Face Recognition Using Eigenfaces

Jatin Ashish Dholakia

16110066

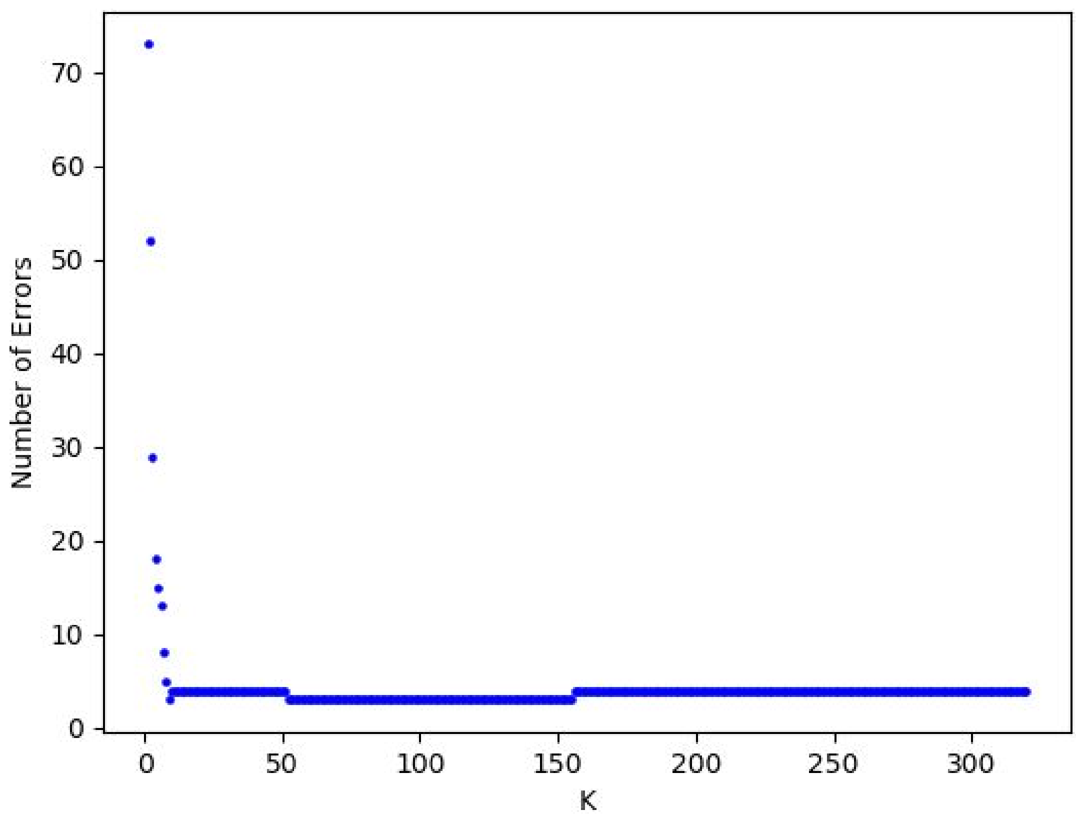
Much of the previous work on Automated Face Recognition ignores the issue of what aspects of face stimulus are important for identification. The paper by Turk and Pentland suggests a method to encode the relevant information of a face image, and compare one face encoding with a database of models encoded similarly. It is done by finding the principal components of the distribution of faces, or the eigenvectors of the covariance matrix of the set of images.

Each face image in the training set can be represented as a linear combination of the eigenfaces. However the eigenfaces that have the largest eigenvalues account for the most variance within the set of face images. Hence K most significant eigenfaces are chosen for computational efficiency.

The database of faces used for this purpose is taken from AT&T Laboratories Cambridge, Speech, Vision and Robotics Group. (Link). It consists of 10 images each of 40 distinct subjects. Each image has a dimension of 112*92. The images are organised in 40 directories and in each of these directories there are 10 different images of that subject. The database can be downloaded from this link.

Calculating Eigenfaces

- From the dataset of 400 images, 320 images (8 from each folder) are used for training and 80 (2 from each folder) are used for testing.
- All the training images are of same dimension M*N (here M=112, N=92). They are unrolled into a vector of size MN*1 and stored in an array. Hence size of array is MN*320.
- For each image I_i , let the corresponding vector be Γ_i. All the Γ_i are arranged column wise in a matrix of size MN*320. Mean is computed as follows:
  - $\Psi = \frac{1}{M} \sum_{n=1}^{M} \Gamma_i$
- The mean is subtracted from all the vectors.
  - $\phi = \Gamma - \Psi$
- Next step is to find the covariance matrix.
  - Covariance matrix is obtained as follows: (It has a size of MN*MN)
  - $C = \phi\phi^T$

- Since the covariance matrix is very large, finding the eigenvalues and eigenvectors directly is not practical. The following sub-steps are implemented. Let the eigenvectors of C be given by u_i and eigenvectors of $\phi^T \phi$ be given by v_i and corresponding eigenvalues by μ_i.
    - The matrix $\phi^T \phi$ is of the size 320*320. Its eigenvectors are found by:
        - $\phi^T \phi v_i = \mu_i v_i$
        - $\phi\phi^T \phi v_i = \mu_i \phi v_i$
        - $C\phi v_i = \mu_i \phi v_i$
        - $C u_i = \mu_i u_i$
    - Hence, the eigenvalues of $\phi^T \phi$ correspond to 320 largest eigenvalues of C. The corresponding eigenvectors of C are given by:
        - $u_i = \phi v_i$
    - The u_i are then normalised such that $\|u_i\| = 1$.
- The dimension of u_i is (MN*320). Out of 320 eigenvectors only K are required and significant for face detection. K is found as follows:
    - For all the values of K from 1 to 320, we compare the predicted output with the expected output and count the number of errors in each case. The smallest value of K which minimises the error is chosen.
    - The graph of K vs No of errors is given below:
    - 

  - The minimum number of errors = 3 for K = 52. Hence K is chosen as 52.
  - Hence dimension of u_i gets reduced to (MN*K)
- The weight matrix Ω is obtained by:. Its dimensions are (K*320)
  - $$\Omega = u_i^T \phi$$

Using Eigenfaces to classify face image

- Let Γ be an unknown face image. It is flattened into a column vector and normalised by subtracting mean.
  - $$\phi = \Gamma - \Psi$$
- It is projected onto the eigenspace as follows:
  - $$\Omega = u_i^T \phi$$
- The L_2 norm of the error between the calculated weights of all the images and the test image is found as follows:
  - $$e_r = \left\| \Omega - \Omega^l \right\|$$
- The index that minimises the error is the closest match to the test image. The class to which it belongs is predicted as the class of the test image.
- Out of the 80 test images (last 2 from each 40 folders), 77 are correctly classified. Accuracy = 96.25%

Threshold for Face Detection and Recognition

Haar Cascade is used to identify whether a face is present in an image or not. If a face is present, it crops the face and resizes it to the size of training images and proceeds further. If no face is found, “No face detected” is returned.

When the L_2 norm is calculated for a test image with all the weights, it is checked whether the minimum of all is below a threshold θ_e. If it is not, it does not lie in the classes observed in the training set. The threshold θ_e is obtained by taking the maximum of all the L_2 norms observed for correctly identified images. It is found to be 3205, hence θ_e is chosen as 3230.

References:

- https://ieeexplore.ieee.org/document/139758/
- https://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html
- http://www.vision.jhu.edu/teaching/vision08/Handouts/case_study_pca1.pdf
- http://ijarcet.org/wp-content/uploads/IJARCET-VOL-1-ISSUE-9-135-139.pdf
- https://www.youtube.com/watch?v=88HdqNDQsEk
- https://github.com/opencv/opencv/tree/master/data/haarcascades