Eigenfaces

Team Family (8)

Aravind Narayanan - 2019102014

Abhayram A Nair - 2019102017

Vaibhav Bhushan - 2019112019

Rishin Chakraborty - 2019112008

LINK TO PROJECT REPOSITORY:
https://github.com/aravind-3105/SMAI-Family-Project

Overall Goals

- The primary goal of this project was to implement the method of Eigenfaces for face recognition by projecting the face images on the feature space (face space) which best represents the variations among distinct faces.
- The face space is defined as the “Eigenfaces ", which are the eigenvectors of the set of faces.
- The model will have the ability to learn new faces in an unsupervised manner.
- We also additionally implement fischerfaces and explore a pre-written LBH method to compare accuracies.

Datasets

We used two datasets for training and testing face recognition algorithms. We also used a collection of images from pinterest as a dataset to demonstrate face detection.

- AT&T dataset - The images are organised in 40 directories (one for each subject), which have names of the form sX, where X indicates the subject number (between 1 and 40). In each of these directories, there are ten different images of that subject, which have names of the form Y.pgm, where Y is the image number for that subject (between 1 and 10)
- yaleB - The Yale Face Database B contains 2432 images of 38 human subjects under different poses and illumination conditions..

Eigenfaces algorithm

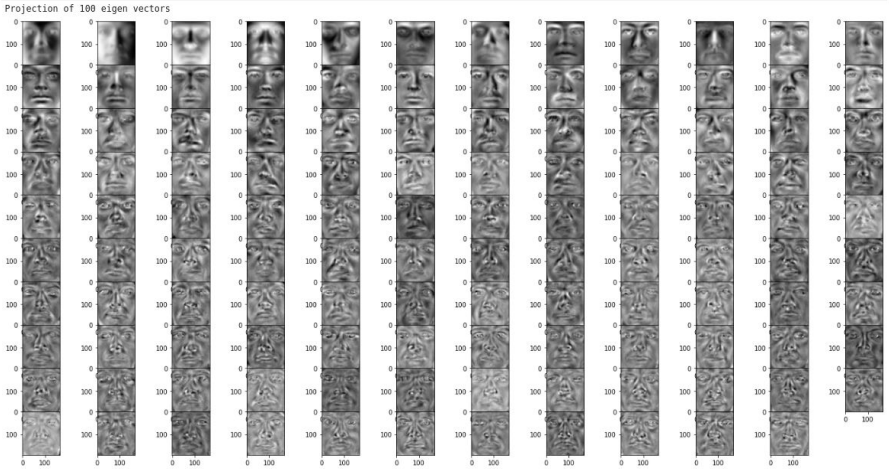
The gist of the approach is that the faces are reduced to vectors that preserve only those characteristics of the face which are useful for recognition. Rest of the faces are expressed as a linear combination of these faces.

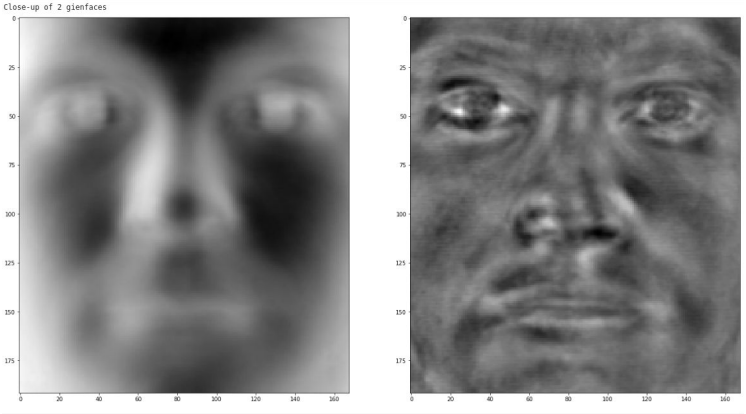
1. Obtain all images given in the dataset and store them into vector X
2. An average ‘face’ is calculated. This average is then subtracted from each of the vectors and the resulting difference is stored in an N^2xM matrix. This matrix is storing the variation from the average for each face vector. We call this matrix A.

Eigenfaces algorithm continuation

3. We calculate the eigenvalues and eigenvectors for the smaller covariance matrix and use this relation to map them to the original covariance matrix. This gives us M eigenvectors.

4. We reduce these M vectors to the K best eigenvectors. These are just the K largest eigenvectors. These will be able to capture maximum variations from the average face. These resultant eigenvectors are called eigenfaces.

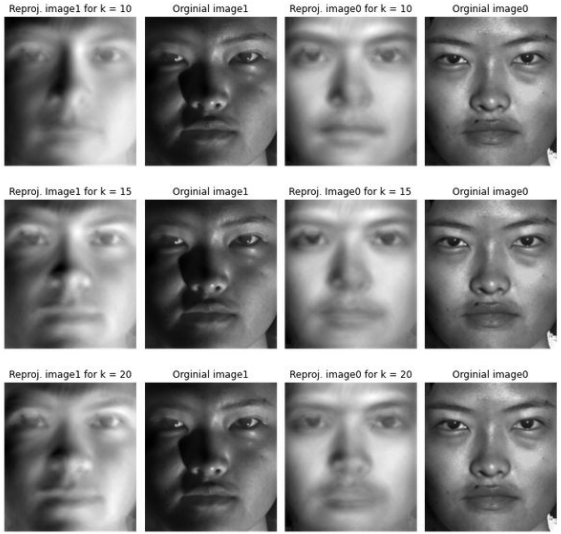




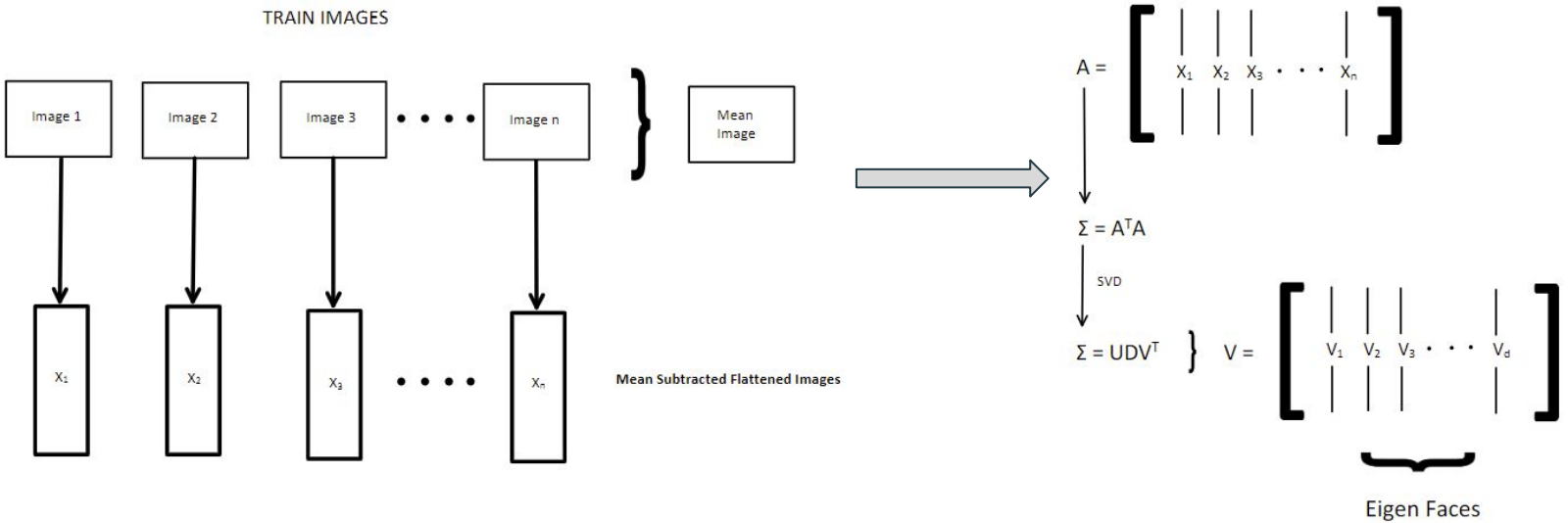
Eigenfaces algorithm continuation

5. All the training set faces can now be represented as a linear combination of the eigenfaces. Each face is now stored as a vector containing the coefficients of this linear combination.

6. Whenever a new image is received, we convert this image into a linear combination of eigenfaces and do a nearest neighbor analysis with existing training set faces in this space to get the final prediction.

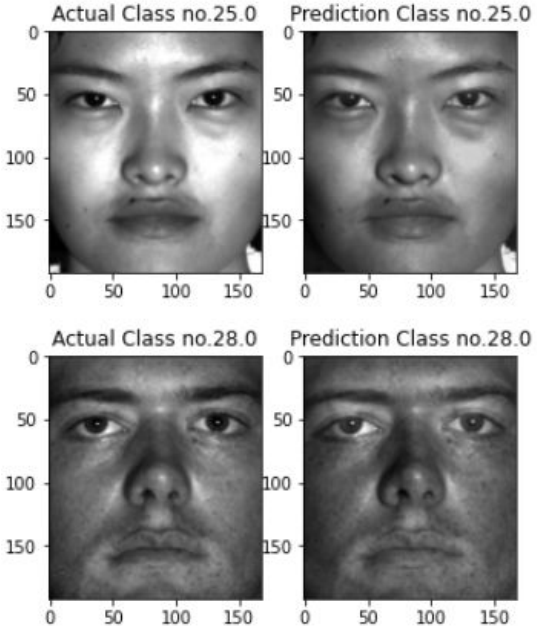


Visualisation of the algorithm

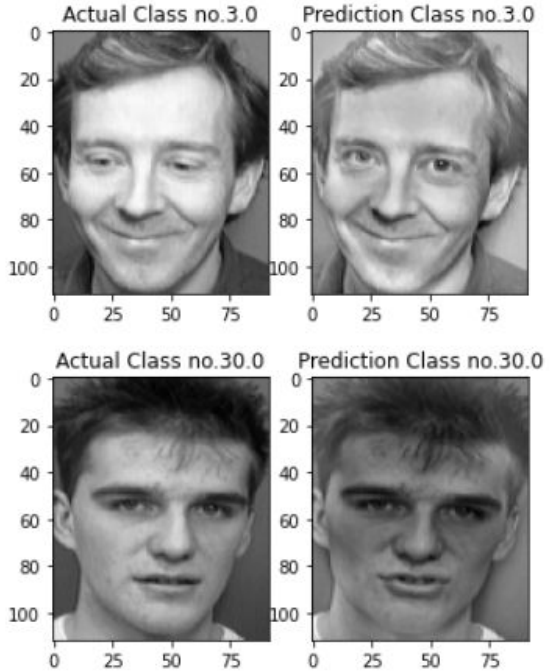


Outputs

Classification by eigenfaces in YaleB dataset:

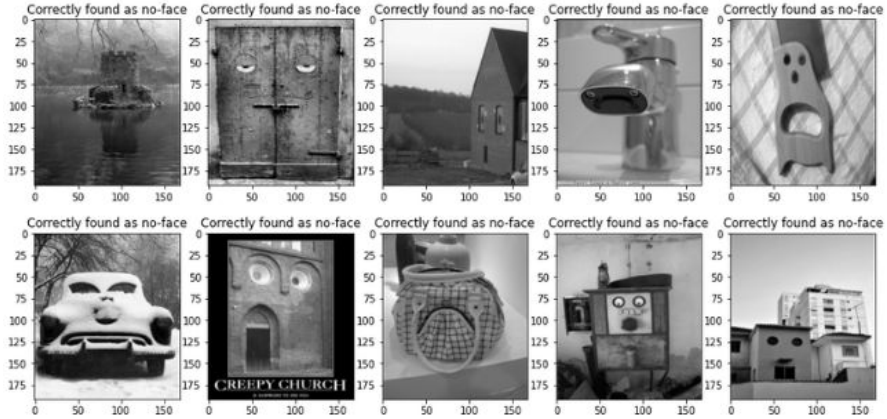


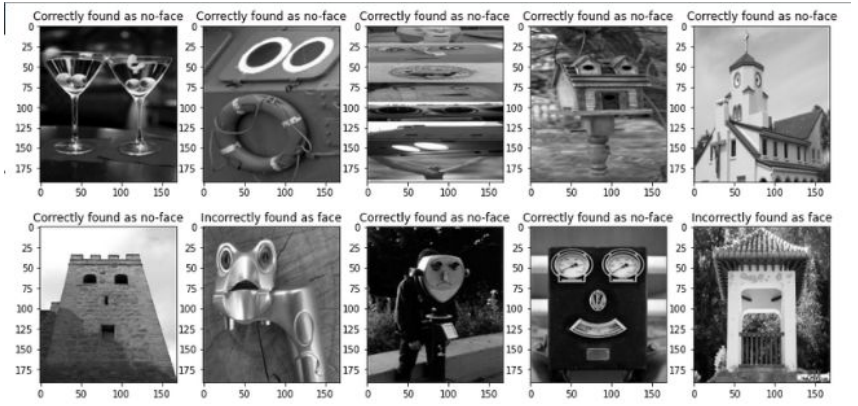
Classification by eigenfaces in AT&T dataset:



Outputs

Checking by eigenfaces for non-faces:





Performance using Eigenfaces

Dataset and Operation	Accuracy
AT&T Face, Face Recognition	95%
Yale B Face, Face Recognition	84.21%
No-Face Dataset, Face Detection	90%
Combination of No-Face and AT&T, Face Detection	81.25%

Fischerfaces Algorithm

We saw that PCA was useful in extraction of facial features for recognition. We need to extract best features which can discriminate between classes. These features (Fischerfaces) then can be used to make prediction given a face image.

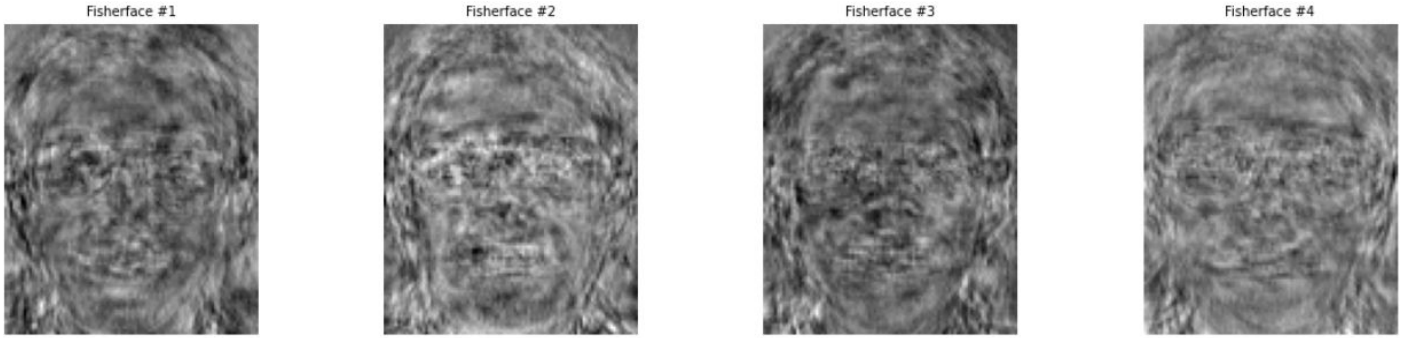
1. We start with converting the images into matrix form (X), where it can be arranged row wise or column wise. Each image has its corresponding class stored in vector (y).
2. Let's say we have c number of classes, N is total number of samples, next we project X into (N-c) dimensions using PCA and obtain eigenvectors W_{PCA} .

Fischerfaces Algorithm contd.

3. We now calculate Sw and Sb , within class & between class scatter respectively, and apply LDA and maximize the Fischer index which is the ratio of the determinant of between class and within class scatter.

4. The solution obtained is set of eigenvectors W_{FLD} , and hence we get $W = W_{PCA} * W_{FLD}$, where W is called Fischer Faces.

Example of Fischerfaces (AT&T database)



Output

Examples from Yale B database

Expected = 20 Predicted = 20

Expected = 25 Predicted = 25

Examples from AT&T database

Expected = 9 Predicted = 9

Expected = 38 Predicted = 38

Performance using Fischerfaces (YaleB)

Trial Number	Accuracy
1	81.70%
2	81.73%
3	73.93%
4	71.43%
5	72.93

Trial Number	Accuracy
6	68.35%
7	79.37%
8	77.48%
9	86.17%
10	67.79

Comparison with other implementations

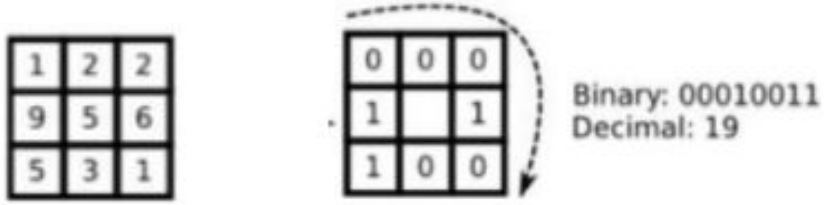
We also trained and tested some other implementations of face recognition on the same datasets to compare our own implementations.

- We tested the OpenCV implementation of Eigenfaces algorithm on the AT&T. We performed 10 trials with a 80/20 split and obtained an average accuracy of 97.5%

- We tested an implementation of the Linear Binary Pattern Histogram algorithm on the AT&T dataset.

Comparison with other methods

The LBPH algorithm can be understood by the following example:



This had an average accuracy of 96.25% on the AT&T dataset.

Project timeline

Timeline	Milestones
26th October 2021	Project Allocation
7th November 2021	Project Proposal
Week - 1	Assigned paper reading and Fischer faces paper formalized the structure of basic implementation
Week - 2	• Implementation of both the core methods from scratch. • Reading about the comparison method(LBPH).
19th November 2021	Mid-Evaluation
Week - 3	Comparison of the accuracies for basic classification
Week - 4	Implementation of LBPH and tabulating accuracies for non-faces and faces
8th December 2021	Final Evaluation

Work Distribution

- **Abhayram -** implementation of PCA, generating eigenfaces, testing on YaleB dataset
- **Aravind-** implementation of eigenfaces on AT&T dataset, comparison of faces, no faces
- **Rishin-** implementation of Fischer faces functions and implementation of the inbuilt functions for eigenfaces
- **Vaibhav -** implement the Fischer faces on the YaleB, LBPH on YaleB

References

1. Face recognition using Eigenfaces (https://ieeexplore.ieee.org/document/139758)
2. Eigenfaces vs. Fisherfaces: recognition using class specific linear projection (https://ieeexplore.ieee.org/document/598228)
3. LBPH Based Improved Face Recognition At Low Resolution (https://ieeexplore.ieee.org/document/8396183)
4. LPBH Implementation (https://github.com/opencv/opencv_contrib/blob/4.x/modules/face/src/lbph_faces.cpp)
5. Eigenfaces using OpenCV (https://github.com/opencv/opencv_contrib/blob/4.x/modules/face/src/eigen_faces.cpp)

Thank you!