Face Recognition Using PCA

Submitted by:
Mohit Kumar Ahuja
Solene Guillaume

Submitted to:
Prof. Desire Sidibe

University of Burgundy
2016-2017

ABSTRACT

In recent years, the demand for face recognition systems has increased very much for identification and the verification purposes. There are different systems like Fingerprint recognition, Iris recognition, Retina recognition, voice recognition and so on. Face recognition, in particular has received a considerable attention in recent years for identification and the verification purposes because of its applications. At present, there are many methods for face recognition.

Applied Math's concepts play crucial role in this time to make the technology easy and efficient. For instance google page ranking using linear algebra, for face recognition using PCA and SVD. Here in our project, we used Principal Components Analysis (PCA) and Singular value decomposition (SVD) for face recognition. In PCA, we find eigenvalues and eigenvectors of the covariance matrix of the set of face images by using the Eigen vectors. We are able to detect the face of the person.

Table of Contents

Chapter 1

Introduction

1.1 Face Recognition

Face recognition (FR) has always remained a major focus because we identify people through their faces, which we can say it as the primary method. Actually, we can ask something's like this, what is Face Recognition? Identifying a person by his face from previous knowledge. Why do we use Face Recognition? To identify or determine people's identity. Obviously, we can say that there are many different identification technologies like fingerprints, user id and password, retina and iris recognition etc. so why do we use only this recognition when we have others? Because when we use most of the other technologies like the above given ones you should stop and press the fingers or enter the password or show your eye but FR can be done while you are walking in or out i.e. there is no need to stop and show your face. Where do we use? The most useful applications are crowd surveillance, video content indexing, personal identification (ex. driver's license), entrance security, etc.

1.2 Principal Component Analysis

Principal Components Analysis (PCA) is a practical and standard statistical tool in modern data analysis that has found application in different areas such as face recognition, image compression and neuroscience. It has been called one of the most precious results from applied linear algebra. The Principal Component Analysis (PCA) is one of the most successful techniques that have been used in image recognition and compression. PCA can be used for many reasons such as Simplification, data Reduction, modeling, outlier detection, variable selection, classification, prediction and so on. But in here the purpose of PCA is to reduce the large dimensionality of the data space (observed variables) to the smaller intrinsic dimensionality of feature space (independent variables), which is needed to describe the data economically.

Singular Value Decomposition

The Singular Value Decomposition (SVD) is a widely used technique to decompose a matrix into several component matrices, exposing many of the useful and interesting properties of the original matrix. The decomposition of a matrix is often called Factorization.

In SVD, we decompose a real matrix A € **R** $m \times n (m > n)$ can be written as the product of three matrices as follows:

$$A = U S V^T$$

The two matrices U €**R** $m \times n$ and V € **R** $n \times n$ are orthonormal. The diagonal elements of S € **R** $n \times n$ are the singular values. The column vectors of U and V are the corresponding singular vectors of each singular value.

Face recognition is one example where principal component analysis has been extensively used. The choice of the Singular Value Decomposition (SVD) is very important due to its vast use in the Principal Component Analysis and because it largely defines its performance.

Chapter 2

Methodology

2.1 Normalization

The normalization steps are done for the scaling, orientation and location variation adjustment to all the images with some predefined features and the feature of images. Normally, all the images are set to a 64x64 window taking some of the important facial features. In here, we will use the following facial features: (1) left eye center, (2) right eye center, (3) tip of nose, (4) left mouth corner, and (5) right mouth corner.

2.2 Algorithm for Normalization

Step 1: We take the predefined feature co-ordinates

Step 2: We consider all the features f_i 's and then we calculate the affine transformation by the equation

$$f_i^p = Af_i + b$$

Where A and b are 6 unknowns

Step 3: We update $\overline{F}$ each time by using SVD

Step 4: We take the average of all the aligned feature locations for each face image and update $\overline{F}$ with this value

Step 5: we compare F_t and.F_t-1 the difference is less than a threshold, then stop and come out of the loop and the final converged $\overline{F}$ gives the affine transformation matrix A and vector b; otherwise, go to step 2.

2.3 Recognition

First we put all the images in the training set and the training set in stored in a single matrix D

$$D = \begin{bmatrix} I_1(1,1) & I_1(1,2) & \dots & I_1(1,N) & \dots & I_1(M,1) & I_1(M,2) & \dots & I_1(M,N) \\ I_2(1,1) & I_2(1,2) & \dots & I_2(1,N) & \dots & I_2(M,1) & I_2(M,2) & \dots & I_2(M,N) \\ & & \vdots & & \vdots & & \vdots & & \\ I_p(1,1) & I_p(1,2) & \dots & I_p(1,N) & \dots & I_p(M,1) & I_p(M,2) & \dots & I_p(M,N) \end{bmatrix}_{p \times d}$$

Now, we compute the covariance matrix of D by

$$\sum = \frac{1}{\mathrm{p}-1} D^T \mathrm{D}$$

Now, we compute the eigenvalues and eigenvectors of D and keep the eigenvectors corresponding to the largest eigenvalues. Each eigenvector has the same dimensionality (number of components) as the original images, and thus can itself be seen as an image. The eigenvectors of this covariance matrix are called Eigen faces. Here we have few samples and many variables. This means that the number of nonzero eigenvalues of the covariance matrix is limited to training images. Therefore, instead of computing the eigenvectors of the very large covariance matrix, we can just compute the eigenvectors of $\Sigma' = \frac{1}{p-1} DD^T$ which is a p x p matrix.

In PCA, we basically try to find eigenvalues and eigenvectors of the covariance matrix, D. We showed that D = (AA^T) / (n-1), and thus finding the eigenvalues and eigenvectors of D is the same as finding the eigenvalues and eigenvectors of AA^T.

In SVD, decompose a matrix A as follows:

$$A = U S V^T$$

Show that:

$$AA^T = U S^2 U^T$$

Where the columns of U contain the eigenvectors of AA^T and the eigenvalues of AA^T are the squares of the singular values in S

Thus SVD gives us the eigenvectors and eigenvalues that we need for PCA.

2.3.1 Recognizing a face

The face is expressed in the face space by its Eigen face coefficients (or weights).We can handle a large input vector now, facial image, only by taking its small weight vector in the face space.

In the recognition phase, we give an image of a known person. The image is labeled with name of the person. First, image is represented by a vector Xj as in the training phase and this vector is projected into the PCA space to get a corresponding feature vector

$$\phi_j = X_j \Phi .$$

To identify, we have to compute a similarity, for example Euclidean distance, between corresponding feature vector and all feature vectors in the training set. The image whose feature vector is most similar to the corresponding feature vector is the output of the face recognition system. If the Euclidean distance between them is equal, then we have correctly identified the person. Otherwise, we have mismatch.

2.4 Algorithm for Recognition

Step 1: Set the number of Principal components

Step 2: Build a training data matrix D from training images

Step 3: We assign name as the label for each training image and form a labels matrix Lt

Step 4: Remove the mean and compute covariance matrix (D). Then compute the PCA of D, which gives PCA transformation matrix

Step 5: Feature vectors of all training images using transformation matrix and store them in a matrix Ft

Step 6: Read test images, and for each test image Iq

– Calculate the feature vector ⍉q using the PCA transformation matrix

– Compute the distance between ⍉q and all training feature vectors, which are stored in Ft

– Find the best match Iz (minimum distance between feature vectors)

– Check if the label of Iq is the same as the label of Iz. Otherwise, increment an error count € by 1.

Step 7: We find the Accuracy

$$accuracy = \left(1 - \frac{\epsilon}{total\ number\ of\ test\ images}\right) * 100$$

Chapter 3

Implementation

3.1 Implementation

We created a student database and which contains their photographs and here we are working with 81 training images. We have taken the mean of all the 81 training faces. We have subtracted these values from each face. We concatenate all the subtracted faces in a single matrix D.

$$D = \begin{bmatrix} I_1(1,1) & I_1(1,2)\ldots & \ldots\ldots\ldots & \ldots..I_1(64,64) \\ I_2(1,1) & I_2(1,2)\ldots & \ldots\ldots\ldots & \ldots..I_2(64,64) \\ \vdots & \vdots & \vdots & \vdots \\ I_{81}(1,1) & & & I_{81}(64,64) \end{bmatrix}$$

We just compute the eigenvectors of $\Sigma' = \frac{1}{p-1} DD^T$. This reduces the dimension to 81x81 in this case. The Eigenvector computed from this matrix will be of size 81x81. Next, we find the Eigen faces. Each face in the training set can be represented as a linear combination of these eigenvectors.

$$\phi_i = X_i \Phi$$

Where Φ are the principal components and *Xi* is the training images

Now, we normalize the image and project this normalized image into the PCA space to get a corresponding feature vector

$$\phi_i = X_i \Phi$$

Now, we calculate the Euclidean Distance between ϕ_j and all feature vectors ϕ_i's in the training set. The image whose, feature vector is similar and, distance is minimum to ϕ_j will be the output of the face recognition system.

Chapter 4

Results

4.1 Normalization

Normalization plays key role in face recognition. In this normalization it will take all the features average to proceed further. But here what we have observed is when taking the F'(F_bar) as initial features of the first image we are getting new normalized images. When trying to match in order obtain the accuracy we are getting different accuracy when we are changing initial features. For instance when we are taking the initial features of dataset (Mohit1) we are getting accuracy around 75.70. But when are changing the initially predefined features we are getting accuracy change. In order to conform ourselves we changed the initial features which are given in the database. For other defined features we observed that accuracy is increasing and decreasing. This time we changed our initial features of other image (Dani2) we observed that accuracy is around 78.9. For other image we get less accuracy around 44.4 (Yemen5) and for Eza4 we got 74.49%.

One point we observed by doing series of experiments we are changing the initial features of while doing normalization are accuracy is not constant with respect to predefined features which we done manually.

We try to figure out the where the problem lies but we unable to make that.

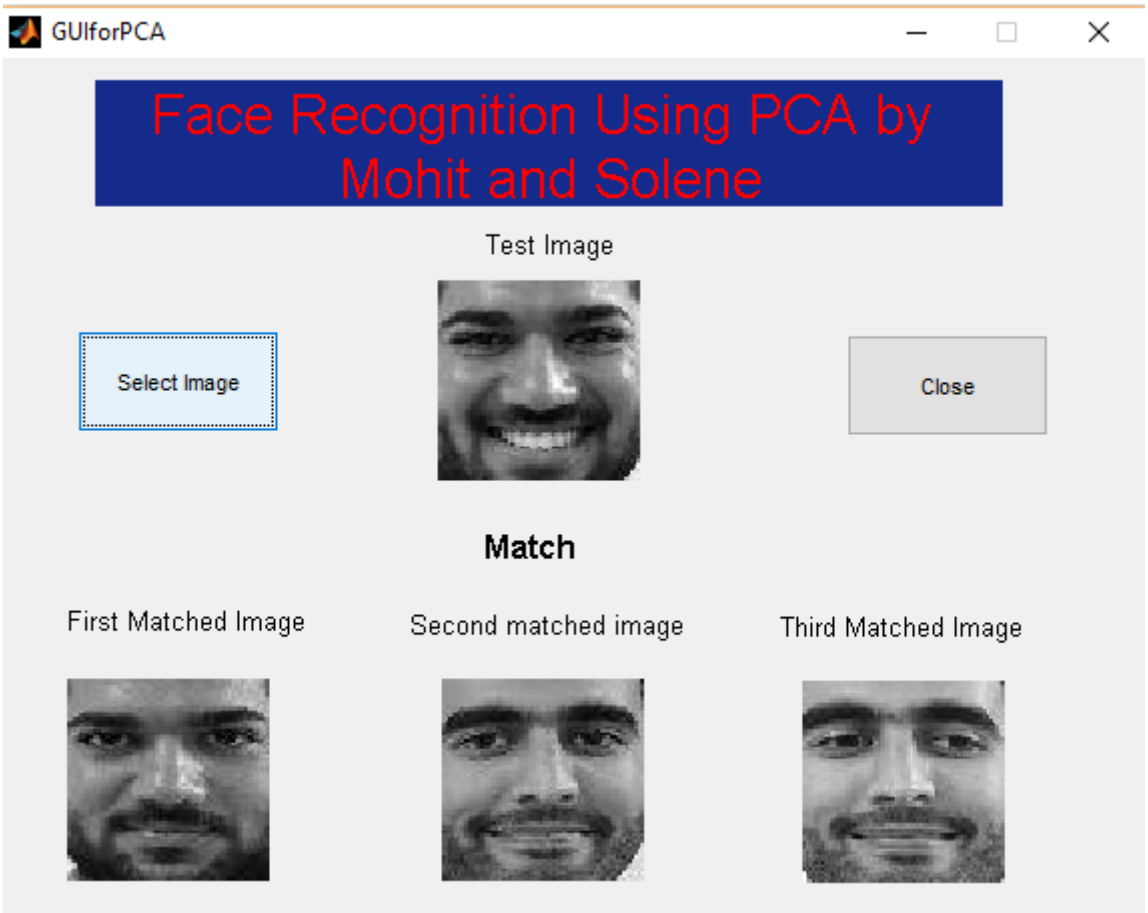
Normalized images look like below images:

Output Figure after Normalization

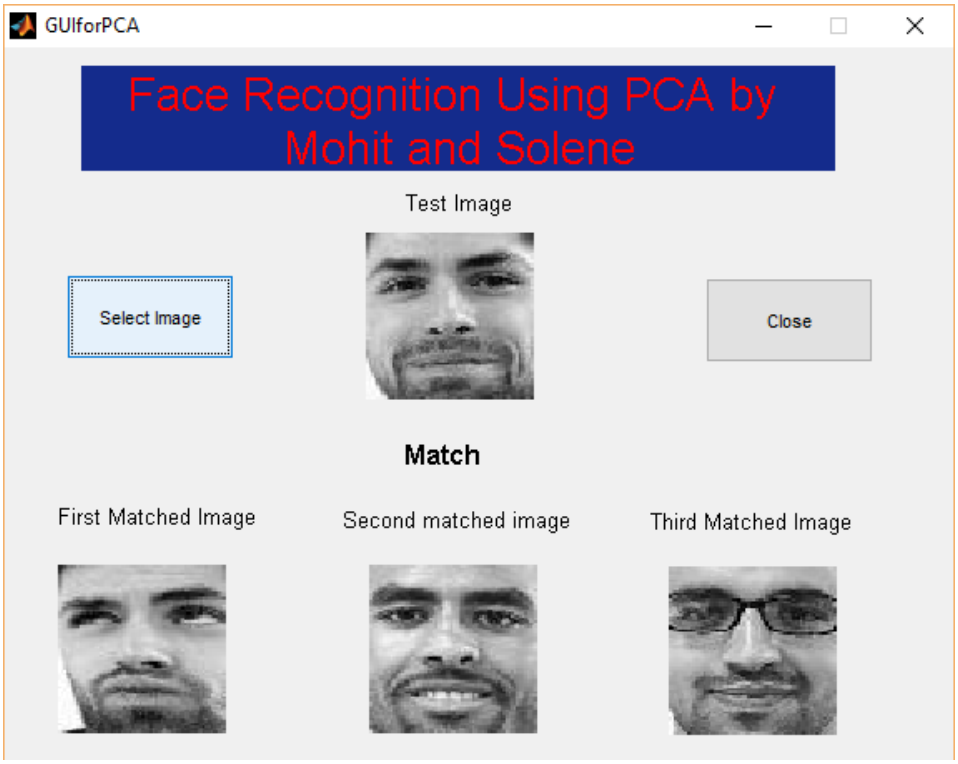
4.2 Face Recognition

In this recognition part after coding the PCA algorithm we did couple of experiment. First we did our experiment how efficient our PCA algorithm is at that point we observe out accuracy is around 76% we find 12 mismatches from the entire dataset (of our class). After doing the normalization of the images we find that accuracy depends on the initial features. In the below images we can observe the matches and mismatches in the both cases.

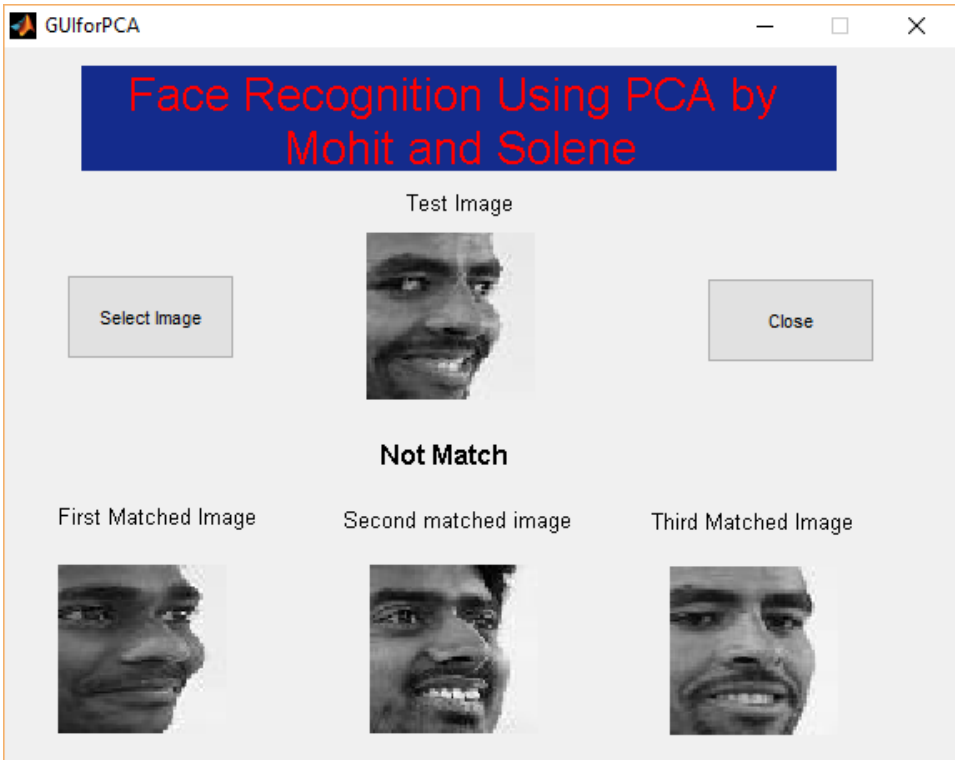
PCA with normalization works well.



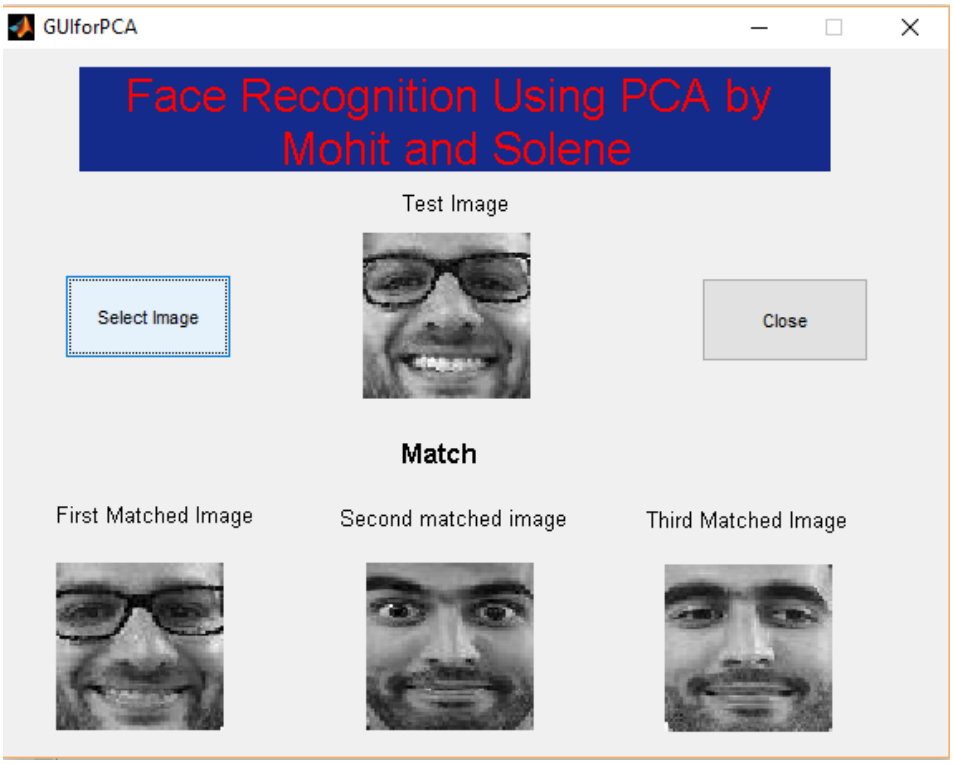
(a) First result with exact match



(b) Second result with exact match



(c) Third result which was Not-matched



(d) Forth result with exact match

4.3 Accuracy

Here we are giving the accuracy of the image with different predefined features

S.No	Predefined Features	Accuracy
1.	Mohit1 (Initial features)	75.70%
2	Dani2	78.9%
3	Yemen5	44.4%
4.	Eza4	74.49

For the data base images with doing normalization we got accuracy around 64%(on average)

Chapter 5

Conclusion

5.1 Conclusion

Principal component Analysis can be used for many purposes we found some of them are to decrease the computational complexity and measure of the covariance between the images. PCA reduces the complexity of computation when there is large number of database of images. These principal components of the Eigen vector of this covariance matrix when concatenated and converted gives the Eigen faces. These Eigen faces are the ghostly faces of the trained set of faces form a face space. This distance gives the location of the image in the Eigen space, which is taken as the output matched image.

5.2 What We Learned

1. If the Initial Features (F_bar) are changed accuracy changes
2. Affine Transformations are used to map images
3. Image can be represented in a vector form using reshae in MATLAB
4. If the intensity of the image pixels correlation increases it reduces its dimensions reduce by projecting along the vector using PCA
5. Data of the give images are easy to do face reorganization but some false data causes troubles
6. If we take covariance matrix as A*A' it leads to higher dimensions takes more time for computing
7. Learned many inbuilt matlab commands and their uses.
8. If there is any wrong data in the given images responsible for the accuracy.
9. Shows an example how applied math's is used in various fields.

5.3 Future Works

1. To find why we are getting different accuracy for different predefined features and steps to fix them.
2. To investigate the steps to get better accuracy.

References

1. Applied Mathematics Homework 4 by DésiréSidibé
2. Springer series in statistics by I.T.Jolliffe
3. Principal Component Analysis by SVANTE WOLD
4. Using the Singular Value Decomposition by Emmett J. Ientilucci
5. Wikipedia.org/wiki/Eigenface
6. Principal Components Analysis (PCA) by FaezehSalehi and Athena Ahmadi
7. Face Recognition using Principle Component Analysis by Kyungnam Kim