Credit Hours System

Cairo University

Faculty of Engineering

Image Processing and Computer Vision

Project Report

Team #7

Ahmed Mohamed Ismail Nabeel	**1180501**
Moaaz Mohamed Elsherbini	**1180528**
Mostafa Ashraf Ahmed Kamal	**1180406**
Nader Youhanna Adib Khalil	**1180477**

Used Algorithms:

- Face Detection: Viola Jones algorithm: Paper
- Face Recognition: Eigenfaces algorithm: Paper

Experiment Results and Analysis

Face Detection:

- 98% accuracy on the Olivetti dataset
- Trained on 837 upfront-face images each of size 24x24 pixels

Points of Strength:

Very high accuracy and only one run over the dataset is required.

Points of weaknesses:

Training takes a lot of time. Also, it needs upfront-close face images dataset which is hard to find on the internet.

Dataset:

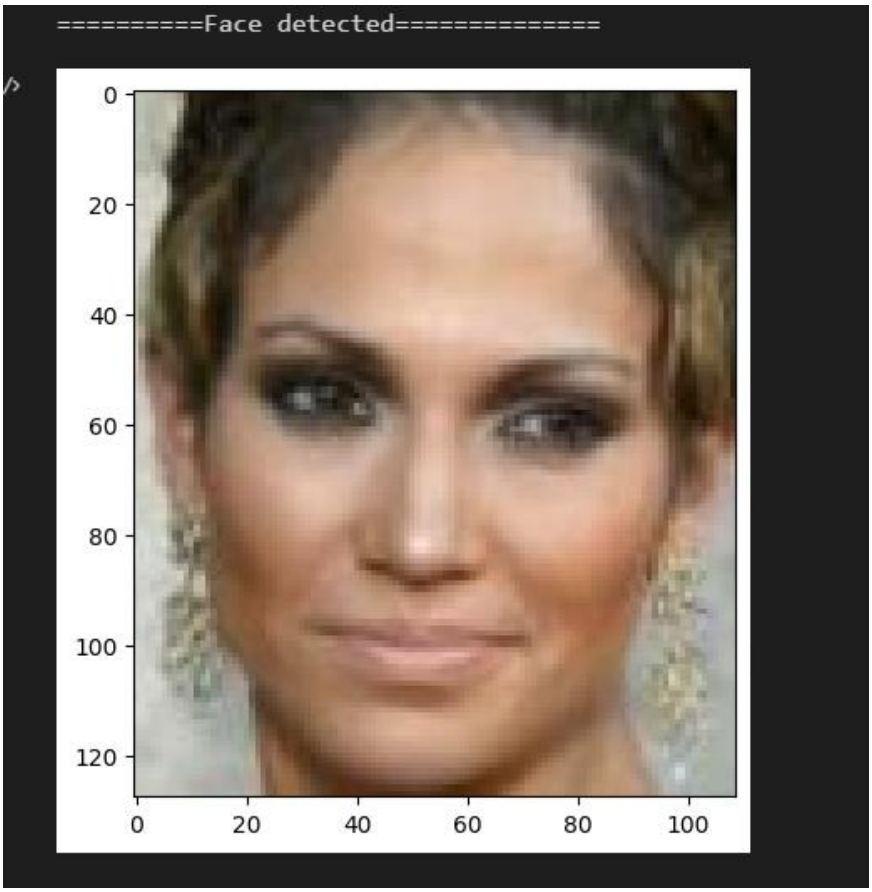
We used the Olivetti faces twice and manually added 37 faces to the training dataset for a total of 837 faces. The 37 faces were manually chosen from LFW dataset and manually cropped to match the required input.

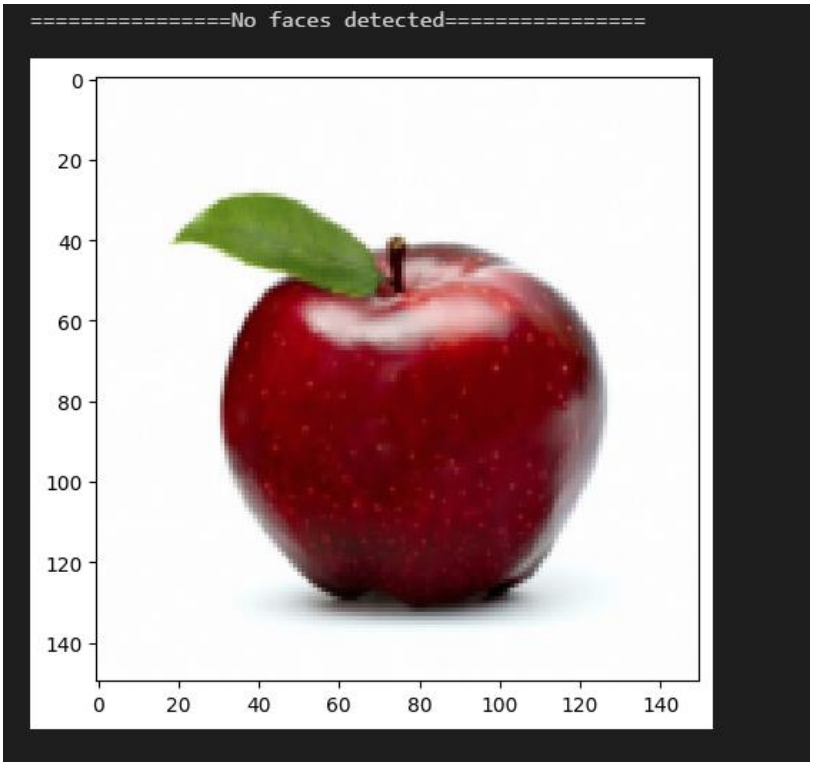
Accuracy:

```
Test Dataset Shape: (160, 24, 24)
Predicted:  0
Target:  1
Predicted:  0
Target:  1
Predicted:  0
Target:  1
Accuracy:  98.125 %
Faces detected: 97 / 100 , Accuracy: 97.0%
Non Faces detected: 60 / 60 , Accuracy: 100.0%
```

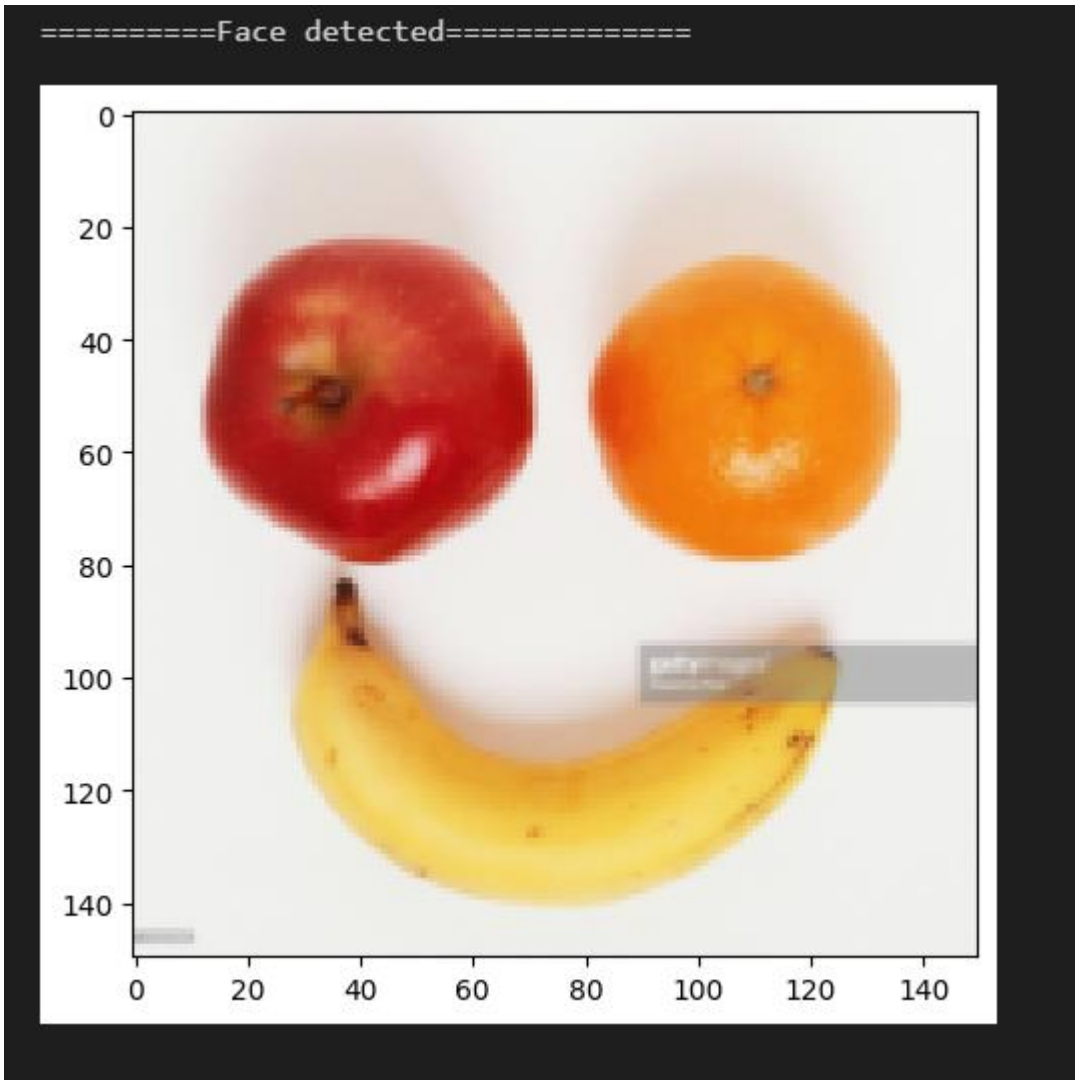
$$\text{Accuracy} = \frac{Number\ of\ correct\ predictions}{Total\ Number\ of\ test\ images} * 100$$

Correct Example:





Incorrect Example:



Face Recognition:

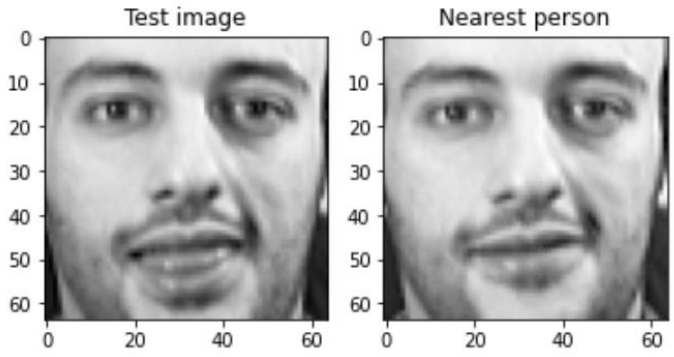
- 97% accuracy on the Olivetti Dataset
- 400 images of 40 people
- Each image is 64 × 64 pixels
- 320 images are used for training
- 80 images are used for testing

Example of two correct recognitions

```
40      plt.imshow(test_image.reshape(height, width), cmap='gray', label = 'Te
41      plt.show()
42      plt.imshow(train_images[nearest_person,:,:], cmap='gray', label='Neare
43      plt.show()
44
```

[39] ✓ 0.2s

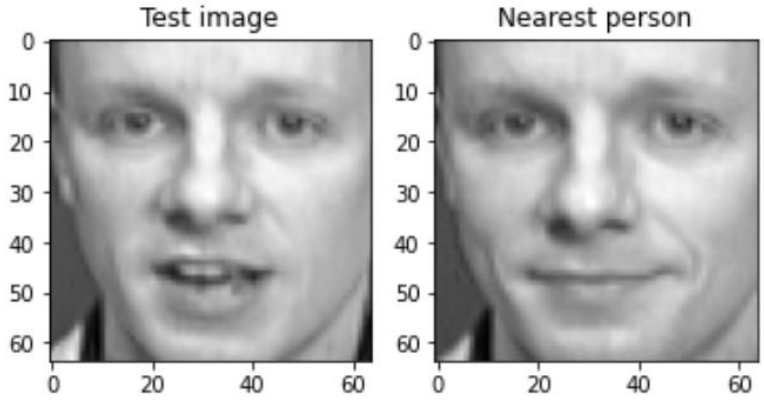
```
Correct
```



```
40      plt.show()
41      print('Incorrect')
42
```

[22] ✓ 0.2s

```
Correct
```

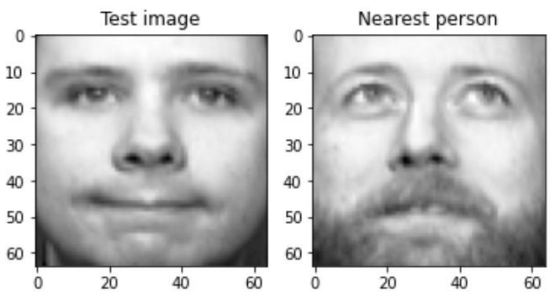


Example of an incorrect recognition

```
40      test_figure = plt.subplot(1, 2, 1)
41      test_figure.set_title('Test image')
42      plt.imshow(test_image.reshape(height, width), cmap='gray', label = 'Test image')
43      true_figure = plt.subplot(1, 2, 2)
44      true_figure.set_title('Nearest person')
45      plt.imshow(train_images[nearest_person,:,:], cmap='gray', label='Nearest person')
46      plt.show()
47
```

[70] ✓ 0.2s

```
Incorrect
```



Accuracy:

```
    threshold = float('inf')
    if (min_distance < threshold):
        if(y_train[nearest_person] == y_test[index]):
            if (min_distance > max_correct_distances):
                max_correct_distances = min_distance
            if (min_distance < min_correct_distances):
                min_correct_distances = min_distance
            sum_correct += min_distance
            count_correct += 1
        else:
            if (min_distance < min_incorrect_distances):
                min_incorrect_distances = min_distance
            if (min_distance > max_incorrect_distances):
                max_incorrect_distances = min_distance
            sum_incorrect += min_distance

print(f'Accuracy: {count_correct/trials*100}%')
```

[93] ✓ 2.3s

```
Accuracy: 96.8%
```

Accuracy is calculated as $\frac{number\ of\ correct\ predictions}{Total\ number\ of\ predictions}$ 1000 trials

- Points of strength: The algoritm does not need much time to train because it sees the training data only once (1 epoch)

- Points of weakness: The images must be close ups of faces. The algorithm did not perform well for Datasets where faces were not the main element in the image (15% accuracy for the LFW Dataset).

  This is expected of the Eigenfaces algorithm, because it does not try to extract certain features from the images that would make the difference between one face and the other. Rather, it makes the assumption that the faces distribution over the whole image space (which is all possible combinations of pixels ≡ all possible $width \times height$ images) is not random. Based on this assumption, the algorithm aims to calculate the Eigenvectors (called Eigenfaces) that best desribe the distribution of face images over the images space.

Work Division

Team Member	Task
Ahmed Mohamed Ismail	Face Detection
Moaz Mohamed Elsherbini	
Mostafa Ashraf Ahmed	Face Recognition
Nader Youhanna Adib	