Face Recognition using Principal Component Analysis

In the language of information theory, we want to extract the relevant information in a face image, encode it as effectively as possible, and compare one face encoding with a database of models encoded similarly. A simple approach to extracting the information contained in an image of a face is to somehow capture the variation in a collection of face images, independent of any judgment of features, and use this information to encode and compare individual face images [1]. Hence in PCA we aim to find the direction where the variance is maximum.

Steps involved in training:

1. Generate the face database:

   Each face image is represented in the form a matrix having m rows and n columns, where each pixel (x,y) such that $x \in m$, and $y \in n$ shows pixel location of the image as well as the direction.

   For the simplicity we are assuming each face image as a column vector, if we have p images then the size of the face database will be mn*p.

   Let's say face database is denoted as $(\text{Face\_Db})_{mn*p}$

2. Mean Calculation:

   Calculate the mean of each observation

   $$M_i = \sum_{i=1}^{mn} \sum_{j=1}^{p} Face\_Db(i,j)$$

   here mean vector will have the dimension $(M)_{mn*1}$

3. Do mean Zero

   Subtract mean face from each face image, let's say this mean zero face data as Δ, and calculated as

   $$(\Delta(i))_{mn*p} = (\text{Face\_Db}(i))_{mn*p} - (M)_{mn*1}$$

   Where $i \in 1,2,3 \ldots\ldots, p$.

4. Calculate Co-Variance of the Mean aligned faces (Δ)

   Here there is a slightly variation in calculation of covariance, generally we prefer to calculate the covariance of data by:

   $$C = \sum_{i=1}^{n} (X_i - \overline{X})\ ((Y_i - \overline{Y}))^t$$

Where $\overline{X}, \overline{Y}$ *are the mean of* X_i *and* Y_i, and C is the covariance matrix. If we will follow the same convention on face data we will get.

$$C\ (mn,mn) = \sum_{z=1}^{mn} \sum_{y=1}^{mn} \sum_{i=1}^{p} (\Delta(z,i) - M_z) * (\Delta(z,i) - M_{y)^t}$$

Here we will get mn direction, which is very hard to compute, store and process. It also increases the program complexity, hence in 1991 Turk and Peterland [1] two researches suggested a new way to calculate the co-variance that is basically known as surrogate covariance, that is:

$$C\ (p,p) = \sum_{z=1}^{mn} \sum_{y=1}^{mn} \sum_{i=1}^{p} (\Delta(z,i) - M_{y)^t} * \ (\Delta(z,i) - M_z)$$

Hence here will get only p * p dimension, which is easy to compute and process, the idea behind computing the surrogate covariance suggested by turk and peterland that, these are only the valid direction where we will get maximum variances, and rest of the directions are insignificant to us. Menas these are direction where we will get the eigenvalues and for rest we will get eigenvalues equal to zero.

5. Do eigenvalue and eigenvector decomposition:

   Now we have covariance matrix $(C)_{p*p}$, *find out the eigenvectors and eigenvalues.* Let we have eigenvector $(V)_{p*p}$ *and eigenvalues* $(\lambda)_{p*p}$.

6. Find the best direction (Generation of feature vectors)

   Now select the best direction from p directions, for this sort the eigenvalues in the descending order and decide a k value, which represents the number of selected eigenvectors to extract k direction from all p direction. On the basis of k value we can generate the Feature vector $(\Psi)_{p*k}$.

7. Generating Eigenfaces:

   For generating the eigenfaces (Φ) project the each mean aligned face to the generated feature vector.

$$(\Phi)_{k*mn} = (\Psi)^t{}_{k*p} * (\Delta)^t{}_{p*mn}$$

8. Generate Signature of Each Face:

   For generating signature of each face (ω), project each mean aligned face to the eigenfaces.

$$(\omega)_{k*i} = (\Phi)_{k*mn} * (\Delta)_{mn*i}$$

Where i ∈ 1, 2, 3, ….., p. hence ω will have the size k * p.

Steps involved in Testing:

1. For testing, let we have an image (I), make it as a column vector say $(I)^1_{mn*1}$
2. Do mean Zero, by subtracting mean face (M) to this test face, say it $(I)^2$.

$$(I)^2_{mn*1} = (I)^1_{mn*1} - (M)_{mn*1}$$

3. Project this mean aligned face $(I)^2$ to eigenfaces (Φ), we will get the projected test face (Ω).

$$(\Omega)_{k*1} = (\Phi)_{k*mn} * (I)^2_{mn*1}$$

4. Now we have projected test face Ω and signature of each face Φ, calculate the Euclidean distance between Ω and each column of Φ. We will end up with 100 distances, whatever the minimum distance; assign that test face to that person's class.

Note: Generate face database as each person should have at least 10 image, having some orientation and expression change, divide it into the training and test set, keeping that thing in mind that training set ∩ test set = Φ (null).

Take 60% data as training set and 40 % data as test set, evaluate your classifier on the given factors.

a) Change the value of k and then, see how it changes the classification accuracy. Plot a graph between accuracy and k value to show the comparative study.
b) Add imposters (who do not belong to the training set) into the test set and then recognize it as the not enrolled person.

Reference

[1] Turk, M.A.; Pentland, A.P.; "Face recognition using eigenfaces," *Computer Vision and Pattern Recognition, 1991. Proceedings CVPR '91. IEEE Computer Society Conference on*, vol., no., pp.586-591, 3-6 Jun 19915.

[2] M. Kirby, L. Sirovich, Application of the Karhunen-Loeve Procedure for the Characterization of Human Faces, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 12, No. 1, January 1990, pp. 103-108.