CS 479

Programming Assignment 3

Jacky Yu

4/23/2024

“I declare that all material in this assignment is my own work except where there is clear acknowledgment or reference to the work of others. I understand that both my report and code may be subjected to plagiarism detection software, and fully accept all consequences if found responsible for plagiarism, as explained in the syllabus, and described in UNR’s Academic Standards Policy: UAM 6,502.”

Section 3a:

Theory: Given a set of faces such that the images are centered in the same position with the same dimensions, eigenfaces can be calculated to create a recognition algorithm based on feature extraction. In order to create the eigenfaces, the images are flattened and used to create a NxM matrix. Sample mean(1) is calculated and then subtracted from all data points in order to find Φ(2) and center the data. In this case, A is the matrix created from flattening the images in creating a matrix from them. ATA trick is used in order to reduce the dimensions of the covariance matrix(3) to make a MxM matrix instead of NxN. Using this covariance matrix, eigenvalues and eigenvectors can be calculated. The eigenvalues from ATA become lambda and the eigenvectors are multiplied by the original matrix and normalized to obtain the actual eigenvectors. K is selected based on the amount of data to preserve and select from the eigenvectors to create the eigencoefficients. Using the set K, a testing image is projected(5) onto the eigencoefficients with the top 50 eigenfaces being selected from least to greatest based on the Mahalanobis distance(4). If 1 of the 50 eigenfaces ID matches the actual ID, it is considered a correct match. This is repeated for all images in the testing folder and the results are stored.

Equation #	Result	Formula
1	$\bar{x}$	$1/M\ \Sigma x_i$
2	Φ	$X_i - \bar{x}$
3	$\sum_X$(Covariance)	$A^T A$
4	Mahalanobis Distance	$min\sum 1/\lambda(y_i - y^j{}_i)^2$

5	Projection	$y_i = \Phi^T u$

A.1) Eigenfaces

From the set of data, the mean faces, the 10 largest values and their face, and the 10 smallest eigenvalues and their face are displayed. Figure 1 shows the results from calculating the average face, figure 2 shows the 10 largest eigenfaces, and figure 3 shows the 10 smallest eigenfaces.

Figure 1: The mean face of the high resolution data

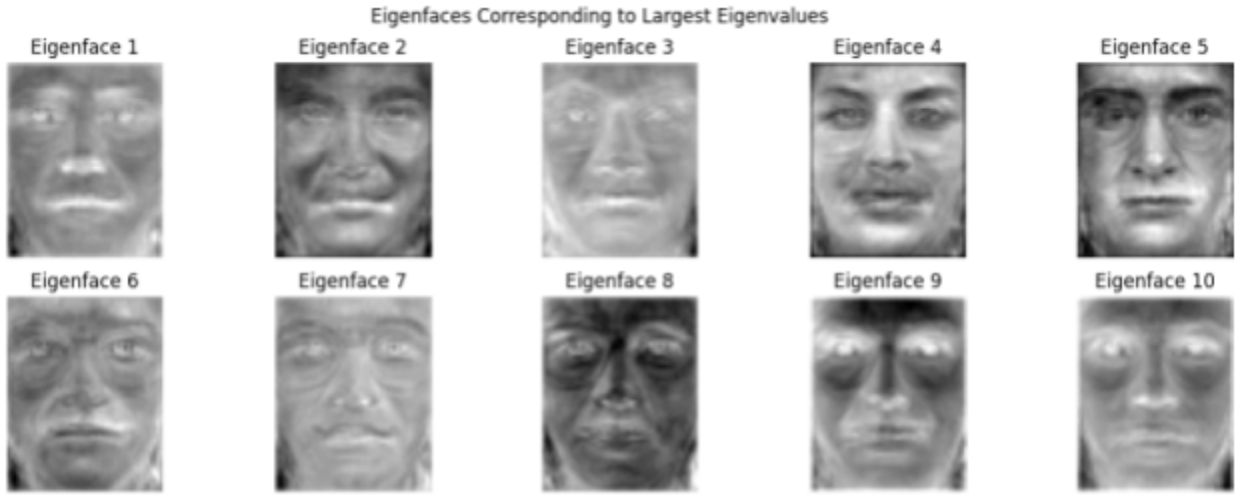

Figure 2: The 10 faces based on the largest eigenvalues

Figure 3: The 10 eigenfaces from the 10 smallest eigenvalues

A.2) Results

Theory: 3 sets of data are created using K values 80%, 90%, and 95%. Using these K values, it is projected onto the testing data set and used to find the accuracy of each calculation. The eigenface for all original faces is then compared with the projected test face with difference in face space being calculated with the Mahalanobis distance(5). All values are stored and organized at least to the greatest matching face ID. Accuracy is calculated by totaling the

number of properly matched ID divided by the total number of faces. This process is repeated for different values or r, where r is the amount of eigenfaces used to find a match.

The results for the accuracy can be seen in figure 4 with the CMC curve. In this CMC curve, the x-axis is the r values, ranging from 1-50. The performance ranges from 0-1, representing the percentage of correct faces within the r values range. Using this graph, it can be seen that preserving more data yields a better recognition of the faces. This is due to the capturing of more variations of each training image allowing a lower value in Mahalanobis distance, resulting in a more accurate result. This comes at the cost of having to use more data points, resulting in longer computations. Figure 5,6, and 7 shows moments where the algorithms match the IDs correctly and moments where the IDs are not properly matched. From these images it can be any matches from the lower amounts are still true, but increasing the number of eigenfaces does not guarantee the face would have a new match.

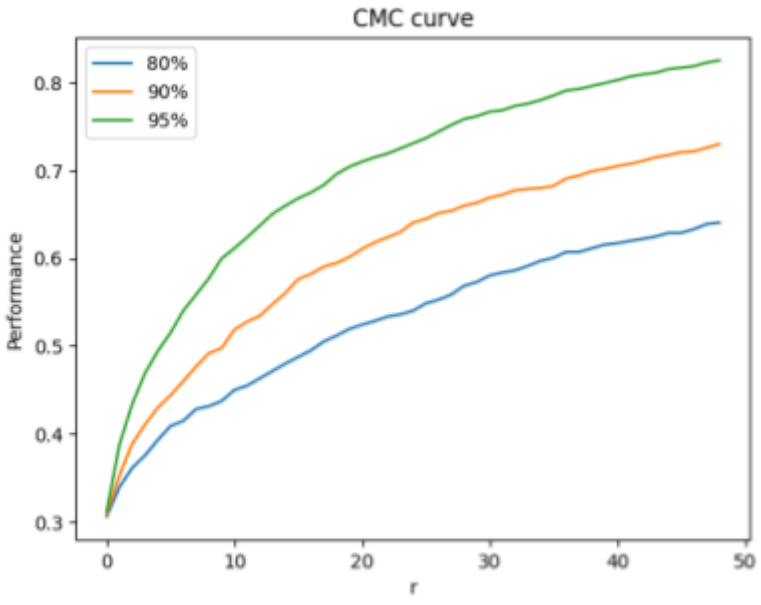

Figure 4: CMC curve using different percentages of Eigenfaces for comparisons.

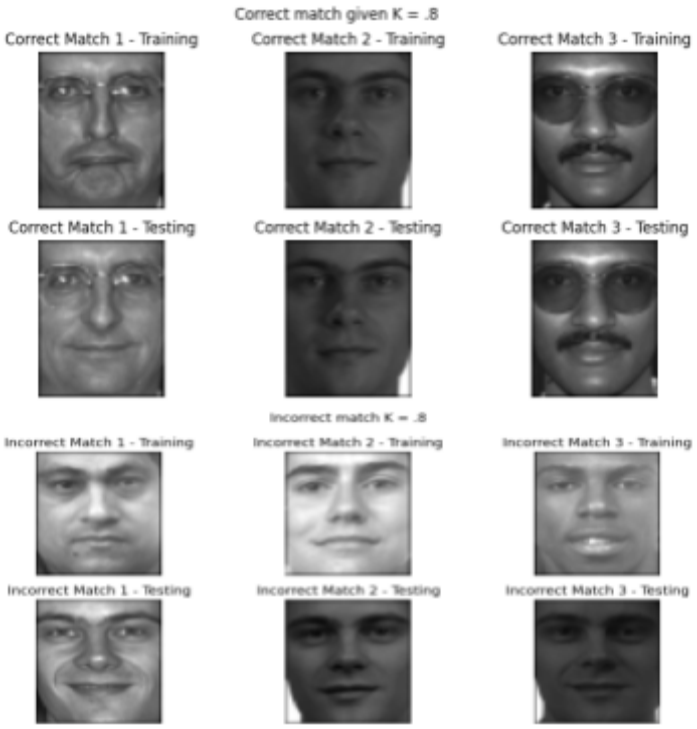

Figure 5: 3 cases for correct matches and incorrect matches at 80%

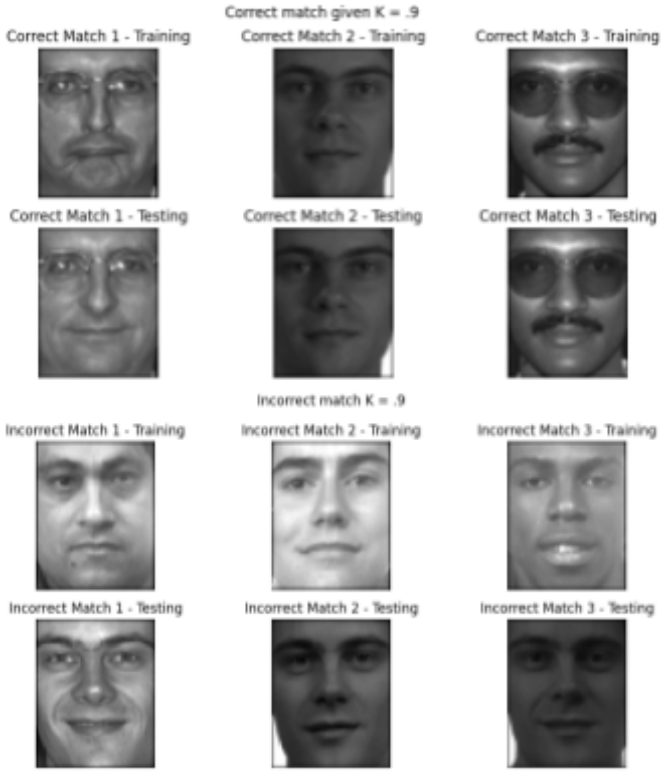

Figure 6: 3 cases for correct matches and incorrect matches at 90%

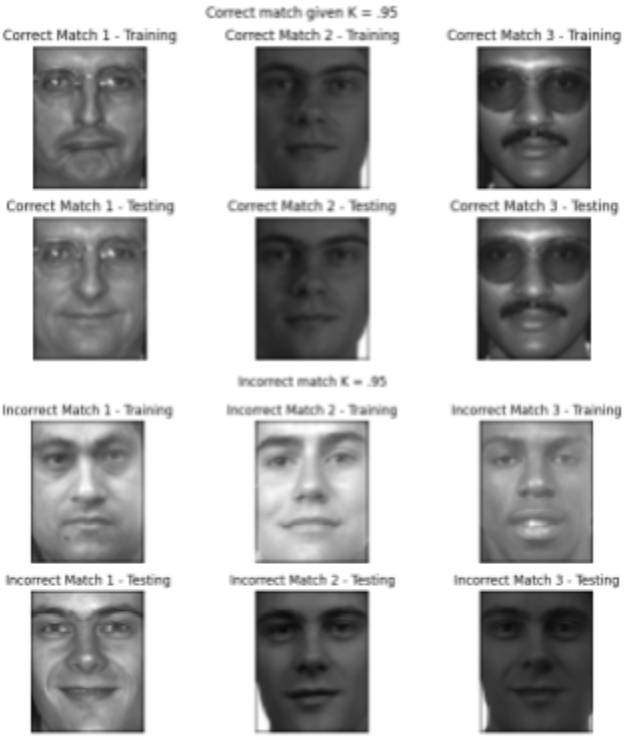

Figure 7: 3 cases for correct matches and incorrect matches at 95%

B) Intruders

Theory: Changing the training data by removing some IDs, creating new values and attempting to detect intruders or IDs not in the data set. With the new Eigenvalues and eigenvectors, calculate the maximum threshold by finding the max possible distance between the training image and testing image, and make intervals. If a value is below the threshold, they are considered an intruder, and if a value is above the threshold they are considered in the database. As the threshold increases, the number of false positives increases and when the threshold decreases, the number of true positives decreases. This is done at 95% data retention.

As seen with figure 8, a ROC curve plotting the false positives against the true positives. Figure 9 shows the FPR and TPR at each threshold. When comparing the 2, it can be seen that TPR decreases more significantly as threshold decreases, while FPR increases when threshold increases. This shows a lower threshold produces a higher positive accuracy for true positives

while a higher threshold can produce higher accuracy at the cost of false positives being more likely. With figure 8, it can be seen that positive accuracy is higher at all times, but they equalize once there are no threshold limits.

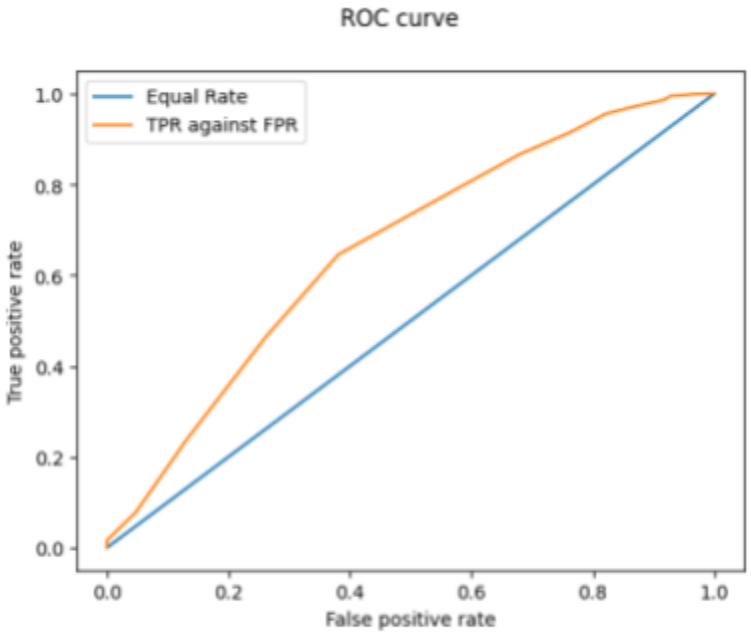

Figure 8: ROC curve of TPR and FPR

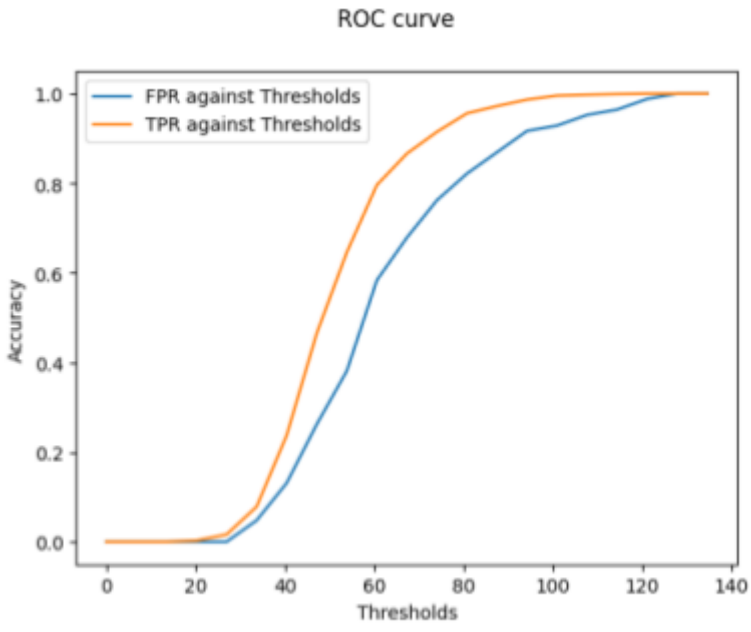

Figure 9: The accuracy at different thresholds for FPR and TPR

Section Extra Credit:

C)

Theory: Given a set of faces with a lower dimensionality/resolution, what effect would it produce on the results in terms of accuracy. Using the same steps from A1, find the CMC curve or the new set of data.

From figure 10, the average face from the data can be seen. Figure 11 and 12 shows the eigenfaces for the 10 largest and 10 smallest eigenvalues. The lower dimensions of the image can be seen with more details being missing.

Figure 10: Average face for the low resolution images

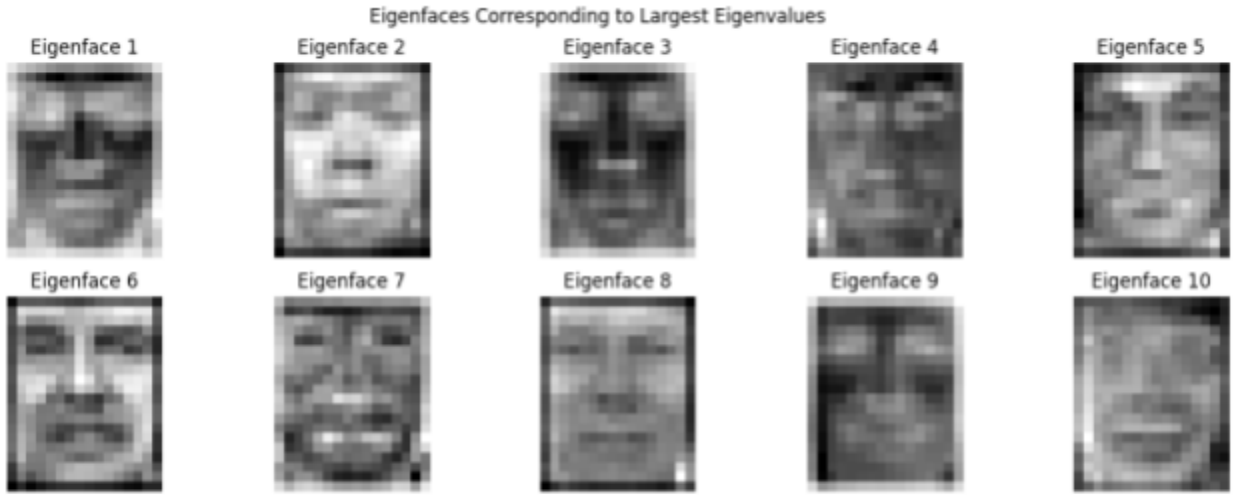

Figure 11: The eigenfaces corresponding to the 10 largest eigenvalues of the low resolution faces

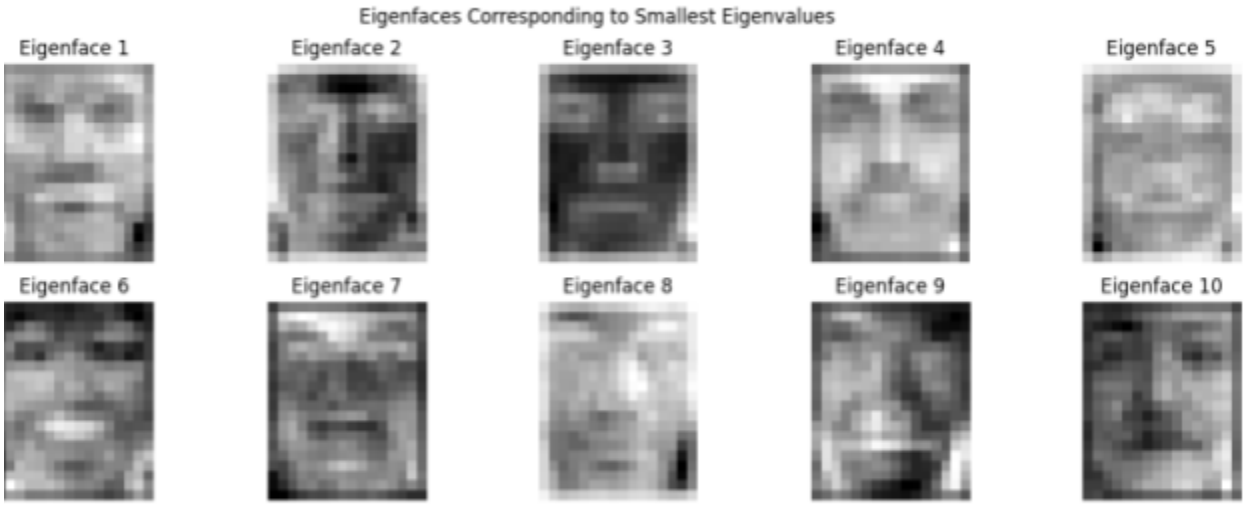

Figure 12: The eigenfaces corresponding to the 10 smallest eigenvalues of the low resolution faces

From figure 13, it can be seen that the CMC curve shows a different trend from the original. While the 80% and 90% shows an increase, the 95% shows a decrease in values. In this case, using 90% of the data creates the most accurate prediction algorithm. Figure 14, 15, and 16 show the correct and incorrect matches for the lower quality images. From these images it can be seen with a low range, images that were originally able to get a match are unable to get a proper match.

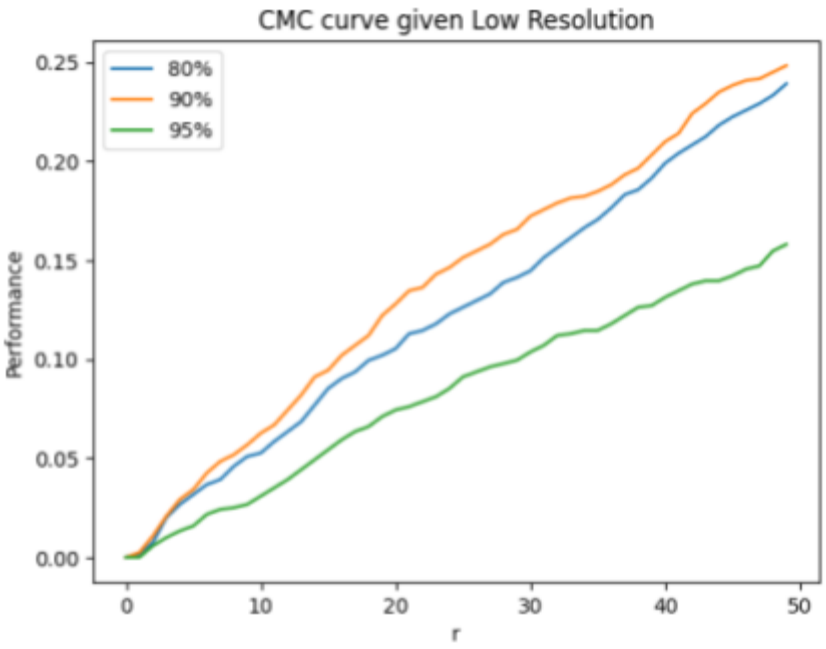

Figure 13: The CMC at different percentages given low resolution images

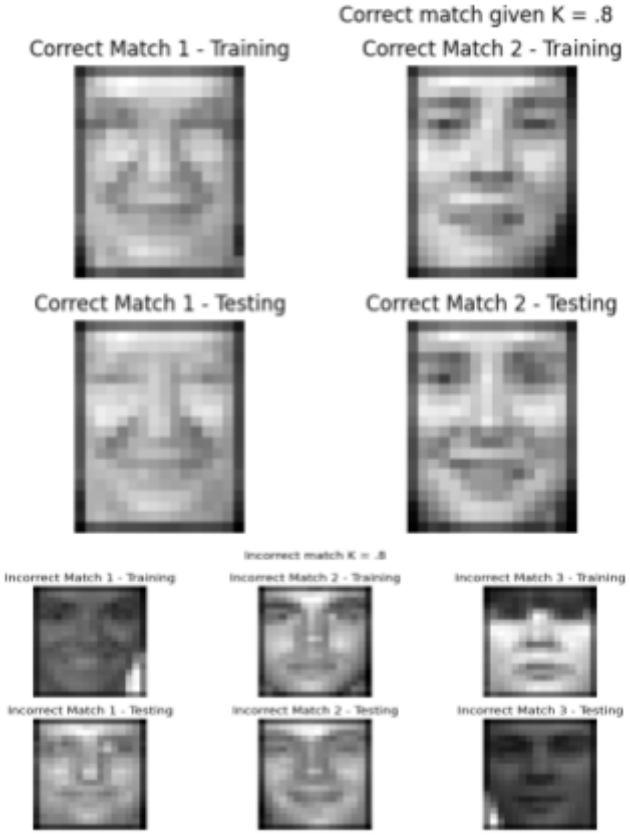

Figure 14: The matching images of the low resolution at 80%

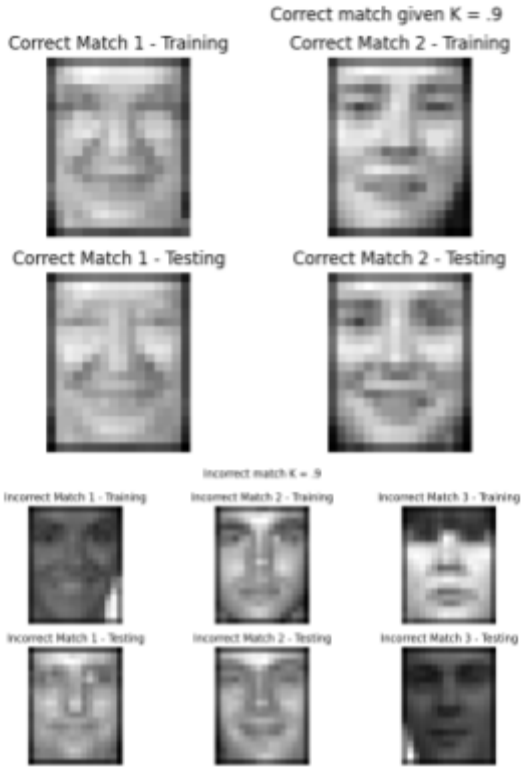

Figure 15: The matching images of the low resolution at 90%

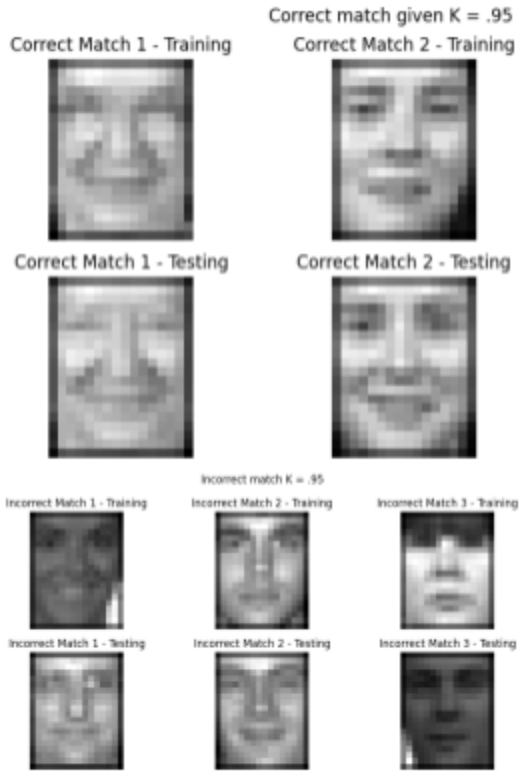

Figure 15: The matching images of the low resolution at 95%

D)

Theory: Given a lower dimension data set, do the same steps in part B of the experiment to detect intruders. Changing the training data by removing some IDs, creating new values and attempting to detect intruders or IDs not in the data set. With the new Eigenvalues and eigenvectors, calculate the maximum threshold by finding the max possible distance between the training image and testing image, and make intervals. If a value is below the threshold, they are considered an intruder, and if a value is above the threshold they are considered in the database. As the threshold increases, the number of false positives increases and when the threshold decreases, the number of true positives decreases. This is done at 95% data retention.

Figure 16 shows the ROC curve as of a result of the 95% data retention. Positive rate is greater than the equal point, the value is not that much different, explaining the previous CMC curve and the lower accuracy of the 95% data. With figure 17, it shows how low the threshold needs to be for the FPR to approach or equal the TPR. This shows that having more data retention is not always better both computationally and results wise.

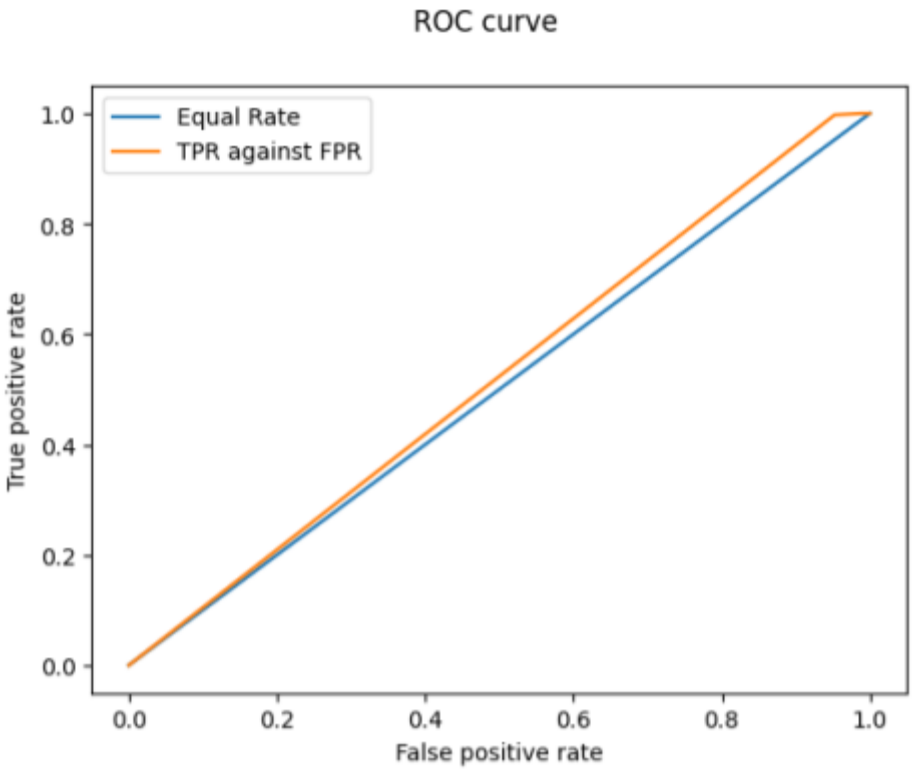

Figure 16: The ROC curve of the intruder detection at 95% data

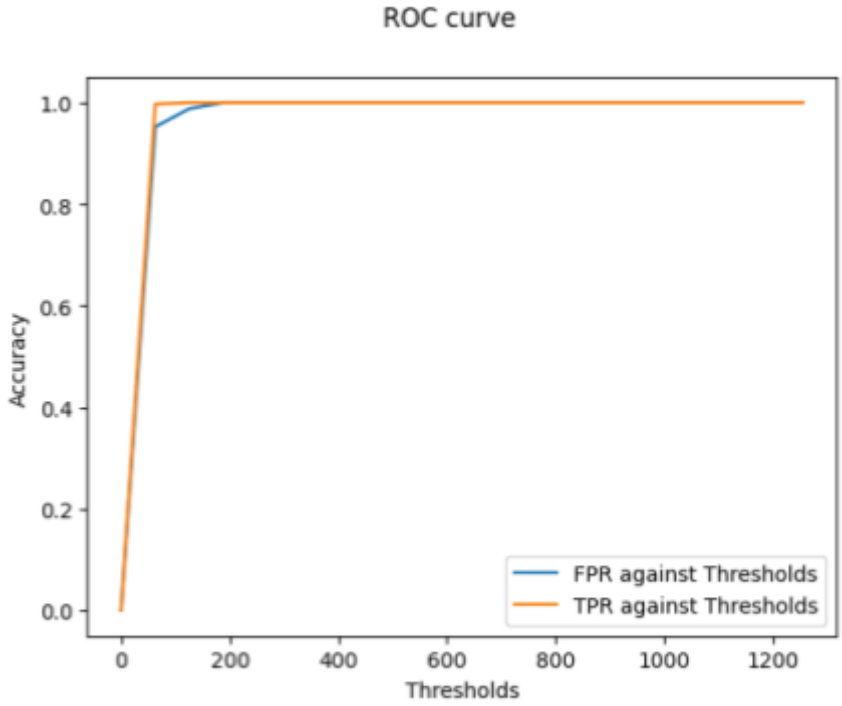

Figure 17: The ROC curve of the FPR and TPR against the thresholds.

E)

Theory: Comparing the results from the high resolution to the low resolution, there is a different pattern in the CMC graph. For higher resolution, the accuracy increases with a larger range and more data retention, while the lower resolution has an increasing accuracy from 80-90%, but drops at 95%. With the ROC curve the values show that at a lower resolution, there is less of a difference at different thresholds while the higher resolution shows a noticeable decrease in TPR and increase in FPR. The reason for this is due to the curse of dimensionality in the data. For the lower resolution, in order to use more of the eigenvectors, a larger amount of data is needed in order to make the test accurate. By increasing the number of features used, the lower resolution doesn't provide enough data in order to make proper predictions. This is why the 90% and 95% shows a decrease in accuracy instead of a usual increase. Overall, the low resolution images produce more inaccurate results compared to the high resolution. Without a larger amount of data, the lack of variations between the images make the results worse with a higher number of eigenvectors.

Compiling:

Library Used: This program uses the hash table library from hashtable-python. In order to run this program, pip install hashtable-python has to be run inside of the terminal to get the required library. Hashtable is used during intruder detection in order to perform a linear search for whether the ID is a intruder or not. In the code, the pip-install hashtable-python is already on the first line. If you are trying to copy the code, install it first and import the library HashTable from hashtable