M.Tech.(Computer Science) Dissertation Series

Polynomial PCA for Face Recognition

A dissertation submitted in partial fulfillment of the requirements for the M.Tech.(Computer Science) degree of the Indian Statistical Institute

By

Swarup Chattopadhyay

Roll No: CS0712

under the supervision of

Prof. C.A. Murthy

Machine Intelligence Unit.

Indian Statistical Institute

203, B.T. Road

Kolkata- 700108

July 17, 2009

Indian Statistical Institute

203,B.T.Road

kolkata-700108

Certificate of Approval

This is certify that this dissertation thesis titled "**Polynomial PCA for Face Recognition**"submitted by Mr. Swarup Chattopadhyay, in partial fulfillment of the requirements for the M.Tech. (Computer Science) degree of the Indian Statistical Institute, Kolkata, embodies the work done under my supervision.

Prof. C.A. Murthy

Machine Intelligence Unit

Indian Statistical Institute,Kolkata

Kolkata-700108

Acknowledgements

It is my privilege and pleasure to convey my gratitude to my guide **Prof.C.A.Murthy** for his support and guidance throughout the duration of dissertation without which it would have been impossible to complete this work.I sincerely thanks all my friends and well wishers who helped me directly or indirectly towards the completion of this work.

Swarup Chattopadhyay

Mtc0712

Indian Statistical Institute

Kolkata-700108

Contents

Chapter 1

Introduction

1.1 PCA

PCA is a useful statistical technique that has found application on fields such as face recognition and image compression,and is a common technique for finding patterns in data of high dimensions.PCA finds the orthogonal directions that account for the highest amount of variance. The data is then projected into the subspace spanned by these directions.

It is a way of identifying patterns in data, and 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, where the luxury of graphical representation is not available, PCA is a powerful tool for analysing data. The other main advantage of PCA is that once you have found these patterns in the data, and you compress the data, ie. by reducing the number of dimensions, without much loss of information. This technique used in image compression. some mathematical and statistical skills are required to understand the process of PCA , such as **covariance**, **standard deviation**, **eigenvectors** and **eigen values**.

The following steps are required to perform a Principal Components analysis on a set of data

- **Step1: get some data**

  At first we collect some data of some dimensions on which PCA will be applied.

- **Step2: Subtract the mean**

  Now we have to subtract the mean from each of the data dimensions. The mean subtracted is the average across each dimension.

- **Step3: Calculate the covariance matrix**

  In this step we calculate the variance covariance matrix of the given data which is a squre matrix.

- **Step4: Calculate the eigenvectors and eigenvalues of the covariance matrix**

  Scince the covariance matrix is squre, we can calculate the eigenvectors and eigenvalues for the matrix. These are rather important, as they tell us useful information about our data. Scince sum of eigenvalues = sum of the variances .

- **Step5: Choosing components and forming a feature vector**

  Here is where the notion of data compression and reduced dimensionality comes into it. In fact, it turns out that the eigenvector with the highest eigenvalue is the **principle component** of the data set. To be price, if we have n dimensions in our data, and so we calculate n eigenvectors and eigenvalues, and then we choose only the first p (largest) eigenvectors, then the final data set has only p dimensions.

- **Step6: Deriving the new data set**

  This the final step in PCA. Once we have chosen the components (eigenvectors) that we wish to keep in our data and formed a feature vector, we simply take the transpose of the vector and multiply it on the left of the original data set, transposed.

  $$\text{FinalData} = \text{RowFetureVector} \times \text{RowDataAdjust},$$

  where RowFetureVector is the matrix with the eigenvectors in the columns transposed so that the eigenvectors are now in the rows, with the most significant eigenvector at the top, and RowDataAdjust is the mean adjusted data transposed, i.e the data items are in each column, with each row holding a separate dimension

1.2 Kernel PCA

Kernel PCA is a nonlinear generalization of PCA

Why Kernel PCA

- The essential idea of KPCA is based on the hope that if we do some non linear mapping of the data points to a higher dimensional(possibly infinite) space we can get better non linear features which are a more natural and compact representation of the data.

How

- Let us consider M input data points of N dimensions and also Consider a nonlinear mapping $\phi(.) : R^N \rightarrow R^H(F)$ from R^N the space of N dimensional data points to some higher dimensional space or feature space$R^H(F)$.
- As of now assume that the mapped data are zero centered (i.e $\sum_{i=1}^{M} \phi(x_i) = 0$). Now we define a dot product matrix K such that $[K]_{ij} = [\phi(x_i).\phi(x_j)]$, K is called the Gram matrix.
- Once we have this mapping KPCA is nothing but Linear PCA done on the points in the higher dimensional space.
- The computational complexity arising from the high dimensionality mapping is mitigated by using the kernel trick.

The details of the Kernel PCA algorithm will be described in the next chapter. Now we are going to discuss something about Kernel Trick and Kernel function.

Kernel Function:

A kernel function K on a space x is a mapping from $X \times X$ to R. Define the kernel function K(x,y) as $K(x, y) = (1 + x_1y_1 + x_2y_2)^2$. Consider the following transformation

$$\phi(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}) = (1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, x_2^2, \sqrt{2}x_1x_2)$$

$$\phi(\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}) = (1, \sqrt{2}y_1, \sqrt{2}y_2, y_1^2, y_2^2, \sqrt{2}y_1y_2)$$

$$\langle\phi(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}), \phi(\begin{bmatrix} y_1 \\ y_2 \end{bmatrix})\rangle = (1 + x_1y_1 + x_2y_2)^2$$

$$= \text{K(x,y)}$$

The inner product can be computed by K without going through the map $\phi(.)$

Example of kernel function

- Polynomial kernel with degree d

$$K(X, Y) = (X^TY + 1)^d$$

- Radial basis function kernel with width σ

$$K(X, Y) = exp(-\|X - Y\|^2/(2\sigma^2))$$

Here in this paper we will mainly discussing about **Polynomial Kernel**

Kernel Trick

- The relationship between the kernel function K and the mapping $\phi(.)$ is

$$K(X, Y) = \langle\phi(X), \phi(Y)\rangle$$

  –This is known as the kernel trick

- In practice, we specify K, thereby specifying $\phi(.)$ indirectly, instead of choosing $\phi(.)$

- Intuitively, K(x,y) represents our desired notion of similarity between data x and y and this is from our prior knowledge.

- K(x,y) needs to satisfy a technical condition (Mercer condition) in order for $\phi(.)$ to exist.

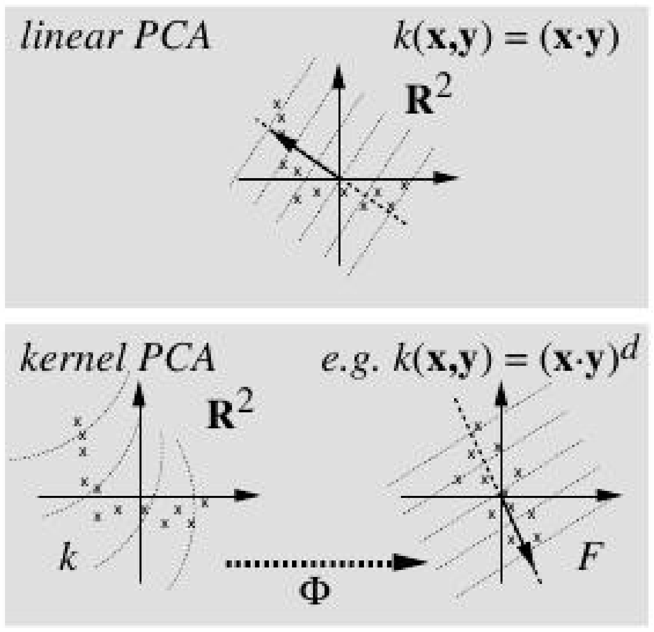

linear PCA
$k(\mathbf{x},\mathbf{y}) = (\mathbf{x}\cdot\mathbf{y})$
$\mathbf{R}^2$
kernel PCA
e.g. $k(\mathbf{x},\mathbf{y}) = (\mathbf{x}\cdot\mathbf{y})^d$
$\mathbf{R}^2$
k
Φ
F

Figure 1.1:

Properties of Kernel PCA: For Mercer kernels, we know that we are in fact doing a standard PCA in F . Consequently, all mathematical and statistical proper- ties of PCA carry over to kernel PCA, with the modi cations that they become statements about a set of points $\phi(X_i)$, i = 1,...,M , in F rather than in R^N . In F , we can thus assert that PCA is the orthogonal basis transformation with the following properties (assuming that the eigenvectors are sorted in descending order of the eigenvalue size):

1. the first $q(q \in 1, ..., M)$ principal components, i.e. projections on eigen- vectors, carry more variance than any other q orthogonal directions.

2. the mean-squared approximation error in representing the observations by the first q principal components is minimal.

3. the principal components are uncorrelated.

4. the first q principal components have maximal mutual information with respect to the inputs (this holds under Gaussianity assumptions, and thus depends on the particular kernel chosen and on the data)

Comparison of Kernel PCA with related methods

- Other generalizations of linear PCA : Hebbian Networks, Autoassociative Multi-Layer Perceptrons, Principal Curves
- **Advantages of Kernel PCA :**

  - No nonlinear optimization is involved.

  - The number of principal components does not need to be specified in advance.
- **Disadvantages of Kernel PCA :**

  - Compared with neural approaches, processing very large number of observations is much more difficult.

  - Compared with the Principal Curve approach, the results are harder to interpret in the input space.

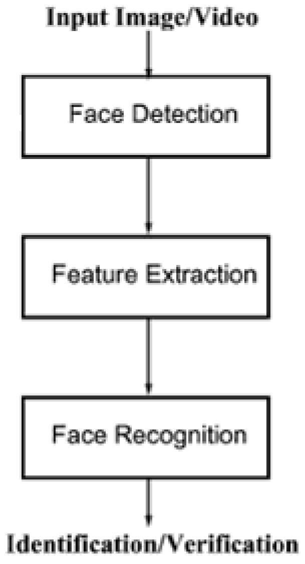


Figure 1.2:

1.3 Face Recognition :

Face recognition is one of biometric methods identifying individuals by the features of face.Given still or video images of a scene, identify or verify one or more persons in the scene using a stored database of faces. The solution to the problem involves the segmentation of faces (face detection) from image of a scene, feature extraction from face region, recognition.

A basic structure of a Face Recognition System is shown in the diagram given above(fig 1.2).

The main parts are typically face detection and face recognition which can it self be decomposed in normalization, feature extraction and classification steps.

Before face recognition is performed, the system should determine whether or not there is a face in a given image or given video, a sequence of images. This process is called face detection. Once a face is detected, face region should be isolated from the scene for the face recognition. The face detection and face extraction are often performed simultaneously.

Feature extraction is to find a specific representation of the data that can highlight relevant information. At the feature extraction stage, the goal is to find an invariant representation of the face image. Usually, an image is represented by a high dimensional vector containing pixel values (holistic representation) or a set of vectors where each vector contains gray levels of a sub-image (local representation).

Different approaches of face recognition for still images in the literature can be categorized into three main groups :

1.Holistic Approach.

2.Feature-Based Approach.

3.Hybrid Approach.

1. Holistic Approach :

In holistic approach, the whole face region is taken into account as input data into face detection system for extracting the features of a face image.

Principal Component Analysis:

One of the feature extraction techniques, based on Principal Component Analysis (PCA), was first used for face recognition by Turk and Pentland [4]. The aim of PCA is to find a representation of the data minimizing the reconstruction error. The PCA finds the orthogonal directions that account for the highest amount of variance. The data is then projected into the subspace spanned by these directions. In practice, the principal component axes are the eigenvectors of the covariance matrix of the data. The corresponding eigen values indicate the proportion of variance of the data projections along each direction.

Linear Discriminant Analysis:

Another feature extraction method used in face recognition is based on Linear Discriminant Analysis (LDA), also known as Fisher Discriminant Analysis [5].

The LDA subspace holds more discriminant features than the PCA subspace. LDA finds a subspace in which the variability is maximized between different class data, and at the same time where variability in the same class data (face images of the same identity) is minimum.

We define the within-class scatter matrix as S_w and between-class scatter matrix as S_b .The goal is to maximize the between-class measure while minimizing the within-class measure. One way to do this is to maximize the ratio $det(S_b)/det(S_w)$. Intuitively, for face recognition, LDA should outperform PCA because it inherently deals with class discrimination. However, Martinez and Kak [5] have shown that PCA might outperform LDA when the number of samples per class is small.

2. Feature-Based Approach :

In feature-based approaches, local features on face such as nose, and then eyes are segmented and then used as input data for structural classifier.

E.g. Pure Geometry Methods.

3. Hybrid Approach :

Use both local features and whole face region to recognize a face.

- **Classification Task :**

  The classification step consists of attributing a class label to the input data. Some of the well known classifiers are nearest neighbor classifier and minimum distance classifier. Current approaches for face recognition often make use of simple image similarity metrics such as the Euclidean distance between the feature vector of reference images and feature vector of the test image. Because of curse of dimensionality problem, the distance metrics are not computed in the image space but in an appropriate subspace such as PCA or LDA. Some more appropriate metrics have been proposed in the literature such as Mahalanobis distance, Normalized correlation.

- **Limitation of the above Approaches :**

  All the above methods concern about the dissimilarity (i.e., generally the metrics like Euclidean distance or Mahalanobis distance) between representations of the query image and the training images. We classify the test face image into the class whose distance is the least from the training images. In this approach the test image always fall into one of the face classes of the database. This process of identification known as closed set identification

  In case, if the test image is not a valid face image (i.e., image of an object or a scene or a face image which is not in our database), it will be classified to one of the face classes of our database, then this type of identification is called open set identification. Here the test image is said to be imposter to the system.

  We have to find a threshold value on the dissimilarity value that will decide whether a test image is either a valid face or not.

1.4 PCA Based Face Recognition :

PCA based face recognition, means we are going to recognise the face by reducing dimensionality.

One approach to deal with high dimensional data is by reducing their dimensionality. Project high dimensional data onto a lower dimensional sub-space using linear or non-linear transformations.

- Each dimensionality reduction technique finds an appropriate transformation by satisfying certain criteria (e.g., information loss, data discrimination, etc.)
- The goal of PCA is to reduce the dimensionality of the data while **retaining as much as possible of the variation present in the dataset.**

Methodology :

- Suppose Γ is an $N^2 \times 1$ vector, corresponding to an $N \times N$ face image I.

- The idea is to represent $\Gamma (\Phi = \Gamma - meanface)$ into a low-dimensional space:

$$\hat{\Phi} - mean = w_1 u_1 + w_2 u_2 + ... + w_K u_K (K << N^2)$$

Step 1 : Obtain face images $I_1, I_2, ..., I_M$ (training faces)

(very important: the face images must be centered and of the same size)

Step 2 : Represent every image I_i asa vector Γ_i

Step 3 : Compute the average face vector Ψ :

$$\Psi = \frac{1}{M} \Sigma_{i=1}^{M} \Gamma_i$$

Step 4 : Subtract the mean face :

$$\Phi_i = \Gamma_i - \Psi$$

Step 5 : Compute the covariance matrix C :

$$C = \frac{1}{M} \Sigma_{n=1}^{M} \Phi_n \Phi_n^T = AA^T \ (N^2 \times N^2 \text{ matrix})$$

where $A = [\Phi_1 \Phi_2 ... \Phi_M]$ ($N^2 \times M$ matrix)

<u>Step 6 :</u> Compute the eigenvectors u_i of AA^T

<u>The matrix AA^T is very large $--\,>$ not practical !!</u>

<u>Step 6.1 :</u> Consider the matrix A^TA ($M \times M$ matrix)

<u>Step 6.2 :</u> Compute the eigenvectors v_i of A^TA

$$A^TAv_i = \mu_i v_i$$

<u>What is the relationship between u_i and v_i ?</u>

$A^TAv_i = \mu_i v_i \Rightarrow AA^TAv_i = \mu_i Av_i \Rightarrow Cu_i = \mu_i u_i$, where $u_i = Av_i$. Thus AA^T and A^TA have the same eigen values and their eigen vectors are related as follows $u_i = Av_i$.

<u>Note 1 :</u> AA^T can have upto N^2 eigenvalues and eigenvectors.

<u>Note 2 :</u> A^TA can have upto M eigenvalues and eigenvectors.

<u>Note 3 :</u> The M eigenvalues of A^TA (along with their corresponding eigenvectors) correspond to the M largest eigenvalues of AA^T (along with their corresponding eigenvectors)

<u>Step 6.3 :</u> Compute the M best eigenvectors of AA^T : $u_i = Av_i$

(important: normalize u_i such that $\| u_i \| = 1$)

<u>Step 7 :</u> Keep only K eigenvectors (corresponding to the K largest eigenvalues)

Classification :

- After extracting features using PCA above, we can apply any classifier like minimum distance classifier, K-nearest neighbor classifier to classify a query image.
- In minimum distance classification ,we first extract the features of the query image by projecting onto above eigen space.
- We classify an query image to a face class for which it contains the training image having minim um distance from the query image.

- In K-nearest neighbor classification (K is predefined), we consider the first K nearest training images and classify into the face class for which the maximum number of neighbors came from.
- We may use Euclidean distance or Mahalanobis distances for calculating the distances.
- **Note.** For increasing the distance b/w classes more and decrease with-in- class distance, we use Linear Discriminant Analysis for feature extraction.

Chapter 2

Kernel Principal Component Analysis

We already know that **Kernel PCA** is a nonlinear generalization of PCA. Here in this chapter we are mainly going to describe the **derivation** part of Kernel PCA algorithm. For a given input data set, first we do the nonlinear trnsformation from input space to some higher dimensional space (i.e in Feature Space) and then apply **PCA** in Feature Space.

- **Principal Component Analysis in Feature Space :**

  Given a set of centered observations $X_k \in \Re^N$, k=1, . . . ,M, $\sum_{k=1}^{M} X_k = 0$, PCA diagonalizes the covariance matrix :

$$C = \frac{1}{M} \sum_{j=1}^{M} X_j X_j^T. \tag{2.1}$$

  Now we apply the nonlinear map Φ from input space $\Re^N$ to Feature Space(F).

$$\Phi : \Re^N \rightarrow F, X \mapsto \Phi(X). \tag{2.2}$$

  Note that the feature space F could have an arbitrary large, possibly infinite dimensionalty. Again, we assume that we are dealing with centered data, $\sum_{k=1}^{M} \Phi(X_K) = 0$. In F , the covariance matrix takes the form

$$\overline{C} = \frac{1}{M} \sum_{j=1}^{M} \Phi(X_j)\Phi(X_j)^T. \tag{2.3}$$

We now have to find eigenvalues $\lambda \geq 0$ and eigenvectors $V \in F \setminus \{0\}$ satisfying $\lambda V = \overline{C}V$.

Again, all solutions V with $\lambda \neq 0$ lie in the span of $\Phi(X_1), ..., \Phi(X_M)$. For us, this has two useful consequences : first, we may instead consider the set of equations

$$\lambda(\Phi(X_k).V) = (\Phi(X_k).\overline{C}V) \tag{2.4}$$

for all k=1,...,M and second, there exists coefficients $\alpha_i(i = 1, ..., M))$ such that

$$V = \sum_{i=1}^{M} \alpha_i \Phi(X_i). \tag{2.5}$$

Combining (2.4) and (2.5), we get

$$\lambda \sum_{i=1}^{M} \alpha_i (\Phi(X_k).\Phi(X_i)) = \frac{1}{M} \sum_{i=1}^{M} \alpha_i \left(\Phi(X_k). \sum_{j=1}^{M} \Phi(X_j)(\Phi(X_j).\Phi(X_i)) \right) \tag{2.6}$$

for all k=1,...,M. Defining an $M \times M$ **Gram Matrix** K by

$$K_{ij} := (\Phi(X_i).\Phi(X_j)) \tag{2.7}$$

this reads

$$M\lambda K\overline{\alpha} = K^2\overline{\alpha} \tag{2.8}$$

where $\overline{\alpha}$ denotes the column vector with entries $\alpha_1, \alpha_2, ..., \alpha_M$. To find solutions of (2.8), we solve the eigen value problem

$$M\lambda\overline{\alpha} = K\overline{\alpha} \tag{2.9}$$

for nonzero eigenvalues.Let $\lambda_1 \leq \lambda_2 \leq ... \leq \lambda_M$ denote the eigenvalues of K, and $\overline{\alpha}^1, ..., \overline{\alpha}^M$ the corresponding complete set of eigen vectors, with λ_p being the first nonzero eigenvalue. We normalize $\overline{\alpha}^p, ..., \overline{\alpha}^M$ by requiring that the corresponding vectors in F be normalized.i.e,

$$(V^k.V^k) = 1, k = 1, ..., M. \tag{2.10}$$

By virtue of (2.5) and (2.9) ,this translates into a normalization condition for $\overline{\alpha}^p, ..., \overline{\alpha}^M$:

$$1 = \sum_{i,j=1}^{M} \alpha_i^k \alpha_j^k (\Phi(X_i).\Phi(X_j)) = \sum_{i,j=1}^{M} \alpha_i^k \alpha_j^k K_{ij} = (\overline{\alpha}^k.K\overline{\alpha}^k) = \lambda_k(\overline{\alpha}^k.\overline{\alpha}^k) \tag{2.11}$$

For the purpose of principal component extraction, we need to compute projections onto the eigenvectors V^k in F (k=p,...,M).Let X be a test point, with an image $\Phi(X)$ in F, then

$$(V^k.\Phi(X)) = \sum_{i=1}^{M} \alpha_i^k (\Phi(X_i).\Phi(X)) \tag{2.12}$$

may be called its nonlinear principal components corresponding to Φ.

Kernel PCA Algorithm :

To perform kernel-based PCA , henceforth reffered to as kernel PCA, the following steps have to be carried out: first, we compute the **Gram Matrix** $K_{ij} = (k(X_i, X_j))_{ij}$. Next we solve (2.9) by diagonalizing K, and normalize the eigenvector expansion coefficients α^n by requiring $\lambda_n(\overline{\alpha}^n.\overline{\alpha}^n) = 1$. To extract the principal components (corresponding to the kernel K) of a test point X, we then compute projections onto the eigenvectors by

$$(V^n.\Phi(X)) = \sum_{i=1}^{M} \alpha_i^n k(X_i, X). \tag{2.13}$$

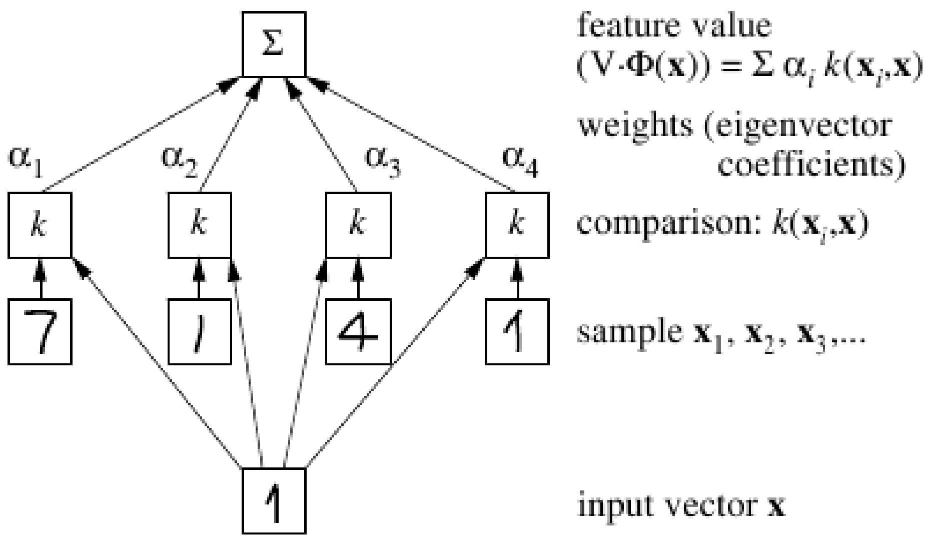


Figure 2.1:

Computational Complexity of KPCA :

- The Gram matrix is an $M \times M$ matrix and finding the eigen values and eigen vectors is of $O(M^3)$ complexity.
- Also the memory storage is of $O(M^2)$
- However in most applications all the eigen vectors are not needed. The number of eigen vectors needed is usually $<< N$. In such cases we can use iterative methods to compute the eigen vectors and the eigen values and the complexity becomes $O(M^2)$.
- $O(M^2)$ complexity to compute the kernel principal components.

Chapter 3

Proposed Modification

3.1 Why Modification

In this paper we are mainly concern about Polynomial Kernel.In KPCA for constructing Gram Matrix, we are consider Polynomial Kernel of degree d i.e $K(X,Y) = (X^TY + 1)^d$.

For the purpose of face recognition,if we are using Kernel PCA (Polynomial Kernel) with Gram Matrix, then in that case we are moving in the following way :

Here we consider each face image as a single vector, i.e each pixel consider as a variable.If in our hand 400 images are there, then first we calculate the Kernel Matrix or Gram Matrix K of order 400×400 by using the following rule :

$$K_{ij} := (\Phi(X_i).\Phi(X_j)) \tag{3.1}$$

Next we are going to find the eigenvalues and eigenvectors of K, and normalize the eigenvector expansion coefficients α^n by requiring $\lambda_n(\overline{\alpha}^n.\overline{\alpha}^n) = 1$. To extract the principal components (corresponding to the polynomial kernel $K(X,Y) = (X^TY + 1)^d$) of a test point X, we then compute projections onto the eigenvectors by

$$(V^n.\Phi(X)) = \sum_{i=1}^{M} \alpha_i^n k(X_i, X). \tag{3.2}$$

In that case we are not using the nonlinear map Φ explicitely, instead of that we are doing everything in the input space whatever we need.**That means we did not impose any condition on the individual pixels of the face images.**

So we are modifying this method by applying some conditions on individuals pixels of the face images. That means in our proposed method we are going to use the nonlinear map Φ explicitely.How we are using the map Φ explicitely and how we are doing the PCA in the transformed space (i.e in Feature space) is given below :

3.2 How

Suppose we are considering 400 images $I_1, I_2, ..., I_{400}$ and each image is of order 92×112. Then we consider each image as a single column vector of dimension $(92 \times 112, 1)$.

Consider the Nonlinear map Φ defined as follows :

$$\Phi : (x_1, x_2) \longrightarrow (x_1, x_2, x_1x_2, x_1^2, x_2^2) \tag{3.3}$$

Now we apply the nonlinear map Φ on each image I.Since each image I is of size $(92 \times 112, 1)$, after applying this nonlinear map Φ on I, the dimension of the transformed vector I' become huze.So we reduce the size of each image first, then apply the nonlinear map Φ on every images.

Suppose $I_1', I_2', ..., I_{400}'$ are the transformed vector after applying Φ on $I_1, I_2, ..., I_{400}$ .Let us consider the matrix A defined as follows : $A = [I_1', I_2', ..., I_{400}']$. Here A is of order $(92 \times 112, 400)$.Now we are going to do the PCA on the set of vectors $I_1', I_2', ..., I_{400}'$ .This is the same as PCA based face recognition process described in chapter 1.

All we are interested to find out the eigenvalue and eigenvectors of the variance covariance matrix $C = AA^T$. Since the matrix C is of order $(92 \times 112, 92 \times 112)$, finding the eigenvalues and eigenvectors of C is quite impossible, so all we are do that , first consider the matrix $C' = A^TA$ and find out the eigenvalues, eigenvectors of C'.Then premultiplying A with C' and in that way find out the eigenvalues, eigenvectors of C. Next we project all the data points(vectors) onto the eigenvectors corresponding to the largest eigenvalues,for reducing the dimension.Then apply the KNN rule (K-nearest neighbour rule) for doing the classification.

Now we do the classification on the set of images (E.x. ORL data set, YALE data set, Other data set) using our proposed method by applying different nonlinear map Φ and in the usual Kernel PCA method using Gram Matrix, and comparing the results.

ORL Data set : This data set contains the face images of 40 different peoples and each people have 10 face images with different expressions. That means this data set has 40 different classes and each class contains 10 members, total 400 face images are there.

YALE Data set : This data set contains the face images of 15 different peoples and each people have 11 face images with different expressions. That means this data set has 15 different classes and each class contains 11 members, total 165 face images are there.

OTHER Data set : This data set contains the face images of 38 different peoples and each people have 10 face images with different expressions. That means this data set has 38 different classes and each class contains 10 members, total 380 face images are there.

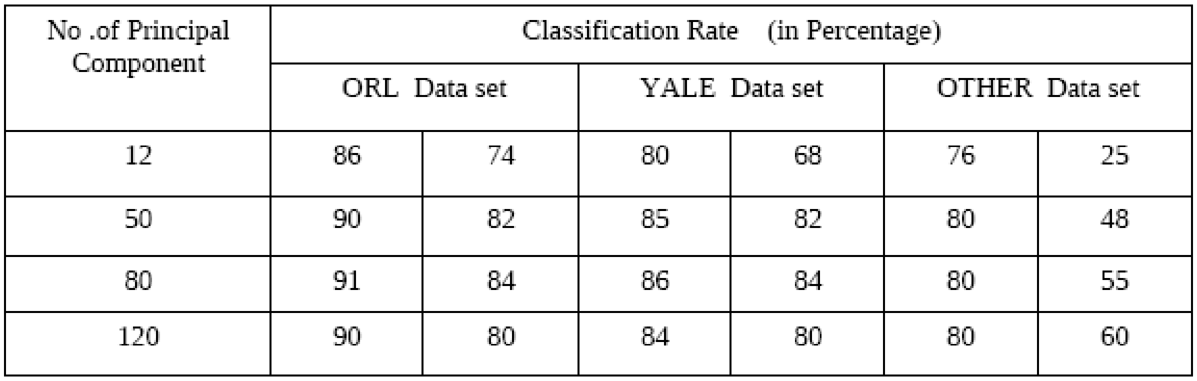
All the above data set contains Gray Lavel images.

3.3 Experiment and Result :

First we apply the nonlinear map Φ defined as follows :

$$\Phi : (x_1, x_2) \longrightarrow (x_1, x_2, x_1x_2, x_1^2, x_2^2) \tag{3.4}$$

in our proposed Polynomial PCA method and comparing the result with the usual Kernel PCA method using Gram Matrix, by considering a polynomial kernel of degree 2
(i.e $K(x, y) = (x^T y + 1)^2$).



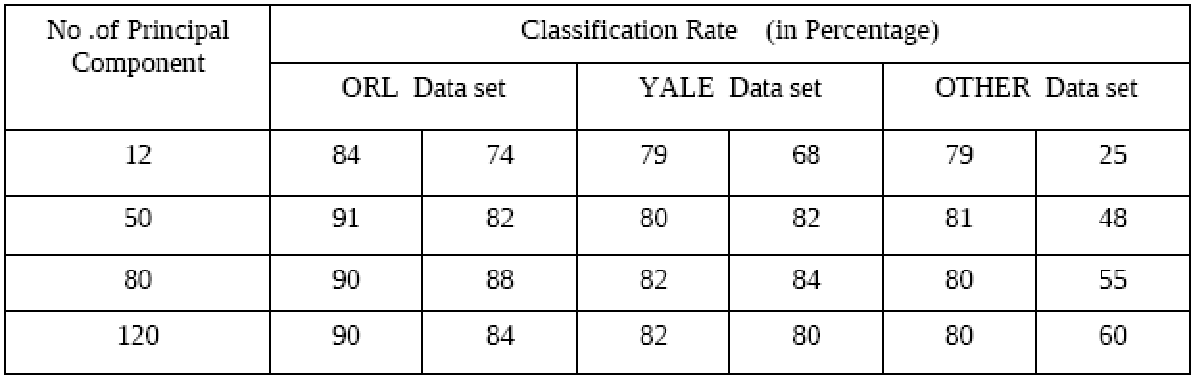
No .of Principal Component	Classification Rate (in Percentage)					
	ORL Data set		YALE Data set		OTHER Data set	
12	86	74	80	68	76	25
50	90	82	85	82	80	48
80	91	84	86	84	80	55
120	90	80	84	80	80	60

Figure 3.1: Classification Rate using our Proposed method VS usual Gram Matrix method

Now we apply the nonlinear map Φ defined as follows :

$$\Phi : (x_1, x_2) \longrightarrow (\sqrt{2}x_1, \sqrt{2}x_2\sqrt{x_1}, \sqrt{x_2}, \sqrt{2}x_1x_2, x_1^2, x_2^2) \tag{3.5}$$

in our proposed Polynomial PCA method and comparing the result with the usual Kernel PCA method using Gram Matrix, by considering a polynomial kernel of degree 2
(i.e $K(x, y) = (x^T y + 1)^2$).



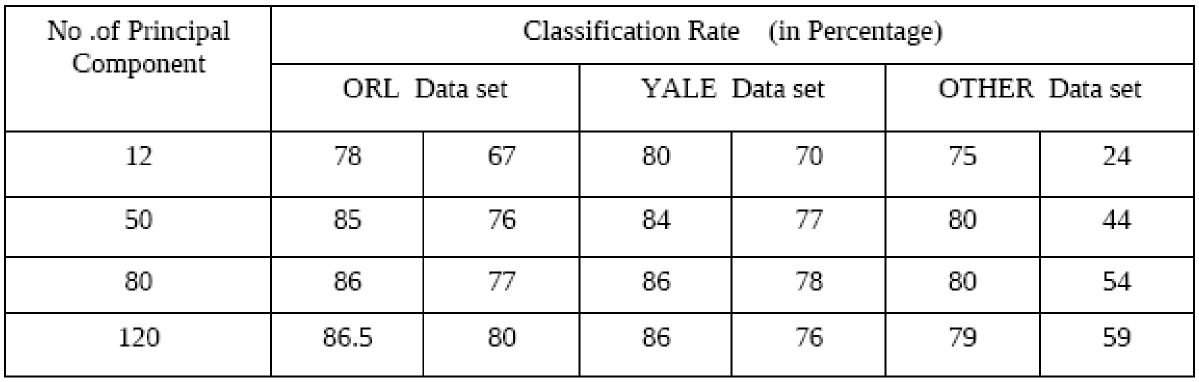
No .of Principal Component	Classification Rate (in Percentage)					
	ORL Data set		YALE Data set		OTHER Data set	
12	84	74	79	68	79	25
50	91	82	80	82	81	48
80	90	88	82	84	80	55
120	90	84	82	80	80	60

Figure 3.2: Classification Rate using our Proposed method VS usual Gram Matrix method

Now we apply the nonlinear map Φ defined as follows :

$$\Phi : (x_1, x_2) \longrightarrow (x_1, x_2, x_1x_2, x_1^2, x_2^2, , x_1^3, x_2^3,) \tag{3.6}$$

in our proposed Polynomial PCA method and comparing the result with the usual Kernel PCA method using Gram Matrix, by considering a polynomial kernel of degree 3
(i.e $K(x, y) = (x^T y + 1)^3$).



No .of Principal Component	Classification Rate (in Percentage)					
	ORL Data set		YALE Data set		OTHER Data set	
12	78	67	80	70	75	24
50	85	76	84	77	80	44
80	86	77	86	78	80	54
120	86.5	80	86	76	79	59

Figure 3.3: Classification Rate using our Proposed method VS usual Gram Matrix method

Chapter 4

New Dimensionality Reduction Method

4.1 Method

Here we are going to introduce a new dimensionality reduction method for face recognition using Polynomial PCA. In the previous chapter we have used the nonlinear map Φ on the reduced images, that means we first reduce the size of each face images and then applying the non linear map Φ over all the images(vectors).Hence we are not working with the full size images. Next we collect all the transformed images(vectors) and apply the PCA method over the nonlinear space.

We are not applying the nonlinear map Φ on the whole images because of the fact that if we consider the non linear map Φ as

$$\Phi : (x_1, x_2) \longrightarrow (x_1, x_2, x_1x_2, x_1^2, x_2^2) \tag{4.1}$$

and apply that map over an image of size 92×112 that mens over a vector v of length $(92 \times 112, 1)$,then the dimension of the transformed vector $\Phi(v)$ becomes huze. The dimension increases because of the term x_1x_2.That neans in an image if we consider pixelwise covariance i.e the covariance of each pixel with each other ,then the dimension increases.If a vector v is of dimension $(n, 1)$ where n=92×112 , then after applying the nonlinear map Φ as described above the dimension of the transformed vector $\Phi(v)$ become $(\frac{(n^2+3n)}{2}, 1)$.

Likewise if we consider 400 images of size 92×112 i.e 400 vectors $v_1, v_2, ..., v_{400}$ each of dimension $(92 \times 112, 1)$ and apply the nonlinear map Φ as in equation 4.1 over all the vectors, then in our hand 400 transformed vector $\Phi(v_1), \Phi(v_2), ..., \Phi(v_{400})$ are there each of dimension $(\frac{(n^2+3n)}{2}, 1)$ where $n = 92 \times 112$.

It is impossible to store all the images in a single matrix or it is not feasible to work with the set of vectors $\Phi(v_1), \Phi(v_2), ..., \Phi(v_{400})$.

So for working with the set of vectors $\Phi(v_1), \Phi(v_2), ..., \Phi(v_{400})$ we need to reduce the dimension of $\Phi(v_1), \Phi(v_2), ..., \Phi(v_{400})$.For that we are using a simple technique. The dimension increases because of the presence of product terms i.e because of pixelwise covariances, so if we reduce the no. of product terms in $\Phi(v_i)$ then our problem become solve.For that we consider each pixel in an image and no need to consider all the product terms (i.e covariances) with other pixels, only consider the product terms (i.e covariances) in some neighbourhood (e.x 6×6,7×7,8×8,9×9) of each pixels , rest are zero.

This is going to be a reasionable assupmption that we can make in an image(face image). i.e in an image I

Assumption : $Cov((i_1, j_1), (i_2, j_2)) = 0$ if either $|i_1 - i_2| \geq r$

or$|j_1 - j_2| \geq r$ where r=6,7,8,9

Under the above assumptions if we apply the nonlinear map Φ as described in equation 4.1 over the set of vectors $v_1, v_2, ..., v_{400}$, then we get a set of transformed vectors $\Phi(v_1), \Phi(v_2), ..., \Phi(v_{400})$ each of dimension approximately $(rn, 1)$ where r=6,7,8,9 and n=(92×112).This dimension is feasible and we able to store all the 400 transformed vectors in a matrix and it is possible to apply PCA over the set of transformed vectors.

Now in this way we are going to apply the Polynomial Kernel PCA (i.e Linear PCA on a nonlinear space or feature space) over a set of face images and do the classification.Next we compare this result with the usual Kernel PCA method using Gram Matrix.

4.2 Experiment and Result :

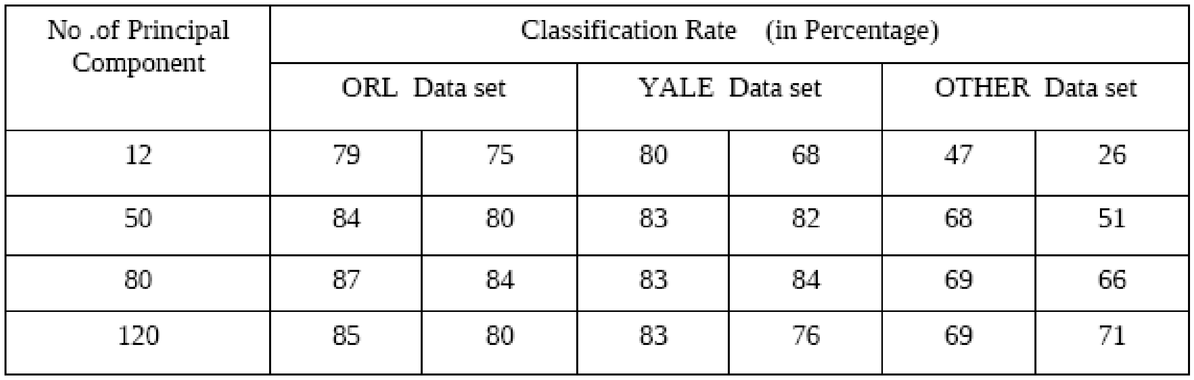
1. First we consider the nonlinear map Φ which consists only linear, squre and product terms.i.e the map

$$\Phi : (x_1, x_2) \longrightarrow (x_1, x_2, x_1x_2, x_1^2, x_2^2) \tag{4.2}$$

Now we apply this nonlinear map Φ in our proposed Polynomial PCA method under the Assumption given below (Note that here we apply the nonlinear map Φ over whole images,we are not reducing the size of each images, we just maintain the Assumption given below) and comparing the result with the usual Kernel PCA method using Gram Matrix,by considering a polynomial kernel of degree 2
(i.e $K(x, y) = (x^T y + 1)^2$).
The size of the neighbourhood can vary.Suppose we are considering 6×6 neghbourhood.Hence our Assumption is

Assumption : $Cov((i_1, j_1), (i_2, j_2)) = 0$ if either $|i_1 - i_2| \geq 6$

or$|j_1 - j_2| \geq 6$



No .of Principal Component	Classification Rate (in Percentage)					
	ORL Data set		YALE Data set		OTHER Data set	
12	79	75	80	68	47	26
50	84	80	83	82	68	51
80	87	84	83	84	69	66
120	85	80	83	76	69	71

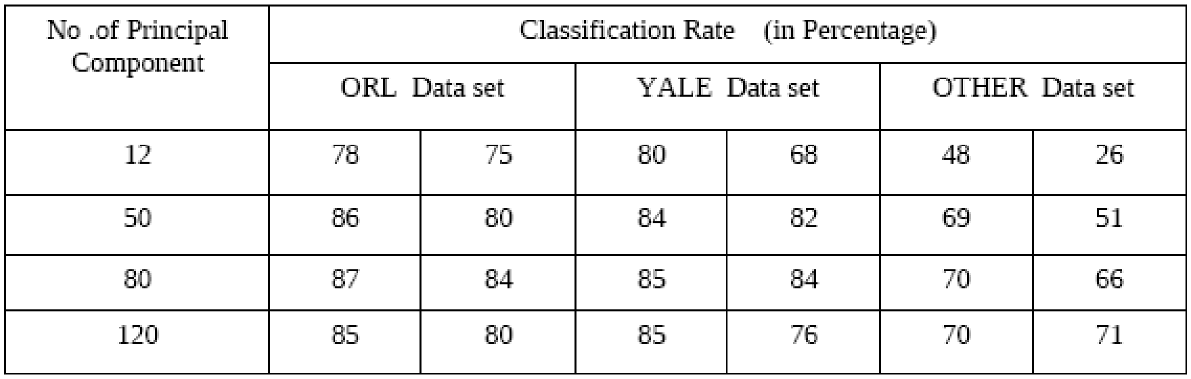
Table 4.1:Classification Rate using our Proposed method under the Assumption of 6×6 nbd VS usual Gram Matrix method

If we consider 7×7 nbd then our Assumption becomes

Assumption : $Cov((i_1, j_1), (i_2, j_2)) = 0$ if either $|i_1 - i_2| \geq 7$

or$|j_1 - j_2| \geq 7$

Under this Assumption by considering the same nonlinear map Φ as described in eq^n 4.2,the output is as follows :



No .of Principal Component	Classification Rate (in Percentage)					
	ORL Data set		YALE Data set		OTHER Data set	
12	78	75	80	68	48	26
50	86	80	84	82	69	51
80	87	84	85	84	70	66
120	85	80	85	76	70	71

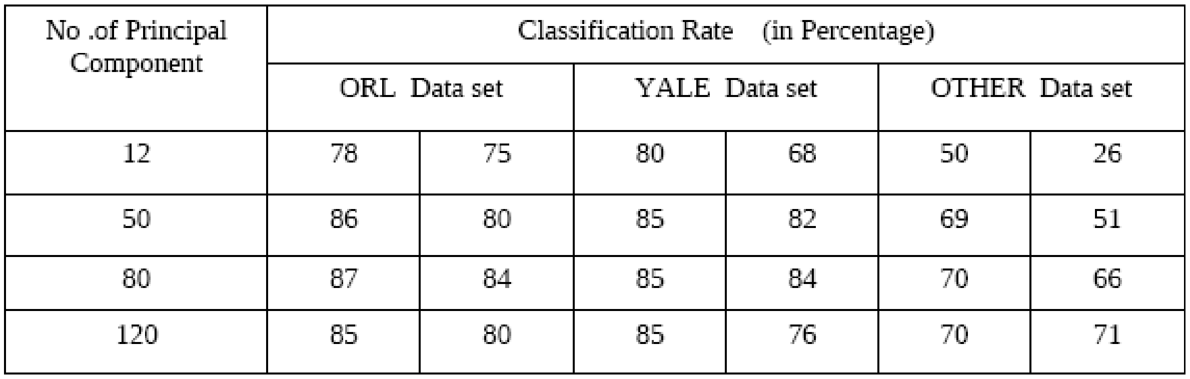
Table 4.2:Classification Rate using our Proposed method under the Assumption of 7×7 nbd VS usual Gram Matrix method

If we consider 8×8 nbd then our Assumption becomes

Assumption : $Cov((i_1, j_1), (i_2, j_2)) = 0$ if either $|i_1 - i_2| \geq 8$

or$|j_1 - j_2| \geq 8$

Under this Assumption by considering the same nonlinear map Φ as described in eq^n 4.2,the output is as follows :



No .of Principal Component	Classification Rate (in Percentage)					
	ORL Data set		YALE Data set		OTHER Data set	
12	78	75	80	68	50	26
50	86	80	85	82	69	51
80	87	84	85	84	70	66
120	85	80	85	76	70	71

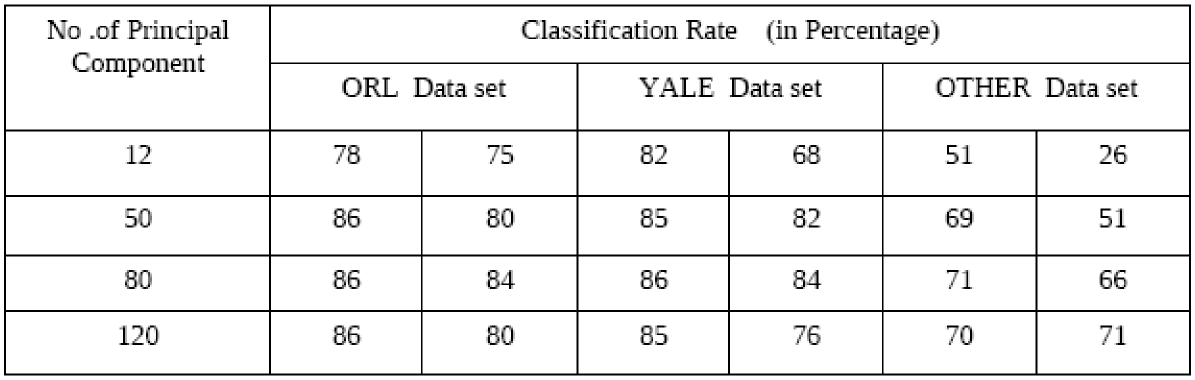
Table 4.3:Classification Rate using our Proposed method under the Assumption of 8×8 nbd VS usual Gram Matrix method

If we consider 9×9 nbd then our Assumption becomes

Assumption : $Cov((i_1, j_1), (i_2, j_2)) = 0$ if either $|i_1 - i_2| \geq 9$

or$|j_1 - j_2| \geq 9$

Under this Assumption by considering the same nonlinear map Φ as described in eq^n 4.2,the output is as follows :



No .of Principal Component	Classification Rate (in Percentage)					
	ORL Data set		YALE Data set		OTHER Data set	
12	78	75	82	68	51	26
50	86	80	85	82	69	51
80	86	84	86	84	71	66
120	86	80	85	76	70	71

Table 4.4:Classification Rate using our Proposed method under the Assumption of 9×9 nbd VS usual Gram Matrix method

2. Secondly we consider the nonlinear map Φ which consists only linear, squre,cubic and product terms.i.e the map

$$\Phi : (x_1, x_2) \longrightarrow (x_1, x_2, x_1x_2, x_1^2, x_2^2, x_1^3, x_2^3) \tag{4.3}$$

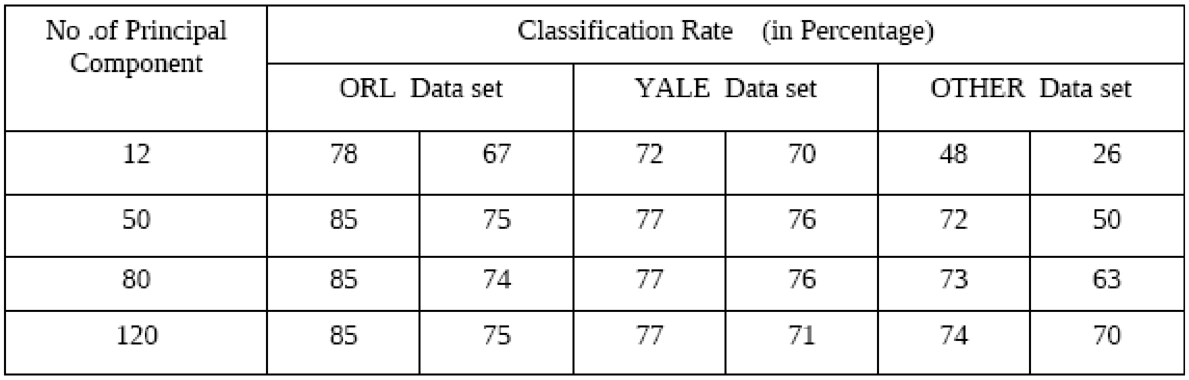
Now we apply this nonlinear map Φ in our proposed Polynomial PCA method under the Assumption given below (Note that here we apply the nonlinear map Φ over whole images,we are not reducing the size of each images, we just maintain the Assumption given below) and comparing the result with the usual Kernel PCA method using Gram Matrix,by considering a polynomial kernel of degree 3
(i.e $K(x, y) = (x^T y + 1)^3$).

The size of the neighbourhood can vary.Suppose we are considering 6×6 neghbourhood.Hence our Assumption is

Assumption : $Cov((i_1, j_1), (i_2, j_2)) = 0$ if either $|i_1 - i_2| \geq 6$

or$|j_1 - j_2| \geq 6$

Under this Assumption by considering the nonlinear map Φ as described in eq^n 4.3,the output is as follows :



No .of Principal Component	Classification Rate (in Percentage)					
	ORL Data set		YALE Data set		OTHER Data set	
12	78	67	72	70	48	26
50	85	75	77	76	72	50
80	85	74	77	76	73	63
120	85	75	77	71	74	70

Table 4.5: Classification Rate using our Proposed method under the Assumption of 6×6 nbd VS usual Gram Matrix method

If we consider 7×7, 8×8 or 9×9 nbd then our Assumption becomes

Assumption : $Cov((i_1, j_1), (i_2, j_2)) = 0$ if either $|i_1 - i_2| \geq r$

or$|j_1 - j_2| \geq r$ where r=7,8,9

Under the above Assumption by considering the same nonlinear map Φ as described in eq^n 4.3, the output does not changes i.e we are getting the same output as in table 4.5.The output changes if we change the nonlinear map Φ.

Chapter 5

Conclusion and Scope for further Reduction

In chapter 3 we proposed a Polynomial PCA method for face recognition.In this method one problem is the high dimensionality. If we want to impose some pixelwise conditions on a face image , then the dimension of that image vector become high.Then practically it is impossible to handle this type of high dimensional vectors .So in chapter 4 we proposed a new technique for reducing the dimension .Now if we apply our proposed polynomial PCA method using the new technique described in chapter4 for face recognition and do the classification using K-NN rule ,then we are getting better output(Classification Rate) from our proposed polynomial PCA method than the usual Kernel PCA method using Gram matrix.

For recognition a face we are using a set of images(Here gray level images) ,each image contains some human faces.For example if we consider ORL data set, which contains 400 face images in which 40 different peoples are there and each people have 10 different face images in different expressions. In our Proposed Polynomial PCA method we are only concern about each pixels of face images,not in the respective positions of the different parts of faces(e.x: nose ,ear,eyes,lips etc).If we are concern about the respective positions and as well as pixelwise informations then our classification rate may come much better than the previous.

Bibliography

[1] **Bernhard Scholkopf, Alexander J. Smola, Klaus-Robert Muller**

KPCA

GMD FIRST

Rudower Chaussee 5, 12489 Berlin, Germany

bs,smola,klaus@ rst.gmd.de

http:// rst.gmd.de/ bs,smola,klaus

[2] **B. Schlkopf, A. Smola, and K. Mller,** ***Non-linear component analysis as a kernel eigenvalue problem,*** Neural Comput., vol. 10, pp. 12991319, 1998.

[3] **Vikas Chandrakant Raykar, Ankur Ankur**, ***Fast Kernel Principal Component Analysis(KPCA) for the Polynomial and Gaussian kernels*** Perceptual Interfaces and Reality Laboratory Institute for Advanced Computer Studies University of Maryland, College Park February 1, 2003

[4] **M.Turk and A.Pentland. Eigenfaces for Recognition,** *Journal of Cognitive Neuroscience.*3 (1), pp 71-86, 1991.

[5] **A. M. Martinez and A.C.Kak.** ***PCA versus LDA***. In IEEE Transactions on pattern analysis and machine intelligence,Vol 23, 2001.

[6] **J. Zhang, Y. Yan, and M. Lades,** ***Face recognition: Eigenface, elastic matching, and neural nets,*** Proc. IEEE, vol. 85, pp. 14231435, Sept. 1997.

[7] **W. ZHAO, R. CHELLAPPA, P. J. PHILLIPS, A. ROSENFELD** ***Face Recognition: A Literature Survey*** University of Maryland, 2003

[8] **Elham Bagherian, Rahmita Wirza O.K. Rahmat**, ***Facial feature extraction for face recognition:*** Department of multimedia University Putra Malaysia, 43400 UPM Serdang, Selangor D.E., Malaysia elhaamb@gmail.com, rahmita@fsktm.upm.edu.my

[9] **Lindsay I Simth**,***A tuturial on Principal Component Analysis*** February 26,2002

[10] **Kwang In Kim,Keechul Jung, and Hang Joon Kim**, ***Face Recognition Using Kernel Principal Component Analysis***

[11] **K. Mller, S. Mika, G. Rtsch, K. Tsuda, and B. Schlkopf,** ***An intro duction to kernel-based learning algorithms,*** IEEE Trans. Neural Net works, vol. 12, pp. 181201, Mar. 2001.

[12] **LI-HONG ZHAO, XI-LI ZHANG, XIN-HE XU**,***Face Recognition base on KPCA with polynomial kernels*** Proceedings of the 2007 International Conference on Wavelet Analysis and Pattern Recognition, Beijing, China, 2-4 Nov. 2007