Chapter 1

Introduction

The use of video surveillance system is becoming more and more important for investigation and deterrent of crimes, and cameras installed in public space are increasing due to an increasing number of crimes and suicide bombings. Therefore, identifying a person on a video tape becomes more and more frequent. However, this can be a formidable task, especially when there are a lot of people in a crowd and the law enforcing agencies have to catch few suspects on the spot.

The problems with facial recognition are rooted in the need for greater processing power. Security agencies have hours of footage available, but they usually don't have resources to handle such large amount of data and to extract information out of it. By breaking CCTV footage into frames, applications take hours and even days to process such vast amount of data. On the other hand, the process of matching a snapshot takes minutes to hours. For this, there is a much need of distributed computing architecture for applications to achieve performance and efficiency in processing images that are very large in size in a short interval of time.

1.1 Problem Statement:

Security agencies have deployed hundreds of surveillance cameras all over the country to detect and recognize suspects, but there is no such system which can shows results in real time. Moreover, identifying a person from a video is even more difficult especially when suspects have to be caught in a crowded area and in minimal amount of time as the real time matching of an image is still a challenge. Other than this, the matching of human faces with the ones present in the database is another great challenge. Some applications require to process images that are of very large size and require that all the processing can be done in a very short interval of time. The throughput of arrival of the images may undergo great and unpredictable variations according to the time and the duration of the image processing and can also undergo great and unpredictable variations according to the image type and size. As the human face evolves over time, the matching of face extracted from the stream with the ones present in the database is another difficult task.

1.2 Proposed Solution:

To combat the problems mentioned above, we are proposing a system that would process all the data on a Cloud powered backend server through which the face detection and recognition can be done in milli to micro seconds. Using the power of cloud, the system can easily achieve the real time efficiency which is the major problem. Using cloud will also help scaling and remotely accessing the system at any desired moment. Meanwhile, to increase the accuracy of facial recognition we are proposing to use Adaboost algorithm and Eigen Faces technique for reasonably accurate facial detection and recognition. Adaboost is the most accurate facial detection algorithm available currently and is also free-of-cost while Eigen faces, which is used for face recognition, is the most commonly used technique currently being used. An estimated, 91% detection and 72% recognition accuracy can be achieved by using these two techniques.

Chapter 2

Related Work and Literature Survey

In this section we will explain the various technologies related to our domain and some related projects done in past.

2.1 Introduction to Cloud computing

Cloud computing is a category of computing solutions in which a technology and/or service lets users access computing resources on demand, as needed, whether the resources are physical or virtual, dedicated, or shared, and no matter how they are accessed (via a direct connection, LAN, WAN, or the Internet). The cloud is often characterized by self-service interfaces that let customers acquire resources when needed as long as needed. Cloud is also the concept behind an approach to building IT services that takes advantage of the growing power of servers and virtualization technologies [1].

2.1.1 How Cloud Computing Works

To understand how does cloud computing work, imagine that the cloud consists of layers, mostly the back-end layers and the front-end or user-end layers. The front-end layers are the ones you see and interact with. The back-end consists of the hardware and the software architecture that fuels the interface you see on the front end. Because the computers are set up to work together, the applications can take advantage of all that computing power as if they were running on one particular machine. Cloud computing also allows for a lot of flexibility. Depending on the demand, you can increase how much of the cloud resources you use without the need for assigning specific hardware for the job, or just reduce the amount of resources assigned to you when they are not necessary. Let's say you're an executive at a large corporation. Your particular responsibilities include making sure that all of your employees have the right hardware and software they need to do their jobs. Buying computers for everyone isn't enough -- you also have to purchase software or software licenses to give employees the tools they require. Instead of installing a suite of software for each computer, you'd only have to load one application. That application would allow workers to log into a Web-based service which hosts all the programs the user would need for his or her job. Remote machines owned by another company would run everything from e-mail to word processing to complex data analysis programs. It's called cloud computing, and it could change the entire computer industry. In a cloud computing system, there's a significant workload shift. Local computers no longer have to do all the heavy lifting when it comes to running applications. The network of computers that make up the cloud handles them instead. Hardware and software demands on the user's side decrease. The only thing the user's computer needs to be able to run is the cloud computing system interface software, which can be as simple as a Web browser, and the cloud's network takes care of the rest.

There's a good chance you've already used some form of cloud computing. If you have an e-mail account with a Web-based e-mail service like Hotmail, Yahoo! Mail or Gmail, then you've had some experience with cloud computing. Instead of running an e-mail program on your computer, you log in to a Web e-mail account remotely. The software and storage for your account doesn't exist on your computer -- it's on the service's computer cloud.

2.1.2 Cloud Computing Architecture

When talking about a cloud computing system, it's helpful to divide it into two sections: the front end and the back end. They connect to each other through a network, usually the Internet. The front end is the side the computer user, or client, sees. The back end is the "cloud" section of the system [1].

The front end includes the client's computer (or computer network) and the application required to access the cloud computing system. Not all cloud computing systems have the same user interface. Services like Web-based e-mail programs leverage existing Web browsers like Internet Explorer or Firefox. Other systems have unique applications that provide network access to clients. On the back end of the system are the various computers, servers and data storage systems that create the "cloud" of computing services. In theory, a cloud computing system could include practically any computer program you can imagine, from data processing to video games. Usually, each application will have its own dedicated server.

A central server administers the system, monitoring traffic and client demands to ensure everything runs smoothly. It follows a set of rules called protocols and uses a special kind of software called middleware. Middleware allows networked computers to communicate with each other. Most of the time, servers don't run at full capacity. That means there's unused processing power going to waste. It's possible to fool a physical server into thinking it's actually multiple servers, each running with its own independent operating system. The technique is called server virtualization. By maximizing the output of individual servers, server virtualization reduces the need for more physical machines. If a cloud computing company has a lot of clients, there's likely to be a high demand for a lot of storage space. Some companies require hundreds of digital storage devices. Cloud computing systems need at least twice the number of storage devices it requires to keep all its clients' information stored. That's because these devices, like all computers, occasionally break down. A cloud computing system must make a copy of all its clients' information and store it on other devices. The copies enable the central server to access backup machines to retrieve data that otherwise would be unreachable. Making copies of data as a backup is called redundancy.

2.1.3 Cloud Computing Applications

The applications of cloud computing are practically limitless. With the right middleware, a cloud computing system could execute all the programs a normal computer could run. Potentially, everything from generic word processing software to customized computer programs designed for a specific company could work on a cloud computing system.

Why would anyone want to rely on another computer system to run programs and store data? Here are just a few reasons:

- Clients would be able to access their applications and data from anywhere at any time. They could access the cloud computing system using any computer linked to the Internet. Data wouldn't be confined to a hard drive on one user's computer or even a corporation's internal network.
- It could bring hardware costs down. Cloud computing systems would reduce the need for advanced hardware on the client side. You wouldn't need to buy the fastest computer with

the most memory, because the cloud system would take care of those needs for you. Instead, you could buy an inexpensive computer terminal. The terminal could include a monitor, input devices like a keyboard and mouse and just enough processing power to run the middleware necessary to connect to the cloud system. You wouldn't need a large hard drive because you'd store all your information on a remote computer.

- Corporations that rely on computers have to make sure they have the right software in place to achieve goals. Cloud computing systems give these organizations company-wide access to computer applications. The companies don't have to buy a set of software or software licenses for every employee. Instead, the company could pay a metered fee to a cloud computing company.
- Servers and digital storage devices take up space. Some companies rent physical space to store servers and databases because they don't have it available on site. Cloud computing gives these companies the option of storing data on someone else's hardware, removing the need for physical space on the front end.
- Corporations might save money on IT support. Streamlined hardware would, in theory, have fewer problems than a network of heterogeneous machines and operating systems.
- If the cloud computing system's back end is a grid computing system, then the client could take advantage of the entire network's processing power. Often, scientists and researchers work with calculations so complex that it would take years for individual computers to complete them. On a grid computing system, the client could send the calculation to the cloud for processing. The cloud system would tap into the processing power of all available computers on the back end, significantly speeding up the calculation.

2.1.4 Cloud Computing Issues

Perhaps the biggest concerns about cloud computing are security and privacy. The idea of handing over important data to another company worries some people. Corporate executives might hesitate to take advantage of a cloud computing system because they can't keep their company's information under lock and key.

The counterargument to this position is that the companies offering cloud computing services live and die by their reputations. It benefits these companies to have reliable security measures in place. Otherwise, the service would lose all its clients. It's in their interest to employ the most advanced techniques to protect their clients' data. Privacy is another matter. If a client can log in from any location to access data and applications, it's possible the client's privacy could be compromised. Cloud computing companies will need to find ways to protect client privacy. One way is to use authentication techniques such as user names and passwords. Another is to employ an authorization format - each user can access only the data and applications relevant to his or her job.

Some questions regarding cloud computing are more philosophical. Does the user or company subscribing to the cloud computing service own the data? Does the cloud computing system, which provides the actual storage space, own it? Is it possible for a cloud computing company to deny a client access to that client's data? Several companies, law firms and universities are debating these and other questions about the nature of cloud computing. How will cloud computing affect other industries? There's a growing concern in the IT industry about how cloud computing could impact the business of computer maintenance and repair. If companies switch to using streamlined computer systems, they'll have fewer IT needs. Some industry experts believe that the need for IT jobs will migrate to the back end of the cloud computing system.

Another area of research in the computer science community is autonomic computing. An autonomic computing system is self-managing, which means the system monitors itself and takes measures to prevent or repair problems. Currently, autonomic computing is mostly theoretical. But, if autonomic computing becomes a reality, it could eliminate the need for many IT maintenance jobs.

2.2 Introduction Facial Recognition

Humans have always had the innate ability to recognize and distinguish between faces, yet computers only recently have shown the same ability. In the mid 1960s, scientists began work on using the computer to recognize human faces. Since then, face recognition has come a long way. In the past, security departments in United States used video surveillance system to pull an image of a card counter, thief or blacklisted individual. It then runs that image through the database to find a match and identify the person. In 2001, the Tampa Police Department installed police cameras equipped with facial recognition technology in their Ybor City nightlife district in an attempt to cut down on crime in the area. The system failed to do the job, and it was scrapped in 2003 due to ineffectiveness. People in the area were seen wearing masks and making obscene gestures, prohibiting the cameras from getting a clear enough shot to identify anyone. Boston's Logan Airport also ran two separate tests of facial recognition systems at its security checkpoints using volunteers. Over a three month period, the results were disappointing.

According to the Electronic Privacy Information Center, the system only had a 61.4 percent accuracy rate, leading airport officials to pursue other security options. Humans have always had the innate ability to recognize and distinguish between faces, yet computers only recently have shown the same ability. In the mid-1960s, scientists began work on using the computer to recognize human faces. Since then, facial recognition software has come a long way [1].

2.2.1 Facial Recognition Technology

Every face has numerous, distinguishable landmarks, the different peaks and valleys that make up facial features. FaceIt defines these landmarks as nodal points. Each human face has approximately 80 nodal points. Some of these measured by the software are:

- Distance between the eyes
- Width of the nose
- Depth of the eye sockets
- The shape of the cheekbones
- The length of the jaw line

These nodal points are measured creating a numerical code, called a faceprint, representing the face in the database [2].

In the past, facial recognition software has relied on a 2D image to compare or identify another 2D image from the database. To be effective and accurate, the image captured needed to be of a face that was looking almost directly at the camera, with little variance of light or facial expression from the image in the database. This created quite a problem. In most instances the images were not taken in a controlled environment. Even the smallest changes in light or orientation could reduce the effectiveness of the system, so they couldn't be matched to any face in the database, leading to a high rate of failure. In the next section, we will look at ways to correct the problem.

2.2.2 D Facial Recognition

A newly-emerging trend in facial recognition software uses a 3D model, which claims to provide more accuracy. Capturing a real-time 3D image of a person's facial surface, 3D facial recognition uses distinctive features of the face -- where rigid tissue and bone is most apparent, such as the curves of the eye socket, nose and chin -- to identify the subject. These areas are all unique and don't change over time.

Using depth and an axis of measurement that is not affected by lighting, 3D facial recognition can even be used in darkness and has the ability to recognize a subject at different view angles with the potential to recognize up to 90 degrees (a face in profile). Using the 3D software, the system goes through a series of steps to verify the identity of an individual.

- **Detection**

  Acquiring an image can be accomplished by digitally scanning an existing photograph (2D) or by using a video image to acquire a live picture of a subject (3D).

- **Alignment**

  Once it detects a face, the system determines the head's position, size and pose. As stated earlier, the subject has the potential to be recognized up to 90 degrees, while with 2D, the head must be turned at least 35 degrees toward the camera.

- **Measurement**

  The system then measures the curves of the face on a sub-millimeter (or microwave) scale and creates a template.

- **Representation**

  The system translates the template into a unique code. This coding gives each template a set of numbers to represent the features on a subject's face.

- **Matching**

  If the image is 3D and the database contains 3D images, then matching will take place without any changes being made to the image. However, there is a challenge currently facing databases that are still in 2D images. 3D provides a live, moving variable subject being compared to a flat, stable image. New technology is addressing this challenge. When a 3D image is taken, different points (usually three) are identified. For example, the outside of the eye, the inside of the eye and the tip of the nose will be pulled out and measured. Once those measurements are in place, an algorithm (a step-by-step procedure) will be applied to the image to convert it to a 2D image. After conversion, the software will then compare the image with the 2D images in the database to find a potential match.

- **Verification or Identification**

  In verification, an image is matched to only one image in the database (1:1). For example, an image taken of a subject may be matched to an image in the Department of Motor Vehicles database to verify the subject is who he says he is. If identification is the goal, then the image is compared to all images in the database resulting in a score for each potential match (1:N). In this instance, you may take an image and compare it to a database of mug shots to identify who the subject is.

2.2.3 Biometric Facial Recognition

The image may not always be verified or identified in facial recognition alone. Identix® has created a new product to help with precision. The development of FaceIt® Argus uses skin biometrics, the uniqueness of skin texture, to yield even more accurate results. The process, called Surface Texture Analysis, works much the same way facial recognition does. A picture is taken of a patch of skin, called a skinprint. That patch is then broken up into smaller blocks. Using algorithms to turn the patch into a mathematical, measurable space, the system will then distinguish any lines, pores and the actual skin texture. It can identify differences between identical twins, which is not yet possible using facial recognition software alone. According to Identix, by combining facial recognition with surface texture analysis, accurate identification can increase by 20 to 25 percent.

FaceIt currently uses three different templates to confirm or identify the subject: vector, local feature analysis and surface texture analysis.

- The vector template is very small and is used for rapid searching over the entire database primarily for one-to-many searching.
- The local feature analysis (LFA) template performs a secondary search of ordered matches following the vector template.
- The surface texture analysis (STA) is the largest of the three. It performs a final pass after the LFA template searches, relying on the skin features in the image.

By combining all three templates, FaceIt has an advantage over other systems. It is relatively insensitive to changes in expression, including blinking, frowning or smiling and has the ability to compensate for mustache or beard growth and the appearance of eyeglasses. The system is also uniform with respect to race and gender. However, it is not a perfect system. There are some factors that could get in the way of recognition, including:

- Significant glare on eyeglasses or wearing sunglasses
- Long hair obscuring the central part of the face
- Poor lighting that would cause the face to be over- or under-exposed
- Lack of resolution (image was taken too far away)

Identix isn't the only company with facial recognition systems available. While most work the same way FaceIt does, there are some variations. For example, a company called Animetrix, Inc. has a product called FACEngine ID SetLight that can correct lighting conditions that cannot normally be used, reducing the risk of false matches. Sensible Vision, Inc. has a product that can secure a computer using facial recognition. The computer will only power on and stay accessible as long as the correct user is in front of the screen. Once the user moves out of the line of sight, the computer is automatically secured from other users.

Due to these strides in technology, facial and skin recognition systems are more widely used than just a few years ago. In the next section, we'll look at where and how they are being used and what's in store for the future.

2.2.4 Applications of Facial Recognition System

In the past, the primary users of facial recognition software have been law enforcement agencies, who used the system to capture random faces in crowds. Some government agencies have also been using the systems for security and to eliminate voter fraud. The U.S. government has recently begun a program called (United States Visitor and Immigrant Status Indicator Technology), aimed at foreign travelers gaining entry to the United States. When a foreign traveler receives his visa, he will submit fingerprints and have his photograph taken. The fingerprints and photograph are checked against a database of known criminals and suspected terrorists. When the traveler arrives in the United States at the port of entry, those same fingerprints and photographs will be used to verify that the person who received the visa is the same person attempting to gain entry.

However, there are now many more situations where the software is becoming popular. As the systems become less expensive, making their use more widespread. They are now compatible with cameras and computers that are already in use by banks and airports. The TSA is currently working on and testing out its Registered Traveler program. The program will provide speedy security screening for passengers who volunteer information and complete a security threat assessment. At the airport there will be specific lines for the Registered Traveler to go through that will move more quickly, verifying the traveler by their facial features. Other potential applications include ATM and check-cashing security. The software is able to quickly verify a customer's face. After a customer consents, the ATM or check-cashing kiosk captures a digital image of him. The FaceIt software then generates a faceprint of the photograph to protect customers against identity theft and fraudulent transactions. By using the facial recognition software, there's no need for a picture ID, bankcard or personal identification number (PIN) to verify a customer's identity. This way businesses can prevent fraud from occurring.

While all the examples above work with the permission of the individual, not all systems are used with your knowledge. In the first section we mentioned that systems were used during the Super Bowl by the Tampa Police, and in Ybor City. These systems were taking pictures of all visitors without their knowledge or their permission. Opponents of the systems note that while they do provide security in some instances, it is not enough to override a sense of liberty and freedom. Many feel that privacy infringement is too great with the use of these systems, but their concerns don't end there. They also point out the risk involved with identity theft. Even facial recognition corporations admit that the more use the technology gets, the higher the likelihood of identity theft or fraud.

2.3 Kannel

To send an auto-generated SMS, the technology we have used is called Kannel. It is an open source WAP and SMS gateway, used widely across the globe both for serving trillions of short messages, Push service indications and mobile internet connectivity.

Chapter 3

Methodology

3.1 Face Detection

During this phase all the faces that appear in an image are detected. For this purpose, we have used the algorithm presented by Viola and Michael Jones because of its low response time and relatively high accuracy. Its low detection time allows it to be used in real time system which is the basic requirement for Cloud Powered Real Time Facial Recognition system.

This algorithm provides three contributions:

- Integral image.
- Adaboost Algrithm
- Cascade classifiers

3.1.1 Integral image

This face detection technique categorizes images based on the value of features. There are many reasons for using features instead of pixels. One is that system using features are faster than pixel based systems. In this system, three types of features are used: Two-rectangles, Three-rectangles and Four-rectangle. The value of a feature is the difference between the sum of pixels of dark and light regions. The areas have the same size and shape and are adjacent as shown in figure 3.1.

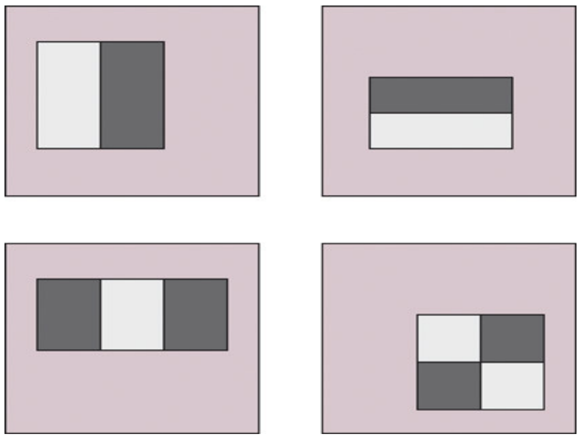


Figure 3.1: **Rectangle features in Viola Jones Algorithm**

Rectangle features can be calculated quickly by using the integral image. Figure 3.2 shows the integral image at location at a point is calculated by summing all the pixels left and above of it.

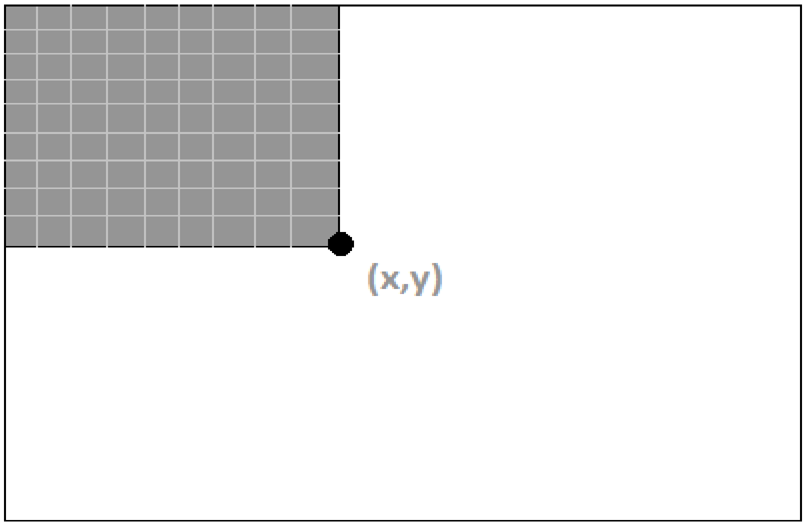


Figure 3.2: Integral image representation

Using the integral image a rectangular sum takes only four calls from the look up table (See figure 3). The difference between two rectangles can be computed in eight references. This technique allows two rectangle features to be calculated in six array references, eight and nine for three and four rectangles as shown in figure 3.3.

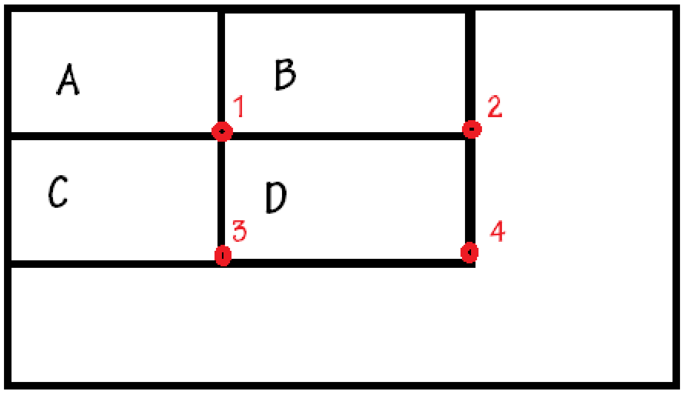


Figure 3.3: Calculating Integral image for a rectangle

3.1.2 Adaboost Algorithm

There are 160,000 exhaustive sets of rectangle features associated with an image sub-window (100x100), which is far larger than the number of pixels. All the features can be calculated efficiently, it will be computationally very expensive and will decrease the speed of the detector. However, it has been found out by experience that by choosing a limited number of features and combining them can result in very efficient detector [3].

In the system, a very of Adaboost is used to find these features and train the classifier. During Adaboost, the problem is to assign large weight to a good classification function and a smaller weight to poor function. For and analogy between weak classifier and features, Adaboost is an effective process to search a small number of good features which have significant variety. Our practical method for completing this analogy is to restrict the weak learner to the set of classification function each of which will depend on a single feature. To achieve this, the weak learning algorithm is designed to select the single rectangle feature which best separates the positive and negative examples. For every feature, the weak learner determines the threshold classification function, such that the minimum number of examples are misclassified. A weak classifier (h (x,f,p,θ)) thus consist of feature (f), threshold (θ) and polarity (p) indicating the direction of inequality, as stated in Eq. (1) below.

$$h(x, f, p, \theta) = \begin{cases} 1 & \text{if } pf(x) < p\theta \\ 0 & \text{otherwise} \end{cases} \qquad \text{.......... Equation (1)}$$

Where x is a 100x100 pixel sub-window of an image.

The figure 3.4 below gives a graphical representation of Adaboost process in which weights are assigned to weak classifiers.

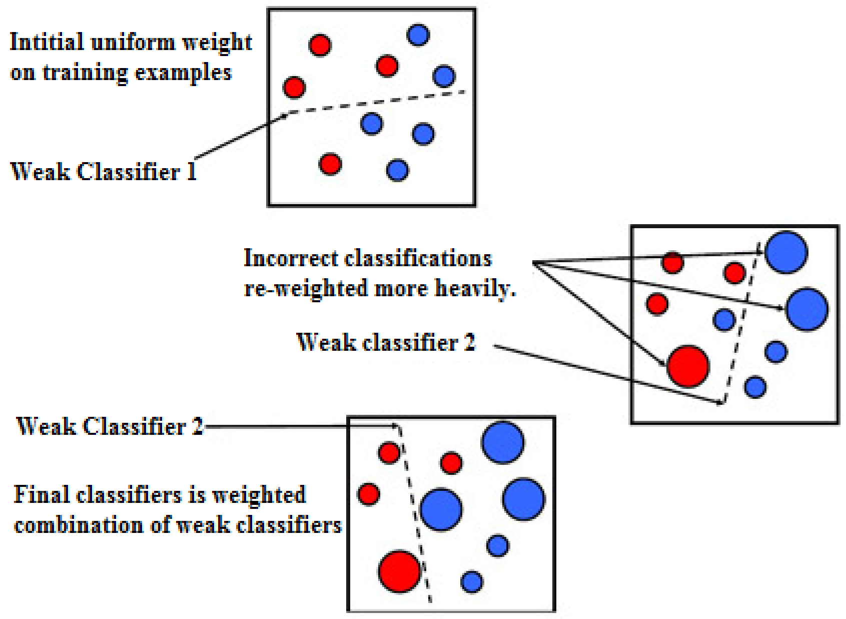

Intitial uniform weight
on training examples
Weak Classifier 1
Incorrect classifications
re-weighted more heavily.
Weak classifier 2
Weak Classifier 2
Final classifiers is weighted
combination of weak classifiers

Figure 3.4: Adaboost Example

3.1.3 Cascade Classifier

In this detector, several classifiers are cascaded to make a fast and efficient system that achieves increased detection performance while radically reducing computation power. The idea is to train simple and small boosted classifiers that reject most of the negative sub windows. Such simpler classifiers are used to reject most of the on face parts before applying the more complex classifiers to achieve false positive rates.

The overall form of the detection process is that of a degenerate decision tree, what we call a cascade as shown in figure 3.5. Positive results from each classifier are fed into the next classifier if a negative result occurs, it is discarded and no further processing is done [4].

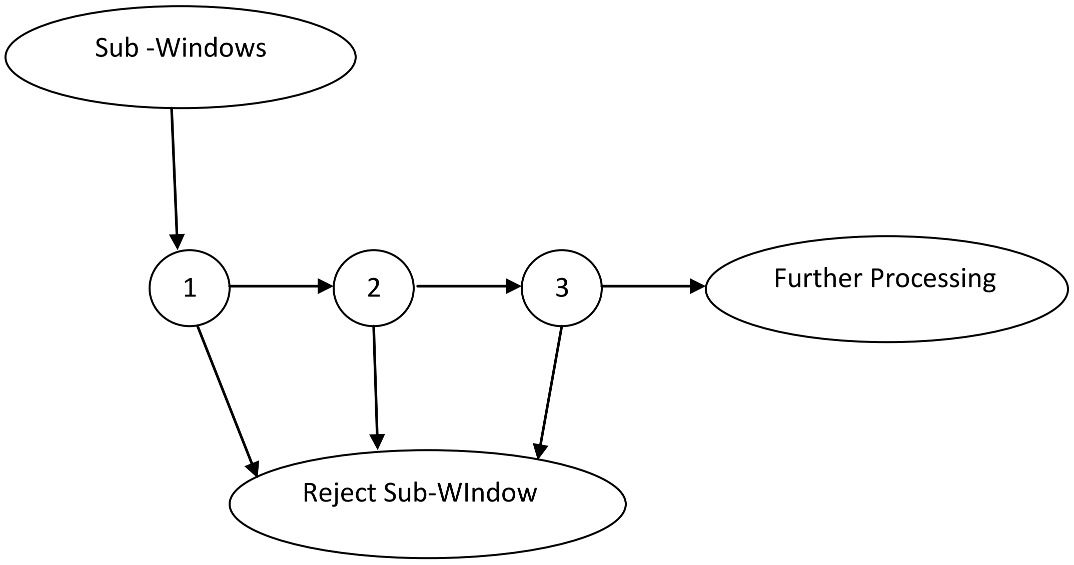


Figure 3.5: Cascaded Classifier

3.1.4 Cascaded Classifier

The cascade is structured keeping in mind that most of the sub windows in an image are non faces. Therefore, the cascade is designed to reject most negative parts of an image as early as possible and leave the complex computation for the promising face containing regions. This results in increased efficiency of the detector.

3.1.5 Structure of the Cascade

The final detector is a 38 layer cascade of classifiers which includes a total of 6060 features the first classifier in the cascade is constructed using two features and rejects about 50% of non faces while correctly detecting close to 100% of faces. The next classifier has 10 features and rejects 80% of non faces while detecting almost 100% of faces. The next two layers are 25 feature classifiers followed by three 50 features classifiers followed by classifiers with different numbers of features.

3.1.6 Scanning the Detector

The final detector is scanned across the image at several scales and positions. Scaling is achieved by scaling the detector itself, instead of the image. The detector is also scanned across different positions. Subsequent points are obtained by shifting the window to some number of pixels T. The shifting is controlled by the scale of the detector: If the current scale is L, the window is shifted by [LT], where [] is round-off operation

3.2 Face recognition

The two available techniques for face recognition are:

1. Recognition using Eigen Faces
2. Recognition using Canny Edge Detector

3.2.1 Recognition Using Eigen Faces

The system functions by projecting face images on to feature space that spans the significant variation among known face images. The significant features are known as Eigen Faces; because they are principal components of the set of faces [5].

3.2.1.1 Introduction to Eigen Faces

In many practical applications the computational models of face recognitions are used by applying it to different problems. Developing such a system is quite difficult because human faces are complex. Face recognition is a very high level task, therefore, broad constraints are applied. “Eigen faces for Recognition” seeks to implement a system capable of efficient, simple, and accurate face recognition in a constrained environment such as white background and good light conditions. Classification is instead performed using a linear combination of characteristic features that are Eigen faces.

3.2.1.2 Eigen Faces

The work done previously ignores the question of which features are important for classification and which are not. It is the basic motivation for using Eigen faces. The use of principal component analysis of the faces tries to answer this question. This technique reduces the dimensionality of training set, leaving only those features that have most significance for recognition. The system starts by loading the training set (ideally a number of images of each subject with different angles and expressions). Eigen vectors and Eigen values are computed on the co-variance matrix of the training images. The N highest Eigen vectors are kept and the rest are ignored. Finally, the whole mean subtracted database is projected onto the face space. This process is shown in a flow chart diagram in figure 3.6 below.

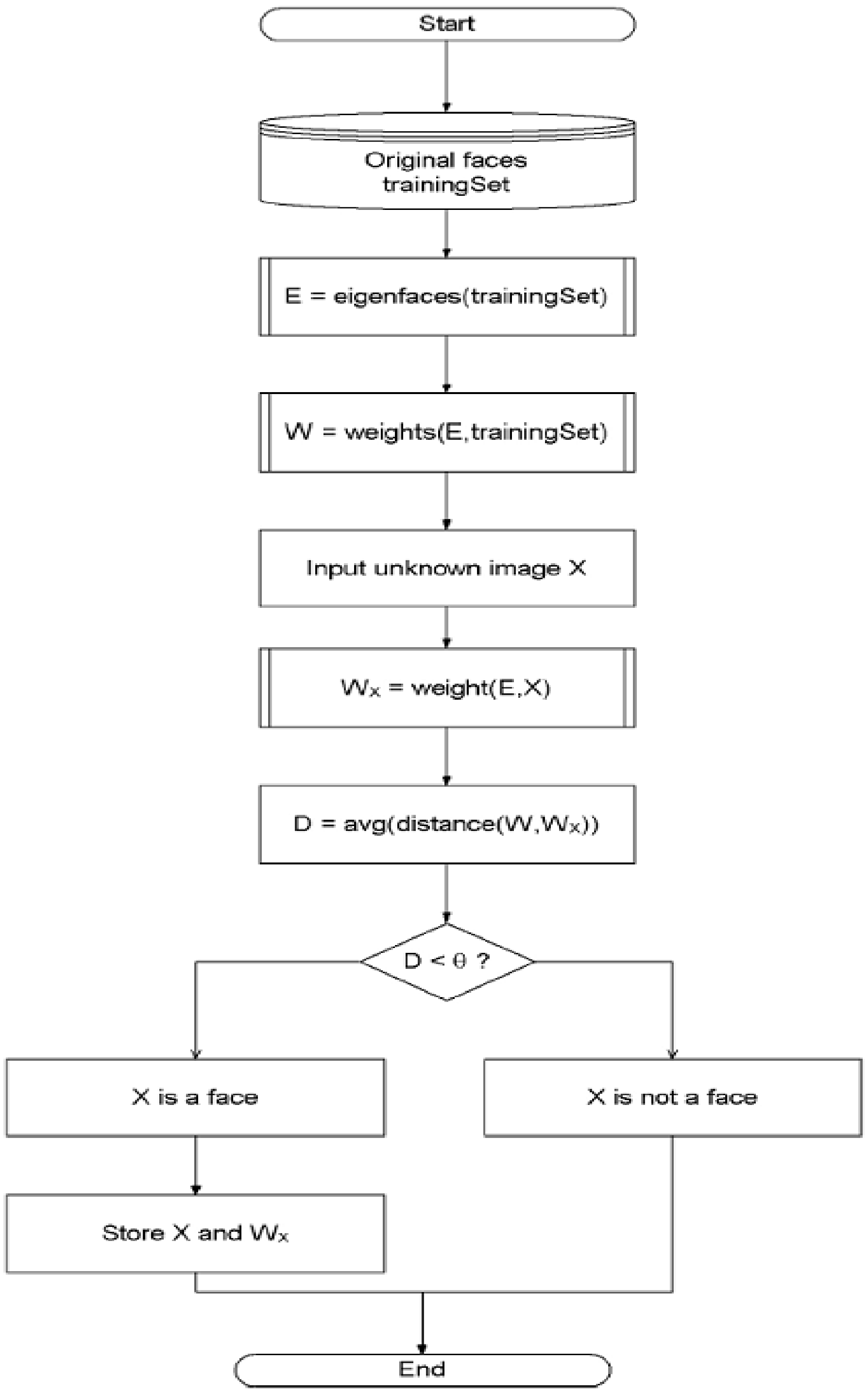

Start
Original faces trainingSet
E = eigenfaces(trainingSet)
W = weights(E,trainingSet)
Input unknown image X
W_x = weight(E,X)
D = avg(distance(W,W_x))
D < θ ?
X is a face
X is not a face
Store X and W_x
End

Figure 3.6: Eigen Faces process flow chart diagram

3.2.1.3 The Eigen Face Approach

The algorithm for the facial recognition using Eigen faces involves the following procedures. Firstly, the original images of the training set are subtracted from the mean of the database and then the covariance matrix of mean subtracted database is calculated. The next step is to compute the Eigen vectors and Eigen values of that covariance matrix and finally the transformation into a set of Eigen faces then on getting an unknown image x, that image is again subtracted from mean and projected onto M_0 Eigen vectors. Consider each image vector as a point in space and calculate an average distance D between the database image vectors and the vector of the unknown image x. the face with the minimum average distance D with a particular image in the database is considered to be the same [6].

3.2.1.4 Calculating Eigen Faces

Eigenfaces are computed using the technique of principle component analysis. The process is divided in the following steps and the general formula for the computation is defined in Eq. (2).

- **Preparation of Database:**
  In this step the faces constituting the training set should be prepared.
- **Subtract the Mean:**
  In this step the calculation of the mean of the database (T_i) is done and this mean is subtracted from each of the original image matrix and stored in another variable (ϕ_i)
- **Calculation of Covariance matrix:**

  After that co variance matrix is calculated as:

$$C = \frac{1}{M} \sum_{n-1}^{M} \Phi_n \Phi_n^T \qquad \text{.......... Equation (2)}$$

- **Calculate the Eigenvectors and Eigenvalues of the covariance matrix**

In this step, the Eigenvectors u_i and the corresponding Eigenvalues $\boldsymbol{\lambda i}$ are calculated. The Eigenvectors must then be normalized so that they are unit vectors, i.e. of length one.

- **Selection of Principal Components**

  From N Eigenvectors u_i, only M_0 should be chosen which have the highest Eigenvalues and they are of the most significance. The higher the value, the more characteristic features of a face does the particular Eigenvector describe. Eigenfaces with low Eigenvalues can be left out, as they are of lesser importance to features of the faces. After M_0Eigenfaces ‘u’ are determined, the ‘training’ phase of the algorithm is finished [7].

3.2.2 Principal Component Analysis

PCA is a useful statistical technique that has applications in many fields such as face recognition and image compression, and is a common technique for finding patterns in data of high dimension. It is a way of expressing the data in such a way as to highlight their similarities and differences. Since patterns in data can be hard to find in data of high dimension. PCA is a powerful tool for analyzing data [8].

The other main advantage of PCA is that once you have found these patterns in the data, and you compress the data, i.e. by reducing the number of dimensions, without much loss of information.

3.2.2.1 Method

The method to calculate PCA includes the following steps. The general formula for the calculation is given in Eq. (3).

- **Get Data**

  First of all get some data whose principal component needs to be calculated.

- **Subtract the mean**

The mean the whole database needs to be subtracted from each component of the database. It is basically the average across each component. So now you have the mean subtracted database which gives a dataset whose mean is zero.

- **Calculate the Covariance Matrix**

Covariance gives the linear relationship between two variables. The covariance can be calculated by the following formula:

$$S = \frac{1}{N}\sum_{n=1}^{N}(x_n - \overline{x})(x_n - \overline{x})^T \quad \text{.......... Equation (3)}$$

n!/(n-2)!*2 covariance values can be calculated for an n-dimensional dataset. It is a useful way to form a covariance matrix by calculating the all possible covariance between each dimension as mentioned above in Eq. (3).

- **Calculate the Eigen values and Eigen vectors of the covariance matrix**

Eigen values and Eigenvectors give very useful information about the data. It is important that all the Eigenvectors must be unit Eigenvectors. Eigenvectors are always perpendicular to each other. But, the more important fact is that, they provide us with information about the patterns in the data. So, by the calculation of Eigenvectors of the covariance matrix, we are able to extract lines that characterize the data.

- **Choosing the components and forming a feature vector**

This is the part of data compression and reducing dimensionality. Eigen values are quite different values. In fact, the Eigenvector with the highest Eigen value is the principle component of the data set. It actually gives the significant relationship between the data components. Once Eigenvectors of the covariance matrix are calculated, the next step is to order them in descending order. This gives you the components in order of significance. Now, it is your choice rather to ignore the components of lesser significance or keep all the components. You do lose some information, but you don't lose much if the Eigen values are small. The final data will have fewer dimensions, if you decide to leave some of the components. If your dataset has m dimensions then you will get m Eigenvectors and Eigen values and upon selection of n Eigen values your final dataset will now only have n dimensions.

- Next part is to form a feature vector, which is just a name for a matrix of vectors. It is formed by placing the n chosen Eigenvectors in a matrix in the columns. The mathematical representation of this is given below in Eq. (4).

$$\boldsymbol{FeatureVector = (e_ig_1\ e_ig_2\ e_ig_3 \ldots\ e_ig_n)} \quad \text{.......... Equation (4)}$$

For our example we have two Eigenvectors so we have two choices either to form a feature vector with both the Eigenvectors.

- **Deriving the new data set**
- This is the last step in PCA. Once we have chosen the Eigenvectors that we want to keep in our data and formed a feature vector, transpose of the vector is taken and multiplied on the left of the original data set, transposed. The mathematical representation of this is given below in Eq. (5).

$$\boldsymbol{Final\ Data = RowFeatureVector \times RowDataAdjust} \quad \text{........ Equation (5)}$$

Where 'Row Feature Vector' is the matrix with the Eigenvectors in the rows, with the most significant Eigenvector at the top, and 'Row Adjust Data' is the mean-adjusted data with the data items in each column and each row having a separate dimension. 'Final Data' is the final dataset with dimensions along rows and the data items in columns.

3.2.3 Improvement of the Original Algorithm

There is a problem with the above described algorithm. The covariance matrix C in step 3 has a dimensionality of N^2x N^2, so we would have N^2 Eigen faces and Eigen values. For a 100

×100 image that means that one must compute a 10000 ×10000 matrix and calculate 10000Eigenfaces. Computationally, this is not very efficient as most of those Eigen faces have not much importance and can be ignored.

Where the matrix A=[ϕ_1,ϕ_2……ϕ_M]. The matrix C however is N^2 x N^2, and determining the N^2 Eigenvectors and Eigen values is a difficult task for large images.

$$\mathbf{L = AA^T} \quad \text{……….. Equation (5.1)}$$

$$\mathbf{A^T V_j = U_j V_j} \quad \text{……….. Equation (5.2)}$$

Combining Equation (5.1) and Equation (5.2). We get,

$$\mathbf{C = AA^T} \quad \text{……….. Equation (5.3)}$$

Where L is a M x M matrix, v_l are M Eigenvectors of L and u are Eigen faces. Note that the covariance matrix C is calculated using the formula C = AA^T, the original formula is given only for the sake of explanation of A as stated above in Equation (5.3). The advantage of this method is that we have to compute only M numbers instead of N^2. Usually, M is very much less than N^2as only a few principal components will be kept. The amount of calculations to be performed is reduced from the number of pixels (N^2 x N^2) to the number of images in the training set (M). In the step 5, the Eigen values allow us to arrange the Eigen faces according to their significance and to use only a subset of M Eigenfaces, the M_0 Eigen faces with the largest Eigen values.

3.2.3.1 Face Recognition

Firstly a new image is subtracted from the mean image and then it is projected into the face space using the following mathematical equation given in Eq. (6).

$$\mathbf{W_k = U^T{}_k\,(T\text{-}U)} \quad \text{……….. Equation (6)}$$

3.2.3.2 Using Euclidean Distance

Euclidean distance or simply 'distance' calculates the *root of square differences* between coordinates of a pair of objects as shown in Eq. (7). It is an ordinary distance between two points that one can measure with a ruler. The formula for Euclidean distance is given by:

$$\boldsymbol{d_{ij}} = \sqrt{\sum_{\boldsymbol{k=1}}^{\boldsymbol{n}} \left(\mathbf{x_{ik}} - \mathbf{x_{jk}}\right)^{\mathbf{2}}} \quad \text{.......... Equation (7)}$$

The Euclidean distance

$$\boldsymbol{\epsilon = ||\Omega - \Omega_k||} \quad \text{.......... Equation (8)}$$

Eq. (8) measures the distance between new image and the rest of the faces in the database. Out of all these distances the image with the minimum distance is said to be recognized with the test image.

3.2.3.3 Using Mahalanobis Distance

Mahalanobis distance is based on correlations between random variables. Identification of different patterns and analysis can be done using this distance. It is a good way to determine the similarity between an unknown sample set and a known one. It is different from the Euclidean distance as it does not dependents on the scale of measurements rather it depends upon the correlations of the data set. It is basically class distance and also called quadratic distance.

Suppose a multivariate vector $x=(x_1, x_2, x_3,\ldots.x_N)^T$ form a group of values with means $(\mu_1,\mu_2,\mu_3,\ldots.\mu_N)^T$ and having a covariance matrix S so the Mahalanobis distance is given by the formula as given below in Eq. (9):

$$\boldsymbol{D_{M(x)}} = \sqrt{(\mathbf{x}-\mathbf{u})\mathbf{S}(\mathbf{x}-\mathbf{u})} \quad \text{.......... Equation (9)}$$

It becomes Euclidean distance if the covariance matrix comes out to be identity , but if it is a diagonal matrix then the measured distance is known as the normalized Euclidean distance The Mahalanobis Distance is a better distance measure for pattern recognition problems. It removes the problems related to scale and correlation that are inherent with the Euclidean Distance by considering the covariance between the variables.

So the same procedure is repeated for calculating the distance between the new image and the rest of the database and applying the formula for Mahalanobis distance. The image with the minimum distance is a match to the new image.

3.2.4 Database

For the purpose of testing of algorithm and running of our system, we developed a database of our classmates. The database contained frontal and upright face images of almost same scale. The image resolution is set at 100 X 100 pixels. All the images are also converted to grayscale to decrease the size of image and increase the performance. The total number images were 4000. Figure 3.7 shows how faces are saved in the database.

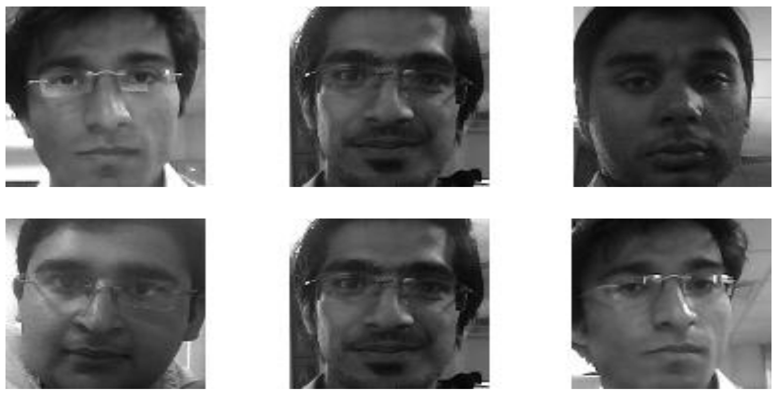


Figure 3.7: Database Images

3.3 Kannel

After detecting and recognizing criminals, the next part is to notify the on-duty guard that a criminal has been spotted. For this, we have used Kannel which is an open source WAP and SMS gateway, used widely across the globe both for serving trillions of short messages, Push service indications and mobile internet connectivity. Figure 3.8 shows the architecture of Kannel technology.

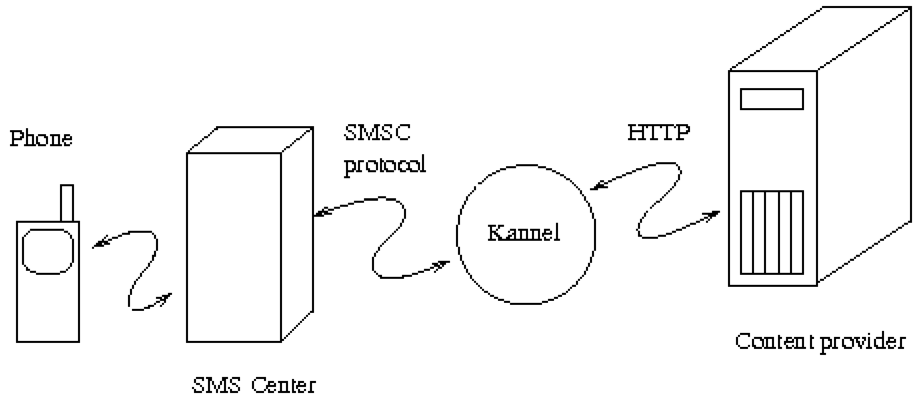


Figure 3.8: Kannel Architecture

Chapter 4

Results:

The following section includes discussion on expected and actually achieved milestones during development of the application “Cloud Powered Real Time Facial Recognition”.

4.1 Milestones

During the development, the system was divided into modules and successful development of each was considered as an achieved milestone. The modules in which the system was divided are as follow:

4.1.1 Database:

The first task during the application development lifecycle is to develop a large scale database which involves the following steps.

1. Creation of image database
2. Optimization of database for optimal query processing.

4.1.2 Video Processing Application:

The major part of developing this system, is to develop the desktop application that takes video streams as input, extract human faces and matches them with the ones present in the database and also trigger a text message. The steps involved in developing this part are:

1. Get video stream from the camera (Network Camera + Webcam) to the computer.
2. Applying face detection techniques to extract faces from the stream.
3. Processing of extracted faces such as adjusting lighting and brightness.
4. Send the processed images to face recognition engine.
5. Get the information from the face recognition engine after the comparison of sent images with the ones present in the database.
6. Sending gathered information to the respective security person through a text message.

4.1.3 Setting up Cloud:

After successful development of video application, the next part was to configure cloud. The technology we choose is called Open Stack. However, we also used an already configured cloud by Huawei, using thin client. Setting up a cloud involves the following steps.

1. Configuration of Open Stack cloud.
2. Deployment of application on cloud.

4.2 Achieved Milestones:

The milestones that were achieved during development of this application are discussed below. Team’s hard work and effort really paid off and we successfully achieved all the defined milestones.

4.2.1 Database Creation:

For the purpose of testing of algorithm and running of our system, we developed a database of our classmates. The database contained frontal and upright face images of almost same scale. The image resolution is set at 100 X 100 pixels as shown in figure 4.1. The total number images were 4000.

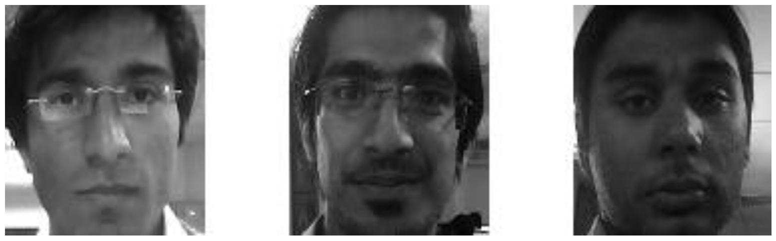


Figure 4.1: Database Images

4.2.2 Optimizating Database:

For efficient processing of the given query, the database was optimized using different optimization algorithms including indexes, B-trees and Binary search trees so that image matching process takes less time and increase overall application efficiency.

4.2.3 Working Of Desktop Application

To get video stream and input it into the application multiple IP Cameras and built-in webcam have been used. We have used algorithm presented by Viola and Michael Jones because of its low response time and relatively high accuracy. Its low detection time allows it to be used in real time system which is the basic requirement for our project. The three main parts of this algorithm are Integral Image, AdaBoost Algorithm and Cascaded Classifiers which have already been discussed. To send the detected faces to the facial recognition engine, client server like architecture has been used. The faces are firstly converted into a byte code and then over to the server for comparing with images present in the database. Figure 4.2 shows the working of video processing desktop application while Table 4.1 shows the comparison between efficiency of different algorithms.

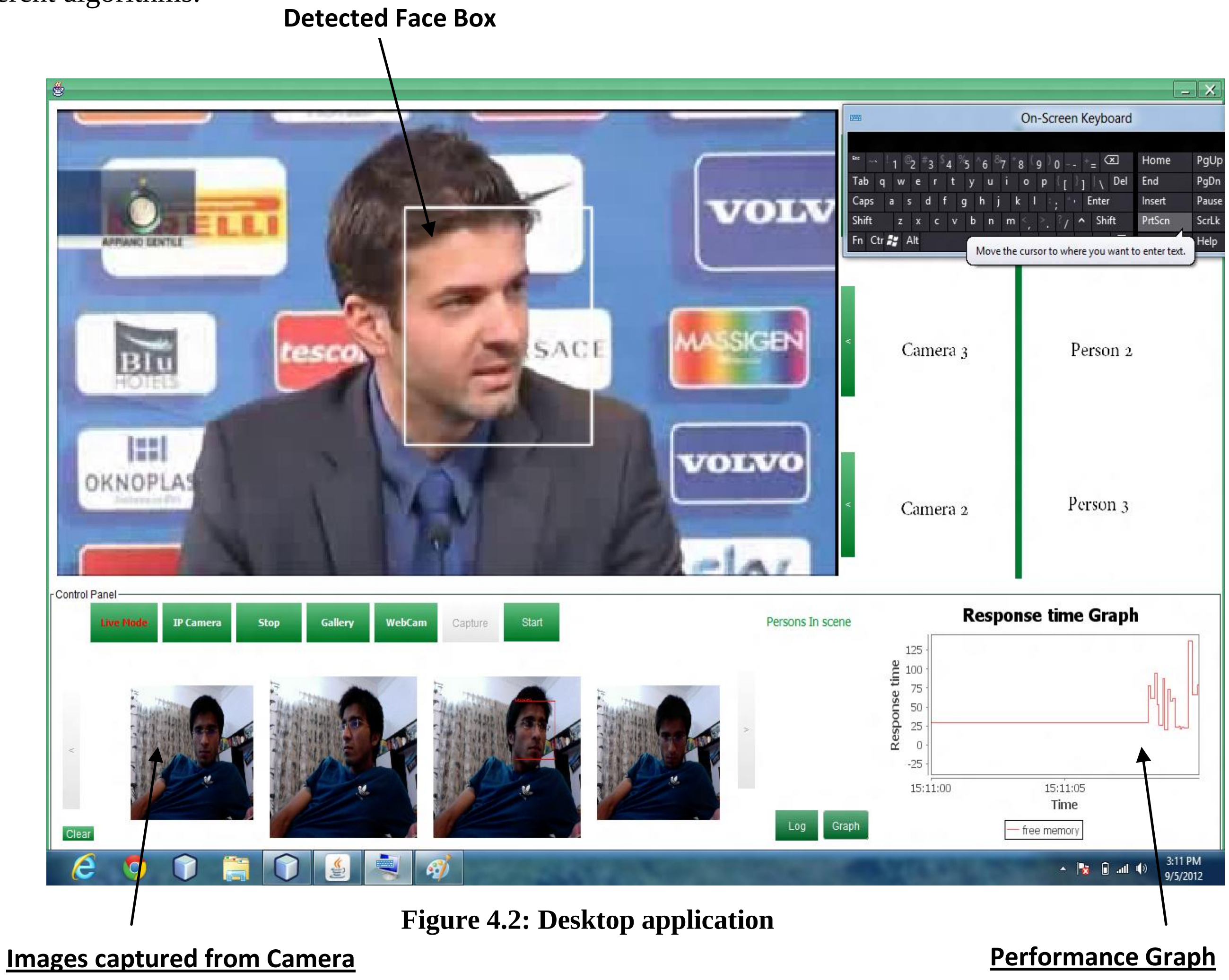


Figure 4.2: Desktop application

Table: 4.1 - Comparison between efficiency of different algorithms

	False Detections							
Detector	**10**	**31**	**50**	**65**	**78**	**95**	**167**	**422**
Viola Jones	76.1%	88.4%	91.4%	92.0%	92.1%	92.9%	93.9%	94.1%
Viola Jones (Voting)	81.1%	89.7%	92.1%	93.1%	93.1%	93.2%	93.7%	-
Rowley-Baluja-Kanade	83.2%	86.0%	-	-	-	89.2%	90.1%	89.9%
Schneiderman-Kanade	-	-	-	94.4%	-	-	-	-
Roth-Yanh-Ahuja	-	-	-	-	94.8%	-	-	-

4.2.4 Facial Recognition:

For facial recognition, we have used Eigen Faces technique. This functions by projecting face images onto feature space that spans the significant variations among known face images. The significant features are known as "Eigenfaces" because they are the eigenvectors (principal components) of the set of faces.

Displaying person's data after correctly recognizing face.

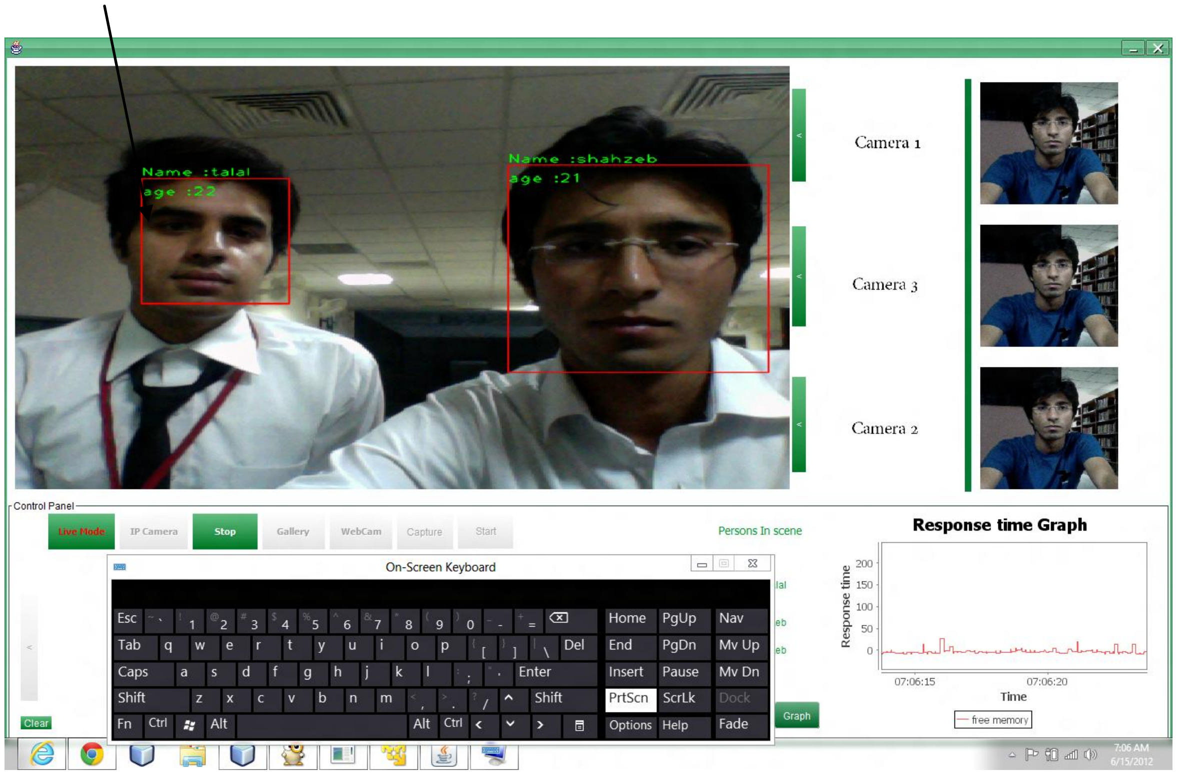


Figure 4.3: Face Recognition is Desktop Application.

Face recognition using eigenfaces technique has been shown below in the figure 4.3 while the comparison between different recognition algorithms has been depicted in table 4.2.

Table 4.2 - Results of Face Recognition using Eigenfaces

Features	N=5	N=10	N=15	N=20	N=25	N=30
Euclidean	87.5%	92.5%	95%	96.5%	96.5%	96.5%
Mahalanobis	92.5%	93%	96%	97.5%	98%	99%

Where N is the number of eigenvectors.

4.2.5 Sending Information Back To Client:

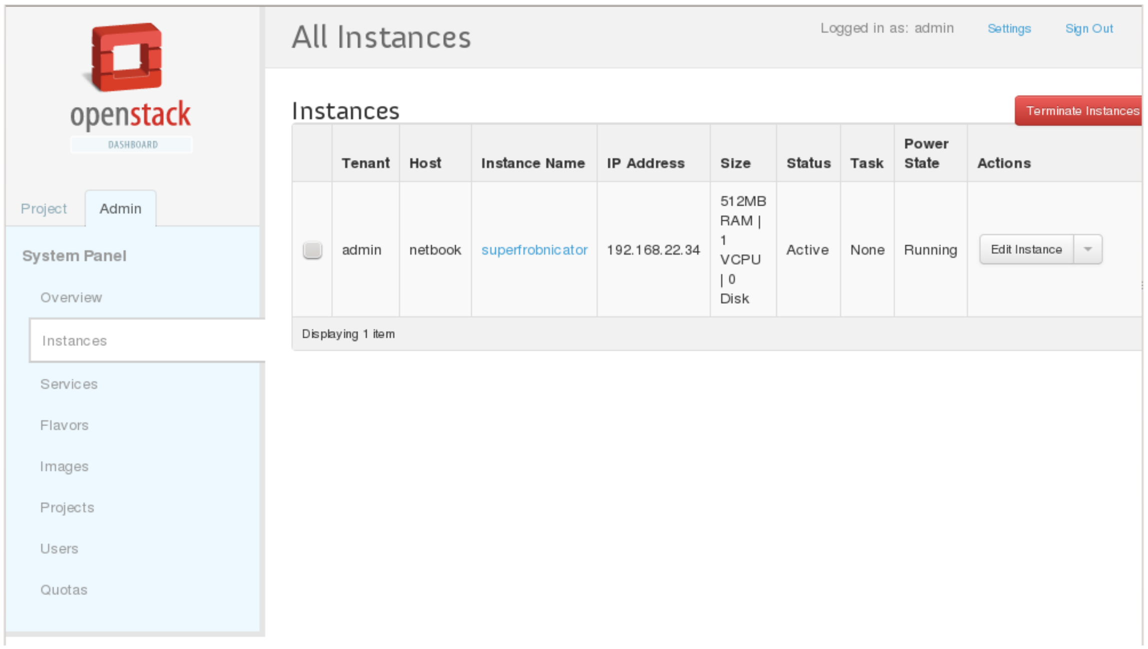
After the successful matching of faces, the respective information of a criminal such as name, age, address and threat level sent back to the client from where a text message is going to be generated to forward that information to the person incharge.

4.2.6 Sending Information From Client to Respective Guard:

After the information has been received at the client end, a text message is sent to the respective security guard. For sending a text message, we have used Kannel. Meanwhile, sending a text message over a normal mobile network is a constraint that affects the performance of the system. We plan to use a satellite phone, through which the responsible person could be notified in less than a second.

4.2.7 Configuring Cloud:

To achieve the real time efficiency we used Open Stack. We choose OpenStack because of its advance networking capabilities, high-performance and widely distributed computing. The configuration was the hard part but we managed to achieve that. Open Stack dashboard screenshot has been attached below in figure 4.4, while the configuration manual has been attached in the appendices.



Instances

	Tenant	Host	Instance Name	IP Address	Size	Status	Task	Power State	Actions
	admin	netbook	superfrobnicator	192.168.22.34	512MB RAM | 1 VCPU | 0 Disk	Active	None	Running	Edit Instance

Displaying 1 item

Figure 4.4: Open Stack Dashboard

4.2.8 Deployment of Application on Cloud:

The last part was the deployment of application on cloud, and we successfully achieved that during project's final defense and Open House. The one thing we noticed while running this application on cloud is that it significantly improved the performance of the system.

4.2.8.1 OpenStack vs. Huawei Cloud

We tested the application on both Open Stack and Huawei cloud. As open stack has limited resources, the application performance was significantly better on Huawei. The comparison is shown with the help of graph in figure 4.5.

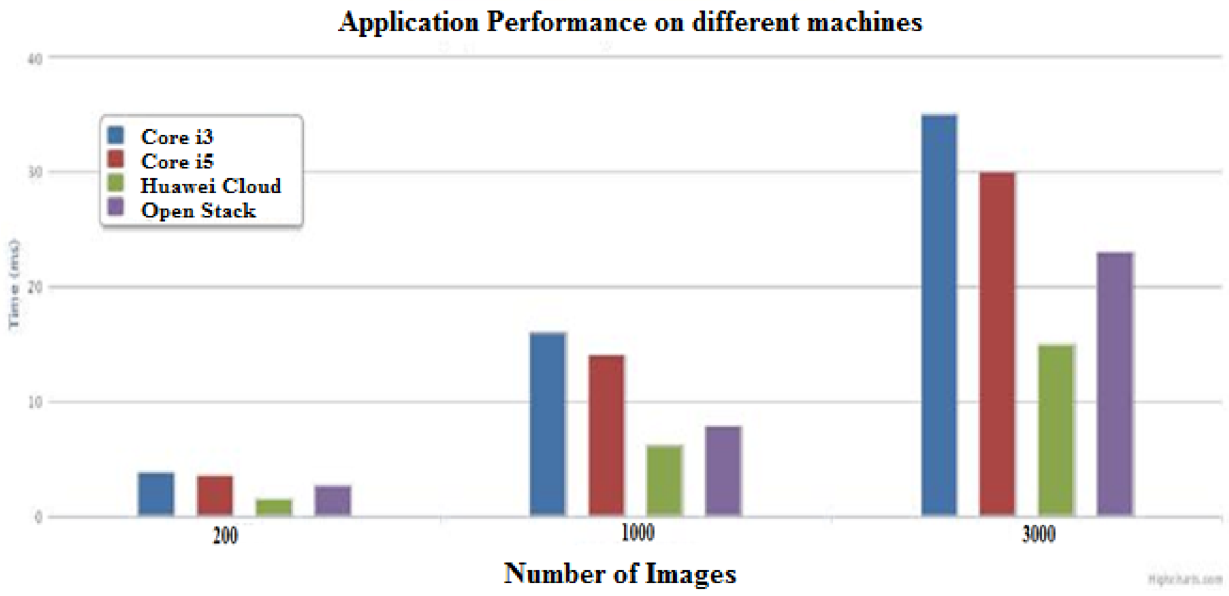


Figure 4.5: Open Stack vs. Huawei Cloud performance graph

The figure above is application's performance chart being run on different machines. It is very clear that application performs best on Huawei cloud.

Chapter 5

Discussion:

5.1 Face Detection and Recognition:

The basic problem that we have analyzed is the fact that Eigenfaces is the most standard technique for face recognition. To solve the problem of this nature like making a real time security system, it is necessary to implement somewhat advanced Face Detection algorithms than Eigen faces to achieve acceptable results. There are a lot of limitations of Eigenfaces technique. This algorithm gives satisfactory results only when the changes in conditions with respect to the trained images are very low for example 200. Even the small amount of change in conditions degrades the results. These conditions include light intensity, scale, size, background, face orientation and change in appearances.

The light intensity of the test image should be almost identical to the images in database. This, in real time, is very difficult to maintain as light changes throughout the day. Hence, algorithms that do not depend upon the light variations is to be used. The background of the image is another hurdle in the success for faces containing some background. The background also produces some features when Eigenfaces is applied and reduces the accuracy of results. The solution to this problem could be to create an environment where there is a white background and the students being instructed to move in front of the camera one at a time. This is, however, very difficult to achieve. Another solution could be that the face localization algorithm be improved in such a way that it only gives the features of the face and discards any background. This can improve the results. Scale and size of the images is another problem. The test face should have an exact scale and size relative to the trained faces. This is a difficult problem to solve because in a real-time system, it is not possible for the subjects to look at the camera at the same position. Therefore, it is difficult to maintain scale and size of the image. Although a detected face image is larger in size than the image in database could be resized, a smaller image when enlarged will result in a change in scale. Keeping the face orientation relatively similar to the database is another daunting task. The test faces should be frontal and upright; any tilted face reduces the accuracy. Face Localization algorithm detects most of the frontal and some semi-frontal faces. If the algorithm localizes a semi-frontal face, the probability of giving a wrong result becomes high. This issue can be resolved by changing the face localization algorithm in a way that it only detects frontal and upright faces. Eigenfaces technique gives the image as a recognized one which has the least distance with the images in database. So, if an image of a person is tested whose image is not present in the database, the system will certainly give a wrong output.

5.2 Running the Application on Cloud:

Image matching and processing in a strict time bound is a compute intensive task. It is not a new problem and security agencies of almost all countries have been using databases solution and algorithms to solve it. However, given large set of data to search and match a particular image especially in real time requires extensive high performance hardware on demand. There are several solutions to this problem but none gives an optimal result duo to limitation of infrastructure and scalability of the underlying hardware. The proposed work takes place in the context of processing a stream of images using Cloud. Each image in the stream has to be processed by several programs, humans, cameras and other devices -including mobile. The throughput of this stream is prone to significant variations. In order to be capable of absorbing such variations a platform of type "Cloud computing" has been chosen to perform the processing [10].

Our project aims to investigate the possibilities of image processing using clouds, where data/ images comes from sensors, cameras, security personnel, internet stream and manual addition. Since the data is coming from varied data sources and it is to be matched, mapped and recognized with the large set of data present at a single location/ cluster to find the relevant information. It is important to use elastic and scalable cloud infrastructure to return the results to the respective device in real time. Such services will be attracted to the cloud not only because they must be highly available, but also because these services generally rely on large data sets that are most conveniently hosted in large datacenters.

Such services will be attracted to the cloud not only because they must be highly available, but also because these services generally rely on large data sets that are most conveniently hosted in large datacenters. This is especially the case for services that combine two or more data sources or other services.

Chapter 6

Conclusion & Future Work:

Implementation of Cloud Powered Real Time Facial Recognition with very good results seems difficult, but with the power of the cloud we have managed to achieve 92% efficiency overall. This can obviously be increased if the resources on cloud are increased.

Our future goals would be to implement an advanced algorithm for face detection that does not falter with the change in light intensity, background and other issues discussed above. Some of the techniques that we came across but could not implement due to shortage of time are AAM and SVM. However, a thorough research that whether these algorithms will work in our constraints is required. Sending a text message over a normal mobile network is another constraint that affects the performance of the system. We plan to use a satellite phone, through which the responsible person could be notified in less than a second. Moreover, the cloud instance we are using at the moment has a limited processing power, and in future we would like to scale it further for better performance. Increasing processing resources increases the performance of the application has been already proved during the testing of our application on different machines. Therefore, increasing the processing capabilities, the time it takes to get input video stream, detect, match faces and send SMS to security guard would get significantly lower. We also plan to make our system commercial by selling it to Universities, Companies and Law enforcement agencies. We have already received offers from organizations who are interested to use our system in their offices. It will be up to the client in what way they want to utilize the system, as it can be used as an attendance system in Universities, Colleges and Organizations as well as a security system. For that, we are already in contact with different marketing agencies to market our system in order to attract businesses who can invest in our product. We have also applied for international business plan competitions such as GIST and Invent, through which we can introduce the product outside Pakistan to global investors in order to get the desired investment.

Chapter 7

References:

[1] Cloud Computing Introduction, Source: http://en.wikipedia.org/wiki/Cloud_computing, Access Date: 30th October, 2011.

[2] An Efficient Method for Face Localization and Recognition in Color Images, Farshid Hajati, Karim Faez, and Saeed Khosh, Source: Ieee.org, Access Date: 30th October, 2011, Published Date: October 8, 2006.

[3] Face detection, Inseong Kim, Joonhyung Shim, and Jinkyu Yang, Source: Stanford.edu, Access Date: 1st November, 2011 , Published Date: Unknown.

[4] Robust Real-time Object Detection, Paul Viola and Michael Jones, Source: International Journal of computer vision, Access Date: 1st November, 2011, Published Date: July 31, 2001.

[5] Face Recognition by Vector Machines, Guo Dongguo, Stan Z. Li, and Kapluk Chan, Source: Ieee.org, Access Date: 21st November, 2011, Published Date: December 2002.

[6] Eigen faces for Recognition, Matthew Turk and Alex Pentland, Source: Journal of Cognitive Neuroscience, Access Date: 21st November, 2011, Published Date: March, 1991.

[7] Eigenface-based facial recognition, Dimitri Pissarenko, Source: Journal of Cognitive Neuroscience, Access Date: 5th December, 2011, Published Date: 13th February, 2003.

[8] Face Recognition Using Kernel Principal Component analysis, Kwang In Kim, Keechul Jung, and Hang Joonkim, Source: Ieee.org, Access Date: 26th December, 2011 , Published Date: December 1, 2002.

[9] Discriminant Analysis of Principal Components, Rama Chellappa, Daniel L.Swets, John Weng, Source: Ieee.org, Access Date: 26th December, 2011 , Published Date: 1998.

[10] Cloud-Powered Facial Recognition Is Terrifying, Jerad Keller, Source: TheAtlantic.com. Access Date: Published Date: 29th September, 2011.

Chapter 8

Appendices:

Open Stack configuration manual: