FACE RECOGNITION USING EIGENVALUES REPORT

1.1 OBJECTIVE:

Design a real-time face recognition system using eigenfaces.

1.2 GOALS:

The primary goal of the task is to implement an algorithm for face recognition by using the method of Eigenvectors or PCA(Principal Component Analysis). The model(once trained on an authorized dataset with significant variation) would be able to learn new faces in an unsupervised manner.

1.3 CONCEPT:

PCA algorithm reduces the dimensionality of a dataset by considering only the principal components using linear projection such that the reconstruction error is minimal.

1.4 DATASET:

Dataset: AT&T "The Database of Faces" (formerly "The ORL Database of Faces")

Description: There are ten different images for 40 distinct subjects. According to “face-rec.org”, the images are taken at different times, varying the lighting and facial expressions.

Dimensions: 92 * 112

Image-format: **.pgm**

Image-format-used: **.png**

1.5 ALGORITHM:

Each image is represented as a linear combination of eigenfaces.

Eigenfaces are the eigenvectors of a set of faces. Each face image in the training set can be represented exactly in terms of the linear combination of eigenfaces

The algorithm is as follows:

1. Process the dataset. Here, it was found that there was a loss of data when we read .pgm files with popular python libraries like rasterio and PILLOW, hence drafted the code to convert .pgm files to .png files.
2. Store the image matrix into an array using the NumPy library and convert the image from W*H to W.H dimension.
3. Split the image matrix into train and test matrix(8 images per subject for train and rest for test in our project).
4. An average face is calculated. The normalized faces are calculated by subtracting the faces from the average face(Fig. 1.1).

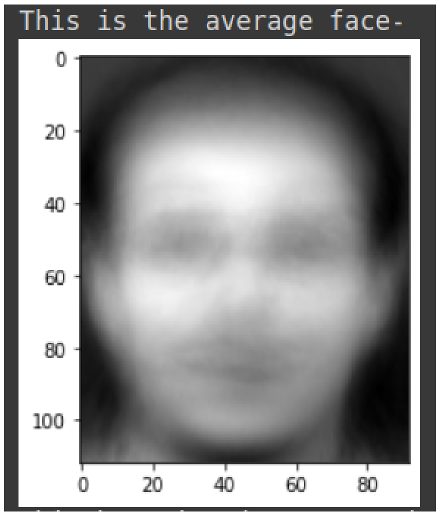

Fig 1.1 Average Image

5. Find the Covariance matrix by taking the dot product of Phi^T and Phi. We are taking the reverse product so that the dimension is reduced from W.H to the training size.
6. Find the eigenvalues and eigenvectors from the covariance matrix using the inbuilt eig functions. We select the largest k(principal components) eigenvalues and the corresponding vectors to create a low-dimensional k-vector space(eigenspace).
7. Projecting the Normalizedtraining images onto the K vector Space.
8. To predict a test image, we convert the test image to the same size as the train images. Here, they are of the same size already.
9. The test images are subtracted from the average face to get the normalized images.
10. Now, we project the test images on our eigenspace and calculate the distance of weights between our test images with each training image.

$$\omega_k = \mathbf{u}_k^T(\mathbf{\Gamma} - \mathbf{\Psi})$$

11. The minimum distance with the train image is the matched image with our test image. However, a threshold is required so that the minimum distance is below that threshold.

Otherwise, it is an unknown face. This threshold is not used(calculated heuristically) as the test contains no unknown face.

$$\epsilon^2 = \|\Phi - \Phi_f\|^2$$

1.6 RESULTS:

The algorithm correctly identifies 76 out of 80 test images.
The accuracy of the model is, therefore, 95%.
This accuracy would be lower for testing a new image.

1.6 REFERENCES:

1. P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001, Kauai, HI, USA, 2001, pp. I-I, doi: 10.1109/CVPR.2001.990517.
2. M. A. Turk and A. P. Pentland, "Face recognition using eigenfaces," Proceedings. 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Maui, HI, USA, 1991, pp. 586-591, doi: 10.1109/CVPR.1991.139758.
3. Pawangfg,“ML|Face Recognition Using Eigenfaces(Using PCA)”, https://www.geeksforgeeks.org/ml-face-recognition-using-eigenfaces-pca-algorithm/.