharma, K. Senthil Kumar, Year Published – 2019
- [4]. Face Recognition with Age, Gender and Emotion Estimations, Authors - Dongqin Gu, Miaomiao Zhu, Lifeng Zhang, Seiichi Serikawa, Year Published – 2020,IEEE
- [5]. Facial Expression Recognition Using Convolutional Neural Network, Authors - Weier Wan, Chenjie Yang, Yang Li, Year Published – 2020,IEEE
- [6]. A Robust Improved Network for Facial Expression Recognition, Authors-Hao Gao, Bo Ma, Year Published- 2020
- [7]. Real-Time Facial Emotion Recognition with Deep Convolutional Neural Network Authors – Lakshmi Bhadana, P.V. Lakshmi, D. RamaKrishna,G. SuryaBharti, Y. VaibhavYear Published- 2020,IEEE
- [8]. Human Emotions Detection Using Hybrid CNN Approach, Authors - Nehmat Sandhu; Aksh
- [9]. U. S. Reddy, A. V. Thota and A. Dharun, "Machine Learning Techniques for Stress Prediction in Working Employees," 2018 IEEE International Conference on Computational Intelligence and Computing Research (ICIC), Madurai, India, 2018, pp. 1-4.

[10]. M. P. Dooshima, E. N. Chidozie, B. J. Ademola, O. O. Sekoni, I. P. Adebayo, “A Predictive Model for the Risk of Mental Illness in Nigeria Using Data Mining”, International Journal of Immunology. Vol. 6, No. 1, 2018, pp. 5-16.

[11] M. Srividya, M. Subramaniam and B. Natarajan, “Behavioral Modeling for Mental Health using Machine Learning Algorithms”, Journal of Medical Systems Vol. 42(5):88 May 2018

[12] S. G. Alonso, I. Torre-Díez, S. Hamrioui,

M. I López-Coronado, D. C. Barreno, L. M. Nozaleda, and M. Franco. “Data Mining Algorithms and Techniques in Mental Health: A Systematic Review”, Journal of Medical Systems 42(9), July 2018

[13] M. A. Haziq Megat S'adan, A. Pampouchidou and F. Meriaudeau, "Deep Learning Techniques

[14] for Depression Assessment," 2018 International Conference on Intelligent and Advanced System (ICIAS), Kuala Lumpur.

[15]]D. Filip & C. Jesus. (2015). “A Neural Network Based Model for Predicting Psychological Conditions”.