# Recognizing Faces using Kernel Eigenfaces and Support Vector Machines

Prospero C. Naval, Jr.
Computer Vision & Machine Intelligence Group
Department of Computer Science
College of Engineering
University of the Philippines-Diliman
pcnaval@up.edu.ph

## ABSTRACT

In face recognition, Principal Component Analysis (PCA) is often used to extract a low dimensional face representation based on the eigenvector of the face image autocorrelation matrix. Kernel Principal Component Analysis (Kernel PCA) has recently been proposed as a non-linear extension of PCA. While PCA is able to discover and represent linearly embedded manifolds, Kernel PCA can extract low dimensional non-linearly embedded manifolds from data, thus providing a more suitable representation for subsequent recognition by a classifier. We provide experimental evidence which show that Kernel PCA performs better than PCA on the ATT Face Dataset when both are used with a linear Support Vector Machine Classifier.

## 1. INTRODUCTION

Thirty years ago, the problem of face recognition by a computer was considered among the hardest in artificial intelligence and computer vision. This past decade has seen great progress in face recognition research beginning with Kohonen's idea of a face representation based on the eigenvectors of the face image's autocorrelation matrix [5]. Turk and Pentland [8], inspired by the findings of Kirby and Sirovich [4], further developed the eigenvector representation and devised a practical method for detecting and recognizing faces which is the basis of many of the current commercial real time face recognition systems.

Face recognition has potential applications in security control, office automation, prevention of fraud, automatic personalization of environments, etc. Face recognition has the advantage of being completely passive and non-intrusive, unlike other biometric techniques such as those using fingerprints, speech, and signature.

There are two main categories of face recognition systems that are available:

- systems that identify a person given a large database of faces (e.g. face database of a large corporation). The system returns a list of names that most likely identifies the query face image. There is usually just one image per person that is stored in the database. The application is usually not real time.
- systems that identify a person given a smaller database of faces so that he/she can gain entry to a particular location or access a particular resource (e.g. computer). Multiple images per person are usually available but real time recognition is necessary.

In this paper, we are interested in developing a simple face recognition algorithm for the second application category. In particular, we want to recognize faces in real time given a set of example images of faces with varying facial expressions, hairstyle, and image background. We will assume that the faces have been localized by a previous localization algorithm, that is, the faces are properly centered in the image.

Many face recognition systems having high recognition rates use Principal Component Analysis (PCA) to convert an input face image into its principal component representation called *eigenface*. The eigenface representation of a face image is then submitted to a classifier such as a $k$-Nearest Neighbor Classifier.

Principal Component Analysis has long been used for extracting structure from high dimensional data sets. Classic PCA literature are found in the works of Pearson [6], Hotelling [1], and Karhunen [3]. Kernel Principal Component Analysis [7] has recently been proposed by Scholkopf, Smola and Muller as a non-linear extension to Principal Component Analysis, drawing inspiration from the idea employed in Support Vector Machines, of implicitly mapping data into a high dimensional feature space and doing what we want done in that feature space.

Face recognition using Kernel Principal Component Analysis was first demonstrated by Yang, et. al. [10] who compared it with PCA. Their experiments showed that a 3rd degree polynomial Kernel PCA with a reduced space of 50 principal components outperforms PCA when a $k$-Nearest Neighbor Classifier is used. We call the face representation produced by Kernel PCA as *kernel eigenface*.

We investigate the performance of Kernel Principal Component Analysis (Kernel PCA) to represent faces using a Soft Margin Support Vector Machine (SVM) for our classifier. To test our approach, we use the ATT Face Dataset (formerly called ORL Face Dataset) available from the Web. To emphasize the power of Kernel PCA for face recognition, no further preprocessing such as histogram equalization (to

compensate for illumination variations) or image warping (to account for facial expression differences) is done on the images.

Visual data are often represented as points in high dimensional space. A $m \times n$ image can be mapped as a vector of dimension $\Re^{N=m\times n}$ by lexicographic ordering of its pixel elements. The intrinsic dimensionality of visual data, however, is much lower and the relevant structure sought in face recognition, as in many computer vision tasks, lies in a low dimensional manifold. Subspace methods such as PCA and Kernel PCA try to extract the low dimensional structure manifold embedded in the the high dimensional raw input visual data.

Principal Component Analysis has been found to be an effective method for face recognition [8]. PCA projects the high dimensional image vector into the subspace spanned by the dominant eigenvectors (eigenvectors whose eigenvalues are large) where the variance of the training images is high, making it easy for the classifier to compute for the correct decision surface.

Principal Component Analysis can discover linearly embedded manifolds and produce a compact orthonormal basis representation. For example, projection of data points onto the first principal component corresponds to a 1-dimensional linear manifold representation.

The manifold structure of the facial recognition task, however, cannot be assumed to be linear since it unlikely that the complications present in this task such as variations in facial expression, illumination, etc. are linear in nature. One promising subspace method that could deal with nonlinearly embedded manifolds is Kernel Principal Component Analysis [7]. Kernel PCA is one of the many algorithms that exploit the following idea:

Data $\mathbf{x}_1, ..., \mathbf{x}_n \in \Re^N$, via the nonlinear mapping

$$\begin{aligned} \Phi : \Re^N &\rightarrow F \\ \mathbf{x} &\mapsto \Phi(\mathbf{x}) \end{aligned}$$

is mapped into a very high (possibly infinite) feature space $F$.

Kernel PCA uses a nonlinear kernel function $\Phi(\mathbf{x})$ to project input data $\mathbf{x}$ into feature space $F$ which is nonlinearly related to the input space and performs PCA in feature space $F$. Although the dimensionality of the kernel feature space $F$ is much higher than that of the input space, kernel methods do not suffer from curse of dimensionality because computation in feature space is carried out implicitly.

## 2. THE EIGENFACE APPROACH

Principal Component Analysis (PCA) is used for extracting relevant features from high-dimensional data sets. It performs an orthogonal transformation of the coordinate system in which the data is originally described. After coordinate transformation, it is often the case that only a subset of the new coordinate values is necessary to describe most of the data. This subset is called the principal components of the data. The principal components possess large variance.

The PCA algorithm is formulated as follows:

Let $\mathbf{x}_i$ be an $m$-dimensional vector obtained from a set of $N$ vectors. The mean vector $\overline{\mathbf{x}}$ of the set is

$$\overline{\mathbf{x}} = E\{\mathbf{x}\} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i$$

The covariance matrix of the set of vectors is

$$\mathbf{R} = E\{(\mathbf{x} - \overline{\mathbf{x}})(\mathbf{x} - \overline{\mathbf{x}})^\top\}$$

Given this covariance matrix we solve the eigenequation

$$\mathbf{R}\mathbf{u} = \lambda\mathbf{u}$$

for positive eigenvalues $\lambda_j$, $(j = 1, \ldots, m)$ with these eigenvalues sorted in decreasing order $(\lambda_j \geq \lambda_{j+1})$. The eigenvectors $\mathbf{u}_j$ are orthonormal vectors.

This eigenequation is usually solved using eigendecomposition or Singular Value Decomposition (SVD) methods.

The principal components are also uncorrelated since they are the components of the orthonormal basis. The first $p(1 \leq p \leq m)$ principal components carry more variance than the other orthogonal directions and they have maximal mutual information with respect to the inputs.

## 3. KERNEL EIGENFACES

Kernel Principal Component Analysis (Kernel PCA) is very similar to Principal Components Analysis. While PCA extracts relevant features from data by performing an orthogonal transformation, Kernel PCA uses a kernel to map the data from input space to a higher dimensional feature space and perform PCA in that feature space. This mapping is not computed explicitly. The kernel must satisfy Mercer's Theorem [7].

Since input space to feature space mapping is nonlinear, the resulting coordinate transformation in input space is not orthogonal. Kernel PCA has the advantage of extracting relevant features from principal components of a non-orthogonal coordinate system having variance greater than what PCA could achieve due to some hidden structure inherent in the data.

The Kernel PCA algorithm is as follows:

Given a set of $N$ vectors $\mathbf{x}_1, \ldots, \mathbf{x}_N$, and an inner product kernel $k(\mathbf{x}_i, \mathbf{x}_j)$ we construct the $N \times N$ kernel matrix $K^{'}$ whose $ij$-th element is $k(\mathbf{x}_i, \mathbf{x}_j)$. We center the mapped data points in feature space using the following equation:

$$\begin{aligned} K_{ij} = K^{'}_{ij} - \frac{1}{N}\sum_{m=1}^{N} 1_{im}K^{'}_{mj} - \frac{1}{N}\sum_{n=1}^{N} K^{'}_{in}1_{nj} \\ + \frac{1}{N^2}\sum_{m=1}^{N}\sum_{n=1}^{N} 1_{im}K^{'}_{mn}1_{nj} \end{aligned}$$

where 1 is an $N \times N$ matrix whose entries are all 1's.

We then solve the eigenvalue problem

$$\mathbf{Ka} = \lambda \mathbf{a}$$

for positive eigenvalues $\lambda_j$ with these eigenvalues sorted in decreasing order ($\lambda_j \geq \lambda_{j+1}$) and normalize the eigenvector coefficients

$$\mathbf{a}^{(n)} \cdot \mathbf{a}^{(n)} = \frac{1}{\lambda_n} \qquad n = 1, \ldots, p \qquad (p \leq N)$$

To extract the $n$-th kernel principal component $q_{(n)}$ of a test image $\mathbf{x}_t$ we use the following formula:

$$q^{(n)} = \sum_{i=1}^{N} a_i^{(n)} k(\mathbf{x}_i, \mathbf{x}_t)$$

Another difference between PCA and Kernel PCA is the maximum number of principal components that we can extract from them. The upper limit for PCA is the dimensionality of the image vector while for Kernel PCA it is the number of samples in the training set. Thus, for Kernel PCA, it is possible for us to use more eigenvector projections than the dimensionality of the input data.

## 4. SUPPORT VECTOR MACHINES

In pattern classification using Support Vector Machines (SVM), examples are mapped to a higher dimensional feature space and a decision hyperplane in feature space that separates the data points is computed. The SVM algorithm [9] computes for the separating hyperplane whose margin of separation between positive and negative examples is maximized.

Given a set of labeled training examples $(\mathbf{z}_1, d_1), ..., (\mathbf{z}_N, d_N)$, the algorithm minimizes the following cost function:

$$\phi(\mathbf{w}, \xi_i) = \frac{1}{2}\mathbf{w}^\top \mathbf{w} + C \sum_i \xi_i$$

subject to the constraints:

$$d_i(\mathbf{w}^\top \mathbf{z}_i - b) \geq 1 - \xi_i \qquad i = 1, 2, \ldots, N$$
$$0 \leq \xi_i \leq C \qquad i = 1, 2, \ldots, N$$

where $\mathbf{w}$ is the weight vector, $b$ the bias that measures the perpendicular distance of the hyperplane from the origin, and $\xi_i$ is a set of slack variables and $C$ is a user-specified complexity parameter.

The presence of the slack variables allows the possibility for some of the examples to be misclassified by the hyperplane as long as there is an upper bound on the number and degree of misclassifications. This type of SVM is called a soft margin SVM [9].

The decision rule for SVM is

$$g(\mathbf{z}) = \sum_{i=1}^{N} \alpha_{o,i} d_i k(\mathbf{z}_i, \mathbf{z}) \qquad i = 1, \ldots, N$$

The examples $\mathbf{z}_i$ for which $\alpha_{o,i}$ is non-zero are called support vectors. These support vectors are those training examples that define the decision hyperplane.

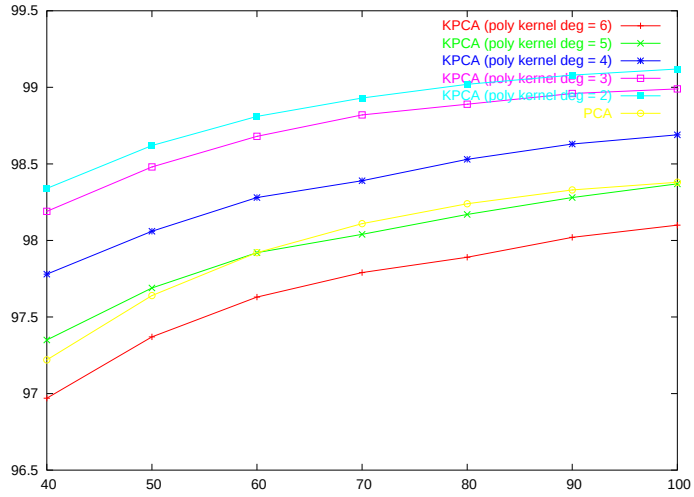


**Figure 1: Recognition Rates with Increasing Dimensionality of Principal Subspace for Different Polynomial Kernel Degree Values**

## 5. EXPERIMENTS

The ATT Face Dataset consists of 10 different greyscale images of 40 different persons. The resolution of each image is 92 by 112 pixels. The images for each person have variations in facial expression (open/closed eyes, smiling/non-smiling), and facial details (glasses/no glasses). The variations in scale is up to 10%. Yang, et. al. [10] also used the ATT Face Dataset for their experiments but derived experimental data only for the $k$-Nearest Neighbor Classifier.

We use a 50/50 partition for training and testing. Each training set contains 200 labeled images of which 5 and 195 are positive and negative examples respectively. We perform a 10-fold cross validation on the dataset.

Feature extraction was performed on the training set of 200 labeled examples using Kernel Principal Component Analysis. In forming the input vector, we down-sampled the images by 4 resulting in an image resolution of 23 $\times$ 28 pixels. The pixels are then normalized by dividing each gray level value by 256.

For the kernel, we tried the polynomial kernel

$$k(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^\top \cdot \mathbf{x}_j)^d$$

The values of $d$ were varied from 1 to 6. For $d = 1$ we simply have the standard PCA.

Computing the kernel components of the full $N \times N (= 200 \times 200)$ matrix required around 2 minutes of computation on a 1.6 GHz Pentium IV machine running Mandrake Linux 9.0. Most of the computation time was spent in solving the eigenequation.

For each image, the top $n$ kernel principal components $q^{(n)}$ corresponding to the kernel were then extracted using the

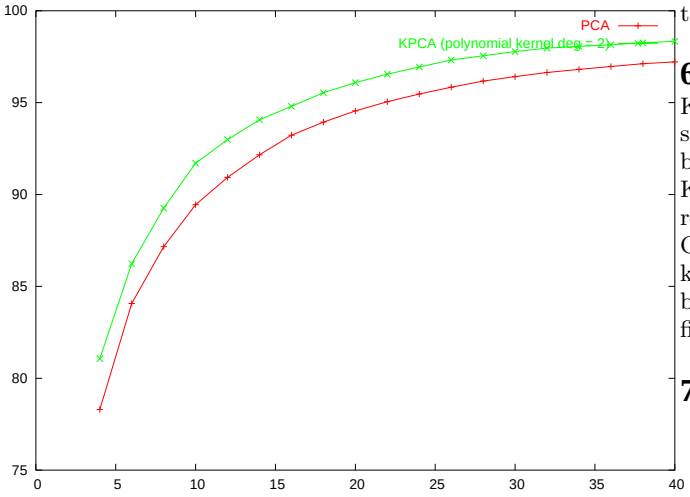


**Figure 2: Comparison of Best Performing Kernel PCA (Polynomial kernel degree = 2) with PCA**

following equation:

$$q^{(n)} = \sum_{i=1}^{N} a_i^{(n)} k(\mathbf{x}_i, \mathbf{x}_t)$$

The principal components were submitted to the Support Vector Machine Classifier for processing.

We used the publicly-available `SVMlight` pattern classifier code written by Thorsten Joachims which implements Vapnik's SVM algorithm for pattern classification [2] .

The results of the experiments for the different values of polynomial kernels are shown in Fig. 1. The best performer was found to be the 2nd degree polynomial Kernel PCA.

We then compared the best performing Kernel PCA with standard PCA for the lower range of principal subspace values. The purpose of this experiment is to evaluate the how compact these two subspace representations are. The results are shown in Fig. 2.

It is evident from our experiments that Kernel PCA performs better than standard PCA by a noticeable margin. The difference is about 1 % for a reduced space of 50 principal components.

We also notice a slight, though progressive decrease in recognition accuracy as the polynomial kernel degree increases. However, Yang et. al. [10], found that 10th degree polynomial kernel also achieved low error rates for the $k$-Nearest Neighbor Classifier.

The second figure is illustrative for it tells us just how much information useful for recognition is contained in the first $n$ principal components for both methods. Using only 4 principal components, the recognition rates for PCA and Kernel PCA are 78.3 % and 81.1 % respectively. This shows us that Kernel PCA provides a more compact face representation than PCA.

## 6. CONCLUSION

Kernel Principal Component Analysis is a non-linear extension of Principal Component Analysis (PCA) which is the basis of the eigenface method. In this paper we argue that Kernel Principal Component Analysis provides a face representation that is more suitable for recognition than PCA. Our experiments show that Kernel PCA with a polynomial kernel outperforms PCA on the face recognition task when both are used with a linear Support Vector Machine classifier.

## 7. REFERENCES

[1] H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24:417–441, 1933.

[2] T. Joachims. *Making Large-Scale SVM Learning Practical. Advances in Kernel Methods - Support Vector Learning*. MIT Press, Cambridge, MA, 1999.

[3] K. Karhunen. Zur spektaltheorie stochastischer prozesse. *Ann. Acad. Sci. Fenn.,*, 34, 1946.

[4] M. Kirby and L. Sirovich. Low-dimensional procedure for the characterization of human faces. *J. Opt. Soc. Am. A*, 4(3):519–524, March 1987.

[5] T. Kohonen. *Self-Organization and Associative Memory*. Springer-Verlag, Berlin, 1989.

[6] K. Pearson. On lines and planes of closest fit to points in space. *Philosophical Magazine*, 2:559–572, 1901.

[7] B. Scholkopf, A. J. Smola, and K. Muller. *Kernel Principal Component Analysis*. MIT Press, Cambridge, MA, 1999.

[8] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3:71–86, 1991.

[9] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer Verlag, New York, 1995.

[10] M. Yang, N. Ahuja, and D. Kriegman. Face recognition using kernel eigenfaces. In *Proceedings of the 2000 International Conference on Image Processing (ICIP 2000)*, pages 37–44, Sept 1994.