Machine Learining for Computer Vision Coursework 1
Face Recognition by PCA

David Angelov
MEng Electrical and Electronic Engineering
Imperial College London
david.angelov12@imperial.ac.uk

Huaqi Qiu
MSc Communication and Signal Processing
Imperial College London
h.qiu15@imperial.ac.uk

1. Question 1

In this coursework, we were given 10 normalized and vectorized face images for each of the 52 different people (referred to as 'class' in this report). Firstly, the given data set was separated into training data and testing data. In this coursework, 80% of data was used for the purpose of training and 20% of data were used for training. The partition was done by randomly selecting 8 out of 10 images for training. Naturally, the remaining 2 images in each class were used for testing. This partition method ensured the randomness of the data sets and full coverage of classes.

Next, we applied PCA to the training data by projecting the faces to so called face space, which is spanned by M eigenfaces. To acquire the eigenfaces, we started with computing the mean face $\bar{\boldsymbol{x}}$ from the n training images x_n by

$$\bar{\boldsymbol{x}} = \frac{1}{N}\sum_{n=1}^{N} \boldsymbol{x_n} \tag{1}$$

and the resulting mean face image is shown in Figure 1.

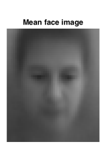

Figure 1. Mean face

Then, the all training faces were normalized by subtracting the mean face

$$\boldsymbol{\phi}_n = \boldsymbol{x_n} - \bar{\boldsymbol{x}} \tag{2}$$

Figure 2. The first 16 eigenfaces visualised

and the matrix $\boldsymbol{A}$ can be acquired by

$$\boldsymbol{A} = [\phi_1, \phi_2, ..., \phi_N] \tag{3}$$

Then the covariance matrix was calculated by

$$\boldsymbol{S} = \frac{1}{N} AA^T \tag{4}$$

The eigenvectors $\boldsymbol{u_i}$ were calculated from the covariance matrix S by $\boldsymbol{Su_i} = \lambda_i \boldsymbol{u_i}$. The dimension of the training data vector x_n is $D = 2576$, denoted by $x_n \in R^D$ (for the data provided for this coursework). To reduce the dimension and maximising the variance of projected data, we selected M of the eigenvectors corresponding to the M largest eigenvalues λ_i as eigenfaces. We found that among all the λ_i ($i = 1, 2, ..., D$), 415 out D eigenvalues were considered large (or non-zero), with order of magnitude ranging from 1 to 5. The remaining $D - 415 = 2161$ of the eigenvalues were considered as zeros with order of magnitude ranging from -10 to -13. In this coursework, we selected the largest $M = 50$ eigenfaces as the basis vectors for the face space. In Figure 2, we visualised 16 out of these 50 eigenfaces. The eigenfaces were visualised by reshaping the x_n vector to a 54×46 matrix.

To apply the PCA method, we projected every normalised training face ϕ_n to the M-dimensional subspace. Namely,

$$\omega_n = [a_{n1}, a_{n2}, ..., a_{nM}] \tag{5}$$

where the projection of nth image on the ith eigenface is denoted by

$$a_{ni} = \phi_n^T \boldsymbol{u_i},\ i = 1, ..., M$$

Thus, the training faces were represented by its projections ω_n.

2. Question 2

It can be noticed from Eq.4 that the dimension of the covariance matrix is $D \times D$, which equals to the number of pixels in an image and is typically large. Under the consideration of computational efficiency, we implemented another proposed method, in which the covariance matrix is calculated by

$$S = \frac{1}{N} A^T A \tag{6}$$

Since $A \in R^{D \times N}$, the covariance matrix now has dimension of $N \times N$, where N equals to the number of images and is typically much smaller than D. As computing eigenvectors of large matrices is computational expensive, this method is preferred if proved equally effective for PCA. The resulting none-zero eigenvalues from Eq.6 had the same value of that using Eq.4. Interestingly, the number of non-zero eigenvalues in this case is also 415. Note that the eigenvectors using this method were converted to the D-dimensional eigenvectors by

$$\boldsymbol{u_i} = A\boldsymbol{v_i} \tag{7}$$

where v_i denotes the eigenvectors calculated using the covariance matrix in Eq.6. Figure 3 shows the visualised 16 eigenfaces. Comparing to the eigenfaces produced by the previous method (Figure 2), we notice that the face contour of the eigenfaces are the same. The difference in grey scale value could be caused by the randomisation of the data set and automatic normalisation by the image display function `imagesc` in MATLAB.

In conclusion, the methods for the computation of covariance matrix in Q1 and Q2 were proven equally effective for PCA, while the latter method is less computational expensive. Hence we used $S = (1/N)A^T A$ to compute the covariance matrix for image recognition in later sections of this coursework.

3. Question 3

Figure 3. The first 16 eigenfaces visualised for question 2

A. Appendix 1 Matlab Code

A.1. Init.m

```
% Please run this script under the root folder

 clearvars -except N;
 close all;

% addpaths
 addpath('./internal');
 addpath('./external');
% addpath('./external/libsvm-3.18/matlab');

% initialise external libraries
 run('external/vlfeat-0.9.18/toolbox/vl_setup.m'); % vlfeat library
% cd('external/libsvm-3.18/matlab'); % libsvm library
% run('make');
% cd('../../..');

% tested on Ubuntu 12.04, 64-bit, Intel  Core ?i7-3820 CPU @ 3.60GHz    8
```

B. matlab code part 2

```
% Please run this script under the root folder

 clearvars -except N;
 close all;

% addpaths
 addpath('./internal');
 addpath('./external');
% addpath('./external/libsvm-3.18/matlab');

% initialise external libraries
 run('external/vlfeat-0.9.18/toolbox/vl_setup.m'); % vlfeat library
% cd('external/libsvm-3.18/matlab'); % libsvm library
% run('make');
% cd('../../..');

% tested on Ubuntu 12.04, 64-bit, Intel  Core ?i7-3820 CPU @ 3.60GHz    8
```