ES 331 Probability and Random Processes Report
Assignment 1 - Face Recognition using Eigenfaces

S Deepak Narayanan, 16110142

Abstract:

The entire motive of this assignment is to use PCA to predict faces from a known dataset of faces. We use a smart way of computing the most important or dominant eigenvectors in this case by a crucial observation. The rest of this report contains the obtained results.

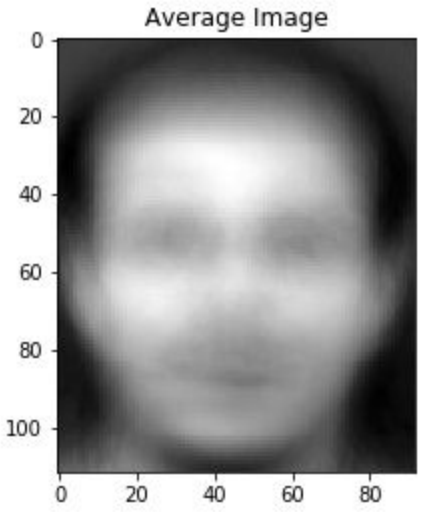
Procedure:

1. Firstly we convert the entire dataset that we have into a long vector.
2. After doing the above step, we then have a set of vectors, which are subsequently divided into two, the train set and the test set. After this particular division, we then make sure that we are handling with the train set.
3. Now in the train set, we perform the most crucial principal component analysis.
4. Here in the train set, first we compute the average long vector among all the possible long vectors that we have.
5. After computing the average long vector, we then go ahead and subtract this mean from each and every other vector present in the dataset. After this particular step is over, we then go ahead and compute the modified covariance matrix as defined below.
6. This particular matrix isn't really the covariance matrix, it is the multiplication of A times A transpose. The subtle observation here is the simple fact that both of them have the same eigenvalues, but for different eigenvectors.
7. We exploit this and compute the top M eigenvectors of the matrix. Here M is the M in the dimension of the modified covariance matrix.
8. After this, we multiply this matrix by the A matrix, which is essentially made of all the images in the training set offset by a particular factor as vectors.
9. We have our eigenfaces. These matrices, in essence, are the eigenfaces, that have a ghostly shape of a face.
10. Now, we compute a vector W for each image. Each term of W is computed as the product of the eigenface vector transpose times the input image, offset by the training mean. This is a characteristic of any image that is an input to our algorithm. We convert it to a vector, then offset it by the mean(basically subtract the mean) and then compute the W vector for the image.
11. We then compute how far these images are from the input images by directly computing the L2 Norm or the Euclidean Norm of the difference between different image classes and the corresponding image.
12. The lowest norm must belong to a class, we report that the image belongs to that class if the lowest norm value is less than a threshold. This is found emperically.
13. We need to also check if the image is a face in itself. That is also done similarly, yet our projection space's distance from the image is computed, instead of W.
14. We first compute the standard vector after lengthening and offsetting and compute the norm between this image and the projection in the face space, that is the space that is

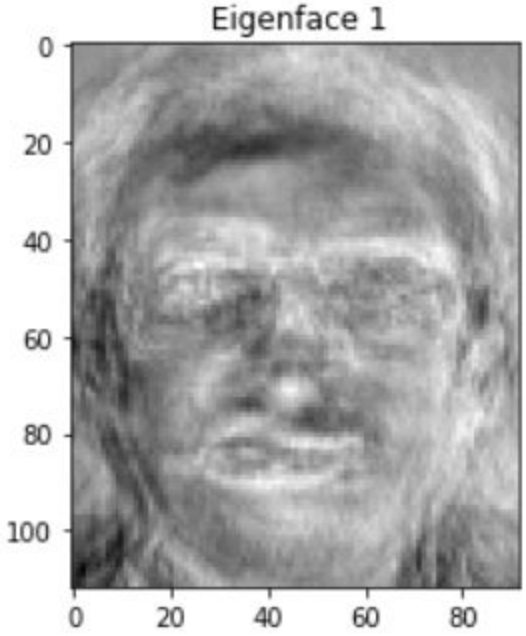
spanned by the eigenfaces. Again, by using a similar binary thresholding, we will identify if the image is a face or not.

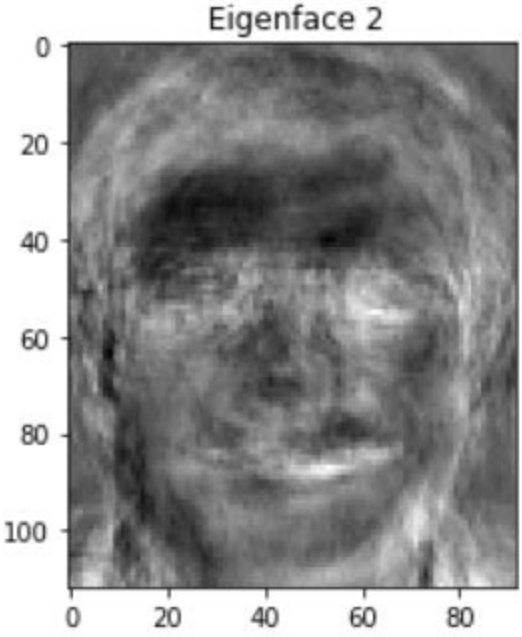
Results:

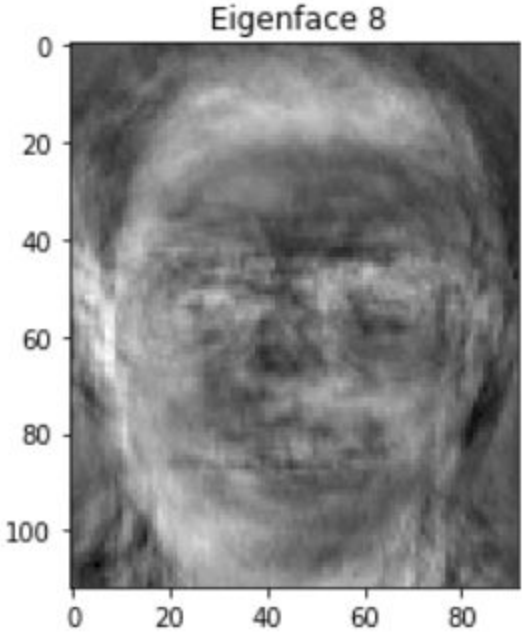
The average face was found out to be:

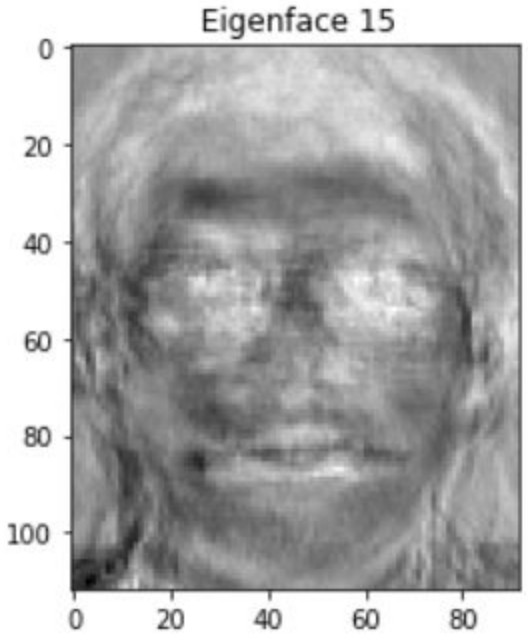


A few of the eigenfaces found were




Eigenface 2
0
20
40
60
80
100
0
20
40
60
80


Eigenface 8
0
20
40
60
80
100
0
20
40
60
80



Conclusion:

1. We use a method that depends on the dataset. For instance is the background is present in excess, there has not been any use of a Haar Cascade type of a mechanism that is in place for this algorithm. Also, it is to be noted that we have this mehtod working only on the given dataset.

References:

1. https://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html
2. http://www.vision.jhu.edu/teaching/vision08/Handouts/case_study_pca1.pdf
3. http://www.face-rec.org/algorithms/pca/jcn.pdf