Eigenfaces for Recognition: Implementation and Review

BDA 492: Data Mining and its Applications
Assignment 1

Manav Prabhakar

July 2021

Abstract

Face recognition can be defined as a method to identify or verify the identity of an individual using their face. For a long time, researchers have been proposing numerous ways in order to optimize face recognition both in terms of computational efficiency and precision or accuracy. We discuss one such approach, the "EigenFace" for facial detection here. Inspired from the theory of Linear Algebra, the idea of eigenvectors, basis and eigenspace, Mathew Turk and Alex Petland had proposed a mechanism for detecting and classifying faces. We have implemented and shall review their approach.

Contents

1 Introduction

The Face plays a major role in our social interaction. Humans use their ability to recognize and classify faces extensively to remember and/or recognize people. We have a one-short learning mechanism and this is what helps us to remember faces unlike traditional Machine Learning and Deep Learning methods which require large datasets to attain good accuracy on unseen datasets. Further, the uses of facial recognition are manifold. It is widely used in surveillance systems, criminal identification, image and video editing tools etc. Both detecting a face and classifying different faces are equally important as both have their relevant use cases. Mathew Turk and Alex Petland [1] have proposed a pre-attentive pattern recognition approach which is independent of three-dimensional geometry. Their model is fast, simple and accurate particularly in constrained environments such as an office or household. Their model is based on an information theory approach that decomposes face images to a small set of characteristic feature images which they have dubbed as "eigenfaces".

2 The Dataset

For implementation purposes, 2 datasets were used.

- Flickr-Faces-HQ Dataset (FFHQ).
- Yale Face Database

Apart from these datasets, a collection of 16 non-face images were used for face vs non-face classification purposes.

2.1 Flickr-Faces-HQ (FFHQ)

It is a high-quality image dataset of human faces, originally created as a benchmark for generative adversarial networks (GAN) [3]. The dataset consists of 70,000 high-quality PNG images at 1024×1024 resolution and contains considerable variation in terms of age, ethnicity and image background. The images were crawled from Flickr, thus inheriting all the biases of that website. Out of these 70,000 images, we used 1000 of them as a training set. This dataset does not classify images belonging to the same person and hence, it was used only for identifying face images from non-face images.

2.2 Yale Face Database

This is similar to Olliveti faces in terms of its composition. It contains 165 grayscale images in GIF format of 15 individuals. There are 11 images per subject, one per different facial expression or configuration: centre-light, w/glasses, happy, left-light, w/no glasses, normal,

right-light, sad, sleepy, surprised, and wink. We use this dataset for both face and non-face classification and subject-wise classification [2].

3 The Approach and Implementation

We first focus on extracting the relevant information/features in a face image, encode it as efficiently as possible and compare this encoding to a dataset with similarly encoded images. In terms of mathematics, we aim at finding the principal components of the distribution of faces, or the eigenvectors of the covariance matrix of the set of face images, treating an image as a point in a higher dimensional space. The eigenvectors can be modeled as the features extracted from the image that characterize the variations between face images.

Attaining eigenfaces:

- Obtain a training set (comprising of face images belonging to different people).
- Calculating the mean face and the principal components (eigenfaces) for the given dataset.
- Calculate the eigenfaces from the training set.
- Store the top K eigenfaces that could span the face space with the least amount of error. These eigenfaces may be changed/updated when the model is encountered with new unseen faces.

3.1 Obtaining Training Set

Face images were obtained from the three datasets and implementation was carried out separately on each dataset. All images were resized to appropriate sizes so as to manage the computational resources while minimizing loss of information due to resizing.

3.2 Mean Face and Principal Components

The next task was to find the mean face and the Principle Components for the face space.

Consider each image is of size (n, n), and we have M such images. Now we unroll the images to shape $(1, n^2)$. So if the image was of the size $(128, 128)$, we now have it of size $(1, 16384)$.

Note: This change in shape has been done just for ease of implementation and is just another way of representing the image and does not account to dimensionality reduction.

Let the training set of face images F be $[F_1, F_2, F_3, ..., F_M]$. with F_i of shape and each image be of size (n, n). Then F will be of shape (M, n^2) (after unrolling).

Then we can define the average face for the dataset as

$$F_{avg} = \frac{1}{M} \times \sum_{i=1}^{i=M} F_i \tag{1}$$

Figure 2 shows the mean face for the datasets used.

Figure 1: **Left:** First 25 training examples from Flickr Dataset showing the images for distinct people. **Right:** 25 training examples from the Yale Faces. Images for 3 subjects can be seen here

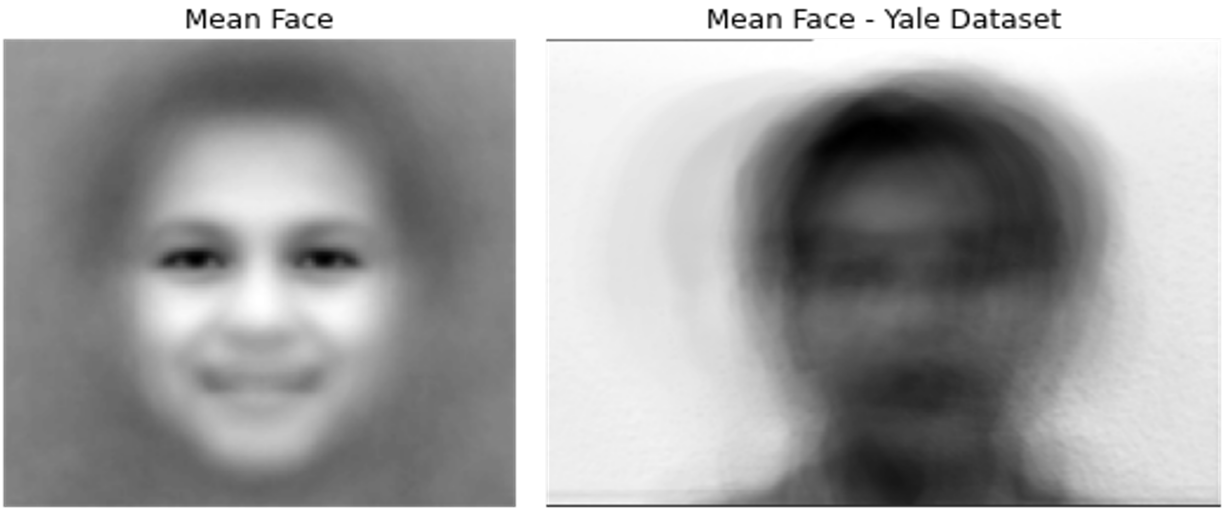

Figure 2: Mean faces obtained for Flickr Dataset (**left**) and for the Yale Dataset (**right**)

Next, we compute the mean subtracted data, Φ by subtracting the mean face from all other faces in the dataset. This is for the purpose of normalization or centring the data on around F_{avg}.

$$\Phi_i = F_i - F_{avg} \tag{2}$$

Now Φ is of the shape (M, n^2)

We now subject Φ to principal component analysis (PCA) to find the set of K vectors that best describe the data.

For PCA, we first find the Covariance Matrix, Θ

$$\Theta = \Phi^T \Phi \tag{3}$$

This, will however result in (n^2, n^2) matrix. Finding eigenvalues and eigenvectors for this will be computationally very expensive. However, we have a workaround for this. We will define our covariance matrix as $\Phi\dot{\Phi}^T$. This will result in an (M, M) matrix. Since, $M << n^2$, computing the the eigenvectors and eigenvalues for these will be simpler. Consider eigenvectors v_i and eigenvalues λ_i such that

$$\Theta_{modified} v_i = \lambda_i v_i \tag{4}$$

where, $\Theta_{modified} = \Phi\Phi^T$

$$\Phi\Phi^T v_i = \lambda_i v_i \tag{5}$$

3.3 Calculating eigenfaces from the training set

Pre-multiplying with Φ^T in equation 5

$$\Phi^T_{(n^2 \times M)} \Phi_{(M \times n^2)} \Phi^T_{(n^2 \times M)} v_i = \lambda_i \Phi^T_{n^2 \times M} v_i \tag{6}$$

From equation 6, we can see that $\Phi^T v_i$ act as eigenvectors to $\Theta = \Phi^T \Phi$

Let $\Phi^T v_i = u_i$, then we have

$$\Phi^T \Phi u_i = \lambda u_i \tag{7}$$

$$\Theta u_i = \lambda u_i \tag{8}$$

Thus, we can determine the linear combination of K training set face images to form eigenfaces (see Figure 3) u_k as

$$u_k = \sum_{i=1}^{M} \Phi_i^T v_{ki} \tag{9}$$

$$\Phi = W u_k^T \tag{10}$$

$$\Phi u_k = W(u_k^T u_k) \tag{11}$$

Since, $(u_k^T u_k)$ is a symmetric matrix it's inverse will exist, we post-multiply with its inverse to obtain W.

$$\Phi u_k (u_k^T u_k)^{-1} = W \tag{12}$$

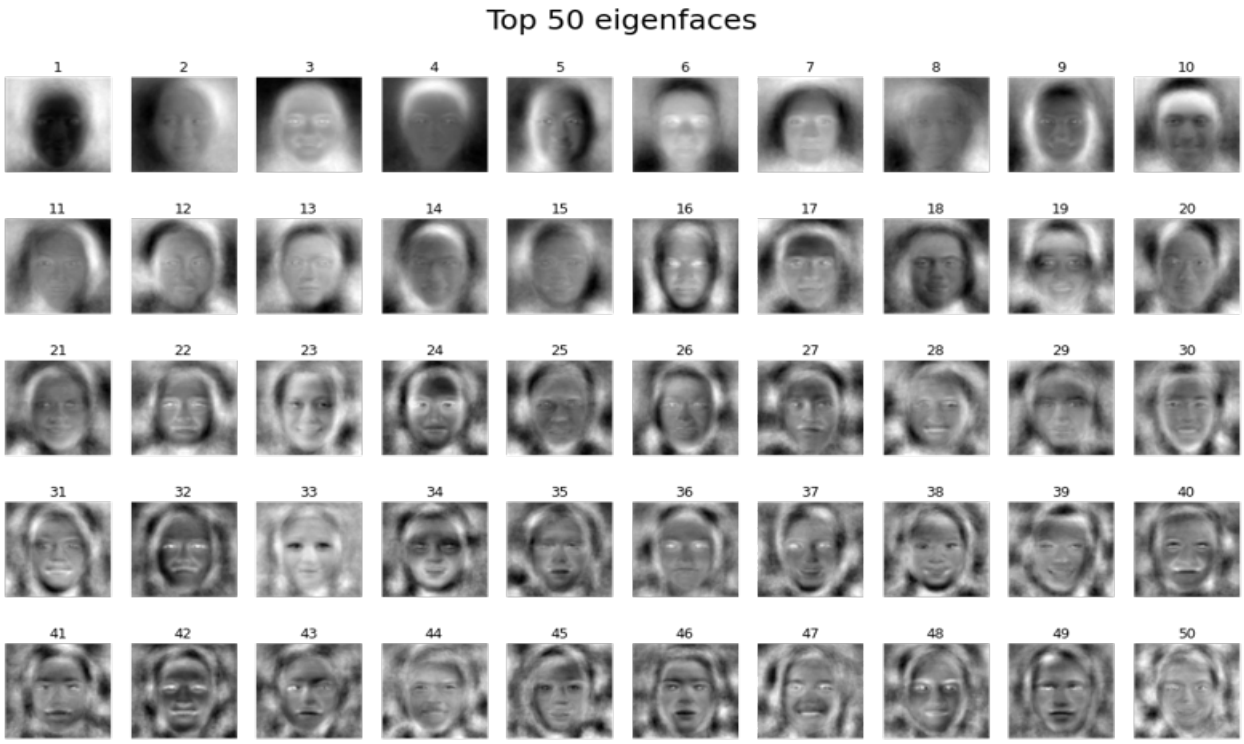

Figure 3: Eigenfaces obtained for Flickr Dataset

3.4 Storing the top K eigenfaces that span the face space

We will select the top K eigenfaces ($K < M$) that span the face space with minimum error. If we select all M eigenfaces, we will be able to denote any image with no loss of information and zero error. On the contrary, when we take $< M$ eigenfaces, we are subject to some loss of information.

Once, we have obtained the eigenfaces, we can store these and use it for further classification.

4 Classification of Images

- Face vs Non-Face image.
- Subject-wise face classification

4.1 Face vs Non-Face

Here, we describe how do we identify face vs non-face images with the help of the eigenfaces that we obtained.

16 non-face images were collected randomly and they were projected onto the face-space with the help of eigenfaces.(see Figure 4) Further, the face images from the training set were also projected onto the face-space. (see Figure 8). We need the weight vectors of these

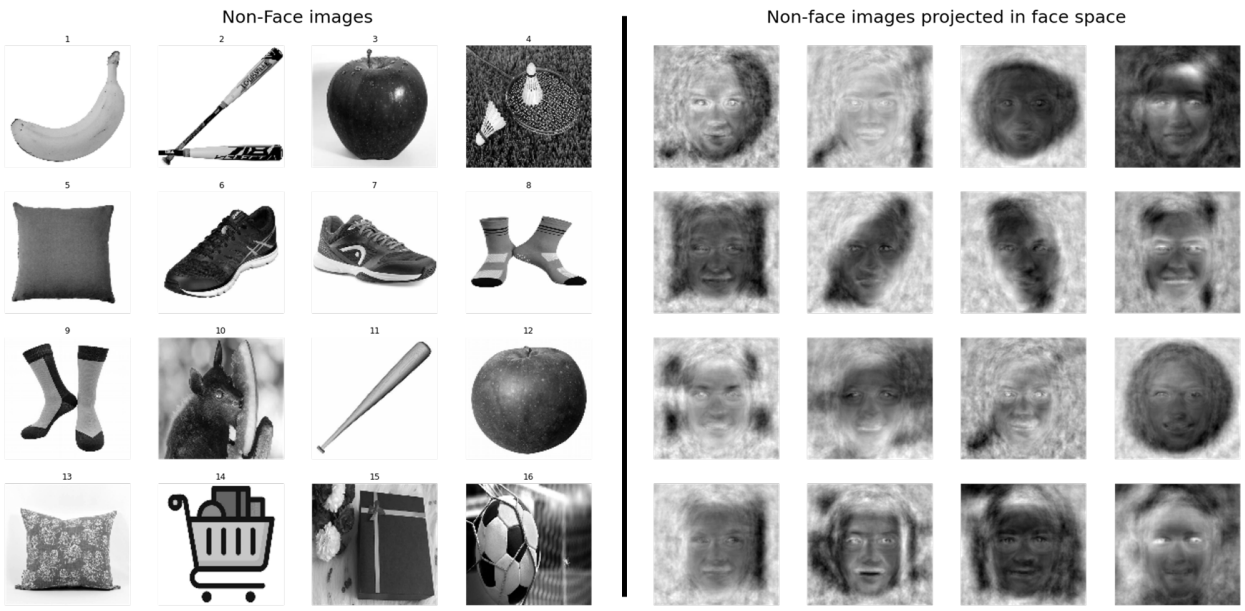

Figure 4: **Left:** Non-face images. **Right:** Non-face images projected onto the face-space

images. The weight vectors can be defined as the vectors that are used to project these images onto the face-space.

The L-2 norm of the weight vectors that were used to project these images were calculated and compared distinguish face images vs non-face images (see Figure 7). We can see from the Figure 4 that the reconstructed images have the eyes, nose and other components of the face even though these were not part of the original image. This further strengthens our claim that the images have been successfully projected onto to face-space.

4.2 Subject-wise face classification

After identifying whether an image represents a face or not, the next task that we accomplished was identifying whose face is there in the image. Yale Face Database has been used for this. The mean norm of all faces for a particular class was calculated and stored. So, now, we have 15 norms, one for each class. Once, we get a new image, we project it onto the face space.

$$W_{test} = (I - F_{avg})u_k(u_k^T u_k)^{-1} \tag{13}$$

The difference between the norm of the weight vector for the new image $||(W_{test}||$ and those of the 15 pre-stored norms is calculated. The image then belongs to the class with the minimum difference. Similar approach is followed for the pre-processing part, wherein mean faces are extracted from each class (see Figure5) and subtracted means (see Figure 6 are calculated to find corresponding weight vectors.

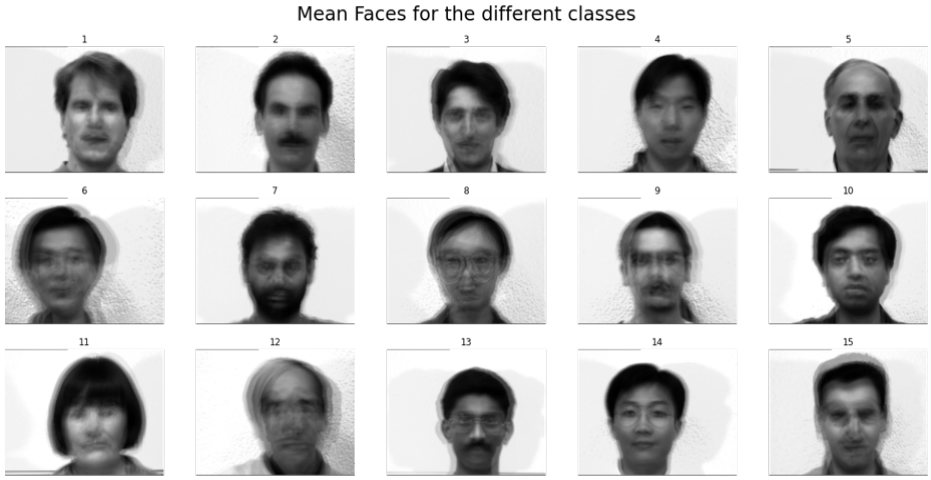

Figure 5: Mean face for each of the 15 classes

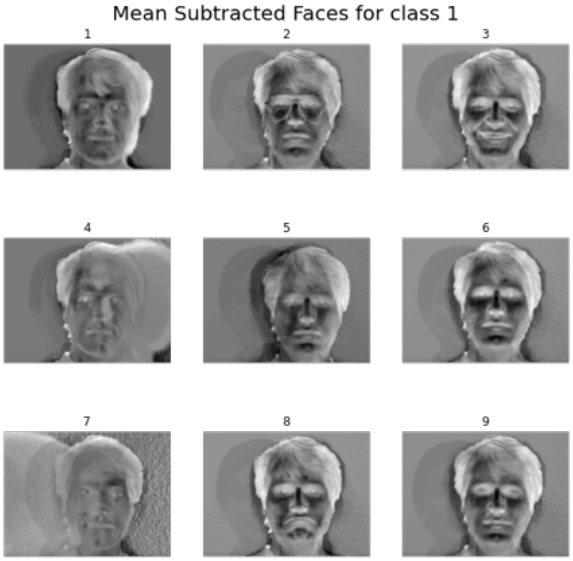

Figure 6: Mean subtracted face for class 1. This data was computed for all classes

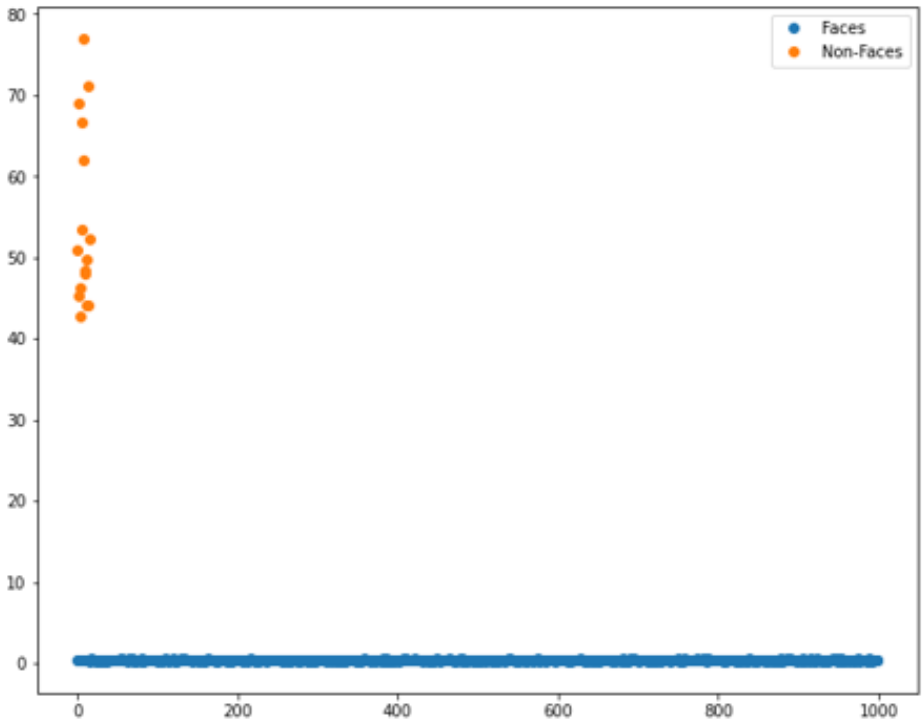

Figure 7: The norms of weights associated with face and non-face images are linearly separable

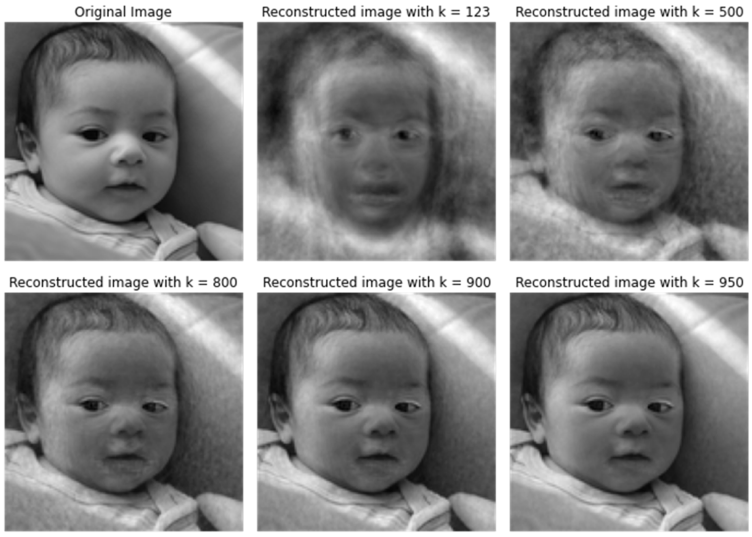

Figure 8: Face images obtained after reconstruction on face-space. The value of k denotes the number of eigenfaces that span the face-space.

5 Results and Analysis

The norm for the non-faces is very high compared to the norm for face-images (see Figure 7. Thus, faces can be separated easily from non-faces with the help of a linear classifier.

The reconstruction of a given face image with the help of eigen-faces is shown in Figure 8. k denotes the number of eigenfaces. We see as k increases the noise in the image decreases. This has been further discussed in section 5.1.

5.1 Reconstruction and Noise

We have 1000 images in our dataset (Flickr). Let this be our image space. We are using PCA, eigenvalues and eigenvectors to span our entire image space with the help of K images ($K < 1000$) such that all 1000 images can be obtained using a linear combination of K images. However, we see that for different values of K, the reconstructed image has some noise. As we increase the value of K, the noise reduces and becomes 0 when K = 1000. This means that the eigen-faces do not span the face-space completely. We are obtaining the eigen-faces using PCA which is a dimensionality reduction technique and dimensionality reduction will always lead to loss of information. Further, when the objective is to use this technique for data compression, if we are saving K = 1000 eigenfaces instead of 1000 images, we are not actually saving any storage space. Thus, to meet our objective, we need to have $K < 1000$. Then we will have K images instead of 1000 and the weights for those 1000 images which will can save a lot of space. In general data compression techniques where we resize the images to a smaller size and then resize them back, there is a lot of loss of information and noise. The image gets blurred. On the contrary, this approach will result in better compression and lesser loss of information.

6 Acknowledgement

To Prof. Santosh Singh, Big Data Analytics Centre for providing an opportunity to understand, implement and analyze this unique technique of Eigenfaces for detection.

7 Conclusion

The approach of face detection using eigenfaces seems to be quite interesting. It is computationally less expensive when compared to the current deep learning models and can give Generative adversarial networks (GANs) a run for their money.

References

[1] Mathew Turk, Alex Petland ; 1991 IEEE computer society conference on computer vision and pattern recognition

[2] P. Belhumeur, J. Hespanha, D. Kriegman, Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection, IEEE Transactions on Pattern Analysis and Machine Intelligence, July 1997, pp. 711-720.

[3] Tero Karras, Samuli Laine, Timo Aila; A Style-Based Generator Architecture for Generative Adversarial Networks