FACE RECOGNITION SYSTEM

PROJECT REPORT

PROF DESIRE SEDIBE

BY
UMAMAHESWARAN RAMAN KUMAR

JANUARY 6, 2017
UNIVERSITY OF BURGUNDY

1. INTRODUCTION

Face Recognition FR is one of the most known/famous applications of Image Analysis and Computer Vision. The basic idea of any FR system is to extract a set of interesting and discriminative features from the face images with the goal of reducing the number of variables. If we need to compare an input image I_p with a set of database images I_1, …,I_q each pixel can be considered as a variable so we have a very high dimensional space(every image is a point in a space dimension $d = MN$, M and N being the size of image). Using Principal Component Analysis(PCA) the problem can be simplified b reducing the dimensionality.

2. METHOD

There are 3 main steps in the Face Recognition system.

i. Normalization
ii. Train Recognition System
iii. Finding Matching Faces

2.1. NORMALIZATION

The training and testing images usually are of different sizes and the features to be identified may or may not in the same location as expected. These images need to be of same size and the features need to be aligned to specific location to use it for recognition system. In this step, we convert the images to a 64 x 64 image and then transform the features to specific locations. Below are the features we consider and their corresponding location in the 64 x 64 normalized image.

i. Left eye – (13, 20)
ii. Right eye – (50, 20)
iii. Nose – (34, 34)
iv. Left mouth corner – (16, 50)
v. Right mouth corner – (48, 50)

All the images used have a corresponding text file containing the coordinates of the required features in the image. This information is used to find the affine transformation used to transform the image to the 64 x 64 sized image with the features aligned to the predetermined locations. The affine transformation is defined by six parameters

$$A = \begin{matrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{matrix} \qquad b = \begin{matrix} b_1 \\ b_2 \end{matrix}$$

Finding the affine transformation that transforms a set of images to normalized images is an iterative process. A feature $f_i = \begin{matrix} x \\ y \end{matrix}$ is mapped to the corresponding predetermined location f_i^P by the equation

$$f_i^P = Af_i + b$$

Since there are 10 equation and 6 unknown we have an over-determined system which is solved using SVD. In matlab the inverse to solve the system is found by using pseudo inverse and the command is *'pinv'*.

The iteration is repeated until the difference in the average of the aligned features of all the images in the current iteration and the previous iteration is less than a threshold value. In this case the threshold value is set to 10. Now we have the affine transformations that maps the faces to 64 x 64 image. These images are then stored in 'Normalized Images' folder created inside the training images folder and testing images folder.

2.2. TRAIN RECOGNITION SYSTEM

An image I_i is treated as a vector X_i in the d-dimensional space ($d = MN$). The vector is obtained by concatenating the rows of the image. If we have *p* training images then the entire training data set is written as a *p x d* matrix *D*, where each row of *D* corresponds to one training image.

$$D = \begin{matrix} X_1 \\ X_2 \\ \vdots \\ X_p \end{matrix}$$

After matrix D is formed we need to subtract the mean from all the pixels. The mean is calculated across each pixel across all the images. PCA (Principal Component Analysis) is the method used to reduce the dimensionality of the problem set. For PCA we need to compute the covariance matrix from *D* matrix which is given by

$$C = \frac{1}{p-1} D^T D$$

After we find the covariance matrix we need to find the eigenvectors for the matrix and keep the *k* eigenvectors corresponding to *k* largest eigenvalues contributing to typically 95% of all the eigenvalues and these are the principal components. After finding the eigenvectors the training images are multiplied by the projection matrix *P* created from the *d* x *k* eigenvectors and we obtain the eigenfaces.

The complexity increase easily as we need to find *d* eigenvectors of the matrix *C*. Since we have few training data the complexity can be reduced by finding the *p* eigenvectors of C' which is given by

$$C' = \frac{1}{p-1} D D^T$$

Then putting the eigenvectors of *C'* as columns of matrix *P'* the eigenvectors of *C* can be determined by $D^T P'$.

2.3. FIND MATCHING FACES

The testing images I_i are represented as vectors X_i and these vectors are projected onto the PCA space to get the corresponding feature vector by multiplying with the projection matrix P. Now to identify the best match we need to take the Euclidean distance between the feature vector of the test image with all the feature vectors of all training images. The best matched image is the one with the least Euclidean distance.

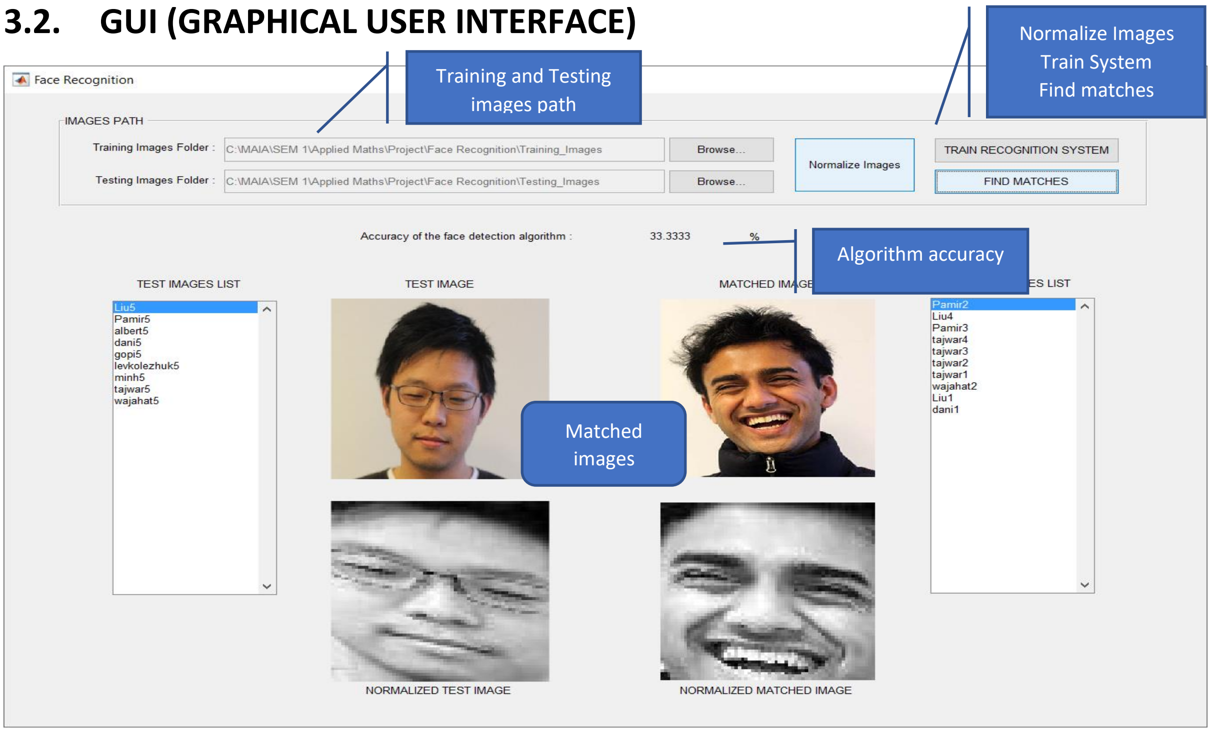
3. IMPLEMENTATION

3.1. IMPORTANT FUNCTIONS

Listed below are few important functions implemented.

i. **normalize_images64x64**
   This function normalizes the training images and test images
ii. **find_affine_transformation**
   This function finds the affine transformation for a given set of features
iii. **apply_affine_transformation**
   This function applies the affine transformation on the image and give the new transformed image
iv. **find_principal_components**
   This function finds the first 'k' principal components of a matrix which corresponds to the given percentage of eigen values
v. **distance**
   This function finds the Euclidean distance between 2 given vectors

3.2. GUI (GRAPHICAL USER INTERFACE)



Training Images Folder : C:\MAIA\SEM 1\Applied Maths\Project\Face Recognition\Training_Images

Testing Images Folder : C:\MAIA\SEM 1\Applied Maths\Project\Face Recognition\Testing_Images

Accuracy of the face detection algorithm : 33.3333 %

Liu5
Pamir5
albert5
dani5
gopi5
levkolezhuk5
minh5
tajwar5
wajahat5

Pamir2
Liu4
Pamir3
tajwar4
tajwar3
tajwar2
tajwar1
wajahat2
Liu1
dani1

4. EXPERIMENTS

Number of training images -> 36

Number of testing images -> 9

Accuracy of algorithm for the given test data set -> 33.33 %

Note: All the images are not considered because the data format of features provided in the text files are incoherent.

5. COMMENTS AND LEARNING OUTCOME

- A large dimension problem can be reduced to distinctly small dimension by using PCA
- GUI programming to handle large sets of data
- The accuracy of the algorithm changes drastically when the number of images are increased or decreased which signifies the algorithm is not fully dependent on the number of training data images alone.