San Jose State University

SJSU ScholarWorks

Spring 2018

Facial Emotion Recognition Using Machine Learning

Nitisha Raut
San Jose State University

Follow this and additional works at: https://scholarworks.sjsu.edu/etd_projects

 Part of the Computer Sciences Commons

Facial Emotion Recognition Using Machine Learning

A Thesis

Presented to

The Faculty of the Department of Computer Science

San José State University

In Partial Fulfillment

Of the Requirements for the Degree

Master of Science

by

Nitisha Raut

May 2018

The Designated Thesis Committee Approves the Thesis Titled

Facial Emotion Recognition Using Machine Learning

by

Nitisha Raut

APPROVED FOR THE DEPARTMENTS OF COMPUTER SCIENCE

SAN JOSE STATE UNIVERSITY

May 2018

Dr. Robert Chun	Department of Computer Science
Dr. Sami Khuri	Department of Computer Science
Dr. Mark Stamp	Department of Computer Science

Abstract

Face detection has been around for ages. Taking a step forward, human emotion displayed by face and felt by brain, captured in either video, electric signal (EEG) or image form can be approximated. Human emotion detection is the need of the hour so that modern artificial intelligent systems can emulate and gauge reactions from face. This can be helpful to make informed decisions be it regarding identification of intent, promotion of offers or security related threats. Recognizing emotions from images or video is a trivial task for human eye, but proves to be very challenging for machines and requires many image processing techniques for feature extraction. Several machine learning algorithms are suitable for this job. Any detection or recognition by machine learning requires training algorithm and then testing them on a suitable dataset. This paper explores a couple of machine learning algorithms as well as feature extraction techniques which would help us in accurate identification of the human emotion.

Acknowledgments

I would like to thank Dr. Robert Chun for his continued support and providing me the guidance necessary to work on this project. I would like to thank my advisor Dr. Robert Chun and committee members Dr. Sami Khuri and Dr. Mark Stamp for teaching me core skills needed to succeed and reviewing my project. And finally I would like to thank my parents for their patience and advice they gave me throughout my life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. Introduction

Human emotion detection is implemented in many areas requiring additional security or information about the person. It can be seen as a second step to face detection where we may be required to set up a second layer of security, where along with the face, the emotion is also detected. This can be useful to verify that the person standing in front of the camera is not just a 2-dimensional representation [1].

Another important domain where we see the importance of emotion detection is for business promotions. Most of the businesses thrive on customer responses to all their products and offers. If an artificial intelligent system can capture and identify real time emotions based on user image or video, they can make a decision on whether the customer liked or disliked the product or offer.

We have seen that security is the main reason for identifying any person. It can be based on finger-print matching, voice recognition, passwords, retina detection etc. Identifying the intent of the person can also be important to avert threats. This can be helpful in vulnerable areas like airports, concerts and major public gatherings which have seen many breaches in recent years.

Human emotions can be classified as: fear, contempt, disgust, anger, surprise, sad, happy, and neutral. These emotions are very subtle. Facial muscle contortions are very minimal and detecting these differences can be very challenging as even a small difference results in different expressions [4]. Also, expressions of different or even the same people might vary for the same emotion, as emotions are hugely context dependent [7]. While we can focus on only those areas of the face which display a maximum of emotions like around the mouth and eyes [3], how we

extract these gestures and categorize them is still an important question. Neural networks and machine learning have been used for these tasks and have obtained good results.

Machine learning algorithms have proven to be very useful in pattern recognition and classification. The most important aspects for any machine learning algorithm are the features. In this paper we will see how the features are extracted and modified for algorithms like Support Vector Machines [1]. We will compare algorithms and the feature extraction techniques from different papers. The human emotion dataset can be a very good example to study the robustness and nature of classification algorithms and how they perform for different types of dataset. Usually before extraction of features for emotion detection, face detection algorithms are applied on the image or the captured frame. We can generalize the emotion detection steps as follows:

1) Dataset preprocessing

2) Face detection

3) Feature extraction

4) Classification based on the features

In this work, we focus on the feature extraction technique and emotion detection based on the extracted features. Section 2 focuses on some important features related to the face. Section 3 gives information on the related work done in this field. Related work covers many of the feature extraction techniques used until now. It also covers some important algorithms which can be used for emotion detection in human faces. Section 4 details the tools and libraries used in the implementation. Section 5 explains the implementation of the proposed feature extraction and emotion detection framework. Section 6 highlights the result of the experiment. Section 7 covers the conclusion and future work.

2. Image Features

We can derive different types of features from the image and normalize it in vector form. We can employ various types of techniques to identify the emotion like calculating the ellipses formed on the face or the angles between different parts like eyes, mouth etc. Following are some of the prominent features which can be used for training machine learning algorithms:

2.1 FACS

Facial Action Coding System is used to give a number to facial moment. Each such number is called as action unit. Combination of action units result in a facial expression. The micro changes in the muscles of the face can be defined by an action unit. For example, a smiling face can be defined in terms of action units as 6 + 12, which simply means movement of AU6 muscle and AU12 muscle results in a happy face. Here Action Unit 6 is cheek raiser and Action Unit 12 is lip corner puller. Facial action coding system based on action units is a good system to determine which facial muscles are involved in which expression. Real time face models can be generated based on them.

Figure 1: Action Units corresponding to different movements in face [15]

3

2.2 Landmarks

Landmarks on the face are very crucial and can be used for face detection and recognition. The same landmarks can also be used in the case of expressions. The Dlib library has a 68 facial landmark detector which gives the position of 68 landmarks on the face.
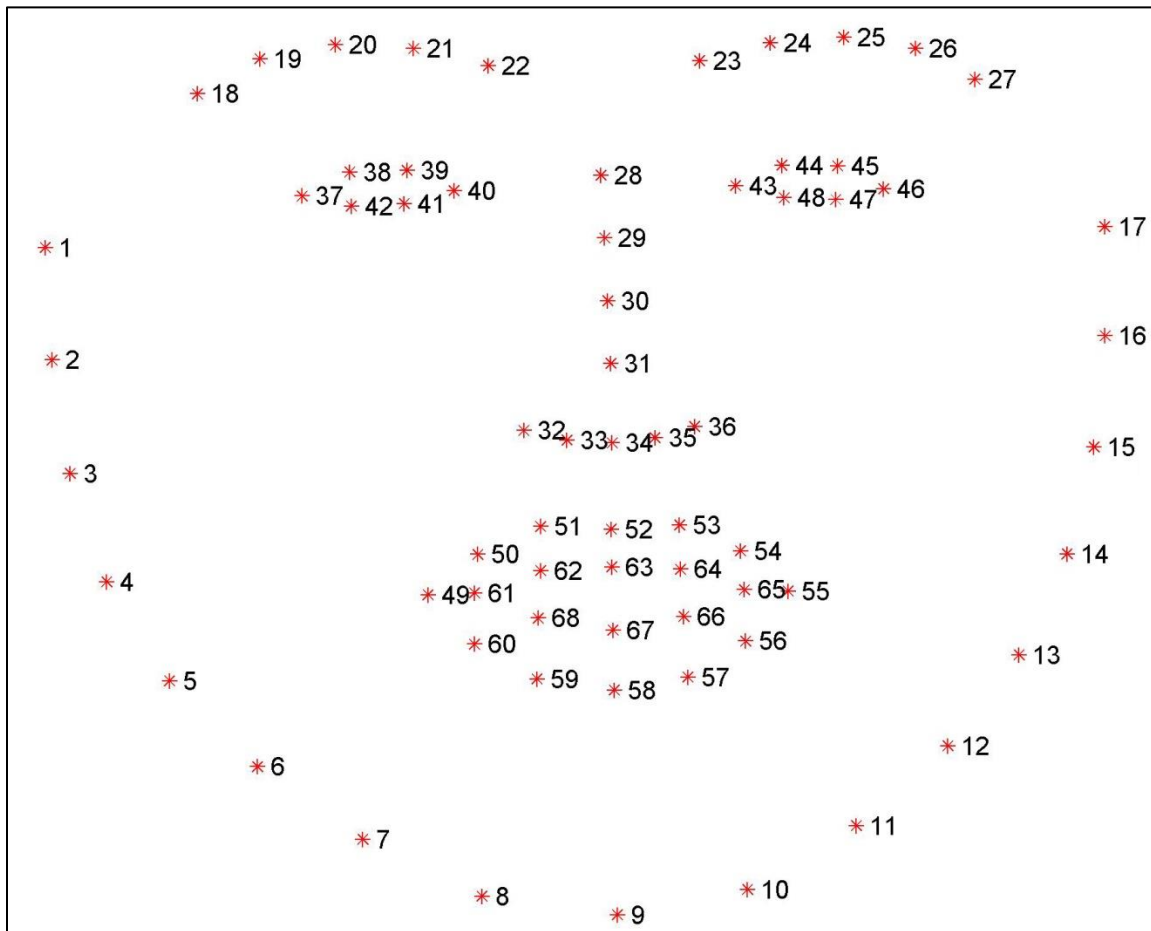
Figure 2: Landmarks on face [18]

Figure 2 shows all the 68 landmarks on face. Using dlib library we can extract the co-ordinates(x,y) of each of the facial points. These 68 points can be divided into specific areas like left eye, right eye, left eyebrow, right eyebrow, mouth, nose and jaw.

2.3 Feature Descriptors

Good features are those which help in identifying the object properly. Usually the images are identified on the basis of corners and edges. For finding corners and edges in images, we have many feature detector algorithms in the OpenCV library such as Harris corner detector. These feature detectors take into account many more factors such as contours, hull and convex. The Key-points are corner points or edges detected by the feature detector algorithm. The feature descriptor describes the area surrounding the key-point. The description can be anything including raw pixel intensities or co-ordinates of the surrounding area. The key-point and descriptor together form a local feature. One example of a feature descriptor is a histogram of oriented gradients. ORB (based on BRIEF), SURF, SIFT etc. are some of the feature descriptor algorithms [25].

3. Related Work

3.1 Feature Extraction Techniques

3.1.1 Ensemble of regression trees

This method uses cascaded regression trees and finds the important positions on the face using images. Pixel intensities are used to distinguish between different parts of the face, identifying 68 facial landmarks [1]. Based on a current estimate of shape, parameter estimation is done by transforming the image in the normal co-ordinate system instead of global. Extracted features are used to re-estimate the shape parameter vectors and are recalculated until convergence [5].

Figure 3: Image with19 feature points [1]

The author [1] uses only 19 features as shown in Figure 3 from the 68 extracted features, focusing only around eyes, mouth and eyebrows.

Ensemble of regression trees was very fast and robust giving 68 features in around 3 milliseconds. The parameters tuned for the algorithm are shown in Table 1 [1]:

Table 1: Parameters for tuning ensemble of regression trees algorithm [1]

Parameter	Value
Cascade depth	15
Tree depth	4
Number of trees per cascade level	500
Number of test splits	20

3.1.1.1 Displacement ratios

Once the features are in place, the displacement ratios of these 19 feature points are calculated using pixel coordinates. Displacement ratios are nothing but the difference in pixel position in the image from initial expression to final expression. Twelve types of distances are calculated as shown in Table 2 [1].

Instead of using these distances directly, displacement ratios are used as these pixel distances may vary depending on the distance between the camera and the person. The dataset used for this experiment was the iBug-300W dataset which has more than 7000 images along with CK + dataset having 593 sequences of facial expressions of 123 different subjects.

Table 2: Distances calculated to determine displacement ratios between different parts of face [1]

Distance	Description of the distances
D1 and D2	Distance between the upper and lower eyelid of the right and left eyes
D3	Distance between the inner points of the left and right eyebrow
D4 and D5	Distance between the nose point and the inner point of the left and right eyebrow
D6 and D8	Distance between the nose point and the right and left mouth corner
D7 and D9	Distance between the nose point and the midpoint of the upper and lower lip
D10	Distance between the right and left mouth corner
D11	Distance between the midpoint of the upper and lower lip
D12	Mouth circumference

3.1.2 Eulerian Motion Magnification (EMM)

Subtle emotions are hard to detect. If we magnify the emotions, there is a possibility of increasing the accuracy of detection. Motion properties such as velocity and acceleration can be used for magnification. Image as a whole is transformed by magnifying changes in properties of amplitude and phase. Based on the properties, there are A-EMM (Amplitude based) and P-EMM (Phase based) motion magnification [4].

3.1.2.1 Amplitude- Based Eulerian Motion Magnification

Suppose $I(x,t)$ is an image profile at location x at time t. If the image undergoes a translational motion with a displacement function δ(t), the image is given by [4]:

$I(x,t) = = f(x + δ(t))$ and $I(x,0) = f(x)$

Pixel intensity *I* of a magnified motion is computed as follows, where *B(x,t)* is differences of intensity and given that *I(x,t) = I(x) + B(x,t)*

$\hat{I}(x,t) = I(x) + \alpha * B(x,t)$, where α = magnification factor
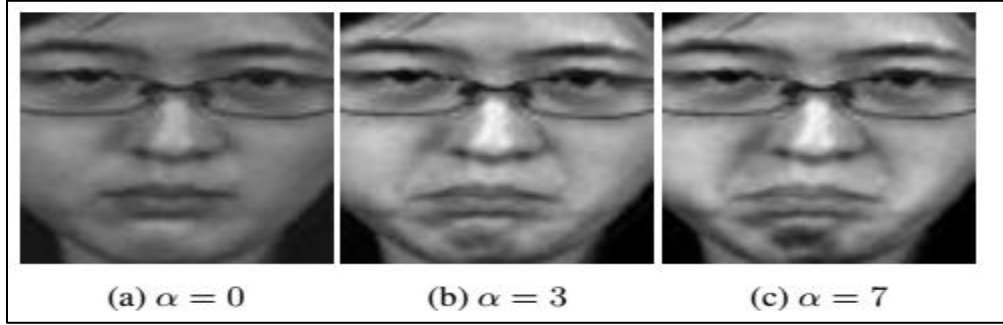
Figure 4: A- EMM using different values of α [4]

Î can be approximated by the first-order Taylor series as follows

$\hat{I}(x, t) \approx f(x) + \sum k\,\alpha\,B(x, t)$, where k is passband of a temporal filter with a

corresponding attenuation factor γk and *B(x, t)* is output of temporal bandpass filter:

$B(x, t) = \sum \gamma k\,\delta(t)\,\delta f(x) / \delta x$

3.1.2.2 Phase – Based Eulerian Motion Magnification

Image profile *I(x,t) = = f(x + δ(t))* for P-EMM can be written as

$f(x + \delta(t)) = \sum_{(\omega=-\infty,\infty)} A_\omega e^{i\omega(x+\delta(t))} = \sum_{(\omega=-\infty,\infty)} I_\omega A_\omega e^{i\omega\delta(t)}$
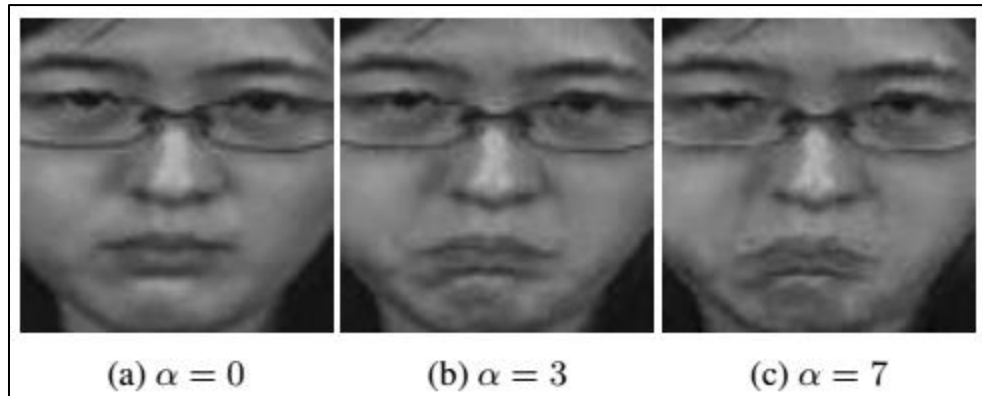
Figure 5: P- EMM using different values of α [4]

9

3.1.2.3 LBP-TOP feature extraction from EMM

Once the expressions are exaggerated using A-EMM or P-EMM and with magnification factor α, Local Binary Pattern with Three Orthogonal Plane (LBP – TOP) is used for feature extraction. For LBP-TOP feature extractions, the following steps are followed [4]:

1) Spatially resize image to 320 x 240 pixels resolution

2) Partition images into non-overlapping 5 x 5 blocks

3) Blocks are stacked up in 3-D volumes

From these blocks, spatial-temporal features and histograms of binary patterns are extracted by LBP-TOP$_{4,4,4,1,1,4.}$ Here, 4,4,4 (first 3 numbers) represent 4-neighbor connections in three orthogonal planes and 1,1,4 (last 3 numbers) are radii of the respective connections [4]. The dataset used for this experiment was CASME2 dataset which is a subtle emotion dataset.

3.1.3 Face detection using Viola-Jones face detector

Images are pre-processed to reduce noise. Also, dataset must have images with equal level of exposure, illumination and brightness. Image enhancement is performed on such images using Histogram equalization techniques. The face detection module is the familiar Viola- Jones technique which uses Ada-boost algorithm. It combines a lot of weak classifiers to form a strong classifier by iterations, where a weak classifier reduces the weighted error rate per iteration [6].

3.1.3.1 LBP technique for feature extraction

Local Binary Pattern (LBP) is a very simple and robust technique for feature extraction and is independent of the illumination differences in the images. A 3x3 matrix is generated and each of the pixels in the matrix is assigned a binary value depending on the center pixel value. These 8- bit binary values form an 8 bit binary number excluding the center pixel value which is finally converted to decimal value [6].

LBP code for a pixel at (x_c, y_c) is given by [6]

$$LBP_{P,R} (x_c, y_c) = \sum_{(P=0,7)} S(g_p-g_c) \, 2^P, \, S(x) = \{1, x >= 0 \text{ and } 0, x < 0\}$$

where,

g_c = gray value of center pixel

g_p = gray value of neighboring pixel of g_c

P = 8 (maximum 8 neighbors for center pixel in a 3x3 matrix)

Hence a pixel can have 256 different values as $2^P = 2^8 = 256$.

Figure 6 shows an input image block with center pixel value as 7. Each value of the surrounding 8 pixels is reassigned by comparing it to the intensity of the center pixel. Pixel values greater than or equal to the center pixel are assigned 1; otherwise 0. Figure 7 shows the binary values assigned to all the 8 pixels, which combined in clockwise direction, gives a binary 8 bit number. Converting the 8-bit number in the Figure 6 into decimal value (11000010) gives us the number 194 [6].

12	15	18		G_7	G_6	G_5
5	7	3	⇨	G_0	G_c	G_4
8	1	2		G_1	G_2	G_3
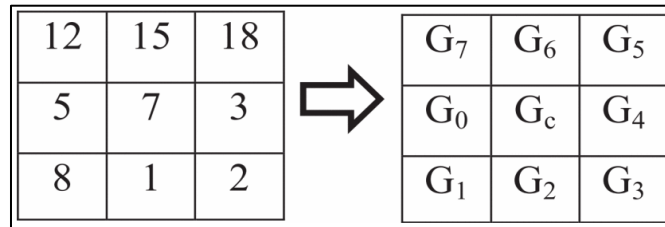
Figure 6: 3x3 initial input matrix [6]

B_7	B_6	B_5		1	1	0
B_0		B_4	⇨	0		0
B_1	B_2	B_3		1	0	0
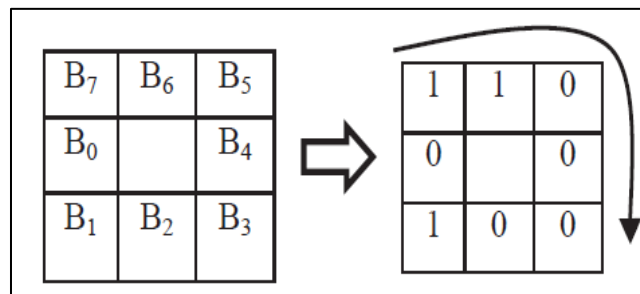
Figure 7: Pixel encoding using LBP [6]

We can encode all such pixel blocks in an image and use the decimal values obtained as image feature vectors for classification. LBP is simple to implement as the binary pattern remains the same in spite of changes in the illumination or brightness. An increase in brightness or illumination condition will result in all the pixel intensities increasing by the same value, keeping the relative difference the same [6].

3.1.4 Using dynamic grid – based HoG features

Histogram of oriented gradient (HoG) is an object detection technique based on edge information. Each detection window can be defined by edge orientations. Image is cropped according to the area of interest on the face. This cropped face is the detection window which is divided into even smaller set of regions called cells. In each cell, for each pixel, a magnitude of edge gradient is computed for each orientation bin, thus forming the local histogram of oriented gradients [7].

As Figure 8 shows, from the face, required region is cropped and divided into a matrix of 8 rows x 6 columns. Pixel size may vary in the grid for each cell.
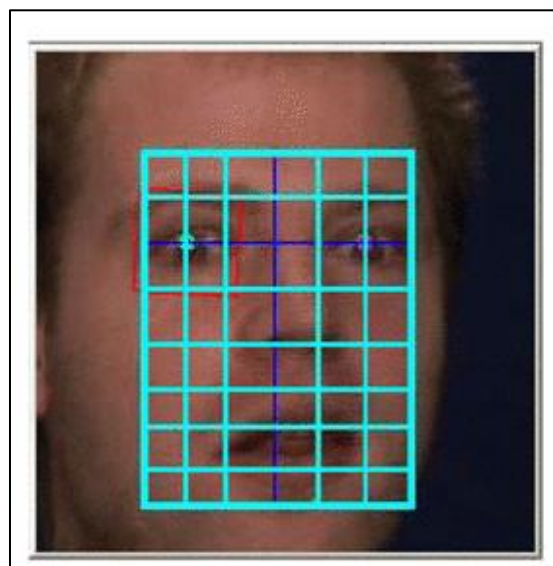
Figure 8: Cropped image divided into cells of 8 rows x 6 columns [7]

A local histogram is generated for each cell. Further, for each block of 2 x 2 cells, 4 histograms are concatenated. Finally we have 12 normalized histograms concatenated into one single global histogram which gives us 432 feature vectors [7].

3.1.5 Geometrical facial features extraction

A set of 19 features are selected empirically by observing the landmark positions on the face and which are more related to human expressions. These 19 features are taken as a subset from an existing marker-less system for landmark identification and localization, which has actual 66 2D features [9] [10]. These 19 features or landmarks on the face are given in Table 3 and Figure 9. Landmark positions in the image space are used to define two set of features: eccentricity features and linear features [9].

3.1.5.1 Eccentricity features

Eccentricity features are based on the concept of ellipses. The eccentricity of ellipses is the amount of deviation of the ellipse from being a circle. Eccentricity is between 0 and 1 for ellipses and 0 if the ellipse is a circle. For example, while smiling, the eccentricity will be greater than 0; but while expressing surprise it will be closer to 0 [9].

Refer to Figure 10. Here, A_M and B_M are the end-points of the major-axis which is the length of the mouth, whereas U_{m1} and D_{m2} are the upper and lower end points of the minor axis respectively. The eccentricity is calculated for the upper and lower half separately. For the upper ellipse (A_M, B_M, $U_{m1)}$ eccentricity is given by,

$$e = \sqrt{a^2 - b^2} / a$$

where,

$$a = B_{Mx} - A_{Mx} / 2$$

$$b = A_{My} - U_{m1y}$$

Table 3: 19 landmark points [9]

No.	Landmark	Label	Region
1	Right Cheilion	A_M	Mouth
2	Left Cheilion	B_M	Mouth
3	Labiale Superius	U_{m1}	Mouth
4	Labiale Inferius	D_{m2}	Mouth
5	Left Exocanthion	Ell_M	Left Eye
6	Right Exocanthion	Elr_M	Left Eye
7	Palpebrale Superius	UEl_{m3}	Left Eye
8	Palpebrale Inferius	DEl_{m4}	Left Eye
9	Left Exocanthion	Erl_M	Right Eye
10	Right Exocanthion	Err_M	Right Eye
11	Palpebrale Superius	UEr_{m5}	Right Eye
12	Palpebrale Inferius	DEr_{m6}	Right Eye
13	Zygofrontale	$EBll_M$	Left Eyebrown
14	Inner Eyebrown	$EBlr_M$	Left Eyebrown
15	Superciliare	$UEBl_{m7}$	Left Eyebrown
16	Inner Eyebrown	$EBrl_M$	Right Eyebrown
17	Zygofrontale	$EBrr_M$	Right Eyebrown
18	Superciliare	$UEBr_{m8}$	Right Eyebrown
19	Subnasale	SN	Nose
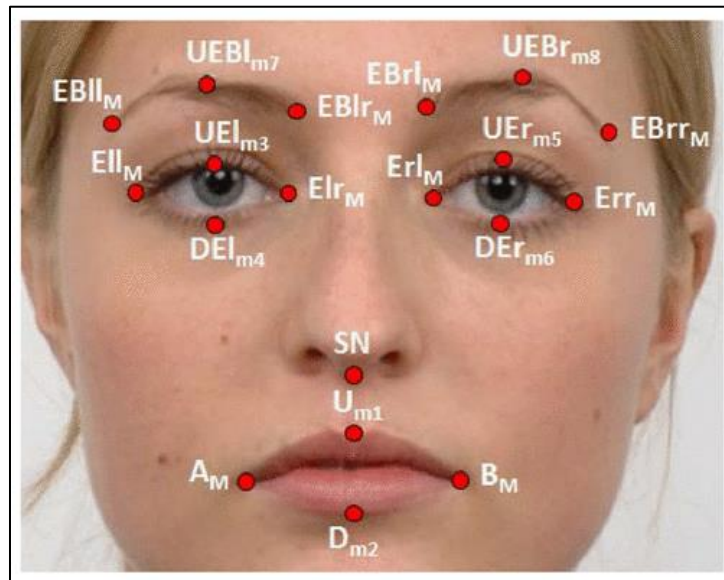
Figure 9: Position of 19 landmark points on face [9]

14

3.5.1.1 Eccentricity extraction algorithm
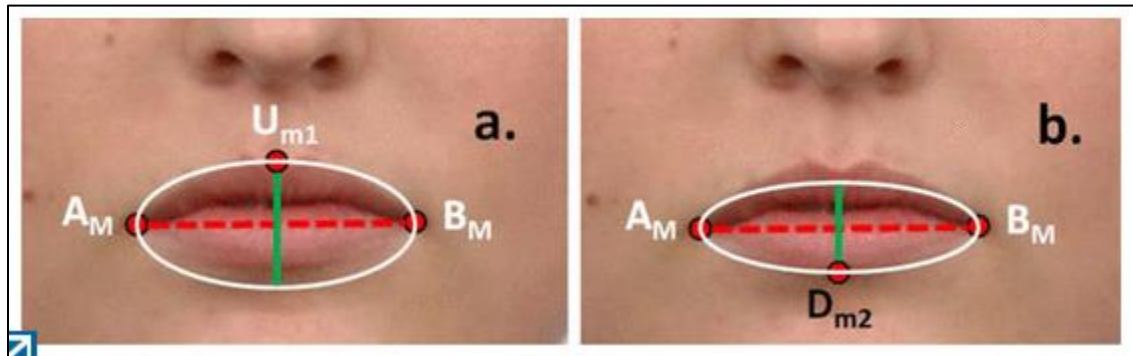
Figure 10: Ellipse for the upper part of mouth (a). Ellipse for lower part of mouth (b) [9]

In the same way eccentricity is calculated for the other 7 ellipses: lower mouth, upper left eye, lower left eye, upper right eye, lower right eye, left eyebrow and right eyebrow. In Figure 11, part a shows eight ellipses, whereas part b shows the change in eccentricity as the person smiles.
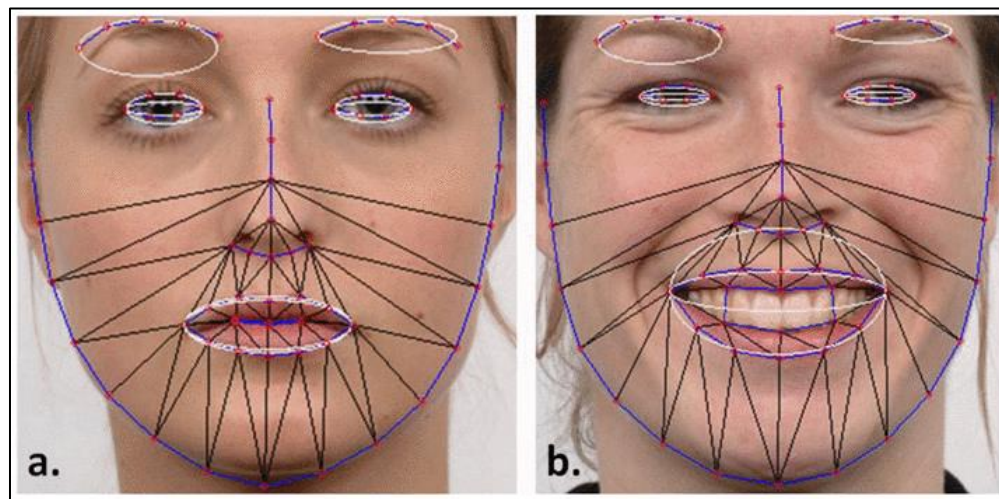
Figure 11: ellipses construction for different areas on face(a).changes in ellipse eccentricity(b) [9]

3.5.1.2 Linear features

Movements that occur during expressing emotions between facial landmarks can be quantitatively evaluated using distances. These are the normalized linear distances.

The following distances are calculated to determine the linear feature vectors [9]:

L_1 – Movements between eyes and eyebrows

L_2 – Movements between mouth and nose

L_3 – Movements between upper and lower mouth points

3.2 Machine Learning Algorithms

Once the dataset is created using the required features, the next important step is to use a good classification algorithm. Support Vector Machines (SVM) are used almost in all cases of multi-class classification of human expressions [1][7][8]. They are combined with one or another feature extraction technique [8].

3.2.1 Support Vector Machines (SVM)

SVM are one of the most powerful classification algorithms. The idea is to find an optimal hyper plane which divides the two classes accurately. There is also a concept of margin, which is the supposed to be maximum from both the classes so as to avoid any overlapping between two classes [11]. Data which is not linearly separable is mapped into a higher dimension to achieve better classification results. Kernel functions such as radial basis function (rbf) and polynomial are used for non-linear data [1].

In case of emotion detection, usually a multi-class SVM is used instead of a binary to detect emotions such as anger, contempt, disgust, fear, happy, sadness and surprise [1]. K-fold cross-validation is used to remove any variances in the database and to compare different machine learning algorithms [9]. In k-fold cross validation, the dataset is divided k times into k slices, and prediction results are averaged over all iterations.

Loconsole et al. [8] used Principal component analysis (PCA) for feature set reduction and then feeding the reduced feature set to SVM. In PCA algorithm, the image feature space is transformed to eigen space using an eigen matrix [2].

Along with kernel specification, SVM has methods for tuning parameters like C and γ [7]. Here, C is the penalty function for misclassification and gamma helps to optimize the decision boundary. Both these parameters affect the accuracy of the classifiers and can be tuned to get optimal results in both binary and multi-class classification.

3.2.2 Hidden Markov Models (HMM)

Hidden Markov Models (HMM) is based on statistics, and are useful for finding hidden structure of data. They are also very popular for emotion detection through speech [12]. Input is a sequence of observed features and there are hidden states corresponding to consecutive events. HMM is expressed as follows [14]:

$\lambda = (A, B, \pi)$

where,

$A = (a_{ij})$ transition probability matrix between the hidden states

$B = (b_{ij})$ observation symbols probability from a state

Π = initial probability of states.

Paper [13] describes the process of developing Code – HMM. It tries to improve on the existing HMM by incorporating some characteristics of multiple classifiers. Also, HMM are used in sequence with algorithms such as k-Nearest Neighbor [13]. Advantage of using both the methods is HMM can do the complex computations and k-NN just have to classify between the given samples. HMM decision is based on biggest output probability which might be mixed with noise, whereas K-NN can add a second layer of classification thereby increasing accuracy.

17

HMM's also used in combination with SVM as Serial Multiple Classifier System to get best results for speech emotion recognition [12]. As SVM directly gives a classification instead of a score, HMM's can be used for training the samples and SVM for classification. Along with multiple classifiers, boosting can also be used as a means of developing a strong classification system where two or more weak classifiers are combined to form a strong classifier [15]. The paper [15] also talks about embedding HMM, i.e. developing a two-dimensional HMM, consisting of super states and embedded states. The data is modeled in two directions by super states and embedded states. For face images, top to bottom features can be super states and right –to-left features can be embedded states.

3.2.3 Other Algorithms

Random Forest Classifiers have also proven to have an upper hand over SVM in some of the cases [9]. Random forests are based on decision trees, but instead of just one classifier, use more forests or classifiers to decide the class of the target variable. Table 4 gives us the result of random forest classifier for detecting 6,7 and 8 emotions. Here, S1 to S5 represent the subset of emotion used for detection.

K-Nearest Neighbor, Linear Discriminant Analysis and Neural Networks (ANN) are some of the algorithms used for classification of prediction of emotion.

Table 4: Random forest Classifier results. * means without considering contemptuous emotion, without considering neutral emotion, *** without considering neutral and contemptuous emotion [9]**

No. tested emotions	S1[%]	S2[%]	S3[%]	S4[%]	S5[%]
8	51	76	80	86	89
7*	61	80	84	88	90
7**	60	81	84	90	92
6***	67	87	89	91	94

4.TOOLS AND LIBRARIES USED

4.1 OpenCV

OpenCV is the library we will be using for image transformation functions such as converting the image to grayscale. It is an open source library and can be used for many image functions and has a wide variety of algorithm implementations. C++ and Python are the languages supported by OpenCV. It is a complete package which can be used with other libraries to form a pipeline for any image extraction or detection framework. The range of functions it supports is enormous, and it also includes algorithms to extract feature descriptors.

4.2 Dlib

Dlib is another powerful image-processing library which can be used in conjunction with Python, C++ and other tools. The main function this library provides is of detecting faces, extracting features, matching features etc. It has also support for other domains like machine learning, threading, GUI and networking.

4.3 Python

Python is a powerful scripting language and is very useful for solving statistical problems involving machine learning algorithms. It has various utility functions which help in pre-processing. Processing is fast and it is supported on almost all platforms. Integration with C++ and other image libraries is very easy, and it has in-built functions and libraries to store and manipulate data of all types. It provides the pandas and numpy framework which helps in manipulation of data as per our need. A good feature set can be created using the numpy arrays which can have n-dimensional data.

4.4 Scikit-learn

Scikit-learn is the machine learning library in python. It comprises of matplotlib, numpy and a wide array of machine learning algorithms. The API is very easy to use and understand. It has many functions to analyze and plot the data. A good feature set can be formed using many of its feature reduction, feature importance and feature selection functions. The algorithm it provides can be used for classification and regression problems and their sub-types.

4.5 Jupyter Notebook

Jupyter Notebook is the IDE to combine python with all the libraries we will be using in our implementation. It is interactive, although some complex computations require time to complete. Plots and images are displayed instantly. It can be used as a one stop for all our requirements, and most of the libraries like Dlib, OpenCV, Scikit-learn can be integrated easily.

4.6 Database

We have used the extended Cohn-Kanade database (CK+) and Radbound Faces database(RaFD). CK+ has around 593 images for 123 subjects. Only 327 files have labeled/identified emotions. It covers all the basic human emotions displayed by the face. The emotions and codes are as follows: 1 – Angry, 2 – Contempt, 4 – Fear, 5 – Happy, 6 – Sadness, 7 – Surprise. The database is widely used for emotion detection research and analysis. There are 3 more folders along with the images. FACS contains action units for each image. Landmark contains AAM tracked facial features of the 68 facial points. Emotion contains emotion label for the 327 files.

Radbound Faces Database [19] is a standard database having equal number of files for all emotions. It has images of 67 subjects displaying 8 emotions: Neutral included. The pictures are taken in 5 different camera poses. Also, the gaze is in 3 directions. We are using only front

facing images for our experiment. We have a total of 536 files with 67 models displaying 8 different emotions.

Figure 12: Various emotions from the Cohn-Kanade database

5. Implementation

A static approach using extracted features and emotion recognition using machine learning is used in this work. The focus is on extracting features using python and image processing libraries and using machine learning algorithms for prediction. Our implementation is divided into three parts. The first part is image pre-processing and face detection. For face detection, inbuilt methods available in dlib library are used. Once the face is detected, the region of interest and important facial features are extracted from it. There are various features which can be used for emotion detection. In this work, the focus is on facial points around the eyes, mouth, eyebrows etc.

We have a multi-class classification problem and not multi-label. There is a subtle difference as a set of features can belong to many labels but only one unique class. The extracted facial features along with SVM are used to detect the multi-class emotions. The papers we have studied focus on SVM as one of the widely used and accepted algorithms for emotion classification. Our database has a total of 7 classes to classify. We have compared our results with logistic regression and random forest to compare the results of different algorithms. The processing pipeline can be visualized as Figure 13.

5.1 Setting up the database

The image files for the CK+ database are in different directories and sub-directories based on the person and session number. Also, not all the images depict emotion; only 327 files have one of the emotion depicted from 1-7. All the files were of type portable networks graphic file(.png). The emotion labels are in the different directory but with the same name as image files. We wrote a small utility function in java which used the emotion file name to pick up the

correct image from the directory and copy it in our final dataset folder. We also appended the name of the emotion file to the image file name. Thus, while parsing the file in our program we will have the emotion label for that file.
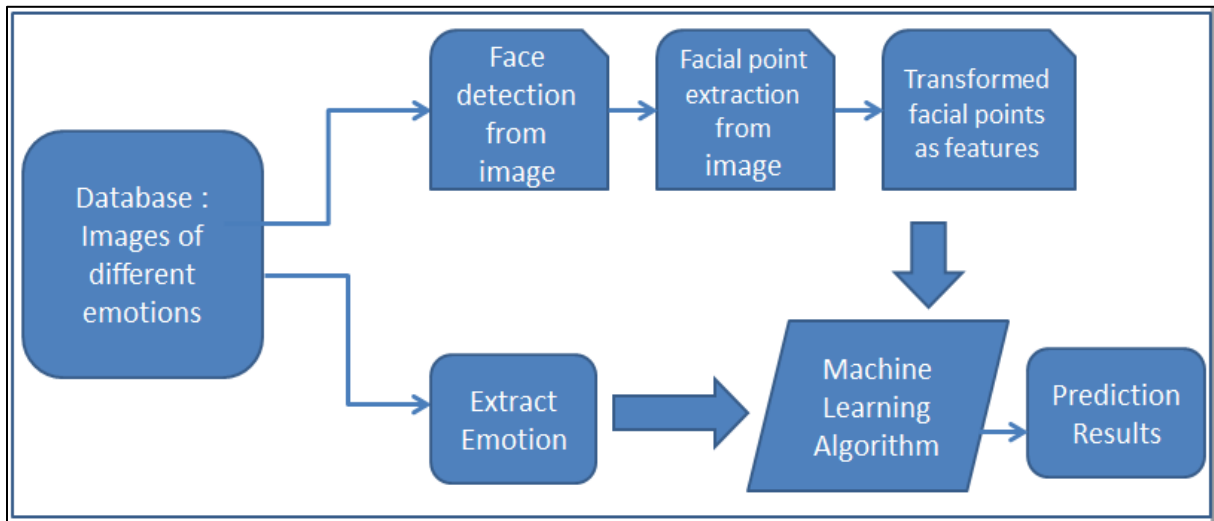
Figure 13: Implementation Pipeline

For example, in the filename S137_001_00000014_7, S137 represents the subject number, 001 the session number, 00000014 the image number in the session and finally 7 represents the emotion the subject is posing.

The dataset we created consisted of only frontal face images, and there was no file with no-emotion (no neutral emotion). The lighting and illumination condition for some images was different. Some images were colored. The processing pipeline for all the images was the same, in spite of the illumination conditions.

For the RaFD database we simply extracted the name of the emotion from image file name which was in jpg format. As this database was standard we had a balanced number of classes for each emotion. Table 5 shows the distribution of different emotion classes.

Table 5: Number of images per class for CK+ database

Emotion	Number of images depicting the emotion
1: Anger	45
2: Contempt	18
3: Disgust	59
4:Fear	25
5: Happy	69
6: Sadness	28
7: Surprise	83
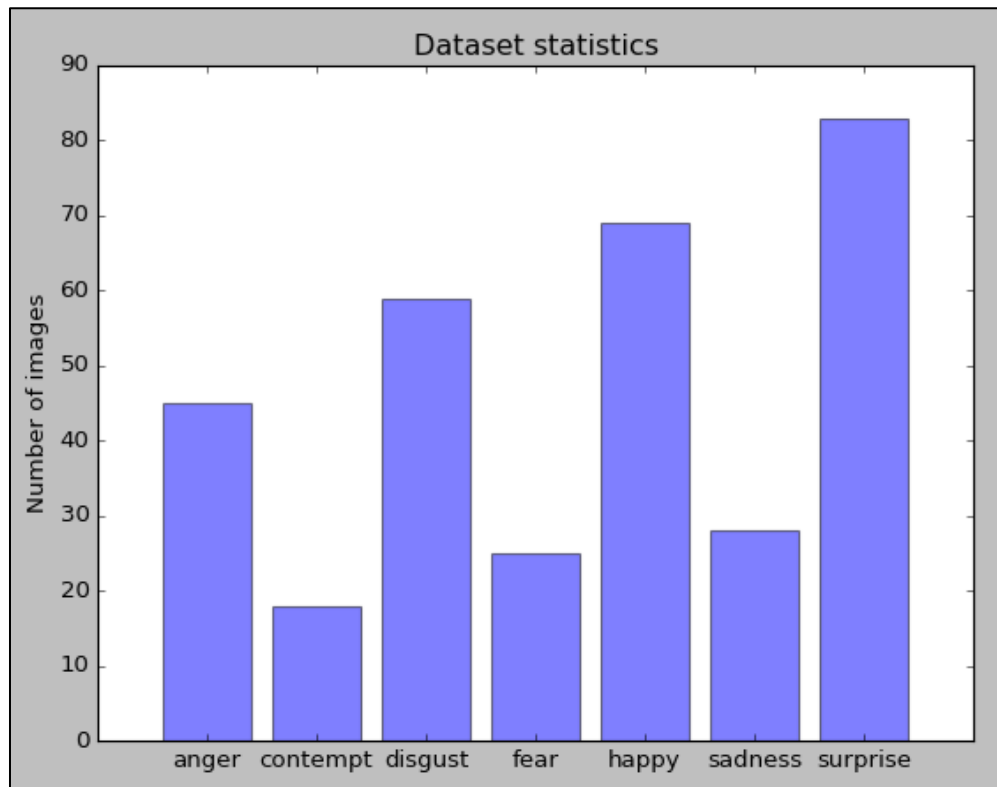
Figure 14: Bar plot of number of samples for each class

From Figure 14 we see that the numbers of classes are not equal and this might result in some classes being misclassified. For example contempt has very less number of samples (18); hence if none of the samples are present in training, it will be difficult to classify the class contempt in the testing data set. Moreover, due to less training samples, the class can also be treated as an outlier. The algorithm can become biased towards the emotion surprise, and classify most images as surprise. For RaFD database there was no bias towards any single class.

5.2 Image processing pipeline

5.2.1 Face detection

Face detection was the first and important part of the processing pipeline. Before further processing, we had to detect the face, even though our images contained only frontal facial expression data. Once the face was detected, it was easier to determine the region of interest and extract features from it.

Figure 15: Original image from the database and detected face from the image

For face detection, we tried many algorithms like Haar-cascades from OpenCV. Finally we settled for face detector based on histogram of oriented gradients from Dlib library. HoG

descriptors along with SVM are used to identify the face from the image. Images are converted to grayscale and resized for uniformity.

5.2.2 Facial feature extraction

For facial feature extraction, we used the 68 landmark facial feature predictor from dlib. The face detector algorithm returns a window(x,y,width,height) which is the detected face. The detected face is passed to the feature predictor algorithm. Figure 16 shows the detected 68 landmarks for a particular face. The predictor function returns the 68 points at the eyes(left and right), mouth, eyebrows(eft and right), nose and jaw. We used numpy array to convert the 68 points to an array of 68 x and y co-ordinates representing their location. These are the facial features we have used to predict emotion.

Figure 16: Detected landmarks from the face

The landmarks are easier to access in numpy array form. Also, from Figure 16 we know the indices of each feature, hence we can focus on a particular feature instead of the entire set.

The feature points are divided as 1-17 for jaw, 49-68 for mouth and so on. So, for instance, if we want to ignore the jaw, we can simply put the x and y co-ordinates for the jaw as 0, while converting the features into numpy array. We also calculated distances and polygon areas for some of the facial landmarks.

5.2.3 Python pipeline

The dataset of 327 files was stored in a directory and each file was processed to create the feature set. As soon as the file was picked up, the name of the file was parsed to extract the emotion label. The emotion label was appended to a list of labels which will form our multi-class target variable. The image was processed for face detection and feature prediction. The features derived from each file were appended to a list which was later converted to a numpy array of dimension 327*68*2. We also had the target classes in the form of a numpy array. Same process was followed for RaFD database.

5.2.4 Machine learning

Once we had created the feature set and the target variable, we used Support Vector Machines to predict the emotions. Sklearn machine library was used to implement the Support Vector Machines (SVM) and Logistic Regression algorithms. The multiclass strategy used was "One-Vs-Rest" for all the algorithms. Logistic regression algorithm was fine tuned for penalty "l1" and "l2". We also fine-tuned the linear kernel to rbf and poly to see the variation in results. Cross-validation technique was used along with SVM to remove any biases in the databases. Initially the dataset was divided as 70% for training and 30% for testing. We tried many other splits such as 80:20 and 70:30. 70:30 split seemed more appealing as our assumption was all classes will be equally represented in the test set. For cross-validation score we initially tested with 4 splits. To improve the results we chose the value 5 and 10, which are standard values for

cross-validation. Random Forest Classifier and Decision Trees were also run on our dataset, but resulted into low accuracy as compared to other algorithms in our experiment; hence we decided to continue with SVM and Logistic Regression.

6. Results

We applied support vector machines to our dataset and predicted the results. The results were interpreted using confusion matrix and accuracy metric. The train:test split was 75:25. We also did cross-validation on the dataset to remove any biases. Value of split was chosen as 4 because the resultant splits will have same number of images as our 25% test set. The results are as follows:

Table 6: Accuracy for 75:25 split and cross-validation

SVM kernel	Accuracy (%)	Cross-Validation Accuracy Score (cv=4)
linear	78.05	0.78(+/- 0.07)
rbf	21.95	0.25(+/- 0.01)
poly	75.61	0.76(+/- 0.06)

In our experiment SVM with linear kernel performed better than other kernels. Rbf gave us the worst performance, whereas poly was as good as linear kernel. We tried to keep the test set % same for both split and cross-validated data so as to have uniformity in results. The mean cross-validation score was also approximately equal to the accuracy score achieved by the split. Figure 17 shows the heat-map of the confusion matrix from our multi-class classification results. On further analysis of the confusion matrix, we see that the diagonals have higher weights; there are a few misclassifications in every class except class 4: Fear.
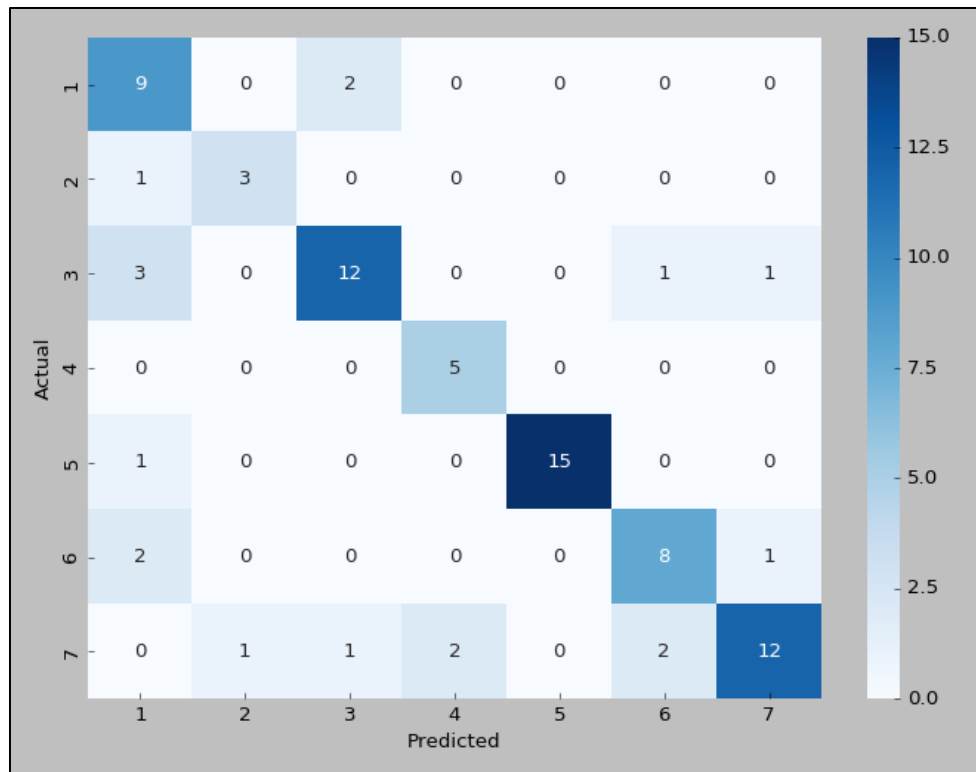
Figure 17: Heat map of actual and predicted values

Table 7: Report of predicted values Vs actual values

Predicted / Actual	Anger	Contempt	Disgust	Fear	Happy	Sadness	Surprise	All
Anger	9	0	2	0	0	0	0	11
Contempt	1	3	0	0	0	0	0	4
Disgust	3	0	12	0	0	1	1	17
Fear	0	0	0	5	0	0	0	5
Happy	1	0	0	0	15	0	0	16
Sadness	2	0	0	0	0	8	1	11
Surprise	0	1	1	2	0	2	12	18
All	16	4	15	7	15	11	14	82

31

Table 7 shows the predicted values and actual values in report format. From this, we can infer the correct number of emotions predicted for each class.

1: Anger – 9/11 – 82%

2: Contempt – 3/4 - 75%

3: Disgust – 12/17 - 70%

4: Fear – 5/5 - 100%

5: Happy – 15/16 - 93%

6: Sadness – 8/11 - 72%

7: Surprise – 12/18 – 66%

Many of the samples were misclassified into other classes such as contempt, disgust, fear and sadness. Fear was detected accurately for all the samples. The samples for disgust and sadness were misclassified as anger in some cases. This can be due to the similarity of features between these emotions.

For our next part, we considered all the 68 facial point features and determined which of these features actually help to determine the emotion. Jaw was not adding any difference in all of the micro-expressions. Hence, we decided to ignore the jaw facial landmark positions and continue with eyes, eyebrows, mouth and nose for our analysis. Our feature set was still the same with dimensions 327*68*2, but all the jaw facial landmarks had co-ordinates as 0. Also, we kept the split ratio of test-train dataset same as above. Following are the results for this experiment.

Table 8: SVM accuracy results without jaw features

SVM kernel	Accuracy (%)	Cross-Validation Accuracy Score (cv=4)
linear	82.93	0.80(+/- 0.04)
rbf	21.95	0.25(+/- 0.01)
poly	80.49	0.78(+/- 0.07)

Our focus is on the cross-validation score as it takes entire database into consideration for training, thus removing any biases. After removing jaw-line features the cross-validation results also increased from 78% to 80%. For this experiment, we used the cross-validation fold value as 4. Further we analyzed the predicted Vs actual results report for this scenario as in Table 9. From Table 9 we see that the detection rate for anger, disgust and surprise has increased, whereas contempt and fear has been misclassified. Detection rate of sadness has been same as the above experiment. A logical assumption will be that the sadness emotion of the subjects, closely resemble that of anger.

To improve the results, we did some fine tuning. We changed the data split to 70:30 and cross validation folds to 5. We use stratified sampling for the split. We also extracted more features from the available 68 facial point landmarks.

Table 9: Report of predicted values Vs actual values without jaw features

Predicted / Actual	Anger	Contempt	Disgust	Fear	Happy	Sadness	Surprise	All
Anger	**10**	1	0	0	0	0	0	**11**
Contempt	1	**2**	0	0	0	0	1	**4**
Disgust	2	0	**14**	0	0	0	1	**17**
Fear	0	0	1	**4**	0	0	0	**5**
Happy	1	1	0	0	**14**	0	0	**16**
Sadness	2	0	0	1	0	**8**	0	**11**
Surprise	0	0	1	0	0	1	**16**	**18**
All	**16**	**4**	**16**	**5**	**14**	**9**	**18**	**82**

Distances :

As we had the 68 co-ordinates we could easily calculate the distance between any two facial points on the face using the distance formula:

$$\sqrt{(x2 - x1)^2 + (y2 - y1)^2}$$

In Figure 18 we show some of the horizontal and vertical distances calculated for the face. In all, we calculated 25 such distances as shown in Table 10. Paper [1] use displacement ratios calculated using the facial landmark instead of directly using the distances.
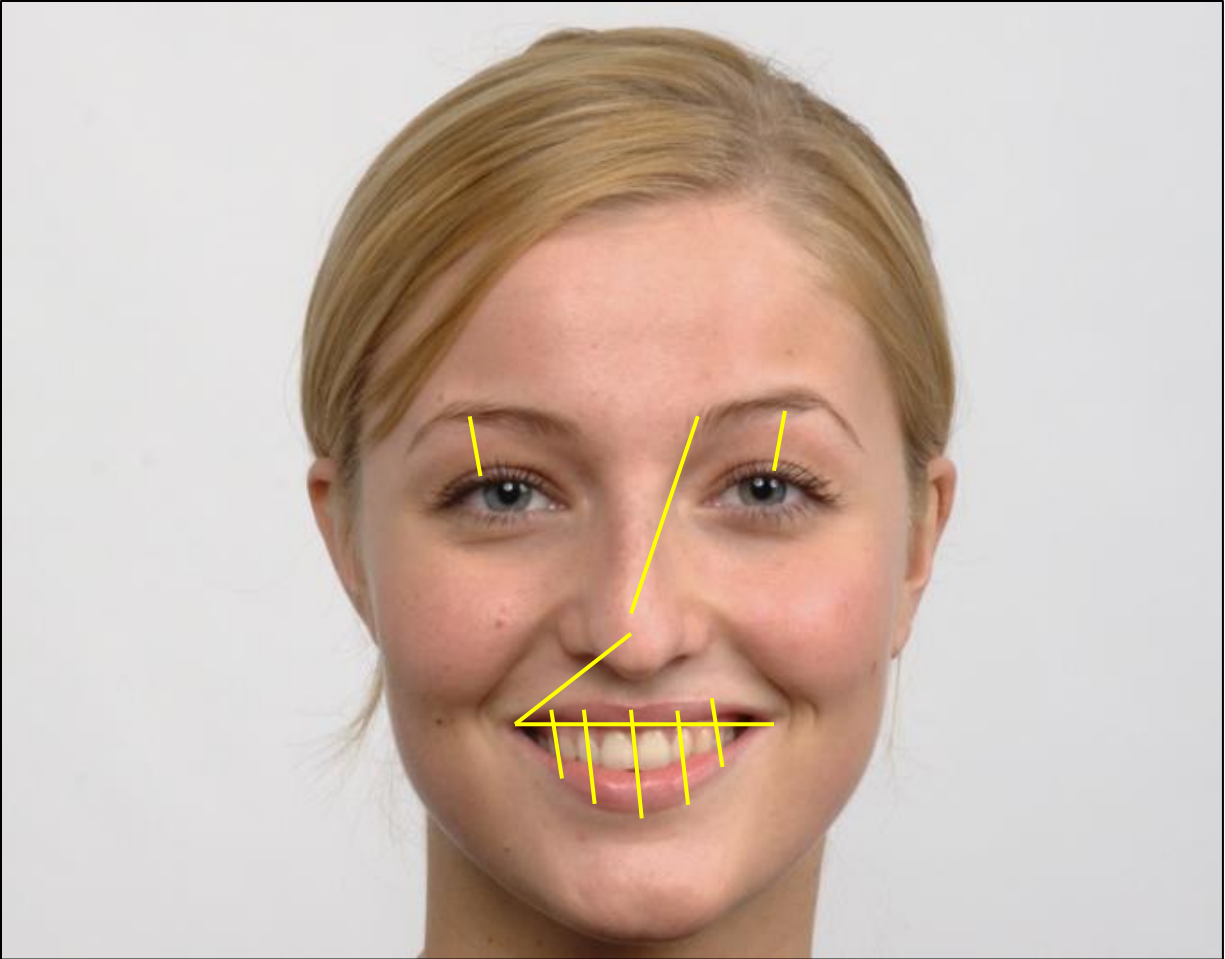
Figure 18: Distances calculated on the face

Areas:

We also calculated areas of the polygon surrounding the eye and the mouth region as shown in Figure 19. Paper [9] uses eccentricities of the ellipses instead of polygon areas. Area of polygon, whose co-ordinates are known is calculated using the formula:

$$|((x1y2 - y1x2) + (x2y3 - y2x3) + \cdots (xny1 - ynx1))/2|$$

We calculate 3 such areas one for each eye and the mouth.

Table 10: Distances calculated on the face

Distances	Facial points
D1,D2,D3, D4,D5, D6, D7 and D8	Distances between the eyebrows and the upper eye-lid
D9,D10,D11 and D12	Distances between the upper eye-lid and lower eye-lid
D13,D14,D15,D16 and D17	Distance between the outer upper lip and outer lower lip
D18 and D25	Distance between nose tip and outer upper lip and outer lower lip
D19 and D20	Distance between nose tip and left and right mouth corner
D21	Distance between the left and right mouth corner
D22	Distance between inner points of both the eyebrows
D23 and D24	Distance between nose tip and the inner points of both the eyebrows.

The prediction results using the 25 distance features and 3 areas (28 features) shows improvement in average accuracy by 2%. Also, along with SVM we use other two algorithms such as logistic regression and random forest classifier to evaluate our results.

Table 11 shows the result of experiment with 28 features consisting of distances and area. We have total 327 samples in a 70:30 split. Cross-validation folds are 5. Kernel for SVM is linear. For this experiment, Logistic Regression(l1 and l2 both) perform better along with Linear SVC(OVR). Logistic regression with penalty l1 gave the best result in terms of cross-validation score which has now improved to 82%..
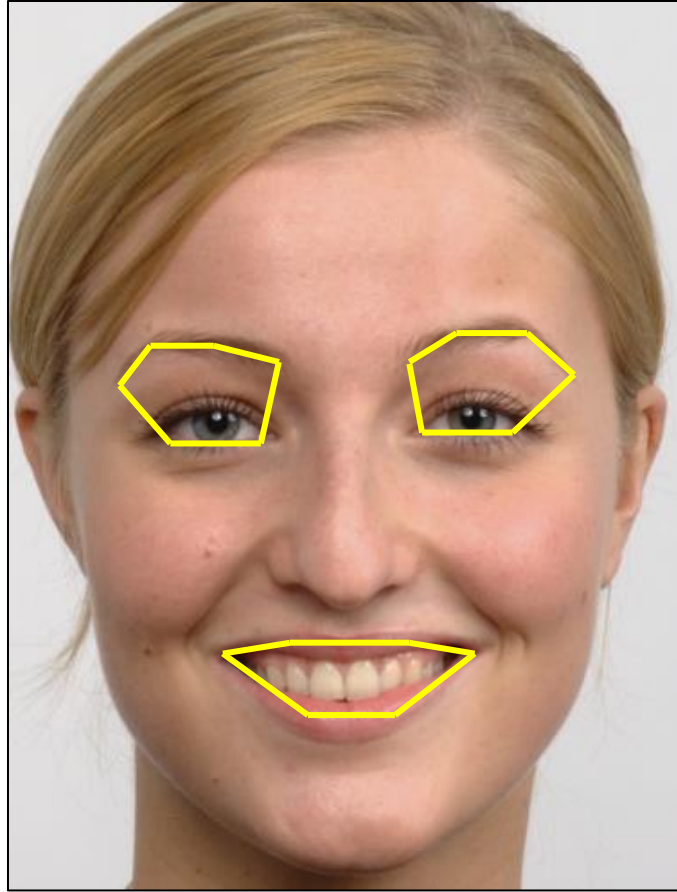
Figure 19: Polygon areas calculated on the face

We use the cross_val_predict method from sklearn to predict the outcomes of all 327 sample files. The cross_val_predict method concatenates the result for each cross-validation result. As in our case, each data-point will be in the test set only once, this method gives the outputs for data points when they were present in the test set. The prediction for such a data point is not over-lapping and not averaged.

Prediction report Table 12 gives us results about how well the logistic regression algorithm did in terms of individual classification of classes for a 70:30 split of the data:

Table 11: Prediction accuracy for different algorithms using 28 features

Algorithm with tuning parameter	Cross - Validation Score	Accuracy (70:30 split)	Accuracy (entire dataset using cross_val_predict)
Logistic Regression(OVR, penalty=l2)	0.80 (+/- 0.11)	78.78788	80.43
Logistic Regression(penalty=l1)	0.82 (+/- 0.07)	78.78788	81.35
Linear SVC(OVR)	0.81 (+/- 0.09)	79.79798	81.35
LinearSVM('crammer_singer')	0.77 (+/- 0.07)	78.78788	77.37
Logistic (multinomial, penalty=l2)	0.62 (+/- 0.13)	67.67677	62.08

Table 12: Actual Vs Predicted report for 28 features, 327 samples using cross_val_predict (Logistic Regression l1)

Predicted / Actual	Anger	Contempt	Disgust	Fear	Happy	Sadness	Surprise	All	Recall Score
Anger	**32**	2	5	1	0	5	0	**45**	71.11
Contempt	4	**10**	0	0	0	1	3	**18**	55.56
Disgust	2	1	**52**	0	1	2	1	**59**	88.14
Fear	0	0	1	**16**	3	1	4	**25**	64
Happy	0	0	1	1	**67**	0	0	**69**	97.10
Sadness	7	1	0	1	0	**14**	5	**28**	50
Surprise	1	2	0	2	0	3	**75**	**83**	90.36
All/Avg	**46**	**16**	**59**	**21**	**71**	**26**	**88**	**327**	81

In the next experiment, we combined all sets of features distances, areas and landmarks and checked the prediction accuracy of the combination of features as shown in Table 13. Following are the tuning parameters used for the entire set of samples and features: Cross-validation split: 5, Test-train split:30:70, Stratify: Yes, Random State = 42, Number of features: 164, Samples: 327

Table 13: Prediction accuracy for different algorithms using 164 features

Algorithm with tuning parameter	Cross - Validation Score	Accuracy	Accuracy (entire dataset using cross_val_predict)
Logistic Regression(OVR, penalty=l2)	0.87 (+/- 0.07)	82.83	86.54
Logistic Regression(penalty=l1)	0.85 (+/- 0.11)	80.81	85.63
Linear SVC(OVR)	0.77 (+/- 0.07)	74.75	76.76
LinearSVM('crammer_singer')	0.84 (+/- 0.07)	84.84	84.40
Logistic (multinomial, penalty=l2)	0.68 (+/- 0.09)	69.69	67.89

We see that the cross-valdiation score for LinearSVM('crammer_singer') and Logistic regression(l2 penalty) has improved significantly with the new feature set. To ratify our results, we used the Radboud Faces Database(RaFD) with 536 frontal images of around 67 models. The difference between CK+ database and RaFD databse is: RaFD has 8 emotions instead of 7. Neutral emotion image is added in the database. Also, the dataset is uniformly distributed, i.e. we have a standard dataset which contains all emotions in equal numbers (67). Table 15 shows the prediction accuracies using RaFD database.

Table 14: Actual Vs Predicted report for 164 features, 327 samples using cross_val_predict (Logistic Regression l2)

Predicted / Actual	Anger	Contempt	Disgust	Fear	Happy	Sadness	Surprise	All	Recall Score
Anger	**34**	1	5	0	0	5	0	**45**	75.56
Contempt	2	**14**	0	0	1	0	1	**18**	77.78
Disgust	1	0	**57**	0	0	1	0	**59**	96.6
Fear	0	0	1	**18**	2	2	2	**25**	72
Happy	0	0	0	0	**68**	1	0	**69**	98.55
Sadness	5	1	0	2	0	**15**	5	**28**	53.57
Surprise	1	2	0	1	0	2	**77**	**83**	92.77
All/Avg	**43**	**18**	**63**	**21**	**71**	**26**	**85**	**327**	87

Table 15: Prediction accuracy for different algorithms using 164 features for RaFD database

Algorithm with tuning parameter	Cross - Validation Score	Accuracy	Accuracy (entire dataset using cross_val_predict)
Logistic Regression(OVR, penalty=l2)	0.85 (+/- 0.07)	85.71	85.45
Logistic Regression(penalty=l1)	0.86 (+/- 0.07)	88.2	85.82
Linear SVC(OVR)	0.79 (+/- 0.09)	80.74	78.92
LinearSVM('crammer_singer')	0.77 (+/- 0.06)	83.23	76.68
Logistic (multinomial, penalty=l2)	0.61 (+/- 0.05)	61.49	61.19

As the database is uniformly distributed, we see further improvement in results. The highest accuracy Logistic regression achieves for a 70:30 split dataset is 88.2 %, whereas the average accuracy has also improved to 86%.

Table 16: Actual Vs Predicted report for 164 features, 536 samples using cross_val_predict for RaFD database (Logistic Regression l1)

Predicted / Actual	Anger	Contempt	Disgust	Fear	Happy	Neutral	Sadness	Surprise	All	Recall Score
Anger	**60**	2	3	0	0	0	2	0	**67**	90
Contempt	2	**56**	0	0	1	4	4	0	**67**	84
Disgust	1	0	**64**	1	0	0	0	1	**67**	96
Fear	0	1	0	**57**	0	3	5	1	**67**	85
Happy	0	1	0	0	**66**	0	0	0	**67**	99
Neutral	2	13	0	0	0	**42**	9	1	**67**	63
Sadness	4	1	0	2	0	4	**55**	1	**67**	82
Surprise	0	1	0	5	0	1	0	**60**	**67**	90
All/Avg	**69**	**75**	**67**	**65**	**67**	**54**	**75**	**64**	**536**	86

As logistic regression with penalty l1/l2 was performing higher as compared to other algorithms until now, we continued with logistic regression and cross-validation split = 10 for both the databases.

Using cross-validation split=10, CK+ database score improved, however RaFD database did not improve as seen from Table 19 and Table 20. Assumption is as RaFD database is

uniformly distributed the cross-validation splits did not improve the results much. However, for

CK+ database there was improvement in recognition of emotions such as anger, contempt and

sadness. The average recall score was 89% after the experiment as seen in Table 17 and Table

18.

Table 17: Scores for CK+ database with CV=10

Algorithm with tuning parameter	Cross - Validation Score	Accuracy	Accuracy (entire dataset using cross_val_predict)
Logistic Regression(OVR, penalty=l2)	0.89 (+/- 0.07)	82.83	88.99
Logistic Regression(penalty=l1)	0.86 (+/- 0.07)	88.2	85.82

Table 18: Prediction table for CK+ database with CV=10

Predicted / Actual	Anger	Contempt	Disgust	Fear	Happy	Sadness	Surprise	All	Recall Score
Anger	37	0	4	0	0	3	1	45	82
Contempt	1	16	0	0	0	1	0	18	89
Disgust	1	0	57	0	0	1	0	59	97
Fear	0	0	1	18	2	2	2	25	72
Happy	0	0	1	0	68	0	0	69	99
Sadness	4	1	0	1	0	18	4	28	64
Surprise	1	1	0	1	0	3	77	83	93
All/Avg	44	18	63	20	70	28	84	327	89

Table 19: Scores for RaFD database with CV=10

Algorithm with tuning parameter	Cross - Validation Score	Accuracy	Accuracy (entire dataset using cross_val_predict)
Logistic Regression(OVR, penalty=l2)	0.85 (+/- 0.09)	85.71	84.51
Logistic Regression(penalty=l1)	0.85 (+/- 0.09)	88.2	85.26

Table 20: Prediction table for RaFD database with CV=10

Predicted / Actual	Anger	Contempt	Disgust	Fear	Happy	Neutral	Sadness	Surprise	All	Recall Score
Anger	58	3	3	0	0	0	3	0	67	87
Contempt	2	55	0	0	0	5	5	0	67	82
Disgust	1	0	64	1	0	0	0	1	67	96
Fear	0	0	0	57	0	1	7	2	67	85
Happy	0	0	0	1	66	0	0	0	67	99
Neutral	1	14	0	0	0	41	10	1	67	61
Sadness	3	1	0	3	0	3	56	1	67	84
Surprise	0	1	0	4	0	2	0	60	67	90
All/Avg	65	74	67	66	66	52	81	65	536	85

For testing out our model, we trained using the CK+ dataset and test using some images

taken from the mobile phone. The logistic regression model was able to identify the basic happy

and surprise emotions. Although, with fearful, contemptuous, sad and disgust the algorithm predicted wrong classes as all the test subjects were not giving same expressions for each of the mentioned emotion. We had 44samples from mobile dataset. Accuracy of 36.36% was achieved by logistic regression (penalty l2). We used only 68 point facial landmarks for this experiment as features.

Figure 20: Sample mobile image : Happy face

Table 21 shows the result of the experiment. As there was no neutral expression in training dataset, it was not identified from the test set. Happy and Surprise were detected the most and most of the other emotions were misclassified as Happy and Disgust.

Table 21: Prediction table for Mobile Image Dataset

Predicted / Actual	Anger	Contempt	Disgust	Fear	Happy	Sadness	Surprise	Neutral	All
Anger	**2**	0	1	0	2	0	0	0	**5**
Contempt	1	**0**	1	0	3	0	0	0	**5**
Disgust	0	0	**2**	0	2	0	1	0	**5**
Fear	0	0	2	**0**	1	0	0	0	**3**
Happy	0	0	1	0	**7**	0	0	0	**8**
Sadness	2	0	2	0	0	**1**	1	0	**6**
Surprise	0	0	0	0	2	0	**4**	0	**6**
Neutral	2	1	2	0	1	0	0	**0**	**6**
All	**7**	**1**	**11**	**0**	**18**	**1**	**6**	**0**	**44**

7. Conclusion and Future Work

Our implementation can roughly be divided into 3 parts:

1) Face detection

2) Feature extraction

3) Classification using machine learning algorithms

Feature extraction was very important part of the experiment. The added distance and area features provided good accuracy for CK+ database (89%). But for cross-database experiment we observed that raw features worked best with Logistic Regression for testing RaFD database and Mobile images dataset. The accuracy was 66% and 36% for both using CK+ dataset as training set. The additional features (distance and area) reduced the accuracy of the experiment for SVM as seen in Table 13 and Table 15.

The algorithm generalized the results from the training set to the testing set better than SVM and other algorithms. The results of the emotion detection algorithm gave average accuracy up to 86% for RaFD database and 87% for CK+ database for cross-validation=5. RaFD dataset had equal number of classes; hence, cross-validation did not help in improving the accuracy of the model.

Table 22 shows our performance as compared to different papers. When compared to Paper[1], which used ORB feature descriptors our method performed better, using only the 68 facial landmark points and the distances and area features [14]. Paper [1] achieved an accuracy of 69.9% without the neutral emotion whereas we achieved average accuracy of 89%. Paper [14] had similar feature extraction technique as ours and their accuracy was slightly(0.78) better than us. Paper [20] used large number of iterations to train the layers and achieved an accuracy of

98.15% with 35000 iterations. Paper [21] also used similar concept of angles and areas and achieved an accuracy of 82.2% and 86.7% for k-NN and CRF respectively.

Table 22: Comparison with different papers

Algorithm and Features	Number of Emotions	Accuracy
CNN Network : FENet/ FeNet with BNLayer No. of Iterations: 5000->35000 [20]	7: Neutral considered instead of contempt	Without BNLayer: 85.25->87.35 / With BNLayer: 97.41 -> 98.15
k-nn and CRF(Angles and Areas) [21]	All 7	k-NN: 82.2/ CRF: 86.7
InceptionResNet with and without CRF(Softmax Layer) [22]	7	With CRF: 93.04/ Without CRF: 85.77
M-CRT [23]	7:Neutral considered instead of contempt	90.72
SVM (Multiclass +binary) (displacement ratios) [14]	7	89.78
SVM (ORB Features) [1]	7	69.9% / 79.1% (with neutral)

We did not focus on face detection in this paper. Our main focus was on feature extraction and analysis of the machine algorithm on the dataset. But accurate face-detection algorithm becomes very important if there are multiple people in the image. If we are determining the emotion of a particular person from a webcam, the webcam should be able to detect all the faces accurately.

For future work, a more robust face detection algorithm coupled with some good features can be researched to improve the results. We focused on only some distances and areas, there can be many more such interesting features on the face which can be statistically calculated and used for training the algorithm. Also, not all the features help to improve the accuracy, some maybe not helpful with the other features. Feature selection and reduction technique can be implemented on the created feature to improve the accuracy of the dataset. We can experiment with facial action coding system or feature descriptors as features or a combination of both of them. Also, we can experiment with different datasets amongst different races. This will give us an idea if the approach is similar for all kinds of faces or if some other features should be extracted to identify the emotion. Applications such as drowsiness detection amongst drivers [1] can be developed using feature selection and cascading different algorithms together.

 Algorithms like logistic regression, linear discriminant analysis and random forest classifier can be fine-tuned to achieve good accuracy and results. Also, metrics such as cross-validation score, recall and f1 score can be used to define the correctness of model and the model can be improved based on these metric results.

References

[1] W. Swinkels, L. Claesen, F. Xiao and H. Shen, "SVM point-based real-time emotion detection," *2017 IEEE Conference on Dependable and Secure Computing*, Taipei, 2017.

[2] Neerja and E. Walia, "Face Recognition Using Improved Fast PCA Algorithm," *2008 Congress on Image and Signal Processing*, Sanya, Hainan, 2008

[3] H. Ebine, Y. Shiga, M. Ikeda and O. Nakamura, "The recognition of facial expressions with automatic detection of the reference face," *2000 Canadian Conference on Electrical and Computer Engineering. Conference Proceedings. Navigating to a New Era (Cat. No.00TH8492)*, Halifax, NS, 2000, pp. 1091-1099 vol.2.

[4] A. C. Le Ngo, Y. H. Oh, R. C. W. Phan and J. See, "Eulerian emotion magnification for subtle expression recognition," 2016 IEEE International Conference on Acoustics, Speech and Signal Processing *(ICASSP)*, Shanghai, 2016

[5] V. Kazemi and J. Sullivan, "One millisecond face alignment with an ensemble of regression trees," *2014 IEEE Conference on Computer Vision and Pattern Recognition*, Columbus, OH, 2014

[6] M. Dahmane and J. Meunier, "Emotion recognition using dynamic grid-based HoG features," *Face and Gesture 2011*, Santa Barbara, CA, 2011

[7] K. M. Rajesh and M. Naveenkumar, "A robust method for face recognition and face emotion detection system using support vector machines," *2016 International Conference on Electrical, Electronics, Communication, Computer and Optimization Techniques (ICEECCOT)*, Mysuru, 2016

[8] C. Loconsole, C. R. Miranda, G. Augusto, A. Frisoli and V. Orvalho, "Real-time emotion recognition novel method for geometrical facial features extraction," *2014 International Conference on Computer Vision Theory and Applications (VISAPP)*, Lisbon, Portugal, 2014

[9] J. M. Saragih, S. Lucey and J. F. Cohn, "Real-time avatar animation from a single image," *Face and Gesture 2011*, Santa Barbara, CA, USA, 2011

[10]   G. T. Kaya, "A Hybrid Model for Classification of Remote Sensing Images With Linear SVM and Support Vector Selection and Adaptation," in *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 6, no. 4, pp. 1988-1997, Aug. 2013

[11]   X. Jiang, "A facial expression recognition model based on HMM," Proceedings of 2011 International Conference on Electronic & Mechanical Engineering and Information Technology, Harbin, Heilongjiang, China, 2011

[12]   J. J. Lee, M. Zia Uddin and T. S. Kim, "spatiotemporal human facial expression recognition using fisher independent component analysis and Hidden Markov Model," *2008 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, Vancouver, BC, 2008

[13]   Xiaoxu Zhou, Xiangsheng Huang, Bin Xu and Yangsheng Wang, "Real-time facial expression recognition based on boosted embedded hidden Markov model," *Image and Graphics (ICIG'04), Third International Conference on*, Hong Kong, China, 2004

[14]   T. Kundu and C. Saravanan, "Advancements and recent trends in emotion recognition using facial image analysis and machine learning models," *2017 International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT)*, Mysuru, 2017, pp. 1-6.

[15]   Y. Nijs, " Children's lying behavior towards personified robots: an experimental study,"

2016, 10.13140/RG.2.1.4612.5847. [Online] Available:

https://www.researchgate.net/publication/297379405_Children's_lying_behavior_towards_pe

rsonified_robots_an_experimental_study

[16]   Open CV Python Tutorials. [Online] Available: http://opencv-python-

tutroals.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_table_of_contents_feature2d/p

y_table_of_contents_feature2d.html

[17]   R. Raja, "Face detection using OpenCV and Python: A beginner's guide," 2017.[Online]

Available: https://www.superdatascience.com/opencv-face-detection/

[18]   A. Rosebrock, "Detect eyes, nose, lips and jaw with dlib, OpenCV, and Python," 2017.

[Online] Available: https://www.pyimagesearch.com/2017/04/10/detect-eyes-nose-lips-jaw-

dlib-opencv-python/

[19]   O. Langner et al, (2010). Presentation and validation of the Radboud Faces

Database. Cognition & Emotion, 24(8), 1377—1388. DOI: 10.1080/02699930903485076

[20]   X. Chen, X. Yang, M. Wang and J. Zou, "Convolution neural network for automatic

facial expression recognition," *2017 International Conference on Applied System Innovation

(ICASI)*, Sapporo, 2017, pp. 814-817. doi: 10.1109/ICASI.2017.7988558

[21]   D. Acevedo, P. Negri, M. E. Buemi, F. G. Fernández and M. Mejail, "A Simple

Geometric-Based Descriptor for Facial Expression Recognition," *2017 12th IEEE

International Conference on Automatic Face & Gesture Recognition (FG 2017)*,

Washington, DC, 2017, pp. 802-808. doi: 10.1109/FG.2017.101

[22]   B. Hasani and M. H. Mahoor, "Spatio-Temporal Facial Expression Recognition Using

Convolutional Neural Networks and Conditional Random Fields," *2017 12th IEEE*

International Conference on Automatic Face & Gesture Recognition (FG 2017), Washington, DC, 2017, pp. 790-795. doi: 10.1109/FG.2017.99

[23]   L. Du and H. Hu, "Modified classification and regression tree for facial expression recognition with using difference expression images," in *Electronics Letters*, vol. 53, no. 9, pp. 590-592, 4 27 2017. doi: 10.1049/el.2017.0731

[24]   P. Lucey, J. Cohn, T. Kanade, J. Saragih, Z. Ambadar and I. Matthews, The Extended Cohn-Kanade Dataset (CK+): A complete expression dataset for action unit and emotion-specified expression. Proceedings of the Third International Workshop on CVPR for Human Communicative Behavior Analysis (CVPR4HB 2010), San Francisco, USA, 94-101.

[25]   A. Mordvintsev and K. Abid, "Feature Detection and Description," 2013. [Online] Available:http://opencvpythontutroals.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_table_of_contents_feature2d/py_table_of_contents_feature2d.html