CHAPTER – 1

INTRODUCTION

1.1 Introduction

A facial recognition system is a technology used for identifying or verifying a person from a digital image or a video frame from a video source. There are multiple methods in which facial recognition systems work, but in general, they work by comparing selected facial features from given image with faces within a database. It is also described as a Biometric Artificial Intelligence based application, that can uniquely identify a person by analysing patterns based on the person's facial textures and shape.[5]

1.2 Motivation

The main motivation behind pursuing this project are as follows

- The proposed system can be effectively used in simplifying the process of Attendance by eliminating the need of any biometric devices like fingerprint or retina scanner with just a digital camera. This will reduce the costs considerably and simplify the process of attendance.
- This system can also be used for tracking a suspect/criminal or for surveillance of dangerous individuals and can help to improve security.
- It has wide range applications in security including security of our personal electronic devices like smartphones, laptops, PCs, etc.
- This technology can be further narrowed down to the recognition and tracking of eyes. This would save power by dimming a screen if viewer is not looking.

1.3 Methodology

- Face Detection has the objective of finding the faces (location and size) in an image and probably extract them to be used by the face recognition algorithm.
- The Images are pre-processed to increase the chances of image being detected by the algorithm. This includes a variant of Histogram equalisation and blurring.
- Each method has a different approach to extract the image information and perform the matching with the input image. In this project we make use of one of the oldest and more popular face recognition algorithms: Local Binary Patterns Histograms (LBPH) and also Haar cascade among which we continue forward with LBPH due to better accuracy obtained.
- The first computational step of the LBPH is to create an intermediate image that describes the original image in a better way, by highlighting the facial characteristics. To do so, the algorithm uses a concept of a sliding window, based on the parameters radius and neighbours.
- Now, using the image generated in the last step, we can use the Grid X and Grid Y parameters to divide the image into multiple grids and detect faces in the image.
- To evaluate how good detection algorithm worked on dataset, we evaluate "**Confusion Matrix"** which is calculating **True positive, True Nagative, False Positive and False Negatives.** Accuracy can later be calculated as:

$$Accuracy = \frac{(TP + TN)}{Total}$$

1.4 Dataset

1.4.1 Cambridge ORL Dataset

Here we've use the Database of Faces, (formerly 'The ORL Database of Faces'), which contains a set of face images taken between April 1992 and April 199. The database was used in the context of a face recognition project carried out in collaboration with the Speech, Vision and Robotics Group of the Cambridge University Engineering Department . There are ten different images of each of 40 distinct subjects. For some subjects, the images were taken at different times, varying

the lighting, facial expressions (open / closed eyes, smiling / not smiling) and facial details (glasses / no glasses). All the images were taken against a dark homogeneous background with the subjects in an upright, frontal position (with tolerance for some side movement) . The final testing will be done on a much larger database which will be composed of many smaller databases from other universities and one database of our own and will contain more than 1000 images. [7]

1.4.2 MNNIT Faces Dataset

This dataset contains the images of Final year students of MNNIT.It consists of more than 1000 photos of __ students. This dataset is used for practical implementation of face recognition . This dataset consists of background which is needed to be removed and multiple faces from each image is extracted to create a dataset free from background noise so that this dataset is designed similar to that of Cambridge ORL dataset.

1.5 Workflow

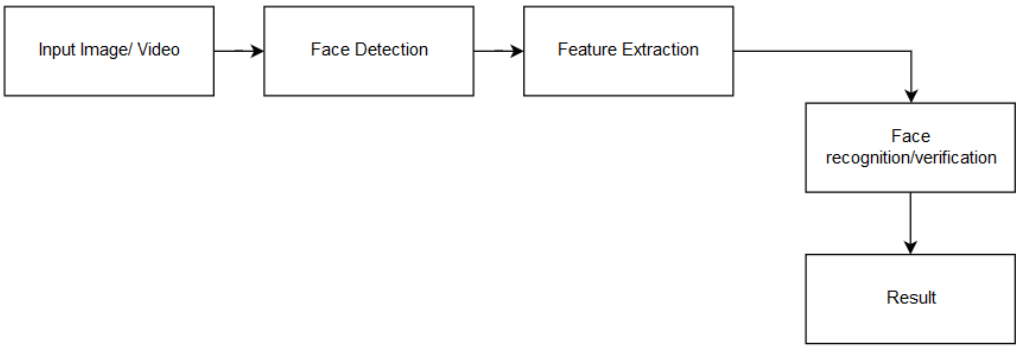

Fig 1.1: Workflow followed

1.6 Application

1.6.1 Biometrics

Biometrics is used in the process of authentication of a person by verifying or identifying that a user requesting a network resource is who he, she is, or it claims to be. It uses the property that a human trait associated with a person itself like structure of data with the incoming data we can verify the identity of a particular person . There

are many types of biometric system like detection and recognition, iris recognition etc., these traits are used for human identification in surveillance system, criminal identification, face details etc. By comparing the existing fingerprint recognition.[5]

1.6.2 Face Detection

Face detection is a computer technology that is being applied for many different applications that require the identification of human faces in digital images or video. It can be regarded as a specific case of object-class detection, where the task is to find the locations and sizes of all objects in an image that belong to a given class[5]. The technology is able to detect frontal or near-frontal faces in a photo, regardless of orientation, lighting conditions or skin color. Face Detection is the first and essential step for face recognition, and it is used to detect faces in the images. It is a part of object detection and can use in many areas such as security, bio-metrics, law enforcement, entertainment, personal safety, etc [5].

Face detection applications use algorithms that decides whether an image is a positive image (face image) or negative image (non-face image). This is called a classifier. To classify a new image correctly, it is trained on hundreds of thousands of face and non-face images. This feature answers the question Where are the faces in this picture?. For each face detected, you get a complete analysis of key points (landmarks) around the eyes, eye brows, jaw, nose and mouth. [6]

Face detection is an important part of face recognition as the first step of automatic face recognition. Because human faces are able to convey many different emotions such as happiness, sadness, interest, excitement, confusion, and intrigue, emotion classification is also being realised.

1.6.3 Face Recognition

The face is an important part of who you are and how people identify you. Except in the case of identical twins, the face is arguably a person's most unique physical characteristics. While humans have the innate ability to recognize and distinguish different faces for millions of years, computers are just now catching up. [5]

Human beings have recognition capabilities that are unparalleled in the modern computing era. These are mainly due to the high degree of interconnectivity, adaptive nature, learning skills and generalization capabilities of the nervous system. The

human brain has numerous highly interconnected biological neurons which, on some specific tasks, can outperform super computers. A child can accurately identify a face, but for a computer it is a cumbersome task. Therefore, the main idea is to engineer a system which can emulate what a child can do. Advancements in computing capability over the past few decades have enabled comparable recognition capabilities from such engineered systems quite successfully. Early face recognition algorithms used simple geometric models, but recently the recognition process has now matured into a science of sophisticated mathematical representations and matching processes. Major advancements and initiatives have propelled face recognition technology into the spotlight. [5]

Face recognition technology can be used in wide range of applications. Computers that detect and recognize faces could be applied to a wide variety of practical applications including criminal identification etc. Face detection and recognition is used in many places nowadays, verifying websites hosting images and social networking sites. Face recognition and detection can be achieved using technologies related to computer science. Features extracted from a face are processed and compared with similarly processed faces present in the database. If a face is recognized it is known or the system may show a similar face existing in database else it is unknown.

In surveillance system if a unknown face appears more than one time then it is stored in database for further recognition. These steps are very useful in criminal identification. In general, face recognition techniques can be divided into two groups based on the face representation they use appearance-based, which uses holistic texture features and is applied to either whole-face or specific face image and feature-based, which uses geometric facial features (mouth, eyebrows, cheeks etc.), and geometric relationships between them.

Chapter -2
PREPROCESSING

2.1 Introduction

It is the first step towards face recognition . The pre-processing is used to increase the chances of face being correctly detected and recognised. The pre-processing steps which we have used includes Histogram Equalization(more specifically Contrast Limited Adaptive Histogram Equalization also known as CLAHE) . The Histogram Equalized image generated has a lot of salt and pepper noise so median blur of filter size 3x3 is used to smoothen the image

2.2 Histogram Equalization

Histogram equalization is a technique for adjusting image intensities to enhance contrast.

Let ‘f’ be a given image represented as a ‘r’ by ‘c’ matrix of integer pixel intensities ranging from 0 to L − 1. L is the number of possible intensity values, often 256. Let p denote the normalized histogram of with a bin for each possible intensity. So

$$p_n = \frac{\text{number of pixels with intensity } n}{\text{total number of pixels}} \qquad n = 0, 1, ..., L-1.$$

Fig 2.1: Normalised Histogram for pixel intensities

The histogram equalized image ‘g’ will be defined by

$$g_{i,j} = \text{floor}((L-1)\sum_{n=0}^{f_{i,j}} p_n),$$

Fig 2.2: Formula for Histogram Equilisation

where floor() rounds down to the nearest integer.[8]

Adaptive histogram equalization (AHE) is a contrast enhancement technique which overcomes the limitations of standard histogram equalization. Unlike ordinary histogram equalization the adaptive method redistributes the lightness values of the image based on several histograms, each corresponding to a distinct section of the image. It is therefore useful for improving the local contrast and enhancing the definitions of edges in each region of an image. However, AHE has a tendency to overamplify noise in relatively homogeneous regions of an image. Contrast limited adaptive histogram equalization (CLAHE) prevents this by limiting the amplification.[8]

2.3 Blur Filters

There are various types of blurring filter available some of them are

- Averaging Filter
- Gaussian Filter
- Median Filter
- Poisson Filter

The Image generated after passing it through CLAHE has salt and pepper noise and Median Filter is best suited for removing this type of noise.

2.3.1 Median Filter

The Median Filter is a non-linear digital filtering technique, often used to remove noise from an image or signal. Such noise reduction is a typical pre-processing step to improve the results of later processing (for example, edge detection on an image). Median filtering is very widely used in digital image processing because, under certain conditions, it preserves edges while removing noise, also having applications in signal processing. The main idea of the median filter is to run through the signal entry by entry, replacing each entry with the median of neighbouring entries. The pattern of neighbours is called the "window", which slides, entry by entry, over the entire signal. For 1D signals, the most obvious window is just the first few preceding and following entries, whereas for 2D (or higher-dimensional) signals such as images, more complex window patterns are possible (such as "box" or "cross" patterns). Note that if the window has an odd number of entries, then the median is simple to define:

it is just the middle value after all the entries in the window are sorted numerically. For an even number of entries, there is more than one possible median.

2.3.2 Resizing

There are basically 2 types of resizing

- Up scaling
- Down scaling

The Images from Cambridge ORL dataset are of 112 x 92 pixel each [7] , after applying our face detection algorithm the size of the newly created images is varying depending on the size of face detected. To bring all the images to the same standard size(here 50 x 50 pixel) some of the images need up scaling and some need downscaling. Those whose size is less than standard size is up scaled and those images whose size is greater than the standard size is down scaled.

Chapter-3
Face Detection

3.1 Introduction

Before we can identify person in an image we need to identify locations where the faces are present, the major problem which we face during face detection is that the faces can be of variable size and in different orientations, the lighting conditions can be different and the faces can be found in different region of image. The face detection was successfully encountered by Paul Voila and Michael Jones.

3.2 Voila Jones Algorithm

The Voila Jones algorithm consists of 4 parts

1. Haar Features
2. Integral Image
3. Ada Boost
4. Cascading

3.2.1 Haar Features

The simple features used are reminiscent of Haar basis functions. More specifically, we use three kinds of features. The value of a two-rectangle feature is the difference between the sum of the pixels within two rectangular regions. The regions have the same size and shape and are horizontally or vertically adjacent .A three-rectangle feature computes the sum within two outside rectangles subtracted from the sum in a center rectangle. Finally a four-rectangle feature computes the difference between diagonal pairs of rectangles. Given that the base resolution of the detector is 24x24, the exhaustive set of rectangle features is quite large, over 180,000.[3]

3.2.2 Integral Image

Rectangle features can be computed very rapidly using an intermediate representation for the image which we call the integral image.The integral image at location(x,y) contains the sum of the pixels above and to the left of (x,y) , inclusive

$$ii(x,y) = \sum_{x' \leq x, y' \leq y} i(x', y'),$$

Fig 3.1: Formula for Integral Image

Using the integral image any rectangular sum can be computed in four array references .Clearly the difference between two rectangular sums can be computed in eight references. Since the two-rectangle features defined above involve adjacent rectangular sums they can be computed in six array references, eight in the case of the three-rectangle features, and nine for four-rectangle features.[3]

3.2.3 Ada Boost

Given a feature set and a training set of positive and negative images, any number of machine learning approaches could be used to learn a classification function. In our system a variant of AdaBoost is used *both* to select a small set of features *and* train the classifier [6]. In its original form, the AdaBoost learning algorithm is used to boost the classification performance of a simple (sometimes called weak)
learning algorithm there are over 180,000 rectangle features associated with each image sub-window, a number far larger than the number of pixels. Even though each feature can be computed very efficiently, computing the complete set is prohibitively expensive. Our hypothesis, which is borne out by experiment, is that a very small number of these features can be combined to form an effective classifier.[3]

3.2.4 Cascading

The adaboost algorithm discussed in the previous step reduces the number of important Haar features to be detected from 180,000 to only 6,000 features. But applying all the 6000 features all at a time will result into slow face detection. So, bunch of features are applied in a level, eliminating the possibility of false detection of Face in an image. So, final algorithm of Voila Jones contains 6000 Haar Feature evaluation over 38 stages of cascade layers.[3]

3.3 Haar Cascade

OpenCV comes with a trainer as well as detector. If you want to train your own classifier for any object like car, planes etc. you can use OpenCV to create one.

OpenCV already contains many pre-trained classifiers for face, eyes, smiles, etc. We have used pre-trained face classifier to detect faces in an image. The pre-trained classifier uses Voila Jones algorithm and Haar features in backend.

3.4 LBP Cascade

Local Binary Pattern (LBP) is a simple yet very efficient texture operator which labels the pixels of an image by thresholding the neighbourhood of each pixel and considers the result as a binary number.

It was first described in 1994 (LBP) and has since been found to be a powerful feature for texture classification. It has further been determined that when LBP is combined with histograms of oriented gradients (HOG) descriptor, it improves the detection performance considerably on some datasets.Using the LBP combined with histograms we can represent the face images with a simple data vector.As LBP is a visual descriptor it can also be used for face recognition tasks, as can be seen in the following step-by-step explanation.[10]

The LBP uses 4 parameters

- **Radius**: the radius is used to build the circular local binary pattern and represents the radius around the central pixel. It is usually set to 1.
- **Neighbors**: the number of sample points to build the circular local binary pattern. Keep in mind: the more sample points you include, the higher the computational cost. It is usually set to 8.
- **Grid X**: the number of cells in the horizontal direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. It is usually set to 8.
- **Grid Y**: the number of cells in the vertical direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. It is usually set to 8.

First, we need to train the algorithm. To do so, we need to use a dataset with the facial images of the people we want to recognize. We need to also set an ID (it may be a number or the name of the person) for each image, so the algorithm will use this information to recognize an input image and give you an output. Images of the same person must have the same ID. With the training set already constructed, let's see the LBPH computational steps.[10]

The first computational step of the LBPH is to create an intermediate image that describes the original image in a better way, by highlighting the facial characteristics. To do so, the algorithm uses a concept of a sliding window, based on the parameters radius and neighbours.

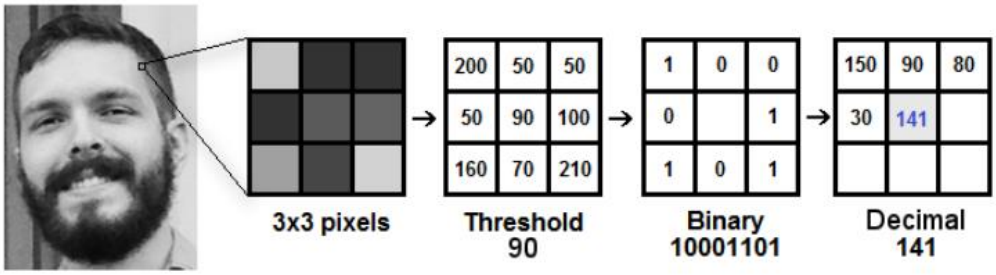

Fig 3.2: Applying LBP operation

Convert the facial Image to We can get part of this image as a window of 3x3 pixels.It can also be represented as a 3x3 matrix containing the intensity of each pixel (0~255).Then, we need to take the central value of the matrix to be used as the threshold. This value will be used to define the new values from the 8 neighbors. For each neighbour of the central value (threshold), we set a new binary value. We set 1 for values equal or higher than the threshold and 0 for values lower than the threshold.

Now, the matrix will contain only binary values (ignoring the central value). We need to concatenate each binary value from each position from the matrix line by line into a new binary value (e.g. 10001101). Note: some authors use other approaches to concatenate the binary values (e.g. clockwise direction), but the final result will be the same. Then, we convert this binary value to a decimal value and set it to the central value of the matrix, which is actually a pixel from the original image. At the end of this procedure (LBP procedure), we have a new image which represents better the

characteristics of the original image.The LBP procedure was expanded to use a different number of radius and neighbors, it is called Circular LBP.[10]

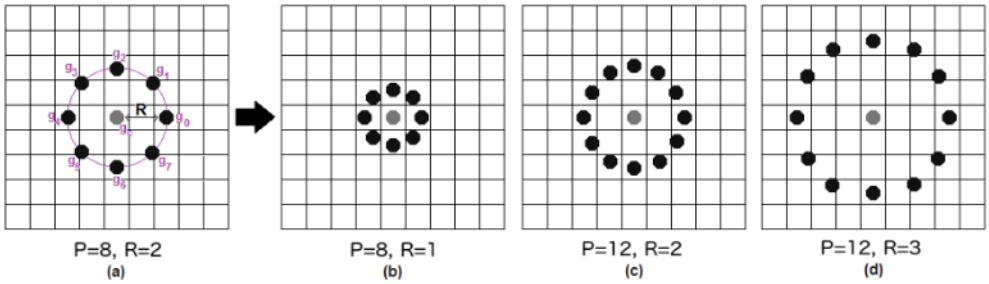

Fig 3.3: Circular LBP

It can be done by using bilinear interpolation. If some data point is between the pixels, it uses the values from the 4 nearest pixels (2x2) to estimate the value of the new data point. Now, using the image generated in the last step, we can use the Grid X and Grid Y parameters to divide the image into multiple grids, as can be seen in the following image

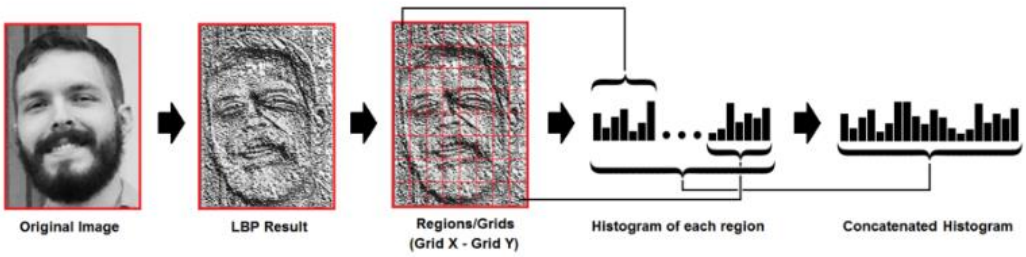

Fig 3.4: Dividing images into multiple grids

As now we have an image in grayscale, each histogram (from each grid) will contain only 256 positions (0~255) representing the occurrences of each pixel intensity.Then, we can concatenate each histogram to create a new and bigger histogram. Supposing we have 8x8 grids, we will have 8x8x256=16.384 positions in the final histogram. The final histogram represents the characteristics of the image original image.

In this step, the algorithm is now trained. Each histogram created is used to represent each image from the training dataset. So, given an input image, we perform the steps again for this new image and creates a histogram which represents the image. So to

find the image that matches the input image we just need to compare two histograms and return the image with the closest histogram.

We can use various approaches to compare the histograms (calculate the distance between two histograms), for example: euclidean distance, chi-square, absolute value, etc. In this example, we can use the Euclidean distance based on the following formula:

$$D = \sqrt{\sum_{i=1}^{n}(hist1_i - hist2_i)^2}$$

Fig 3.5: Euclidean distance formula

So the algorithm output is the ID from the image with the closest histogram. The algorithm should also return the calculated distance, which can be used as a 'confidence' measurement. Note: don't be fooled about the 'confidence' name, as lower confidences are better because it means the distance between the two histograms is closer. We can then use a threshold and the 'confidence' to automatically estimate if the algorithm has correctly recognized the image. We can assume that the algorithm has successfully recognized if the confidence is lower than the threshold defined.

3.5 Face detection Results

We used 3 algorithms for detecting faces from an image and they are Haar cascade, Haar cascade advanced and lbpcascade. Among these lbpcascade performed better than other algorithms. Some face detection results are shown below:

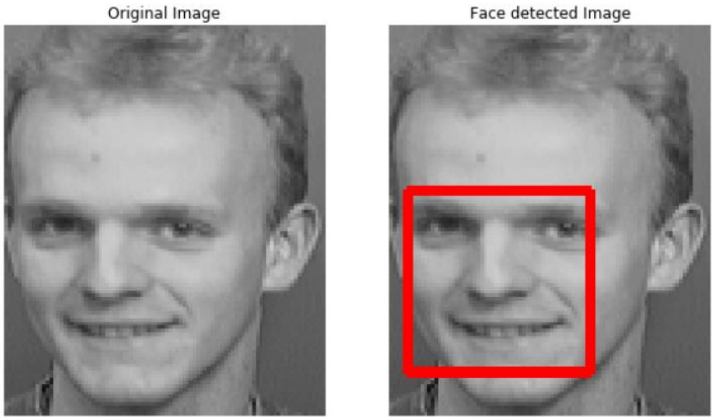

Fig 3.6: Face detection result number 1

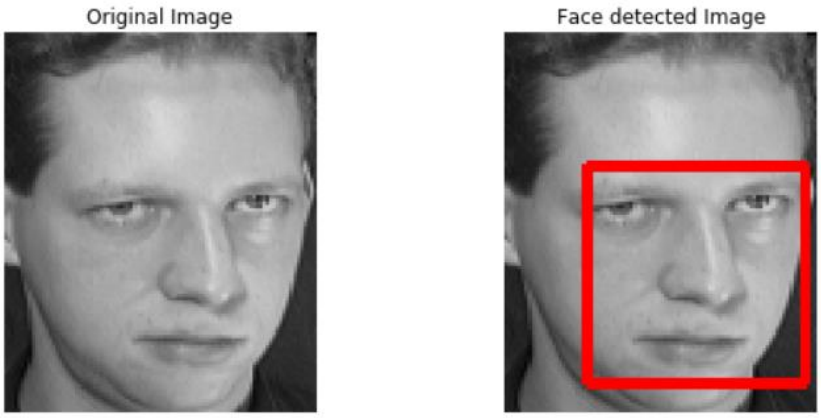

Fig 3.7: Face detection result number 2

3.6 Summary of Face detection algorithm used

Below table summarises the training models used and its performance matrix which includes evaluating Confusion matrix and accuracy from it. The face detection models were trained on Cambridge ORL dataset of total 390 images.

Table 3.1: Face detection algorithm summary

Algorithm	True Positive	False Positive	False Negative	Accuracy
Haar Cascade	362	0	28	92.82%
Haar Cascade Advanced	363	0	27	93.07%
Lbp Cascade	375	0	15	96.15%

Chapter 4

FEATURE EXTRACTION

4.1 Introduction

Humans can naturally identify a face by looking at certain obvious features like colour of faces, colour of hair, shape of Eyes, Nose, etc. But these features are not obvious for the computer which processes image as matrix of pixel intensity. Certain Feature extraction steps are taken, which includes:

- The segment of the image where faces were recognized by the algorithm is cropped form the image and resized to a fixed dimensions in our case 50x50 pixels.
- The resized image is converted to a image vector form by flattening it to 2500x1 dimension.
- This image vector is then passed through a trained Convolutional Neural Network (CNN) which output 128 dimensional feature vector for that image. Unlike other CNN models, this does not generate classification labels for inputs but instead it is trained to find 128 features from the face region which can best describe the image. The use of CNN for feature extraction is related to **Deep Metric Learning.**

4.2 Deep Metric Learning

Deep metric learning (DML) is an emerging field in metric learning by introducing deep neural network. Taking advantage of the nonlinear feature representation learning ability of deep learning and discrimination power of metric learning, DML is widely applied in various computer vision tasks. Existing DML algorithms can be broadly classified into two categories:

- Convolutional neural networks (CNNs) combined with metric loss
- Multilayer perceptron (MLP) (or Fully connected layers) combined with metric loss.

In the first kind of method, metric loss, i.e. pairwise loss (triplet loss) is employed to train a CNN with a structure of two (three) sub-networks. In these algorithms, the role of metric learning is only to optimize the deep neural networks for learning a good feature representation. The second kind of DML method is similar to the first category, but the difference is that instead of CNN, MLP is chosen to learn a set of hierarchical nonlinear transformations. Hand-crafted or pre-extracted deep features shall be input into this kind of method, which is also applicable to non-image data.

Chapter -5

LEARNING ALGORITHMS

5.1 Introduction

Once all the preprocessing is done various models can be built from these images using Artificial Intelligence algorithms from machine learning domain for recognition of a person from a given photo/video.

5.2 Machine Learning

Machine learning (ML) is a field of artificial intelligence that uses statistical techniques to give computer systems the ability to "learn" (e.g., progressively improve performance on a specific task) from data, without being explicitly programmed. Machine learning explores the study and construction of algorithms that can learn from and make predictions on data – such algorithms overcome following strictly static program instructions by making data-driven predictions or decisions through building a model from sample inputs. Machine learning is employed in a range of computing tasks where designing and programming explicit algorithms with good performance is difficult or infeasible; example applications include email filtering, detection of network intruders, and computer vision.

Machine learning algorithms are often categorized as supervised or unsupervised.

- **Supervised machine learning algorithms** can apply what has been learned in the past to new data using labelled examples to predict future events. Starting from the analysis of a known training dataset, the learning algorithm produces an inferred function to make predictions about the output values. The system is able to provide targets for any new input after sufficient training. The learning algorithm can also compare its output with the correct, intended output and find errors in order to modify the model accordingly.
- In contrast, **unsupervised machine learning algorithms** are used when the information used to train is neither classified nor labelled. Unsupervised learning studies how systems can infer a function to describe a hidden structure from unlabelled data. The system doesn’t figure out the right output,

but it explores the data and can draw inferences from datasets to describe hidden structures from unlabelled data.

5.3 Support Vector Machines

SVM is one of the most popular classifiers found in literature. In machine learning, support vector machines (SVMs, also support vector networks) are supervised learning models with associated learning algorithms that analyse data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier.

In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.

5.3.1 Motivation

Classifying data is a common task in machine learning. Suppose some given data points each belong to one of two classes, and the goal is to decide which class a new data point will be in. In the case of support vector machines, a data point is viewed as a p-dimensional vector (a list of p numbers), and we want to know whether we can separate such points with a (p−1)-dimensional hyperplane. This is called a linear classifier. There are many hyperplanes that might classify the data. One reasonable choice as the best hyperplane is the one that represents the largest separation, or margin, between the two classes. So we choose the hyperplane so that the distance from it to the nearest data point on each side is maximized. If such a hyperplane exists, it is known as the maximum-margin hyperplane and the linear classifier it defines is known as a maximum margin classifier; or equivalently, the perceptron of optimal stability.

5.3.2 Result

The model trains on series of feature vector generated for all iamges. The model took around 35 minutes to train and achieved an accuracy of 97.43%

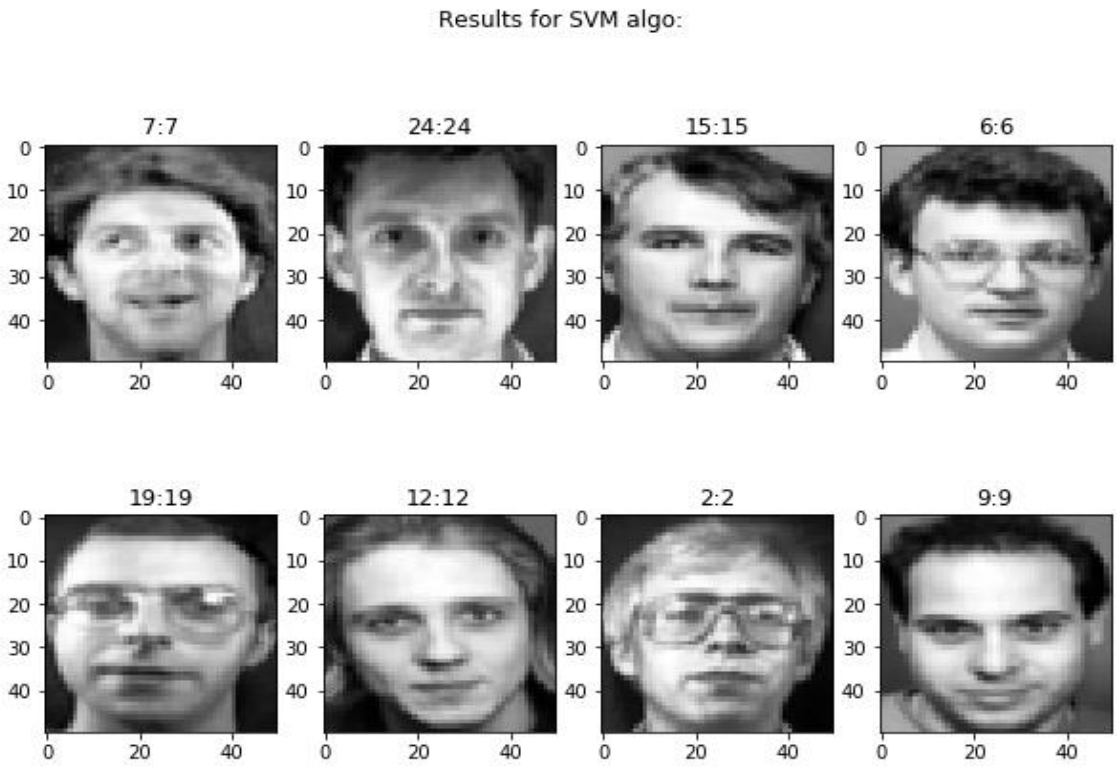

Fig 5.1: SVM Result

5.4 Random Forest

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks, that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of overfitting to their training set.

Decision trees are a popular method for various machine learning tasks. Tree learning "comes closest to meeting the requirements for serving as an off-the-shelf procedure for data mining because it is invariant under scaling and various other transformations of feature values, is robust to inclusion of irrelevant features, and produces inspectable models. However, they are seldom accurate".

In particular, trees that are grown very deep tend to learn highly irregular patterns: they overfit their training sets, i.e. have low bias, but very high variance. Random forests are a way of averaging multiple deep decision trees, trained on different parts of the same training set, with the goal of reducing the variance. This

comes at the expense of a small increase in the bias and some loss of interpretability, but generally greatly boosts the performance in the final model.

5.4.1 Result

The model took around 30 minutes to train and gave an accuracy of 91.02%.

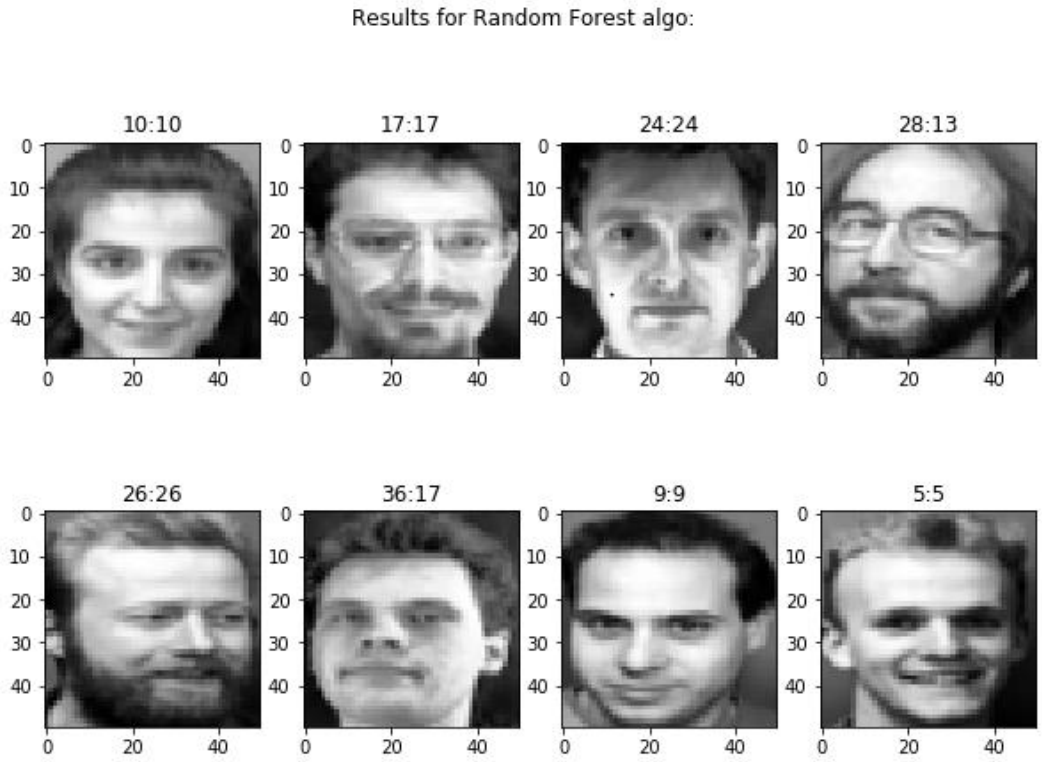

Fig 5.2: Random Forest Result

5.5 Decision Trees

A decision tree is a decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements. Decision trees are commonly used in operations research, specifically in decision analysis, to help identify a strategy most likely to reach a goal, but are also a popular tool in machine learning.

A decision tree is a flowchart-like structure in which each internal node represents a "test" on an attribute (e.g. whether a coin flip comes up heads or tails), each branch represents the outcome of the test, and each leaf node represents a class label (decision taken after computing all attributes). The paths from root to leaf represent classification rules. In decision analysis, a decision tree and the closely

related influence diagram are used as a visual and analytical decision support tool, where the expected values (or expected utility) of competing alternatives are calculated.

A decision tree consists of three types of nodes:

- Decision nodes – typically represented by squares
- Chance nodes – typically represented by circles
- End nodes – typically represented by triangles

Decision trees are commonly used in operations research and operations management. If, in practice, decisions have to be taken online with no recall under incomplete knowledge, a decision tree should be paralleled by a probability model as a best choice model or online selection model algorithm. Another use of decision trees is as a descriptive means for calculating conditional probabilities.

5.5.1 Result

The model took only 1 minute to train and achieved an accuracy of 56.41%.

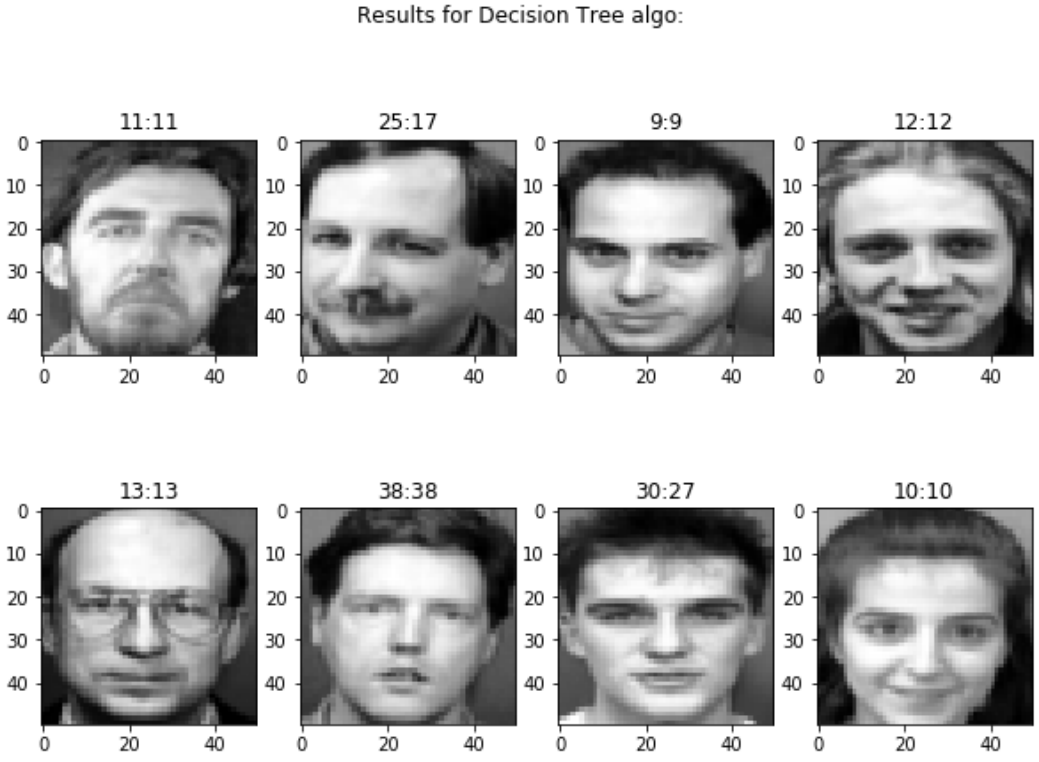

Fig 5.3: Decision Tree Result

5.6 Logistic Regression

Logistic Regression is the appropriate regression analysis to conduct when the dependent variable is dichotomous (binary). Like all regression analyses, the logistic regression is a predictive analysis. Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables. It is used to estimate discrete values (Binary values like 0/1, yes/no, true/false) based on given set of independent variables. In simple words, it predicts the probability of occurrence of an event by fitting data to a logit function. Hence, it is also known as logit regression. Since, it predicts the probability, its output values lies between 0 and 1 (as expected).

The binary logistic regression model has extensions to more than two levels of the dependent variable: categorical outputs with more than two values are modelled by multinomial logistic regression, and if the multiple categories are ordered, by ordinal logistic regression, for example the proportional odds ordinal logistic model. The model itself simply models probability of output in terms of input, and does not perform statistical classification (it is not a classifier), though it can be used to make a classifier, for instance by choosing a cut-off value and classifying inputs with probability greater than the cutoff as one class, below the cutoff as the other, this is a common way to make a binary classifier. The coefficients are generally not computed by a closed-form expression, unlike linear least squares. The logistic regression as a general statistical model was originally developed and popularized primarily by Joseph Berkson, beginning in Berkson (1944), where he coined “logit”.

Logistic regression can be binomial, ordinal or multinomial. Binomial or binary logistic regression deals with situations in which the observed outcome for a dependent variable can have only two possible types, "0" and "1" (which may represent, for example, "dead" vs. "alive" or "win" vs. "loss"). Multinomial logistic regression deals with situations where the outcome can have three or more possible types (e.g., "disease A" vs. "disease B" vs. "disease C") that are not ordered. Ordinal logistic regression deals with dependent variables that are ordered.

5.6.1 Result

The model took around 25 minutes to train and gave an accuracy of 97.43 %

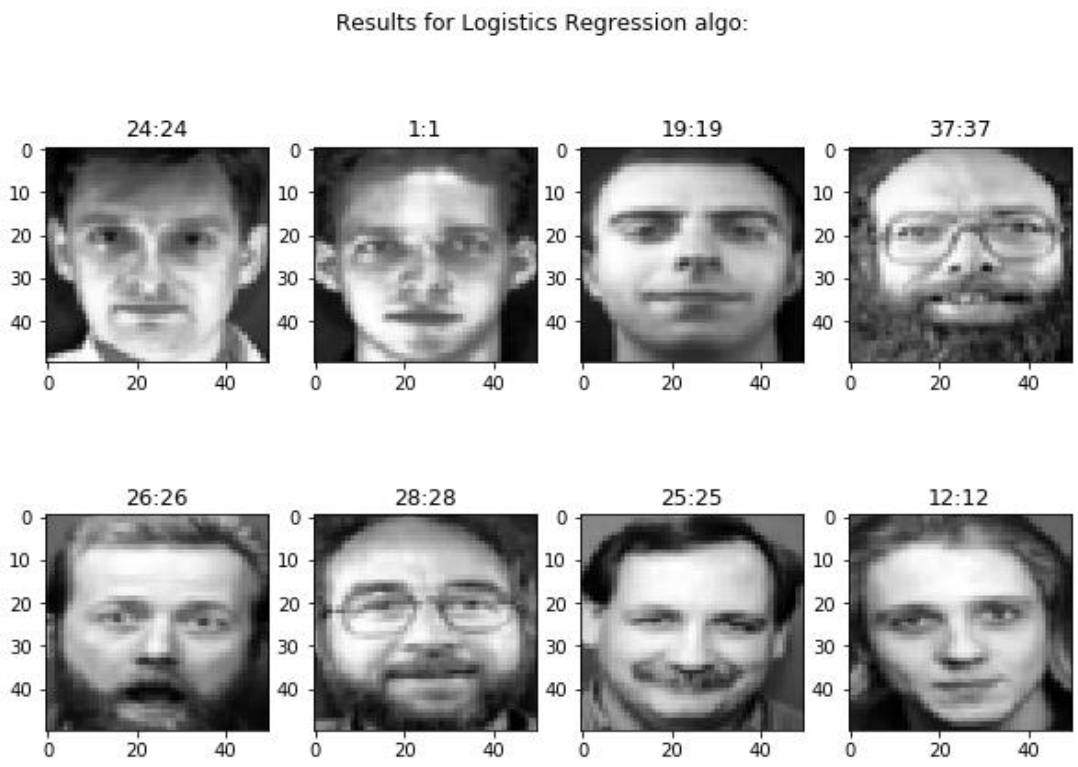

Fig 5.4: Logistic Regression Result

5.7 Naïve Bayes:

Naïve Bayesian Classifier (NB Classifier) is an advanced classification technique that is a self-learning or machine learning technique that uses Bayes Theorem of conditional probability to predict the class of an observation. NB Classifier uses Bayes theorem of conditional probability assuming the predictor variables are independent. It also assumes the continuous variables to follow a normal distribution.

It is a simple algorithm and works very well if the predictor variables are independent to each other. In the absence of large data, the algorithm some time work more efficiently than other complex classification algorithms.

This technique is more useful when it is more important to know the most likely class rather than knowing the actual probabilities for various classes. One example of successful application of this algorithm is in the applications of classifying a mail as spam or not.

5.7.1 Result

The model took around 20 minutes to train and gave an accuracy of about 88.46%

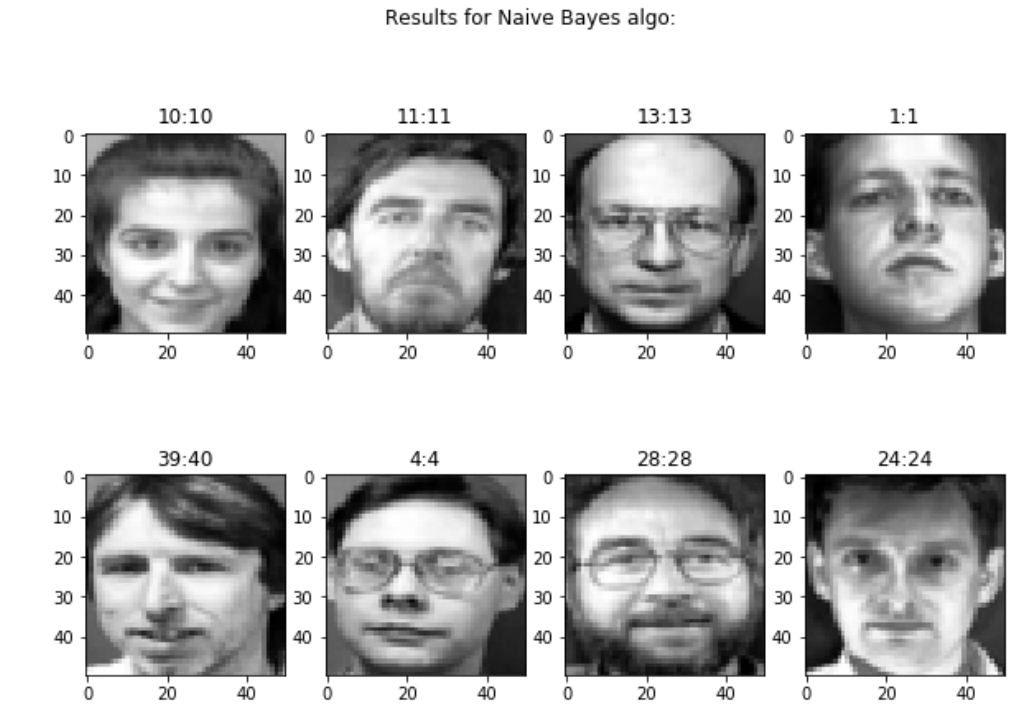

Fig 5.5: Naïve Bayes Result

5.8 Summary of Trained models:

Following Table compiles the accuracies obtained from each of the five models we trained.

Table 5.1: Training Models used and Accuracy

Training Model	**Accuracy**
Naïve Bayes	88.46%
Logistic Regression	97.43%
Decision Tree	56.41%
Random Forest	91.02%
Support Vector Machine	97.43%

CHAPTER – 6

RESULTS & CONCLUSION

6.1 RESULTS

After training our model and getting good accuracy for the Cambridge ORL Dataset we trained our algorithm on MNNIT Faces Database. The Algorithm worked pretty well for the images taken from various events with multiple faces in it. The training images were taken from Informal Photoshoot and Rukhsat while the testing was done on the images taken during the Formal Photoshoot of the batch. While testing we found that the accuracy was about 95%.

Some of the results are shown in the corresponding images.

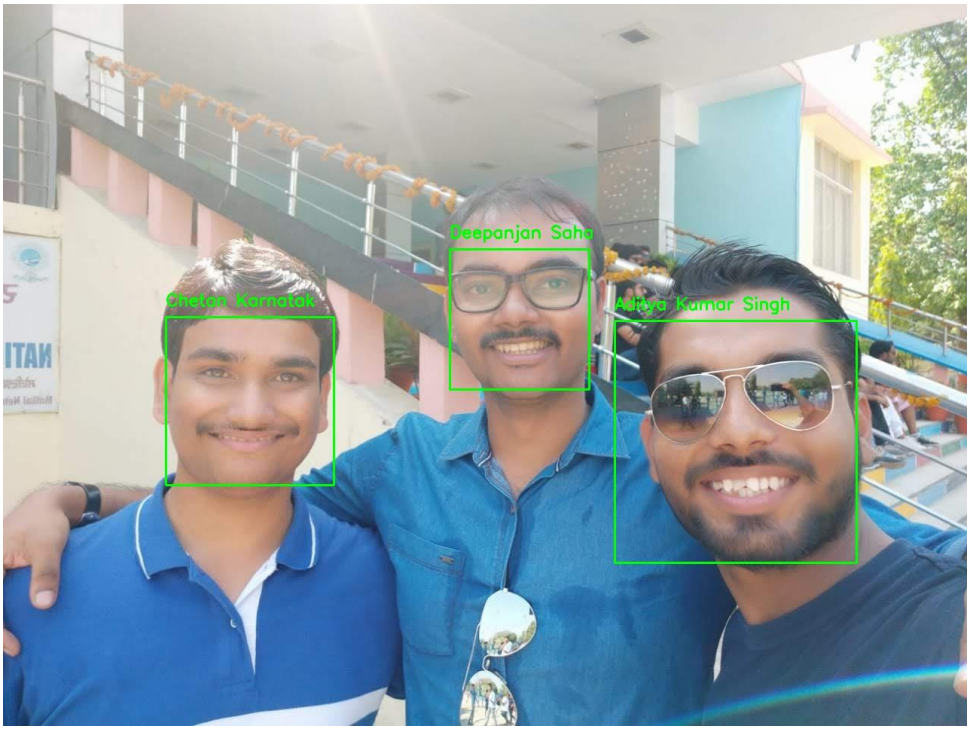

Fig 6.1: Face Recognition result on a random photo

Fig 6.2: Face Recognition result on a random photo

Fig 6.3: Face Recognition result on a random photo

6.2 Conclusion

From the above results we draw the conclusions that the trained algorithm is very good in identifying people in different images and does so with about 95% accuracy. The algorithm misidentifies some of the faces due to unavailability of those faces in the database and the similarities in facial features of some people. The above algorithm can be further improved by using more advanced machine learning algorithm to about 99%. The accuracy can be further improved by providing the learning algorithm with much larger training data.

6.3 Future Scope

With ongoing advances in the fields of machine learning this algorithm can be further improved and the accuracy can be increased further. As more data becomes available the algorithm can get better and the accuracy be improved. For the ease of use for general people we can design a GUI based system for the proposed algorithm and making it usable for the people. The said algorithm requires a lot of processing and has a lot of hardware requirements (GPU) for the processing which then can enable us to implement face identification for live video feed. We can also develop an App for the same algorithm so as to increase the reach and make it much more usable.

References

[1] Zhen, Chenggang, and Yingmei Su. "Research about human face recognition technology." Test and Measurement, 2009. ICTM'09. International Conference on. Vol. 1. IEEE, 2009.

[2] Guo, Guodong, Stan Z. Li, and Kapluk Chan. "Face recognition by support vector machines." Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference on. IEEE, 2000.

[3] Viola, Paul, and Michael Jones. "Rapid object detection using a boosted cascade of simple features." Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on. Vol. 1. IEEE, 2001.

[4] Amaro, E. Garcia, Marco Aurelio Nuño-Maganda, and Miguel Morales-Sandoval. "Evaluation of machine learning techniques for face detection and recognition." CONIELECOMP 2012, 22nd International Conference on Electrical Communications and Computers. IEEE, 2012.

[5] Wikipedia contributors. (Saturday, April 20, 2019, 19:31:53). **"Facial recognition system".** In Wikipedia, The Free Encyclopedia. Retrieved 12:00, April 23, 2019, from (https://en.wikipedia.org/wiki/Facial_recognition_system)

[6] Wikipedia contributors. (Saturday, April 20, 2019, 19:32:27). "**Face detection**". In Wikipedia, The Free Encyclopedia. Retrieved 1:30, April 23, 2019, from (https://en.wikipedia.org/wiki/Face_detection)

[7] Cambridge University Computer Laboratory. (Friday, July 21, 2006, 14:01:58). "**The Database of faces"**. In archive of AT&T Laboratories cambridge. Retrieved 12:40, January 18, 2019, from (https://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html)

[8] UCI Math. (Thursday, Sepember 2, 2010, 1:01:58). "**Histogram Equilisation-UCI Math"**. In UCI labs. Retrieved 1:00, April 23, 2019, from (https://www.math.uci.edu/icamp/courses/math77c/demos/hist_eq.pdf)

[9] Wikipedia contributors. (Saturday, April 20, 2019, 19:29:11). "**Median Filter**". In Wikipedia, The Free Encyclopedia. Retrieved 2:00, April 23, 2019, from (https://en.wikipedia.org/wiki/Median_filter)

[10] Towards Data science contributors. (Sunday, April 21, 2019, 19:29:11). "**Face Recognition: Understanding LBPH Algorithm**". In towardsdatascience.com. Retrieved 2:00, April 23, 2019, from (https://towardsdatascience.com/face-recognition-how-lbph-works-90ec258c3d6b)