Biometric Implementation of Bayesian Networks for Face Recognition on Android Mobile Devices

Musundire Daniel
Department of Computer Science,
National University of Science and Technology, Zimbabwe
daniel.musundire@nust.ac.zw

Chilumani Khesani Richard
Department of Computer Science,
National University of Science and Technology, Zimbabwe
khesani.richard.chilumani@nust.ac.zw

Thambo Nyathi
Department of Computer Science,
National University Of Science and Technology, Zimbabwe
thambo.nyathi@nust.ac.zw

Abstract. *Face recognition is a field that receives attention and inquiry by Artificial Intelligence, Computer Security and Intelligent Data Processing researchers. Its application as a biometric has also grown due to the growth of the camera technologies. The digital camera's resolution and noise reduction in images makes the camera view the world in more or less the same as a human being. However, this technology has been difficult to implement on mobile devices, phones in particular. This has been chiefly because of the computational complexity of the algorithms used. The memory size and processing speed of the mobile phones has been a constraint as well. There has been a huge growth experienced in the mobile phones' industry with respect to the gadgets' processing power. Face detection has been implemented using other statistical methods but this research uses the Bayesian network. This method is graphical and uses probabilistic inference. This helps in reasoning with incomplete and unknown information. The Bayesian Network is applied on data that would have acquired from a face image using Principal Component Analysis (PCA). PCA is used to reduce the dimensions of the image. The co-variances matrix from the principal components is used in the Bayesian network. The Bayesian network uses prior knowledge and the weights on the nodes are modified using weights formulated by a probability distribution function. This means, unlike other neural networks, the weights of a Bayesian network are not constants. We formulate our problem as the maximum a posteriori (MAP) estimate of a properly defined probability distribution function (PDF). A Bayesian network is used to represent the PDF as well as the domain knowledge needed for interpretation (Kumar and Desai, 1996). The development is done for Android-based mobile phones. The Android is an open source mobile phone operating system that runs on a Linux kernel. In this paper, we seek to improve face recognition for mobile devices by using dedicated statistical generative models. Gaussian Mixture Models (GMM) and Hidden Markov Models (HMM), which are classical generative models, have been successfully proven in face recognition. The GMM and HMM based models treat the facial features under investigation as independent. This is not consistent for a face because facial features are related to each other. Bayesian networks can be used to analyse and process the face as a graphical representation of dependent features. It only detects the frontal view of the face, that is, the face to be detected has to be showing the frontal features of the face. An appearance-based recognition and detection method is used. The prototype was built for Android-based mobile phones. Mobile phones which run on Symbian, Windows and Apple (iOS) operating systems are not going to be covered.*

Keywords: android mobile device, Bayesian network, digital camera, face recognition.

INTRODUCTION

Computer vision is a branch of computer science concerned with analysing images to extract information about the environment. Much like the human visual system, embedded computer vision systems perform these same visual functions in a wide variety of products. The device can be trained using feature-based or appearance-based models. Bayesian networks use features' causal relationships to create a full image structure. The resource optimisation ability that exists in Android-based devices makes them the preferred for the dissertation. Face recognition, as an authentication tool, involves accepting or rejection of a face by the testing application. This involves providing an image, known as a probe which is tested using known images, that is, gallery to determine whether the face is an impostor or not.

Face recognition is a field that has received so much attention from scholars and the manufacturers. This is due to the growth that has been realised in what the mobile phone can do. Google has made it possible to do a lot of things using their Android phones. Android is open source and runs on a Linux kernel. This means processing is more optimised in the Android-based mobile phones than others. Using its processing power and advantages inherent in the Linux kernel, it would be worthy to research on the use of Bayesian networks in the mobile phone. The Android market is growing and it is becoming one of the most used mobile phone operating systems. Recently, the Zimbabwe's most common mobile phone, GTide also rolled out a brand that runs on Android. Every transaction is coming to be done on a mobile phone due to its portability. This makes it of necessity that we have the best of biometric applications running on the phone to enhance security.

Computer vision algorithms require a lot of processing power and device memory. This leads to remote face recognition where the mobile phone sends the face for recognition by a server. The server runs the face recognition algorithm. This means the face recognition is only possible when a connection exists with the server. If the face is being used for authentication, then delays are expected during gallery search and matching. There has been growth in mobile phones processing speed and memory. This makes it possible to improve the performance of face recognition systems running on mobile phones. The face recognition system can be made locally implemented on the mobile phone due to their high processing speeds. Appearance-based algorithms can be used in the mobile phones rather than the simple and less efficient feature-based algorithms. High performance is ensured by Expectation Maximisation learning in Bayesian Principal Component Analysis (PCA). The system then uses the Junction tree inference for the face recognition. Bayesian networks create explicit relationships between facial features. The relationships can then be encoded through the network structure. When using the classical statistical models, facial features are treated as independent of each other.

This paper discusses the state of the art in the mobile and computer vision industry. We also present the most suitable methodology used to achieve the prototype design and the results. Models based on Bayesian Networks were used to bring the relationship between the facial features and thus improve on analysis. Better analysis would mean better recognition accuracy. The relationship of facial features can be improved by use of prior knowledge. For this purpose, Bayesian Networks provide an instinctive framework: they allow the encoding of causal relationships between different kinds of random variables. This enables us to express correlations between different sources of information. The correlations give the measure of similarity between faces.

LITERATURE REVIEW

Face detection is the interface between the probe and the system. The face is a highly non-rigid object; this is so because the same face can adopt different expressions. This makes it more difficult to detect the face than other rigid objects for the detection should cater for the non-rigid features. There are many other factors that make face detection a very difficult endeavour. Face detection can be easily affected by the following factors:

i. Pose (Out of Plane Rotation): the image may be rotated such that the frontal view is at an angle. The IMREN has to detect the face regardless of the pose.
ii. Presence or absence of structural components: this includes the presence of things like beard or eye spectacles. This can affect the detection of the face single such structures cause extra variation especially on the recognition. The *IMREN* should be able to detect a face even in the event of such structures.
iii. Facial expression: a frowning face changes the facial features. It does not only introduce wrinkles but the distances between the facial features may be slightly altered. The appearance of the face directly affects its detection.
iv. Occlusion: faces may be partially occluded by other objects and this means the system has to detect the face without a complete face.
v. Imaging conditions: this includes light intensity and distribution. The light source may also contribute to some variation in the appearance of the image. This problem can be reduced by having to change the images to gray-scale. This also helps because it reduces the colour pixels that the system has to compare and process. The camera sensors' characteristics are directly responsible of the image being processed.

The *IMREN* creates a conditional dependent relationship between facial features. The number of the facial features taken into consideration affects the number of nodes in the network. This means that is also increases the Bayesian network's complexity even though it improves its efficiency. The *IMREN* uses the features that are necessary for general face distinction. This involves eyes, nose, mouth and complexion.

Using the face as a biometric requires a camera that good resolution. This is mainly because we use the details of the image for authentication and that can be risky. For us to get enough detail, that means we would require cameras which have very high resolution. However, it is not always the case that high resolution leads to good face recognition. This is so because high resolution means more pixels per area and this means more computation. The high the computation the more our algorithms become prone to error. This leaves us in a fix. In this, we recommend a design of camera sensors that have clustered pixel processing. This would mean the camera sensor processing the pixels as 2D arrays. This makes the processing faster and more efficient even for the algorithms which process the image. Processing images in form of pixel arrays means less processing by the image processing algorithms.

The mobile phone has moved from being a mere calling machine to a multi-purpose tool. From social secrets to banks accounts details, security concerns may be a beneficiary of the face recognition biometric. This is particularly true as people begin using these phones to replace laptops and doing more and more of their financial transactions on their phones (Vercillo, 2009).

For an application to do face recognition, it should be able to detect a face from a provided image, known as a probe. The application should be able to detect a face and separate if from its background. There can be more than one face in the image and the application should detect all the faces. This requires that the application be trained to know how faces look like. The face detection methods are generalised in four classes:

i. Knowledge-based methods

This method encodes the human knowledge of what constitutes a typical face. This is usually by using the relationship between facial features. These methods are designed mainly for face localization.

ii. Feature invariant approaches

The approach aims at finding structural features of a face that exist even when the pose, viewpoint or lighting conditions vary.

iii. Template matching methods

This is when several standard patterns are stored to describe the face as a whole or the facial features separately.

iv. Appearance-based methods

This is the approach we used in the prototype developed in this research. In contrast to template matching, the models (or templates) are learned from a set of training images which should capture the representative variability of facial appearance. These learned models are then used for detection. These methods are designed mainly for face detection. The appearance-based methods rely on techniques from statistical analysis and machine learning to find the relevant characteristics of face and non-face images. The learned characteristics are in the form of distribution models or discriminate functions that are consequently used for face detection. Meanwhile, dimensionality reduction is usually carried out for the sake of computation efficiency and detection efficacy. Many appearance-based methods can be understood in a probabilistic framework. An image or feature vector derived from an image is viewed as a random variable x, and this random variable is characterized for faces and non-faces by the class-conditional density functions p (x|face) and p (x|nonface). Bayesian classification or maximum likelihood can be used to classify a candidate image location as face or non-face. Unfortunately, a straight forward implementation of Bayesian classification is made impossible by the high dimensionality of x. This is the problem that is solved by dimension reduction (discussed in the above sections) (Sebe, et al, 2005).

The photometric stereo technique consists of obtaining several pictures of the same subject in different illumination conditions and extracting the 3D geometry by assuming a Lambertian reflection model. Although a minimum of only three images with non coplanar light sources are needed for photometric stereo, more images can improve results in several ways. We assume that the facial surface, represented as a function, is viewed from a given position along the z-axis. (Bronstein, 2003)

Photometric stereo recovers per-pixel estimates of surface orientation from images of a surface under varying lighting conditions. Transforming reflectance based on recovered normal directions is useful for enhancing the appearance of subtle surface detail. A high-speed video camera, computer controlled light sources and fast GPU implementations of the algorithms enable achievement of real-time photometric stereo and reflectance transformation. By applying standard image processing methods to the computed normal image, the surface detail can be enhanced at different frequencies. Real-time analysis of surface roughness for metrology can also be performed from the extracted normal field.
Many high-speed video cameras can capture images at 1000 frame per seconds or more, and they have already been used for analyzing high-speed phenomenon. Nevertheless, in many fields, it is necessary to observe high-speed phenomena as three-dimensional shapes at a high-speed frame rate; however, most of the high-speed video cameras can only record high-speed phenomena as two-dimensional image sequences (Shigeyuki, 2007).

A pixel in the image plane defines the viewing vector s. The vector direction determines the stripe direction vector v', lying in both the image plane and in the viewing plane. The real tangential vector of the projected stripe v_1 is perpendicular to the normal $c = v' * s$ of the viewing plane and to the normal p of the stripe projection plane. Assuming parallel projection, we obtain

$$v_1 = c * p$$

Acquiring a second image of the scene with a rotated stripe illumination relative to the first one, allows calculating a second tangential vector v_2. Next, the surface normal is computed according to

$$n = v_1 * v_2$$

Winkelbach and Wahl propose to use a single lighting pattern to estimate the surface normal from the local directions and widths of the projected strip

The design of the Bayesian networks are defined by acyclic directed graphs (ADG) in which nodes represent random variables and arcs/edges represent direct probabilistic dependences among them. The edges induce a factorization of the joint probability distribution. There are no feedback loops in the graph. The network gives a qualitative illustration of the interactions among the set of variables that it models. The structure of the directed graph can mimic the causal structure of the modelled domain. When the structure is causal, it gives a useful, modular insight into the interactions among the variables and allows for prediction of effects of external manipulation. A Bayesian network specifies a set of variables and a factorization of the joint probability distribution in a very straightforward way. Let the variables in the network be denoted by capital letters, for example, X_i, and their corresponding specific values by lower case letters, for example, x_i. A variable has a predecessor within a network. Predecessors are also known as parents. Suppose that the immediate parents of a variable X_i are $X_{\pi(i)}$ and that the values of the parents are $x_{\pi(i)}$. Then the joint distribution is factored as

$$P(X_1 = x_1, X_2 = x_2, \dots X_n = x_n) = \prod_i P\left(X_i = x_i \middle| X_{\pi(i)} = x_{\pi(i)}\right)$$

We refer to a variable and its immediate predecessors as a family in the graph. There two distinct types of variables in a Bayesian network graph, that is, the observed variables and the hidden variables. The observed variables are also known as observations. These are the physical quantities of the network. The values of the observations are known but hidden variables are not known. In the case that hidden variables are present, one may still reason in concrete terms by assuming various assignments of values to the hidden variables to obtain concrete data cases. Concrete variable assignments are the key to the inference algorithms. Inference is ensured by finding the single likeliest assignment to all the hidden variables in a network, and training involves summing over all possible variable assignments (Guestrin, 2007).

In order to get numerical values for specific assignments of values to the variables, it is necessary to define concrete conditional probability functions, and here again there is a wide degree of flexibility. Some of the typical functions include tabular representations for discrete variables, and Gaussian distributions for real-valued variables. The set of operations that can be done with a graph depends on the types of variables in the graph, and the conditional probability functions that are used. In particular, the presence of hidden real-valued variables significantly complicates the associated algorithms and in some cases makes them computationally intractable. However, stochastic simulation methods are usually used to provide fast approximation to the required probabilities. The *IMREN* uses evidence weighting. Evidence weighting weights each sample with the likelihood associated with the observed evidence (Chang and Tian, 2002).

The network has to be computing the following (Heckerman, et al, 1995):

i. The probability of the observed variable values: $P(o) = \sum_h P(o,h)$
ii. The likeliest hidden variable values: $argmax_h P(o,h)$
iii. Parameter estimation from the set of observations $\{o_k\}$ via EM: $argmax_\theta \prod_k P(o_k|\theta)$; where o are the observed variables and h are the hidden variables.

The Bayesian Networks framework allows designing generative models capable of encoding correlations between different features extracted from a face image. Actually, we would like to consider that certain features (i.e. observations) are generated by the same *causes* (Einhauser, 2007). Consider the feature vectors corresponding to the eyes for instance. It is very likely that these two observations were generated by the same hidden cause. At least, considering that these features are completely independent is, in our opinion, not true.

Once a Bayesian network is constructed, it can be used to make inferences about the variables in the model. The conditionals given in a Bayesian network representation specify the *prior* joint distribution of the variables (Shenoy and Nadkarni, 2004). If we observe (or learn about) the values of some variables, then such observations can be represented by tables where we assign 1 for the observed values and 0 for the unobserved values. Then the product of all tables (conditionals and observations) gives the (un-normalized) *posterior* joint distribution of the variables. Thus, the joint distribution of variables changes each time we learn new information about the variables.The calculation of probabilities is called inference (Ghahramani, 2001). Inference is an important task in Bayesian Networks. It consists in computing probabilities of interest, once evidence has been entered into the network (i.e. when one or more variables have been observed).

RESEARCH METHODOLOGY

Low-fidelity prototyping were used at the preliminary stages of the user interface design. This is when the user interface screen is drawn on paper using a pen. It is a natural way of viewing the screen and easy to understand. This can then be done using computer-aided design (CAD). This is known as high-fidelity prototyping. Prototypes make a software made using RAD to have smooth functionality and easy-to-use interface. System was analysed in the light of its functional and non-functional requirements.

Research Framework

A face recognition system is made up of two main subsystems; one for face detection and the other for the actual recognition. Face detection is for identification of a face in an image. Certain images have more than one face on it, making it difficult for recognition when the recognition uses only one face. The image used as a probe may have a complicated background that makes it difficult to isolate the facial features from the rest of the image. The *IMREN* uses a classifier trained using thousands of faces for face detection. For a more efficient analysis, the face images should go through dimensionality reduction. This is to make sure that the dimensions presented for processing are compact. Principal Component Analysis (PCA) is used for dimensionality reduction is the *IMREN*.

Research Methods

The software development methodology involves iterative development and the construction of prototypes. Rapid application development is a term originally used to describe a software development process introduced by James Martin in 1991. RAD is for fast system development. This reduces the cost of production by reducing the time dedicated to the development. A balance should be found on speed and system quality even at a relatively low investment cost. It emphasizes on the business need delivery and software engineering excellence is not prioritized. The methodology also emphasizes on the deadlines, known as “time boxes”. System requirements can be streamlined to reduce development time without moving the deadline. This is done without losing the fundamentals of the system under development. There is software component reuse in using RAD. The primary disadvantage of component is the possibility of inheriting problems from the old software. It also presents maintenance challenge in the case of a lack of expertise.

The other basic principles are:

i. Top-down approach is used to solve problem. The problem is broken down into smaller problems. This makes the problem analysis easier and faster.
ii. Attempts to reduce inherent project risk by breaking a project into smaller segments and providing more ease-of-change during the development process. An object oriented approach is then adopted in which the segments are viewed and programmed as weakly coupled entities.
iii. Generally includes joint application design (JAD), where users are intensely involved in system design, via consensus building in either structured workshops, or electronically facilitated interaction.
iv. Produces documentation necessary to facilitate future development and maintenance.

Standard systems analysis and design methods can be fitted into this software process method.

PROTOTYPE RESULTS

The face image is taken from a camera. Camera is used as the image input devices. This means the system has to be able to process images directly from the camera. The user may want to train the Bayesian network using the images being taken. This means each image being captured in input to some Bayesian network training program. The user may just use the camera for a probe and then do the face recognition. Figure 1 below show the architecture of the *IMREN.*

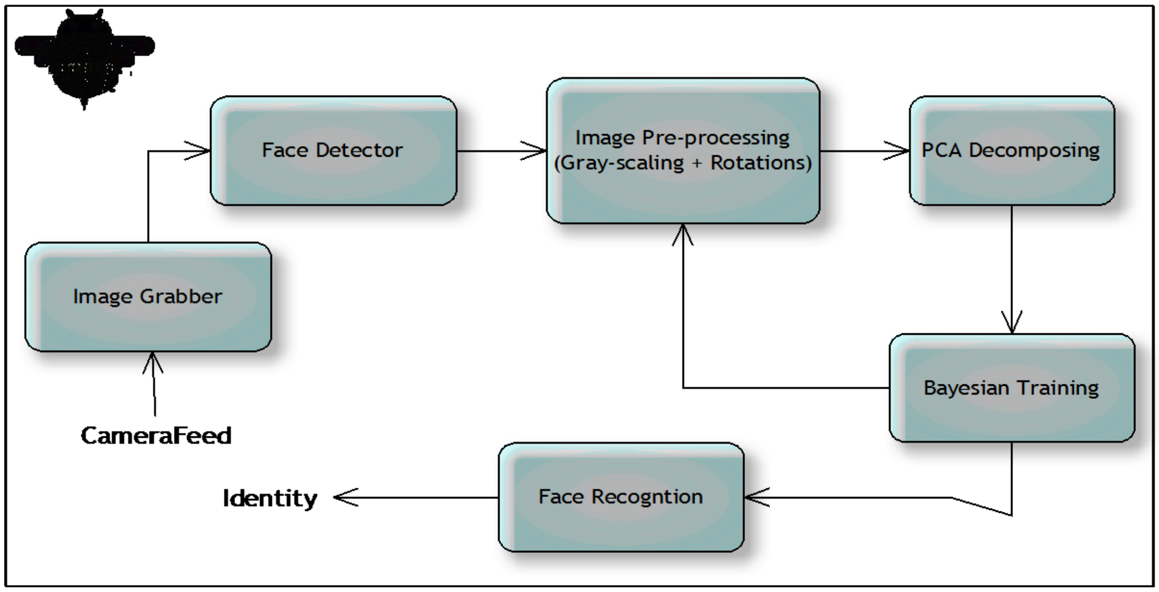

Figure 1 : The IMREN System Architecture

The *IMREN* is consists of six major processing modules. The flow of control is shown the diagram above. The Image Grabber module is responsible for making sure that the image is captured from the camera preview. After the image has been captured, the system “looks” for a face on the image. This is done by the face detector, which returns the user to the camera (by exiting) if a face is not detected. The detector then passes the 2D face image to the Pre-processing module. The pre-processing is for ensuring that there is uniformity on the face images being input. This involves gray-scaling the picture so that there is no lighting problems on the images. The processed image is decomposed into an eigenface space by using Principal Component Analysis (PCA). Bayesian networks can either be data-based or knowledged-based.

A Bayesian network is then trained according to the data in the eigenfaces space. This is known as Bayesian PCA

The *IMREN* system was implemented using Java(TM) language. The Java is used as JavaCV, which is an OpenCV wrapper. It allows the use of C/C++ classes in OpenCV using Java. This is done through JavaCPP. JavaCPP provides efficient access to native C++ inside Java, not unlike the way some C/C++ compilers interact with assembly language. JavaCPP replaced the use of JNI. This works with all the Java implementations including Android. The other approaches are SWIG, JNIWrapper, Platform Invoke, JNI Direct, JNA, JniMarshall, J/Invoke, HawJNI and BridJ. JavaCPP supports many features of the C++ language often considered problematic, including overloaded operators, template classes and functions, member function pointers, callback functions, nested struct definitions, variable length arguments, nested namespaces, large data structures containing arbitrary cycles, multiple inheritance, passing/returning by value/reference/vector, anonymous unions, bit fields, exceptions, destructors and garbage collection. JavaCPP interfaces with a lot of other classes that can be used in Android. JavaCV is therefore highly compatible with Android. Some researchers even say Java is equivalent to Android and this has led to a lot of handshaking between Java and Android. The application is compiled by the Eclipse's Java compiler. Eclipse has the Java Development Tools (JDT) Core component which has the incremental Java compiler. The JDT project contributes a set of plug-ins that adds the capabilities of a full-featured Java IDE to the Eclipse platform. The JDT plug-ins provides APIs so that they can themselves be further extended by other tool builders. The compiling produces standard Java byte code (*.class* files). These files are then converted into Android's *.dex*. The conversation is lossless; this is because Java and Android have equivalent byte-code/class formats. The Dalvik virtual machine implementation in Android is similar to the Java Virtual Machine.

The *IMREN* used a trained classifier for face detection. The classifier was trained using a thousand faces. The classifier data is stored as an xml file. In this case, the classifier is stored in an OpenCV 2.2 folder. The path is passed to the Loader.extractResource() method. This checks if there are facial features in the image.

The human can be taken as a combination of dependent features. The features are related to each other and their relationship is defined to form a single face. The characteristics of a face are a function of the properties of the singular features. Two different faces can have the similar nose features. The BN should be able to use the tree network structure to attach a nose to a mouth in an indirect dependent relationship. An indirect dependent relationship is when the nodes are connected via another node.
Bayes' Probability theory: Mathematical Foundation
Mathematically, Bayes' rule states

$$\boldsymbol{posterior} = \frac{\boldsymbol{likelihood} \times \boldsymbol{prior}}{\boldsymbol{marginal\ likehood}} \qquad (3.1)$$

Or, in symbols,

$$\mathbf{P(R = r|e) = \frac{P(e|R=r)P(R=r)}{P(e)}} \quad (3.2)$$

,

where P(R=r|e) denotes the probability that random variable R has value r given evidence e.

The denominator is just a normalising constant that ensures the posterior adds up to 1; it can be computed by summing up the numerator over all possible values of R, that is, P(e) = P(R=0, e) + P(R=1, e) + ... = sum_r P(e | R=r) P(R=r).This is called the marginal likelihood (since we marginalize out over R), and gives the prior probability of the evidence. The *likelihood* function is the conditional density of the observations given a true value of an observation. According to the Likelihood Principle (LP), the likelihood function contains all the information brought by the observations, e, about the quantity, r. The *prior* is the density function of the quantity r. It is called a prior since it quantifies our belief or knowledge about r before observing the measurements. Through the prior, external knowledge about the quantity r can be incorporated into the estimation problem. The marginal likelihood is also known as the density function of the observation. This can be assumed constant after observing the data.

Thus, the posterior density can be written as,

$$P(R = r|e) \propto P(e|R = r)P(R = r) \text{ OR Posterior} \propto \text{Likelihood} \times \text{Prior} \quad (3.3)$$

This is referred to as the unnormalised posterior, for the density function or the marginal likelihood was our normalising constant. Thus, the posterior combines the data information and any external information. Having constructed the posterior, a sample from it is selected as the final Bayesian estimate of the quantity r. Contrary to non-Bayesian or frequentist approaches, which rely only the data for inference, Bayesian approaches combine the information brought by the data and any external knowledge represented by the prior to provide improved estimates.
After detecting the face, the face should be captured for the training or detection. This requires some rigorous pre-processing since the picture should be decomposed. The pixel colour components should be analysed. Consider a thumbnail of 18x18 (324 pixels) face pattern. There are 256^{324} possible combinations of gray values. This means $2^{8*324}=2^{2592}$ combinations. The dimensions of the face have to be reduced to facilitate processing. Principal Component Analysis was used for this.

Extract relevant information in a face image [Principal Components] and encode that information in a suitable data structure. In mathematical terms we want to find eigen vectors and eigen values of a covariance matrix of images. Where one image is just a single point in high dimensional space [n * n], where n * n are the dimensions of an image. There can be many eigen vectors for a covariance matrix but very few of them are the principle one's. Though each eigen vector can be used for finding different amount of variations among the face image. But we are only interested in principal eigen vectors because these can account for substantial variations among a bunch of images. They can show the most significant relationship between the data dimensions. Eigenvectors with highest eigen values are the principle component of the Image set. We may lose some information if we ignore the components of lesser significance. But if the eigen values are small then we won't lose much. Using those set of eigen vectors we can construct eigenfaces. The following steps show how we obtain egein faces from a sample of face images:

i. Collect a bunch [say 10] of sample face images. Dimensions of all images should be same. An image can be stored in an array of n*n dimensions [**Γ**] which can be considered as an image vector.

   $\{\Gamma_i | \boldsymbol{i} = \mathbf{1}, \dots, \boldsymbol{M}\}$ **,** where M is the number of images.

ii. Find the average image of bunch of images.

$$\Psi = \frac{\mathbf{1}}{\boldsymbol{M}} \sum_{\boldsymbol{i}=\mathbf{1}}^{\boldsymbol{M}} \Gamma_{\boldsymbol{i}} \quad (3.4)$$

iii. Find the deviated [avg - img1, avg - img2, ..., avg - img.n] images .

$$\boldsymbol{\Phi}_{\boldsymbol{i}} = \Gamma_{\boldsymbol{i}} - \Psi; \boldsymbol{i} = \mathbf{1}, \dots, \boldsymbol{M} \quad (3.5)$$

iv. Calculate the covariance matrix.

$C=AA^T$, **(3.6)**

$$\mathbf{C} = \begin{pmatrix} \mathbf{c(1,1)} & \cdots & \mathbf{c(1,d)} \\ \vdots & \ddots & \vdots \\ \mathbf{c(d,1)} & \cdots & \mathbf{c(d,d)} \end{pmatrix}$$

Where $A = [\Phi_1 \cdots \Phi_M]$;

But the problem with this approach is that we may not be able to complete this operation for a bunch of images because covariance matrix is very huge. For Example, Covariance matrix, where dimension of a image = 256 * 256, should consists of [256 * 256] rows and same numbers of columns. So it is very hard or may be practically impossible to store that matrix and also seeing that the matrix requires a lot of computational resources.
So for solving this problem we can first compute the matrix L.

$$\mathbf{L} = [\mathbf{A}^{\mathbf{T}}\mathbf{A}] \qquad (3.7)$$

And then find the eigen vectors [v] related to it

$$\mathbf{v}_l(l = 1, \ldots, M)$$

Eigen Vectors for Covariance matrix C can be found by

$$\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_M] = [\mathbf{\Phi}_1, \ldots, \mathbf{\Phi}_M] = [\mathbf{v}_1, \ldots, \mathbf{v}_M] = \mathbf{A} \cdot \mathbf{V} \qquad (3.8)$$

, where $\mathbf{u}_l(\mathbf{l} = \mathbf{1}, \ldots, \mathbf{M})$ are the Eigen Vectors for C.

Using these eigenvectors, we can construct eigenfaces. But we are interested in the eigenvectors with high eigenvalues. So eigenvectors with less than a threshold eigenvalue can be dropped .So we keep only those images which correspond to the highest eigenvalues. This set of images is called as face space.
The same paradigm also leads us into considering that all the observations extracted from the same face image are generated by the same identity. The face is the root node in our model and has no parent. The diagram below shows the conceptual model.

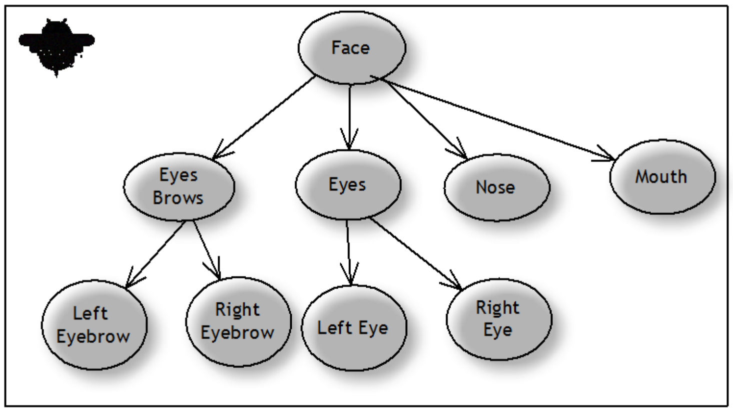

Figure 2: The Face Bayesian Network Conceptual Model

From the model above, it can be deduced that there are two levels in the network. The bottom level consists of the features that are observed on the face. These are the observed variables of the networks. The second level (linked to the root node) has the nodes that represent the hidden variables. These are the variables which cause the observed variables. For instance the eye causes two children, that is, the left eye and the right eye. the design done, our network has 3 layers. The Face is the root node. The network was designed using causal relationship approach. The network has one hidden. The hidden layer is made up of the non-observed variables. The observed variables are the children nodes. The values of these variables change the posterior probability. A decrease in the posterior probability is a reduction in the belief of the network in prediction. The Bayesian network makes sure we can still reason even when there are missing variables. The combination of PCA and Bayesian networks is the Bayesian PCA and is using probabilities in contrast to mere values in PCA. The Junction tree was used for the inference. When training the network, the weights are update by a probability distribution function.

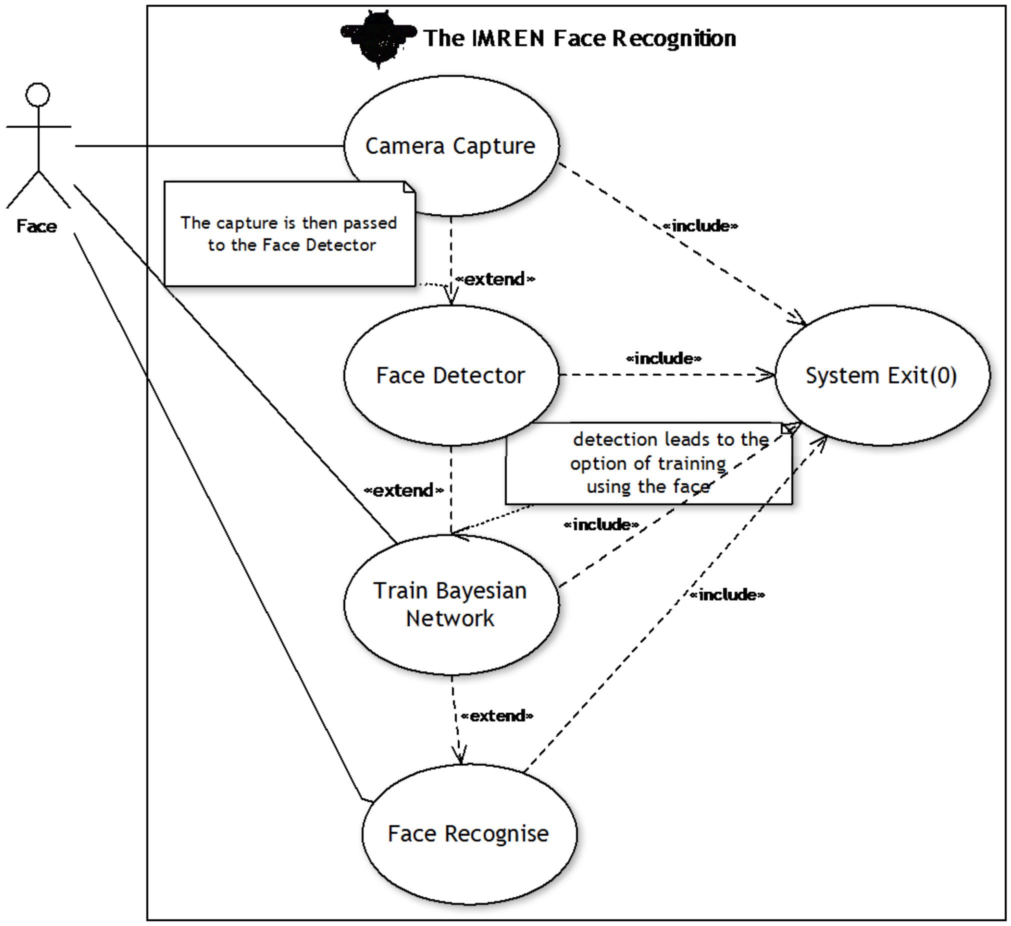

Figure 3: The IMREM Face Recognition Use Case Diagram

The chief of the non-functional requirements being that the system is supposed to take reasonable time to respond to the user's request. The system is to be easy to use and query. The analysis on the Bayesian network and PCA seeks to make sure the system perform and fulfil it functional requirement, chief being face recognition. In the other cases where Eigenfaces have been applied for face recognition, Euclidean Distances have been used. This is when the face is recognised as its nearest neighbour inside an image space.

CONCLUSION

The prototype used in this research uses the appearance-based method of face detection. This means face templates were used to train the system on variability of facial appearance. Much of the work on computer vision has been done to run on the resourceful computers as opposed to smaller mobile phones. This has been regardless of the tremendous growth in processing speed and memory size that the mobile phones have enjoyed in recent years. It is in this light that this research seeks to show the feasibility of using neural networks for computer vision in mobile phones.

REFERENCES

[1.] Chang, K and Tiang, Z. 2002, *Efficient Inference for Mixed Bayesian Networks*, viewed 5 May 2011, http://isif.org/fusion/proceedings/fusion02CD/pdffiles/papers

[2.] Einhauser, B. 2007, 'A Guide to Computer Vision', *Image Analysis with Statistical Models*, viewed 26 May 2011, <www.staff.uni-marburg.de/~einhaeus/IASM07/IASM07_l1.pdf >.

[3.] Ghahramani, Z., 2001, 'International Journal of Pattern Recognition and Artiificial Intelligence', *An Introduction to Hidden Markov Models and Bayesian Networks* , pp. 9-42, viewed 13 May 2011, < http://www.gatsby.ucl.ac.uk/~zoubin/papers/ijprai.ps.gz >.

[4.] Guestrin, C. 2007, *Bayesian Networks (Structure) Learning*, Machine Learning Department and Computer Science Department Carnegie Mellon University, viewed 16 April 2011, < www.cs.cmu.edu/~guestrin/Class/10701/.../structurelearn-clustering.pdf >.

[5.] Heckerman, D. , Mamdani, A. and Wellman, M. 1995, 'Real World Applications of Bayesian Networks', *Communication of the Acc*, vol 38, no. 3, pp. 24-68, viewed 20 March 2011, < www-ksl.stanford.edu>

[6.] Kumar, S.K. and Desai U.B. 1996, '3rd IEEE International Conference on Image Processing', *Image Interpretation using Bayesian Networks,*vol 18, no. 1, pp. 74-77, viewed 15 March 2011, <citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.18.4655>

[7.] Sebe N., Tian Q., Loupias, E., Lew, M. and Huang, T.S, 2003, "Evaluation of Salient Points Techniques", *Image and Vision Computing, Special Issue on Machine Vision*, Vol. 21, No. 13-14, pp. 1087-1095, viewed 15 May 2011, < http://www.ifp.illinois.edu/~qitian/e_paper/ivc/ivc.pdf>

[8.] Shigeyuki, O., Sato M., and Ishii S. 2003, 'ICANN/ICONIP '03 on Artificial Intelligence and neural information processing', *Prior hyperparameters in Bayesian PCA*, vol. 2714, pp. 271 -279, viewed 2 May 2011, <www.springerlink.com/index/bafpvvfw8hcyyrrd.pdf>

[9.] Vercillo, K. 2009, *Mobile Phone Trends You May Want To Jump on the Bandwagon For*, Hubspages, viewed 2 April 2011, http://hubpages.com/hub/Mobile-Phone-Trends-You-May-Want-To-Jump-on-the-Bandwagon-For