Face Recognition using Eigen Faces: Project Proposal

Team Number: 11
Team Name: 21 din mein CG double
Team Members:

- Tushar Choudhary (2019111020)

- Suyash Vardhan Mathur (2019114006)

- Tejas Chaudhari (2019111013)

- Rutvij Menavlikar (2019111032)

Repository Link: https://github.com/Rutvij-1/Eigen-Faces-SMAI-M21-project

Problem Statement

Developing a computational model of face recognition is quite difficult, because faces are complex,
multidimensional, and meaningful visual stimuli. Unlike most early visual functions, for which we may construct detailed models of retinal or striate activity, face recognition is a very high level task for which computational approaches can currently only suggest broad constraints on the corresponding neural activity. Our aim is to develop a computational model of face recognition which is fast, reasonably simple, and accurate in constrained environments such as an office or a household.

Goals and Approach

The goal of this project is to develop a method for automated face recognition through the eigenvalues of distribution of faces(or, the "eigenfaces"). This project follows the work of *Matthew A. Turk et al*, and tries to capture the variation in the features of various faces, and uses those features to represent and detect newer faces.
Thus, the approach in this project would be to use the principal components of the distribution of faces, i.e. the eigenvectors of the covariance matrix of the set of face images. To reduce computational cost, we only consider first M eigenfaces. Now, any since any human face can be represented as a linear combination of these

eigenfaces, we can recognise faces based upon the values of those weights.
The face recognition process can be summarised in the following steps:

1. **Initialisation:** This step involves acquiring the training set of face images and calculating the eigenfaces. For this step, we first shift all the face images by the "average face", denoted by ϕ_i. Then, we obtain M orthogonal vectors, the **eigenfaces** u_n with eigenvalues λ_k of the covariance matrix $C = \frac{1}{M}\sum_{n=1}^{M} \phi_n\phi_n^T = AA^T$, where $A = [\phi_1\phi_2...\phi_M]$. **Note:** Since the matrix C is of order N^2, we instead solve the eigenvectors of the $M \times M$ matrix and take linear combination of the resulting vectors to solve the eigenvectors of the C matrix.

2. When a new face image is encountered, we calculate a set of corresponding weights for the M eigenfaces by projecting the new face image on each of the eigenfaces. For this projection, we perform the operation $\omega_k = u_k^T(\gamma - \psi)$ for k = 1,2,... M. Thus, we get the weights as the vector $\Omega^T = [\omega_1\omega_2...\omega_M]$, describing the contribution of each eigenface.

3. Now, we determine if the image is a face by checking if the image is sufficiently close to the space spanned by the eigenfaces("face space"). To do this, we compute the Eucledian distance $\epsilon_k = ||\Omega - \Omega_k||$, where Ω_k describes the kth face class, and reject the image as not a face if the distance is greater than a certain threshold.

4. If the image is recognized as a face, then classify the weight pattern as a "known" or "unknown" person. This is done by classifying the face into the class which has the minimum corresponding value of ϵ_k.

5. If the same unknown face is seen several times, then its characteristic weight pattern is calculated and incorporated into the known faces.

Dataset

We will be using a dataset prepared by Centro Universitário da FEI. Datasets consist of faces of 200 people and each person has two frontal images (one with a neutral expression and the other with a smiling facial expression), there are 400 full frontal face images manually registered and cropped.
Link to the dataset: https://github.com/lychengr3x/Face-Recognition-Using-Eigenfaces/tree/master/dataset

Expected Deliverables

1. **Code**: A working method of face recognition for a given image.

2. **Documentation**

3. **Sample outputs**

4. **Accuracy study**: With respect to hyper-parameters.

5. **Presentation**: Presentation, description of the project along with instructions to run the demo.

6. **Report**: Detailed report of the code, the outputs obtained and the conclusions derived from the accuracy study.

Milestones & Timeline

Aa Timeline	≣ Milestone
26 oct	Project allocation
7 nov	Project proposal submission
week 1	Relevant research and discussions, initial project layout with TA review
week 2	Implementing the basic pipeline
Nov 17th - 20th	Mid-evaluation
week 3	Compiling results
week 4	Testing with hyper-parameters and getting all deliverables ready
Dec 1st - Dec 4th	Final presentation
Dec 4th (11:59 PM)	Final submission

Work Distribution

Aa Work	≣ Done By
Project proposal	Equal and based on everyone's inputs
Initial research work	independent and equal
Implementing the basic pipeline	Pair 1 will do first half and PCA, pair 2 will do second half
Debugging, testing and compiling results	Pair 1
Getting deliverables ready	Pair 2
Documentations	Pair 2

Pair 1 - Tushar and Suyash
Pair 2 - Rutvij and Tejas

This work distribution is our initial estimate, and might be changed to maintain equal workload for everyone.