Face Recognition

Somay Jain
201101054, Bachelor of Technology, CSE
IIIT Hyderabad
somay.jain@student.iiit.ac.in

Tanuj Sharma
201101138, Bachelor of Technology, CSE
IIIT Hyderabad
tanuj.sharma@student.iiit.ac.in

***Abstract* — In this report we describe a face recognition method based on PCA (Principal Component Analysis). The method consists of two steps – first, we project the face image from the original vector subspace via PCA, second we use KNN and SVM to classify the images. Using Yale, CMU-PIE and SMAI-students dataset we demonstrate a significant improvement when principal components rather than original images are fed to KNN classifier. The hybrid classifier using PCA and KNN/SVM provides a useful framework for image recognition tasks.**

Keywords—face recognition, PCA, SVM, K-NN, ROC curve

I. Introduction

The problem of automatic face recognition is a composite task that involves detection and location of faces in a cluttered background, normalization, recognition and verification. In modern face recognition requirements like speed and accuracy are more important. Assuming that segmentation and normalization of images have been done, we focus on the subtask of person recognition and verification and demonstrate the performance using YALE, CMU-PIE and SMAI student dataset.

There have been many methods proposed for face recognition [1] [2]. In this report we follow closely the paper published by Matthew Turk and Alex Pentland [1]. The scheme discussed by paper is based on an information theory approach that decomposes face images into a small set of characteristic feature images called “eigenfaces”, which may be thought of as the principal components of the initial training set of face images. Recognition is performed by projecting a new image into the subspace spanned by the eigenfaces (“face space”) and then classifying the face by comparing its position in face space with the positions of known individuals.

The Principal Component Analysis (PCA) is one of the most successful techniques that have been used in image recognition and compression. PCA is a statistical method under the broad title of factor analysis. The purpose of PCA is to

Background and Related Work

Within the last several years, numerous algorithms have been proposed for face recognition. While much progress has been made toward recognizing faces under small variations in lighting, facial expressions and pose, reliable techniques for recognition under extreme variations have proven elusive.

Much of the work in face recognition in early years has focused on detecting individual features such as the eyes, mouth, nose and head outline. Such approaches have proven difficult to extend to multiple views, and are fragile.

Others have approached automated face recognition by characterizing a face by a set of geometric parameters and performing pattern recognition based on these parameters. It involves calculation of a set of facial parameters from a single image, and using a pattern classification technique to match the face from a known set, a purely statistical approach depending primarily on local histogram analysis and gray-scale values.

II. Methods Used

Recognition portrayed in this report first reduces the dimension of training set of face images using PCA and then classification is done using KNN and SVM.

A. PCA

The Principal Component Analysis (PCA) is one of the most successful techniques that have been used in image recognition and compression. PCA is a statistical method under the broad title of factor analysis. The purpose of PCA is to reduce the dimensionality of feature space (independent variables), which are needed to describe the data economically. This is the case when there is a strong correlation between observed variables. The main idea of using PCA for face recognition is to express the large 1-D vector of pixels constructed from 2-D facial image into the compact principal components of the feature space. This can be called eigenspace projection. Eigenspace is calculated by identifying the eigenvectors of the covariance matrix derived from a set of facial images (vectors).

Mathematics of PCA (in face recognition) –

A 2D facial image can be represented as 1-D vector by concatenating each row (or column) into a long thin vector. Let’s suppose we have M vectors of size N (= rows of image * columns of image) representing a set of sampled images. *pj*'s represent the pixel values.

$$x_i = [\, p_1 \quad \cdots \quad p_N \,]^T, \qquad i = 1, \ldots\ldots, M$$

The images are mean centered by subtracting the mean image from each image vector. Let m represent the mean image.

$$m = \frac{1}{M} \sum_{i=1}^{M} x_i$$

And let w_i be defined as mean centered image

$$w_i = x_i - m$$

Our goal is to find a set of e_i's which have the largest possible projection onto each of the w_i's. We wish to find a set of M orthonormal vectors e_i for which the quantity

$$\lambda_i = \frac{1}{M} \sum_{n=1}^{M} (e_i^T\, w_n)^2$$

is maximised with the orthonormality constraint

$$e_l^T e_k = \delta_{lk}$$

It has been shown that the ei's and Λi's are given by the eigenvectors and eigenvalues of the covariance matrix

$$C = WW^T$$

Where W is a matrix composed of the column vectors w_i placed side by side. The size of C is $N \; X \; N$ which could be enormous. It is not practical to solve for the eigenvectors of C directly. A common theorem in linear algebra states that the vectors e_i and scalars λ_i can be obtained by solving for the eigenvectors and eigenvalues of the $M \; X \; M$ matrix W^TW. Let d_i and l_i be the eigenvectors and eigenvalues of W^TW, respectively.

$$W^T W d_i = \mu_i \,(\, W\, d_i \,)$$

Which means that the first $M - 1$ eigenvectors e_i, and eigenvalues λ_i of WW^T are given by Wd_i and μ_i, respectively. Wd_i needs to be normalized in order to be equal to e_i. Since we only sum up a fininte number of image vectors M, the rank of covariance matrix cannot exceed $M - 1$.

The eigenvectors corresponding to nonzero eigenvalues of the covariance matrix produce an orthonormal basis for the subspace within which most image data can be represented with a small amount of error. The eigenvectors are sorted from high to low according to their corresponding eigenvalues. The eigenvector associated with the largest eigenvalue is one that reflects the greatest variance in the image. That is, the smallest eigenvalue is associated with the eigenvector that finds the least variance. They decrease in exponential fashion, meaning that the roughly 90% of the total variance is contained in the first 5% to 10% of the dimensions.

A facial image can be projected onto M' (<< M) dimensions by computing

$$\Omega = [\, v_1 \; v_2 \; v_3 \ldots\ldots\ldots v_{M'} \,]^T$$

Where $v_i = e_i^T w_i$. v_i is the i^{th} coordinate of the facial image in the new space, which came to be the principal component. The vectors e_i are also images, so called *eigenimages*, or *eigenfaces* in our case, which was first named by [1]. They can be viewed as images and indeed look like faces. So, Ω describes the contribution of each eigenface in representing the facial image by treating the eigenfaces as a basis set for facial images. The simplest method for determining which face class provides the best description of an input facial image is to find the face class k that minimizes the Euclidean distance.

$$\epsilon_k = \| \Omega - \Omega_k \|$$

Where Ω_k is a vector describing the kth face class. If ϵ_k is less than some predefined threshold θ, a face is classified as belonging to the class k.

B. *Support Vector Machine Classification*

In machine learning, support vector machines (SVMs) are supervised learning models with associated learning algorithms that analyze data and recognize patterns, used for classification and regression analysis. The basic SVM takes a set of input data and predicts, for each given input, which of two possible classes forms the output, making it a non-probabilistic binary linear classifier.

1. *Binary Classification*

SVMs belong to SVMs belong to the class of maximum margin classifiers. They perform pattern recognition between two classes by finding a decision surface that has maximum distance to the closest points in the training set which are termed support vectors. We start with a training set of points $x_i \in R^n$, where each point x_i belongs to one of two classes identified by the label $y_i = \{-1, 1\}$. Assuming linearly separable data1, the goal of maximum margin classification is to separate the two classes by a hyperplane such that the distance to the support vectors is maximized. This hyperplane is called the optimal separating hyperplane (OSH). The OSH has the form:

$$f(x) = \sum_{i=1}^{l} \alpha_i \, y_i \, x_i \, . \; x + b$$

The coefficients α_i and the b in the above equation are the solutions of a quadratic programming problem. Classification of a new data point x is performed by computing the sign of right side of thr above equation. /in the following we will use

$$d(x) = \frac{\sum_{i=1}^{l} \alpha_i y_i x_i \, . \; x + b}{\left\| \sum_{i=1}^{l} \alpha_i y_i x_i \right\|}$$

to perform multi-class classification. The sign of d is the classification result for **x**, and **|d|** is the distance from **x** to the hyperplane. Intuitively, the farther away a point is from the decision surface, i.e. the larger **|d|** , the more reliable the classification result.

The entire construction can be extended to the case of nonlinear separating surfaces. Each point **x** in the input space is mapped to a point $z = \Phi(x)$ of a higher dimensional space, called the feature space, where the data are separated by a hyperplane. The key property in this construction is that the mapping Φ(.) is subject to the condition that the dot product of two points in the feature space Φ(x). Φ(y) can be rewritten as a kernel function K(x,y) .
The decision surface has the equation :

$$f(x) = \sum_{i=1}^{l} \alpha_i \, y_i \, x_i \, . \, x + b$$

Again, the coefficients α_i and b are the solutions of a quadratic programming problem. Note that f(x) does not depend on the dimensionality of the feature space.
An important family of kernel functions is the polynomial kernel :

$$K(x, y) = (1 + x.y)^T$$

Where *d* is a degree of polynomial. In this case the components of the mapping Φ(x) are all the possible monomials of input component upto degree d.

2. *Multi Class Classification*

There are two basic strategies for solving q-class problems with SVMs:

i) In the one-vs-all approach SVMs are trained. Each of the SVMs separates a single class from all remaining classes.
ii) In the pairwise approach *q(q–1)/2* machines are trained. Each SVM separates a pair of classes. The pairwise classifiers are arranged in trees, where each tree node represents an SVM. A bottom-up tree similar to the elimination tree used in tennis tournaments was originally proposed in for recognition of 3-D objects and was applied to face recognition.

There is no theoretical analysis of the two strategies with respect to classification performance. Regarding the training effort, the one-vs-all approach is preferable since only q SVMs have to be trained compared to *q(q–1)/2* SVMs in the pairwise approach. The run-time complexity of the two strategies is similar: The one-vs-all approach requires the evaluation of *q – 1* SVMs. Recent experiments on person recognition show similar classification performances for the two strategies. Since the number of classes in face recognition can be rather large we opted for the one-vs-all strategy where the number of SVMs is linear with the number of classes.

C. *K Nearest Neighbours Classification*

k-nearest neighbors algorithm (k-NN) is a non-parametric method for classification and regression, that predicts objects' "values" or class memberships based on the k closest training examples in the feature space. k-NN is a type of instance-based learning, or lazy learning where the function is only approximated locally and all computation is deferred until classification. The k-nearest neighbor algorithm is amongst the simplest of all machine learning algorithms: an object is classified by a majority vote of its neighbors, with the object being assigned to the class most common amongst its k nearest neighbors (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of that single nearest neighbor.

The training examples are vectors in a multidimensional feature space, each with a class label. The training phase of the algorithm consists only of storing the feature vectors and class labels of the training samples.

In the classification phase, *k* is a user-defined constant, and an unlabeled vector (a query or test point) is classified by assigning the label which is most frequent among the *k* training samples nearest to that query point.

The best choice of k depends upon the data; generally, larger values of k reduce the effect of noise on the classification, but make boundaries between classes less distinct. A good k can be selected by various heuristic techniques. The special case where the class is predicted to be the class of the closest training sample (i.e. when k = 1) is called the nearest neighbor algorithm.

The accuracy of the k-NN algorithm can be severely degraded by the presence of noisy or irrelevant features, or if the feature scales are not consistent with their importance. In binary (two class) classification problems, it is helpful to choose k to be an odd number as this avoids tied votes. One popular way of choosing the empirically optimal k in this setting is via bootstrap method.

III. FACE RECOGNITION

Once the eigenfaces have been computed, several types of decision can be made depending on the application. What we call face recognition is a broad term which may be furthered specified to one of the following tasks :

- *Identification* where the labels of individuals must be obtained
- *Reconstruction* where given badly constructed input image, output the reconstructed image based on the training data set

- *Verification* where given a input image and a claimed class of it, output “Yes” if given image belongs to the given class, else “No”.

A. Identification

The eigenfaces images calculated from the eigenvectors span a basis set with which to describe face images.

Since the eigenfaces seem adequate for describing face images under very controlled conditions, a smaller M’ is sufficient for identification, since accurate reconstruction of image is not a requirement.

Given a N X N image data set, M images, and k eigenfaces u_k, a new face image τ is transformed into its eigenface components by a simple operation,

$$\omega_k = \mu_k^T\ (\tau - \psi), \quad where\ \psi\ is\ the\ mean\ image$$

For *k = 1,…., M’* . This describes a set of point by point image multiplications and summations, operations performed at approximately frame rate on current image processing hardware.

The weights form a vector $\Omega^T = [\ \omega_1, \omega_2, \omega_3, \dots\ \omega_{M'}\ ]$ That describes the contribution of each eigenface in representing the input face image, treating the eigenfaces as a basis set for face images. The vector may then be used in a standard pattern recognition algorithm to find which of a number of predefined face classes, if any, best describes the face. The simplest method for determining which face class provides the best description of an input face image is to find the face class *k* that minimizes the Euclidean distance

$$\epsilon_k^2 = \ ||\Omega - \Omega_k||^2$$

Where Ω_k is a vector describing the kth face class. This classification is also called as k-NN classification.

Support Vector Machines (SVMs) can also be used to classify the given vector Ω_k into its class.

B. Reconstruction

Reconstruction problem can be best described as reconstructing an ill-formed image using principal features from the eigenfaces.

The eigenfaces calculated from the principal component analysis of the training set can also be used to generate a new image from a test image using principal components in eigenfaces.

We seek to project an input face *v* onto the eigenface space to obtain

$$v' = \sum_{i=1}^{k} a_i \mu_i\ ,$$

Where for *i = 1, 2, 3, 4…k*

$$a_i = \mu_i^T\ v_i$$

Note that the mean face Ψ has to be added to v' to get the final reconstructed face image.

C. Verification

In the verification subtask, given an image and a claimed class of it, we have to respond with an affirmative or negative answer correspondingly if the image lies in the claimed class.

ROC curve and operating point threshold is used to verify a given image.

ROC Curve :-

ROC curve, is a graphical plot which illustrates the performance of a binary classifier system as its discrimination threshold is varied. It is created by plotting the fraction of true positives out of the total actual positives (TPR = true positive rate) vs. the fraction of false positives out of the total actual negatives (FPR = false positive rate), at various threshold settings.

ROC analysis provides tools to select possibly optimal models and to discard suboptimal ones independently from (and prior to specifying) the cost context or the class distribution.

Algorithm –

- Find the ROC plot of image pairs (one from training set, other from testing set), classify them as 1 (if in the same class, 0 if in different class). Use Euclidean distance as scores for the labels.
- Find the optimum threshold from the operating point values.
- Find the face image representation of the given test image by projecting into eigenface basis span set.
- Find the nearest neighbour from the face images of training set which has the minimum Euclidean distance from the face image of input image.
- If this distance is greater than optimum threshold, return “No”.
- If the claimed label matches with the label of the closest neighbor return “Yes”, else “No”.

IV. Results And Performance

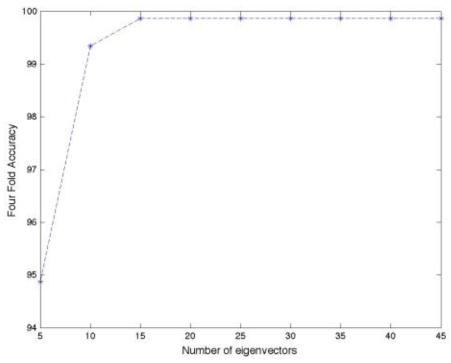
1. Yale Dataset –

   Description –
   The dataset contains images of 38 people with varying Azimuthal and Elevation angles.

For our analysis, we have selected 20 images for each person with Azimuthal and Elevation angles ranging from [-25 to 25].

a) Identification

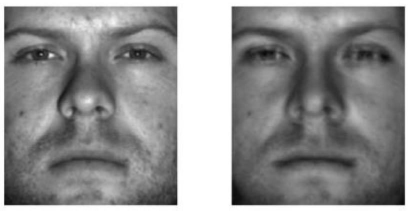
On carrying out PCA using 4-fold validation on this dataset, we get the following plot of Accuracy v/s number of eigenvectors –



On increasing the number of eigenvectors, the accuracy increases until the number of eigenvectors reach 15. Hence, 15 eigenvectors are sufficient to cover the entire image space and hence, the dimensionality can be reduced to 15.

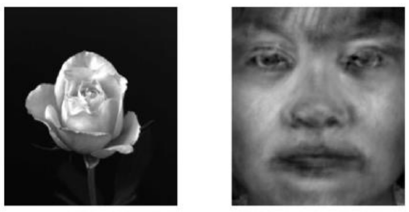
b) Reconstruction

Given a face image from the same dataset, it is possible to reconstruct it using limited number of eigenfaces.

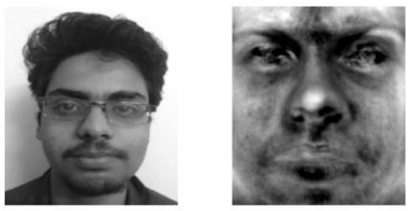


The input image on the left is reconstructed using top 25 eigenfaces.

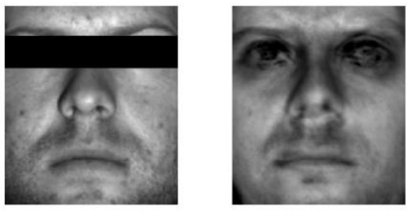
However, on giving a non-face image for reconstruction, it tries to reconstruct it in the face space and tries to introduce a face into the image.



On giving a face image which is not present in the dataset, it fails to reconstruct it and we get similar result.

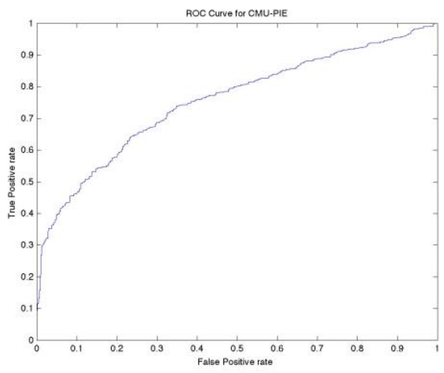


When the input image contains some missing, it tries to reconstruct it using the best features possible.



c) Verification

The ROC Curve used to find the optimal threshold value for the verification of a test image –



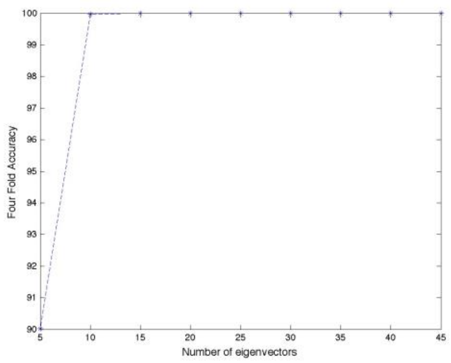
2. CMU PIE Dataset –

Description –

The CMU PIE dataset contains 42 images of 68 people in varying pose, illumination and expression.

a) Identification

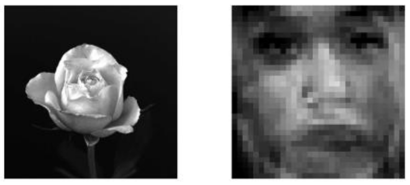
On carrying out PCA using 4-fold validation on this dataset, we get the following plot of accuracy v/s number of eigenvectors –



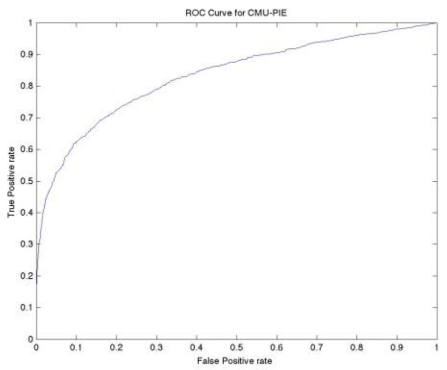
On increasing the number of eigenvectors, the accuracy increases until the number of eigenvectors reach 10. Hence, 10 eigenvectors are sufficient to cover the entire image space and hence, the dimensionality can be reduced to 10.

b) Reconstruction

On giving a non-face image for reconstruction, it tries to reconstruct it in the face space and tries to introduce a face into the image.

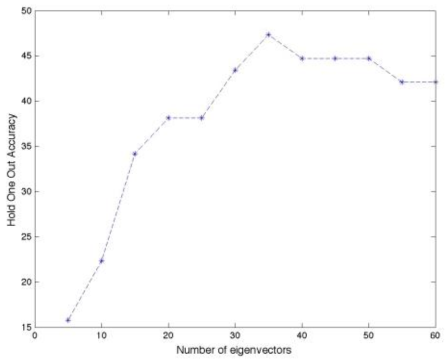


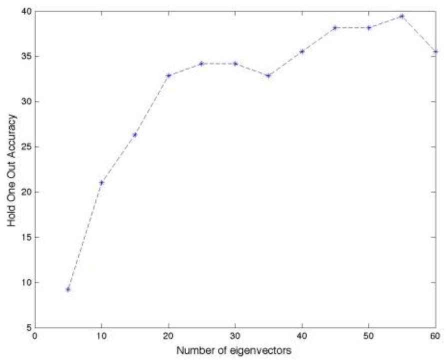
c) Verification –



3. Students SMAI dataset

a) *Identification*





ACKNOWLEDGMENT

Sincerest thanks to our course instructor *Prof Anoop Namboodri* and our entire course TA's, for making this project successful.

REFERENCES

[1] Matthew Turk and Alex Pentland, "Eigenfaces for Recgnition" Vision and Modelling Group, The Media Laboratory, Massachusetts Institute Of Technology(*references*)

[2] Peter N. Belhumeur, Joao P. Hespanha, and David J. Kriegman, "Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection", IEEE transactions on pattern analysis and machine intelligence, vol. 19, no. 7, july 1997

[3] Zhao, W.; Chellappa, R.; Krishnaswamy, A., "Discriminant analysis of principal components for face recognition," Automatic Face and Gesture Recognition, 1998. Proceedings. Third IEEE International Conference on , vol., no., pp.336,341, 14-16 Apr 1998