FACE RECOGNITION

Fan Zhang

Ben Overzat

COMPUTER VISION

Contents

1. ABSTRACT

This project involves the implementation of a face recognition algorithm. Based on a labeled training set of face images, test images can be classified to determine whether they contain a face, and which face specifically from the training set. The algorithm, known as the eigenface algorithm, is implemented in MATLAB using a training and testing set of images of students in this course and famous individuals, as well as a few images without any faces. The algorithm seeks to reduce the dimensionality of images into a representation of a small number of coefficients, and uses these coefficients for classification. The accuracy of the algorithm on the testing set is examined to demonstrate its correct implementation.

2. INTRODUCTION

2.1 Eigenface Algorithm

The eigenface algorithm involves performing principal component analysis (PCA) on images to reduce their dimensionality and represent them by a relatively small number of coefficients. These coefficients can be interpreted as a set of features or attributes, and a simple nearest-neighbor classification scheme can be used to classify test images based on the coefficients of the images in the training set.

For a given grayscale image defined by a *W*-by-*H* matrix of intensity values, the matrix elements are formed into a single *WH*-dimensional column vector. The image can thus be interpreted as a single point in *WH*-dimensional space. To reduce the dimensionality of the set of training images, we consider the variance of the column vectors *x* corresponding to the set of training images along the direction of an arbitrary *WH*-dimensional unit vector *v*:

$$\mathrm{var}(v) = v^T A v, \quad A = \sum_x (x - \bar{x})(x - \bar{x})^T$$

Here, $\bar{x}$ is the mean of all vectors in the training set. It is known from linear algebra that the vectors that maximize this variance are the eigenvectors corresponding to the largest eigenvalues of the matrix *A*. Therefore the components of the training vectors along the direction of these eigenvectors encode the most variation among the training images. We can then reduce the dimensionality of the training vectors by finding their components along these eigenvectors. These are known as the principal components of the vectors. We select the *K* eigenvectors (normalized to unit magnitude) corresponding to the largest *K* eigenvalues of *A*, and find the components a_k of each training vector *x* along each eigenvector v_k as follows:

$$a_k = (x - \bar{x})^T v_k$$

The set of *K* eigenvectors used for PCA can be interpreted as defining a *K*-dimensional hyperplane that best fits the *WH*-dimensional points defined by the training vectors.

Then, given a testing image to classify, we can determine whether the image contains a face by finding the distance between the image's corresponding point in the *WH*-dimensional space and the hyperplane defined by the *K* eigenvectors from the training set. Mathematically, for an image vector *x*, this distance can be found as follows:

$$\left\| x - \left(\bar{x} + \sum_{k=1}^{K} a_k \; v_k \right) \right\|_2$$

where a_k are the components of the testing image along each eigenvector. If this distance is less than a certain threshold, the image can be considered as containing a face. We can then classify the image as a specific person's face by finding the nearest neighbor in *K*-dimensional space as defined by the principal components (i.e. the a_k values). If the distance to the nearest neighbor is larger than a certain threshold, the image can be considered as containing a face that was not part of the training set.

The selection of the number *K* of eigenvectors to use depends on the distribution of eigenvalues. An examination of the decay of eigenvalues when plotted from largest to smallest can reveal the point at which eigenvalues become small enough that the variation along these dimensions is negligible.

2.2 Data Sets

We demonstrate the eigenface algorithm using a set of training and testing images comprised of the faces of students in this course as well as faces of famous individuals. Specifically, we choose a training set of 40 images and a testing set of 20 images. Of the 40 training images, 20 are of students in the class and 20 are of other famous people, such as celebrities and former presidents. There are two training images each for ten students, and 20 different famous people's faces included in this training set. Of the 20 testing images, 9 are of students, 8 are of famous people, and 3 are not faces at all. The testing images of the students are completely different images from the training ones, and the testing images of the famous people are the same as the training images but with a radial illumination field applied. These testing images are darker in the center and lighter toward the edges, compared to the training images where the lighting appears relatively uniform. All images originally were of size 640 by 480, but the size was reduced to 60 by 40 before processing. Fig. 2.1 shows the training set, and Fig. 2.2 shows the testing set.

Fig 2.1 Training Set

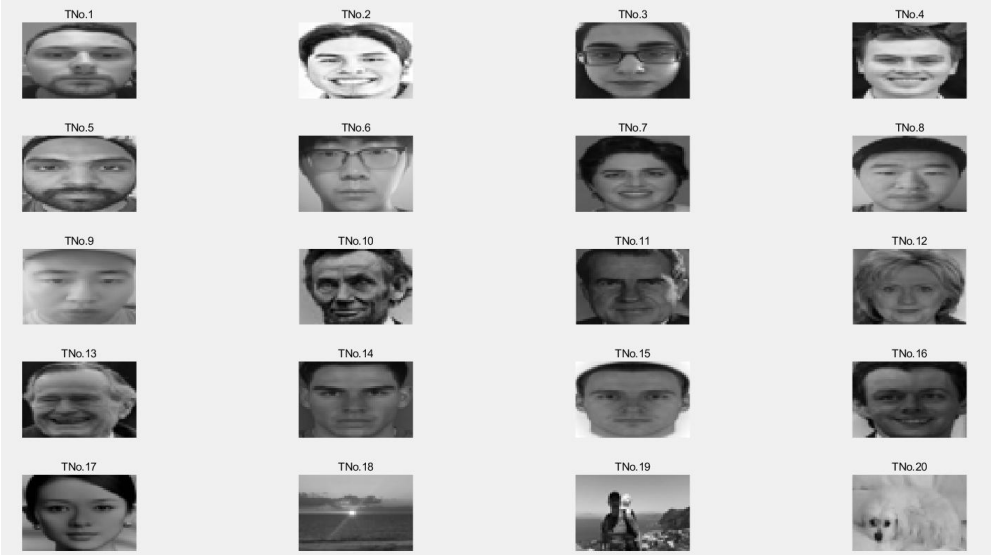

Fig 2.2 Testing Set

3. RESULTS AND ANALYSIS

We have introduced the algorithm principle and data set of this project in detail before, now let's show the results of this project. After reading the data of the training set, let's first get the mean image, as shown in the figure 3.1. The acquisition of mean image is for obtaining matrix A in the next step.

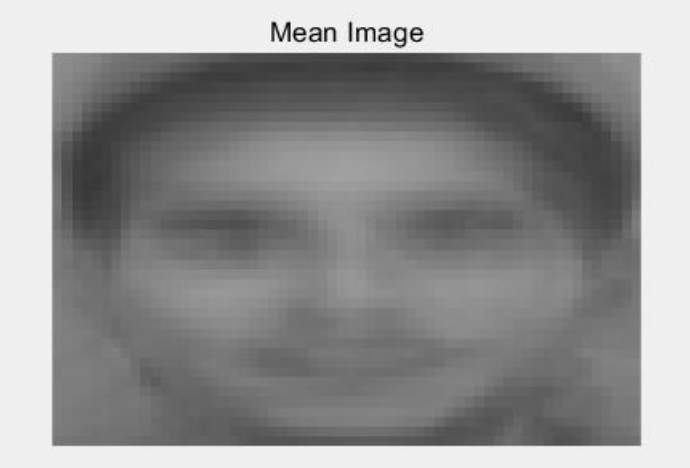

Fig 3.1 Mean image

By using mean image, we can get matrix A and use the PCA to extract the eigenvectors of A. Then by selecting the first 39 larger eigenvectors (here we set k = 39), and we get 39 feature faces, as shown in the figure 3.2.

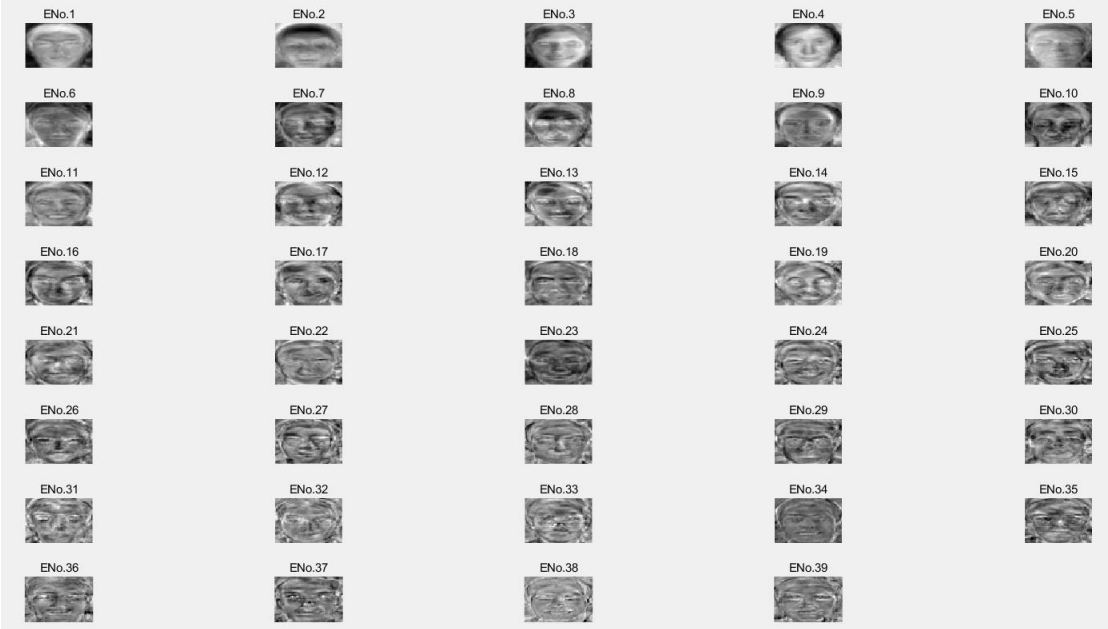

Fig 3.2 Eigenfaces

Fig. 3.3 shows the 100 largest eigenvalues of the matrix A. It is clear that by the 40th largest eigenvalue, the values are very close to zero. Therefore choosing 39 eigenvalues is sufficient to represent the variation between faces in our training set.

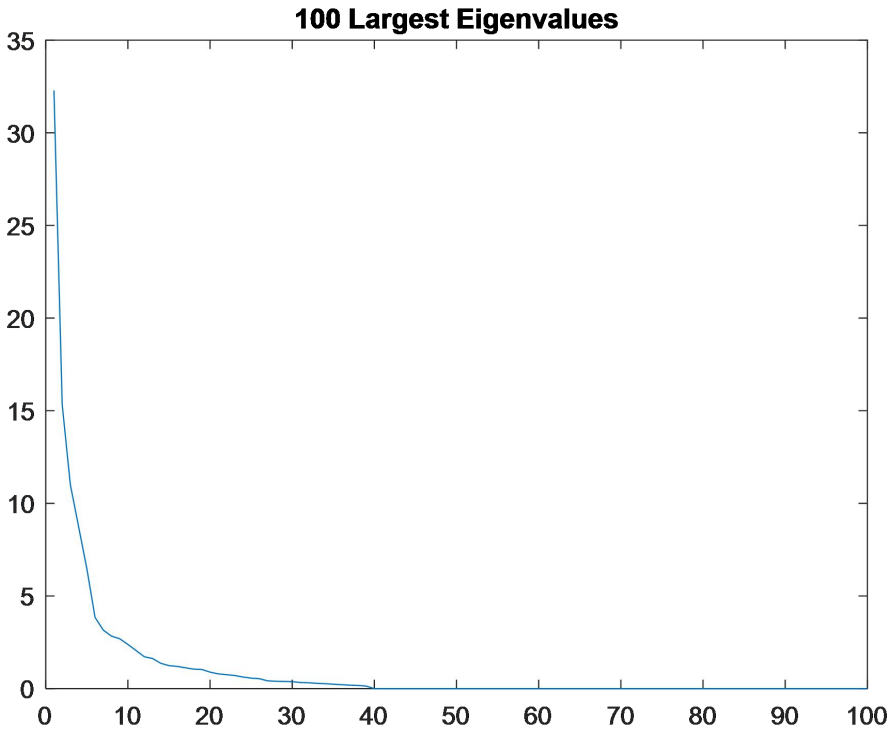

Fig. 3.3 Largest 100 eigenvalues of matrix A.

Using the 39 eigenvectors, we can obtain the coefficients (size 39 x 40) for the training and (size 39 x 20) for testing subjects, which means that we project image data into feature space. Here we show the matrix in color, as shown in the figure 3.4 and 3.5.

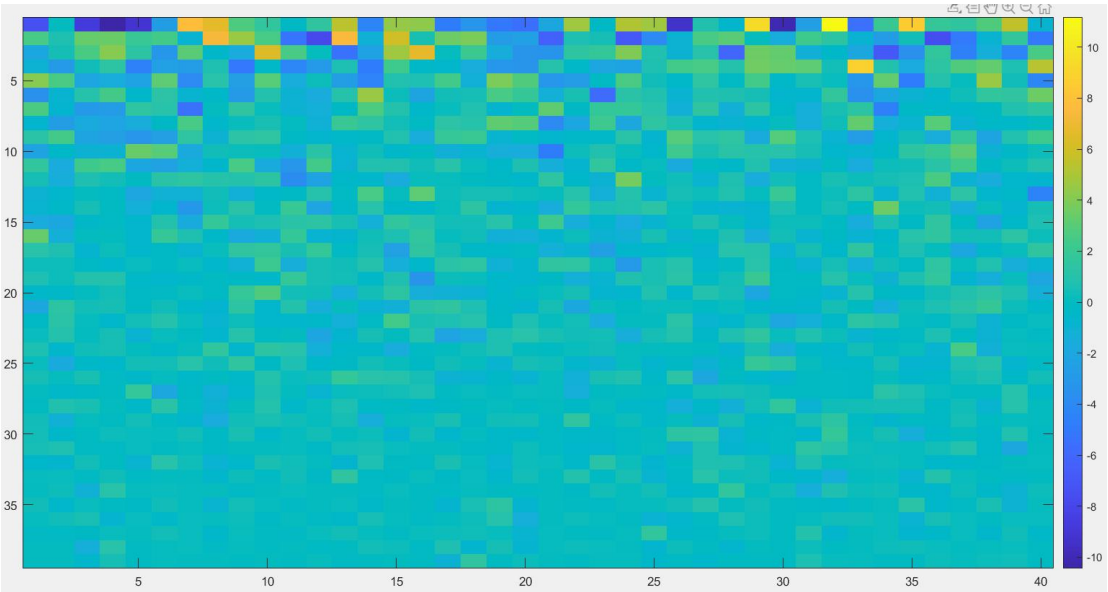

Fig 3.4 Training coefficient

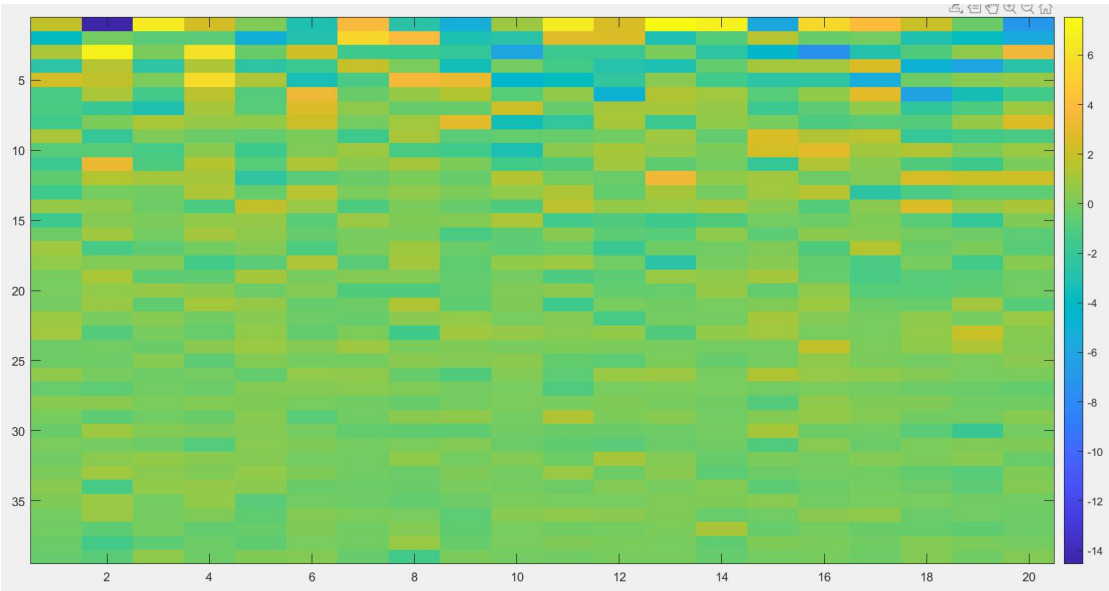

Fig 3.5 Testing coefficient

If you want to see more details, you can download the. mat file below.

TrainingCoefficient.mat

TestingCoefficient.mat

And then, we use the testing coefficient to determine if each image contains a face by calculating delta x (Suppose x is a new image, delta x is equal to the norm of x minus x's projection on the eigenfaces). And we take threshold as 8 and compare delta x with the threshold. When it is less than 8, we think that face is detected, when it is greater than 8, we think that no face is detected. The table 3.1 shows the result of delta x of each image.

Image number	Delta x	Results
TNo.1	6.5118	Is a face
TNo.2	4.3619	Is a face
TNo.3	4.8118	Is a face
TNo.4	6.943	Is a face
TNo.5	4.5293	Is a face
TNo.6	4.2014	Is a face
TNo.7	3.5113	Is a face
TNo.8	2.3735	Is a face
TNo.9	2.0253	Is a face
TNo.10	1.6129	Is a face
TNo.11	1.0716	Is a face
TNo.12	1.4954	Is a face

TNo.13	1.2032	Is a face
TNo.14	1.3053	Is a face
TNo.15	1.5112	Is a face
TNo.16	1.6181	Is a face
TNo.17	1.8294	Is a face
TNo.18	8.4212	Not a face
TNo.19	8.9954	Not a face
TNo.20	4.9055	Is a face

Table 3.1 Delta x and results

Finally, we can use each vector in test coefficient and each vector of training coefficient to calculate Euclidean distance, which is recorded as distance. Show the results and mark the unmatched images as unrecognized face by observing the image matching results. The distance and matching results of each image are shown in the table 3.2 and figure 3.6.

Image number	Distance	Matching results
TNo.1	6.5118	Match the RNo.2
TNo.2	9.247	Match the RNo.4
TNo.3	7.1966	Match the RNo.7
TNo.4	8.7107	Match the RNo.10
TNo.5	5.9798	Match the RNo.11
TNo.6	6.8758	Match the RNo.14
TNo.7	6.0134	Unrecognized face
TNo.8	3.3904	Match the RNo.17
TNo.9	2.1633	Match the RNo.20
TNo.10	6.2058	Match the RNo.21
TNo.11	4.7116	Match the RNo.22
TNo.12	5.9532	Match the RNo.23
TNo.13	6.4413	Match the RNo.24
TNo.14	5.8333	Match the RNo.25
TNo.15	6.3501	Match the RNo.26
TNo.16	7.7666	Unrecognized face
TNo.17	8.5729	Unrecognized face

TNo.18	N/A	Not a face
TNo.19	N/A	Not a face
TNo.20	9.7505	Mismatch the RNo.19

Table 3.2 Distance and matching results

In Figure 3.6, the pictures with the title beginning with TNo are all test pictures, the pictures with the title beginning with RNo are all pictures of the training set, and the picture below each test picture is a picture of the matching training set. When the faces do not match each other, the picture below will be named unrecognized face, and when the test picture is not a face, the test picture will be named not a face.The images circled in red are the wrong result, and the others are all the right results.

From the figure 3.6, we can see that TNo.7, TNo.16 and TNo.17 are not correctly recognized, and other test images containing human faces are correctly recognized. However, TNo.18 and TNo.19, which do not contain human faces, are correctly treated as not a face. However, TNo.20 is mistakenly considered as a face, which leads to wrong matching, which may be caused by the lack of training set data. So now the accuracy is 16 / 20 * 100% = 80%.

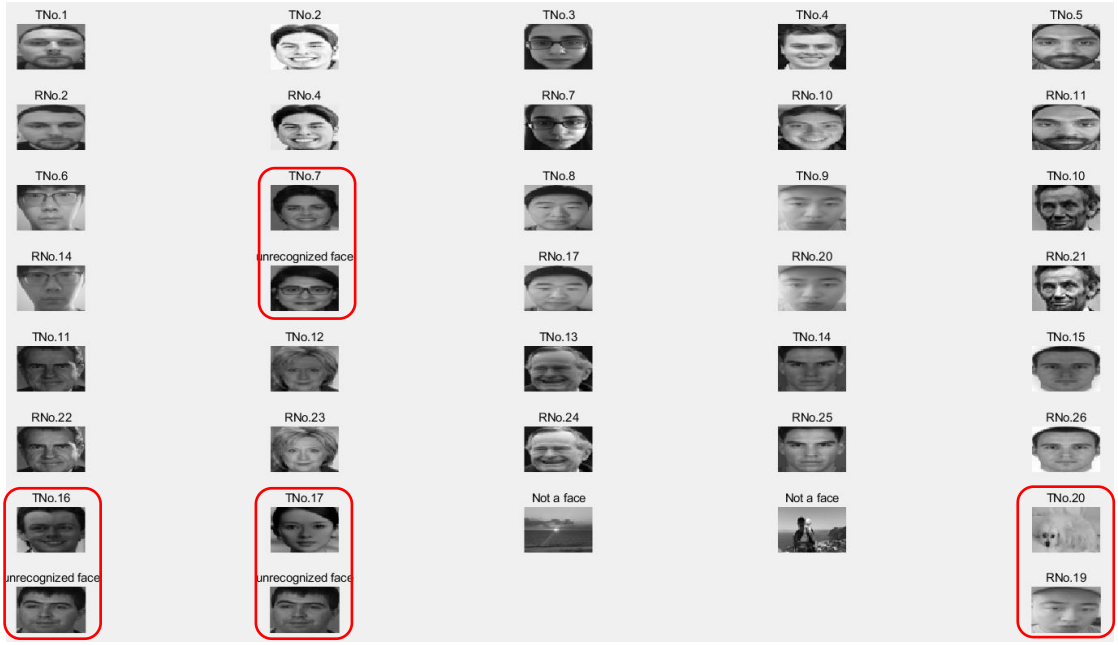

Fig 3.6 Matching results

Now let's discuss how to improve the accuracy of our results. By analyzing the test pictures that are not matched correctly, we can find that TNo. 7 and its matched picture does have some similarities, but the reason why they are not matched correctly may be that there are too few related pictures in the training set, so it can't recognize Victoria's face correctly. Since we don't have more pictures here to improve our training set, we need to think about the pictures themselves to optimize the results.

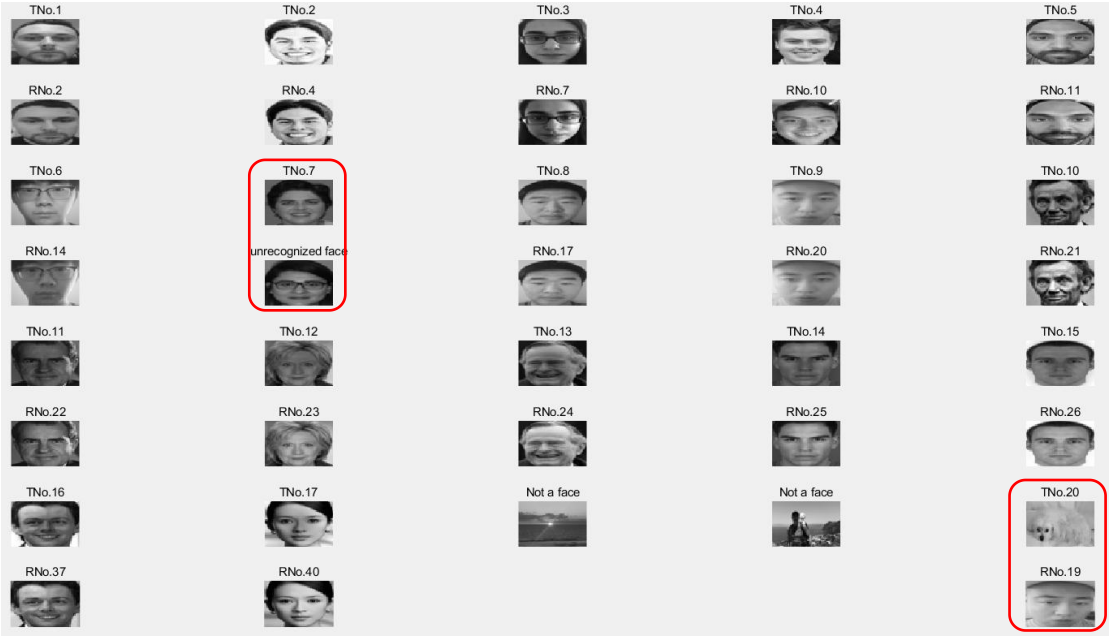

Fig 3.7 Matching results

For the images TNo.16 and TNo.17 that are not matched correctly, this may be because of the illumination. Because the illumination of these two images is too dark, they are mistakenly matched to other faces, which shows that the feature face algorithm is sensitive to the change of illumination. In fact, we can solve this problem perfectly with a trick. By introducing a illumination coefficient b, let b = 1.5, which means that the picture is 1.5 times brighter, we can process the two pictures of TNo.16 and TNo.17 in testing set. After that, you can match the correct result, as shown in the figure 3.7.

By applying this trick, the accuracy is increased to 18 / 20 * 100% = 90%, which is much better than the previous results.

4. EXPLORE THE ROBUSTNESS OF EIGENFACE ALGORITHM TO GEOMETRIC TRANSFORMATION

In this part, we want to study whether the feature face algorithm can still maintain the original accuracy if we apply some geometric transformations to the test set images from TNo.10 to TNo.17, such as rotation, translation, scaling, shear and mirror transformation. It needs to be explained here that we only transform these eight pictures because the illumination transformation of these eight pictures is uniformly dark, which is easy to remove. Here we just want to explore the influence of the geometric transformation of the picture on the algorithm of eigenface, so we need to remove the influence of illumination on its accuracy. Our expectation here is that the rotation, shear and vertical mirror transform will have a greater impact on the accuracy, and horizontal mirror transform will have a small impact, and the scaling and translation transform will have no impact.

First of all, we only make rotation transformation for the pictures, and rotate each picture of the test set 90 degrees clockwise, the effect is as shown in the figure 4.1.

Fig 4.1 Rotation transformation

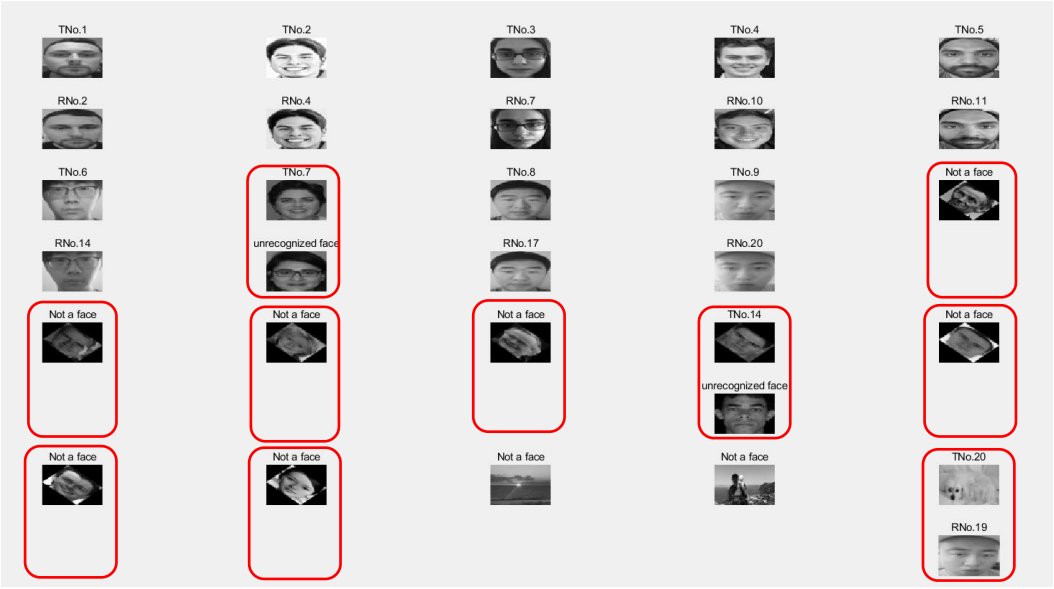

Fig 4.2 Matching results

After the rotation transformation, in order to eliminate the influence of illumination on the accuracy, we uniformly enhance the brightness of these eight pictures, because the brightness of these eight pictures is dark, and then recognize them again, and the result is as shown in the figure 4.2. The images circled in red are the wrong result, and the others are all the right results. We can observe that seven of the eight test images that are rotated were judged not to be a face, and the TNo.14 is a unrecognized face. But other images without rotation transformation are still correct, and the accuracy is 10 / 20 x 100%= 50%, which shows that the eigenface algorithm is not robust to rotation transformation.

And then we translate the same eight test images, as shown in the figure 4.3.

Fig 4.3 Translation transformation

Here we also eliminate the influence of illumination, and then carry out face recognition again. The results are as shown in the figure 4.4. The images circled in red are the wrong result, and the others are all the right results.

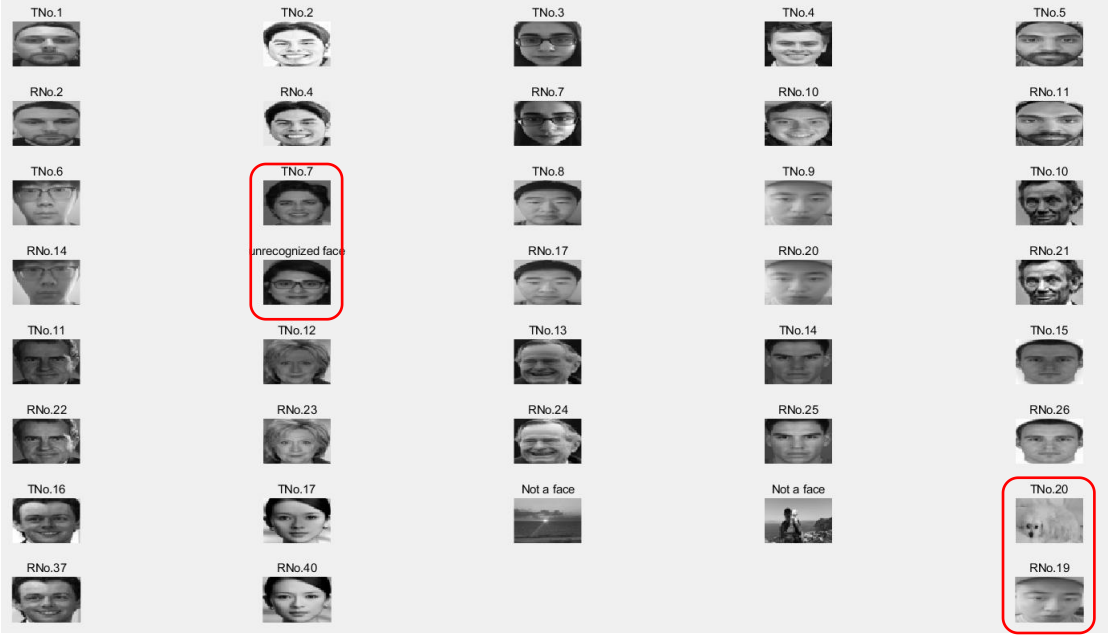

Fig 4.4 Matching results

From figure 4.4, we can see that the results are the same as that without translation transformation, so translation transformation has no impact on the accuracy of eigenface algorithm. The accuracy is still 90%.

And then we reduce the picture to half of its original size, as shown in the figure 4.5. The size of each image has changed from 640x480 to 320x240.

Fig 4.5 Scaling transformation

Eliminate the influence of illumination, and then do face recognition again. The results are as shown in the figure 4.6. The images circled in red are the wrong result, and the others are all the right results.

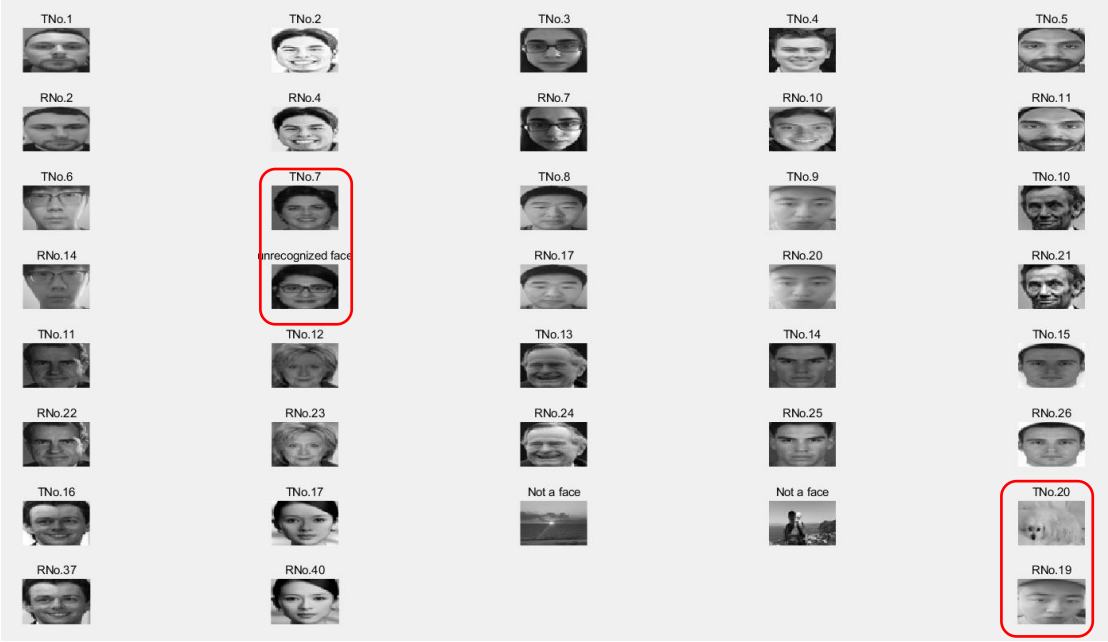

Fig 4.6 Matching results

From figure 4.6, we can see that the results are also the same as that without scaling transformation, so the eigenface algorithm is robust to scaling transformation. The accuracy is still 90%.

We do the shear transformation to the eight images, as shown in the figure 4.7.

Fig 4.7 Shear transformation

We continue to use the eigenface algorithm for face recognition. The result is shown in the figure 4.8. The images circled in red are the wrong result, and the others are all the right results.

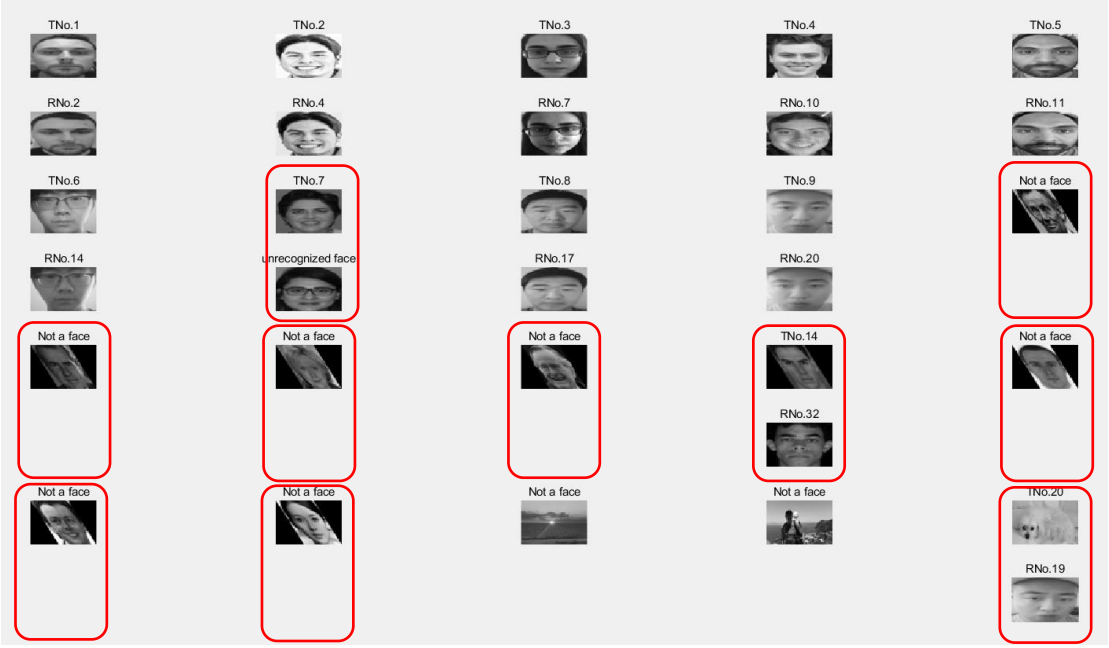

Fig 4.8 Matching results

From the figure 4.8, we can observe that the result is the same as the result of rotation transformation, which may be because these two kinds of transformation will lead to black areas in the image that are not originally part of the image. It might be these black areas that seriously affect the accuracy of the eigenface algorithm. The accuracy is 10/20*100%=50%.

Next, we do the horizontal mirror transform to the images as shown in the figure 4.9.

Fig 4.9 Horizontal mirror transformation

Eliminate the influence of illumination, and then do face recognition again. The results are as shown in the figure 4.10.

Fig 4.10 Matching results

From the figure 4.10, we can observe that only two of the eight images with horizontal mirror transformation are unrecognized, which shows that the horizontal mirror transformation does have a certain impact on the feature face algorithm, but the impact is not great. Now the accuracy is 16/20*100%=80%.

Finally, we do the vertical mirror transform to these 8 images , as shown in the figure 4.11.

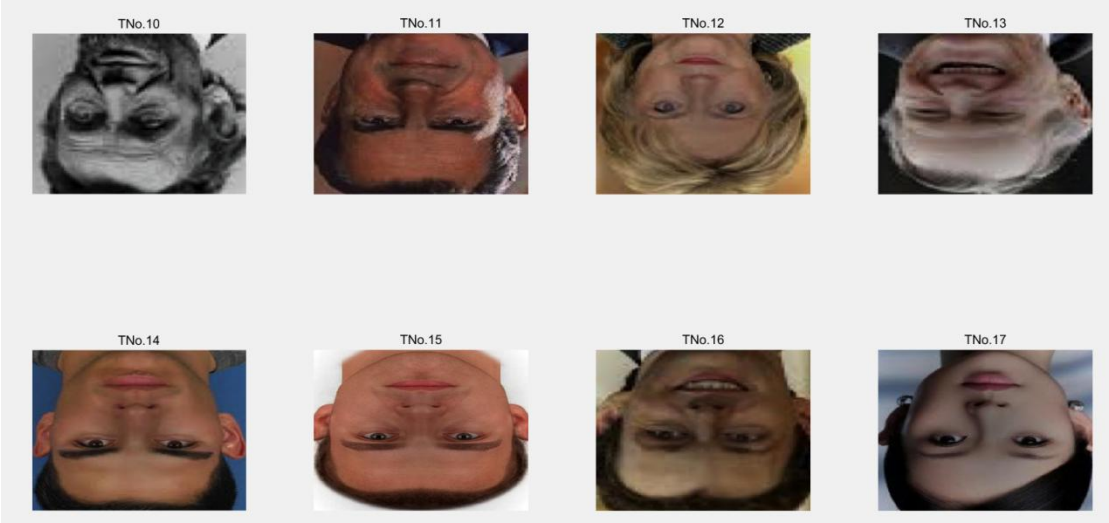

Fig 4.11 Vertical mirror transformation

Use the eigenface algorithm for face recognition again. The result is shown in the figure 4.12. The results are as shown in the figure 4.10.

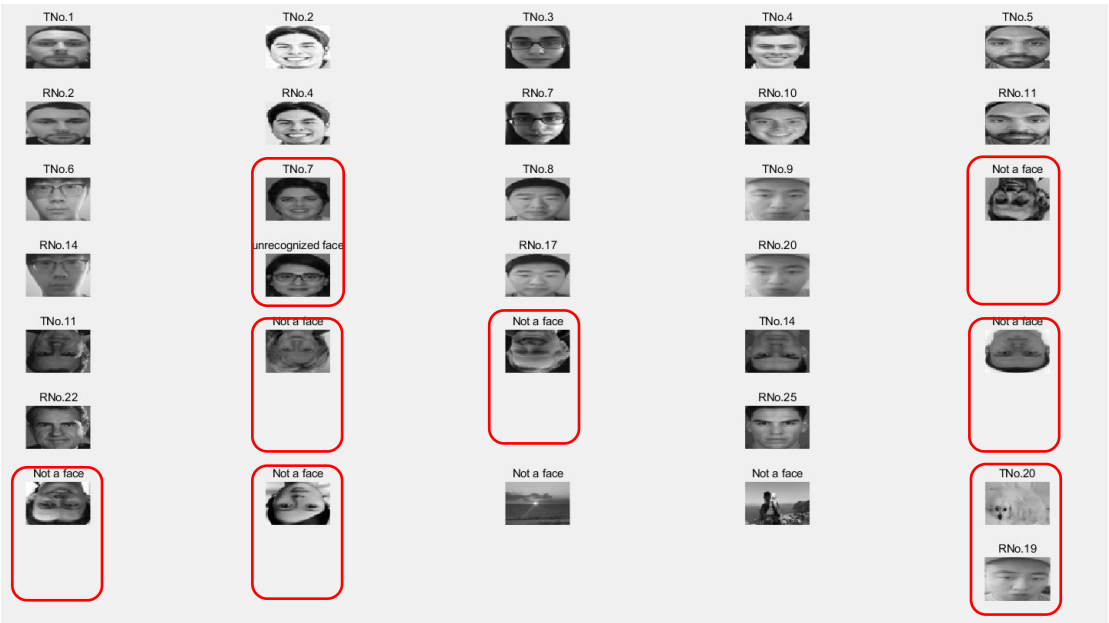

Fig 4.12Matching results

From the figure 4.12, we can observe that it's different from the horizontal image transformation, the accuracy here is greatly affected, which shows that the eigenface algorithm is more sensitive to the vertical image transformation. Now the accuracy is 12/20*100%=60%.

To compare the accuracy of no transformation and six transformations, we draw a histogram, as shown in the figure 4.13. This result is basically consistent with our expectation.

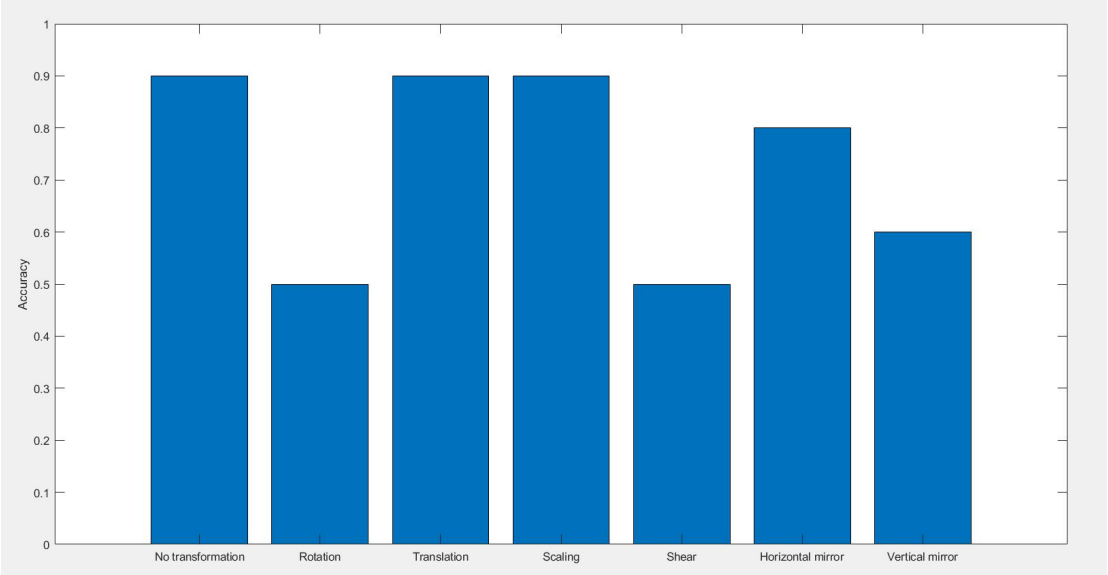

Fig 4.13 Comparison

In conclusion, from the figure 4.13, it can be seen that the eigenface algorithm is not robust to rotation, shear and vertical mirror transformation, which might be due to the loss of face features caused by these three transformations or the great difference between the face and the face in the training set. Thus, if we want to apply the eigenface algorithm to face recognition when there is rotation transformation in the image data of the test set, we should eliminate rotation, shear and vertical mirror transformation as much as possible to improve the accuracy of the results.

5. CONCLUSION

This project demonstrated the implementation of the eigenface algorithm with a training set of 40 images and a testing set of 20 images. The algorithm performed PCA on the training set, and used this to reduce the dimensionality of the training images and represent them with a small number of coefficients. This allowed for efficient and accurate classification of testing images. Initially, an accuracy rate of 80% was achieved, but it was found that applying a constant illumination coefficient to some testing images that appeared too dark resulted in their correct classification, increasing the accuracy to 90%. The incorrect classifications that remained were due to significant differences in the appearance of students across multiple headshots. Then, the robustness of the algorithm to various image transformations was explored, and it was discovered the the algorithm does not hold up well to shearing, rotation, and mirroring transformations. However, this relatively simple face recognition algorithm proved to be largely effective with small variations between the training and testing faces and some limited transformations.