CS549/RBE549 Computer Vision
Face Recognition
Assignment 3

John S. Nafziger, Ph.D.
Adjunct Professor, Computer Science
jsnafziger@wpi.edu

Due 06-November-2017

1 Introduction

The purpose of this MATLAB based assignment is to implement the Eigenface and Fisherface methods for recognizing faces. You will be using face images from the Yale Face Database B where there are 10 faces under 64 lighting conditions, see Figure 1. Using your implementation, you will evaluate the ability of the algorithm to handle lighting conditions of the probe images that differ from those in the training images.

Specific instructions are in the following sections and on the Canvas site. Each section contains instructions that provide overview, specifics on write-up, rubrics and hints. In addition,

- Download the supplied starter matlab code and image data from Dropbox
- Prepare a project report (Section 5) using naming convention:
  FirstName_LastName_HW3.pdf

Figure 1: Example Faces from Database.

- Include all the code you implement for each of the sections. Remember to set relative paths so that we can run your code out of the box.
- Combine code and project report into **FirstName_LastName_HW3.zip** before uploading to canvas.

2 Relevant Papers

- M. Turk, A. Pentland (1991) "Face recognition using eigenfaces", Proc. IEEE Conference on Computer Vision and Pattern Recognition. pp. 586591.
- M. Turk, A. Pentland (1991) "Eigenfaces for recognition", Journal of Cognitive Neuroscience 3 (1): 7186.
- P.N. Belhumeur, J.P. Hespanha, D.J. Kriegman (1997) "Eigenfaces vs. Fisherfaces: recognition using class specific linear projection", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol: 19(7)

3 Recognition with Eigenfaces (PCA)

3.1 Step 1: PCA

Take each 50 50 pixel training image and vectorize into a 2500-dimensional vector. Then perform principal component analysis (PCA) on the entire set of training image vectors, retaining the first d principal components. Use the trick (discussed in class) for avoiding the full 2500x2500 covariance matrix. The d eigenvectors (when reshaped and displayed as images) are the Eigenfaces. You will display the top eigenfaces as images in your report.

3.2 Step 2: Classification

For each of your training images, project to the d-dimensional Eigenspace. Once this is done, classification will be performed by nearest neighbor with the L2 (Euclidean distance) metric in the Eigenspace.

3.3 Step 3: Evaluation

Evaluate your algorithm on the frontal pose of the ten people in the Yale Face Database B. These images have been cropped and aligned for you. From the set of all ten individuals, we will consider 5 subsets, indexed as below:

Set 1. person*01.png to person*07.png
Set 2. person*08.png to person*19.png
Set 3. person*20.png to person*31.png
Set 4. person*32.png to person*45.png

Set 5. person*46.png to person*64.png

4 Recognition with Fisherfaces (FLD)

Extend your Eigenface algorithm to perform the Fisherface algorithm discussed in lecture.

4.1 Step 1

Project the training images via Eigenfaces to a n c dimensional space (n=number of images, c=number of different people.

4.2 Step 2

Apply Fishers Linear Discriminant to obtain c 1 dimensional feature vector for each image.

5 Project Report Writeup

Please submit a project report (PDF) describing your algorithm and any decisions you made to write your algorithm a particular way. Then you will show and discuss the results of your algorithm. Points for each section listed in parenthesis.

5.1 Eigenface algorithm

- Train your Eigenface algorithm with d = 9 and d = 30 on all images in subset 1 (70 images). Then, evaluate your algorithm on subsets 1-5, and report the error rates in a table (error rate for each subset). For subset 1, you would expect perfect recognition because it is used for training. Include pseudocode (or equations) for the Eigenface recognition algorithm. (15%)

- Display the top 9 eigenvectors (eigenfaces) after training. I suggest using subplot(3,3,k) with imagesc and axis image, axis off, colormap gray and save the figure as a png. (5%)

- For both d = 9 and d = 30, display one original and the corresponding reconstructed face from each subset. The reconstructed face is constructed from the mean vector and from the eigenfaces and their coefficients. I suggest using subplot(2, 5, k) to display all five original and reconstructed faces in one figure. (5%)

5.2 Fisherface algorithm

- Evaluate the Fisherface algorithm. Train your Fisherface algorithm with c =10 and c =31 on all images in subset 1, and classify subsets 1-5 as in Part A. Report the error rates in a table (error rate for each subset). Use the same table as in Part A. Include pseudocode or equations for the Fisherface algorithm. (15%)

- Now retrain PCA and FLD using both subset 1 and subset 5 as training data. Test on all subsets. (10%)

6 Notes

- You may NOT use built-in PCA functions like pcacov or princcomp.
- Use eig/svd for computing the PCA and FLD subspaces. Be careful about whether the first or last columns correspond to the largest eigenvalues, as it depends on the way that you call eig.
- You should pre-normalize each face by subtracting its mean and dividing by its standard deviation, as this will lead to better performance.
- You are encouraged to explore further on your own. Try different sized subspaces, or try training on different subsets and see how it impacts the others. Look at the smallest eigenvectors or at reconstruction error as the number of eigenvectors vary. You dont have to include these extra explorations in your report.

Acknowledgements

The data and basic design of this assignment are from David Kriegman.

7 Helper code

Please use the included helper code to read and process the face images, Figure 2. See the code comments for the input and output arguments. Images are returned in Cell arrays. See: `https://www.mathworks.com/help/matlab/cell-arrays.html` for more information.

```
% ---------------------------------
% ---------------------------------
% Function readFaceImages()
% A helper function for reading images from an image Directory
% D. Hoiem
% Useage:   [im, person, number, subset] = readFaceImages('
   faces')
% Input:    imdir - Data Directory (String)
% Ouput:    im - an array of images in {cell} format
%           number - lighting condition index [1,64]
%           person - person index [1,10]
%           subset - grouping based on lighting condition [1,5]
% ---------------------------------
function [im, person, number, subset] = readFaceImages(imdir)

files = dir(fullfile(imdir, '*.png'));
for f = 1:numel(files)
  fn = files(f).name;
  person(f) = str2num(fn(7:8));
  number(f) = str2num(fn(10:11));
  if number(f) <= 7
    subset(f) = 1;
  elseif number(f) <= 19
    subset(f) = 2;
  elseif number(f) <= 31
    subset(f) = 3;
  elseif number(f) <= 45
    subset(f) = 4;
  elseif number(f) <= 64
    subset(f) = 5;
  end
  im{f} = im2single(imread(fullfile(imdir, fn)));
end
```

Figure 2: Read Face Images Helper Code.