In the Name of God
Project-2
SVD Face Recognition
Department of Electrical Engineering - Shiraz University

Instructor: Dr. Hamed Masnadi-Shirazi Spring 2014

1 Introduction

The object of this project is to build a face recognition system using the SVD decomposition. A face recognition system, tries to recognize a persons face among a number of faces it has previously seen. A simple face recognition system can be built by taking a number of pictures of a persons face and storing them in memory. Every time an individual appears before the system, a picture is taken and compared to the set of faces in memory. The closest match to the individuals face is found and the individuals identity is declared. A good system should be able to recognize a persons face under different lighting conditions and be robust to changes in the persons pose and appearance. A good system should also require minimum computation and memory so that it can be expanded to recognize millions of faces.

METHOD-1: The first method that comes to mind could be as follows: Take 5 pictures of every persons face under different lighting and pose. If every picture is 112 by 92 pixels, we can vectorize the image and save it as a $112 \times 92 = 10304$ array in memory. So for every person we have 5 arrays of 10304 numbers. If we want to recognize faces among a population of 40 people, then we need to save $40 \times 5 = 200$ arrays of 10304 numbers. We can save all of this as a matrix with 10304 rows and 200 columns for a total of $200 \times 10304 = 2060800$ numbers in memory. We call this our traninig data. Every time the system needs to recognize a person, the system takes a 112 by 92 pixel picture of the persons face and this image is vectorized to form an 10304 array (or vector). We call this a test data point. The distance between this test vector and a train vector in memory is computed as

$$di = ||I - I_i||_2 \tag{1}$$

where I is the image vector of the unknown face, I_i is the ith trin image vector in memory and d_i is the distance between the unknown vector I and the known vector I_i. The smallest d_i is found and the system declares the unkown faces identity to be the same as that of I_i with smallest d_i. If a new person is added to the population, the system needs to add another 5 pictures to its memory for a total of $5 \times 10304 = 51520$ new numbers in memory.

METHOD-2: In this method rather than store the entire training dataset of faces in memory, we learn the orthonormal basis for our set of faces and only store these limited number of basis vectors in memory. If A is the train data matrix with 10304 rows and 200 columns, we find the SVD decomposition of this matrix to find U, S and V matrixes as described in class. We showed in class that the columns of U form an orthonormal basis for the columns (faces) of A. We also showed that the diagonal elements of S are the singular values and that throwing away the smaller singular values will not greatly affect our representation of A but significantly reduce noise. If we decide to keep only the first 10 singular values, then we only need to store a 10304 by 10 matrix in memory. Each face in the train set is projected onto these 10 basis vectors and is represented as 10 numbers. So we also store a small 10 by 200 matrix in memory. This is our new train set. Every time the system needs to recognize a person, the system takes a 112 by 92 pixel picture of the persons face and this image is vectorized to form an 10304 array (or vector). We call this a test data point. This test vector is then projected onto the 10 basis vectors and is now represented as a 10 dimensional array. The distance between this 10 dimensional test vector and a vector in the train set is found as

$$di = ||P - P_i||_2 \tag{2}$$

where P is the projected image vector of the unknown face, P_i is the ith projected train image vector in memory and d_i is the distance between the unknown vector P and the known vector P_i. The smallest d_i is found and the system declares the unkown faces identity to be the same as that of P_i with smallest d_i. If a new person is added to the papulation, the system needs to add another 5 pictures to its memory which are projected onto the 10 basis vectors for a total of $5 \times 10 = 50$ new numbers in memory.

2 General Assignment

In this project you are to do the following. For the given set of face images, you are to build a face recognition system using both METHOD-1 and METHOD-2. You are to then test both systems and compare their performance.

3 Matlab details

STEP 1. Downlaod and UNZIP the face dataset. You should have 40 folders and each folder should have 10 .png images in it. Each folder contains 10 images of a persons face under different lighting and pose. display the first image in the first folder using the Matlab command below and include its image in your report:

```
I=imread(['C:\orl_faces\s' int2str(1) '\' int2str(1) '.pgm' ]);
I=im2double(I);
figure, imshow(I);
```

How many rows (r) and columns (c) does the matrix I have?(Hint: use size.m function in Matlab). Now vectorize the image using the code below:

```
I=reshape(I,r*c,1);
```

We will use the first 5 images in each folder as the train set and the last 5 images in each folder as the test set. So you should vectorize $40 \times 5 = 200$ images and save them as a 10304 by 200 train matrix.

STEP 2. Find the mean of the training vectors and subtract it from the traning data to get the normalized trainig data matrix D. Hint: use the mean.m Matlab function. (We subtract the mean to center the data and so that our SVD method will be equivalent to the well known PCA method for face recognition.)

STEP 3. Make the test data matrix by including the last 5 images in each folder in vectorized form in a 10304 by 200 train matrix. Normalize the test matrix.

STEP 4. Implement METHOD-1 and find how many faces in the test data set are recognized correctly and how many are not recognized correctly. Report your results.

STEP 5. Find the SVD of your normalized training data. Use the Matlab function below:

```
[U S V]=svd(D);
```

STEP 6. Plot the singular values. What is the largest singular value? What is the smallest singular value? What do you think is a good point for truncating the SVD?

STEP 7. Use the code below to see the first 10 basis vectors as images.

```
imshow(  reshape(U(:,i),r,c),[]  )
```

These are called eigen-faces. Do they look like a good representation of the faces in the dataset? Include these 10 eigen-faces in your report. Look at the last 10 basis vectors. Do they look like they could be discarded? Include them in your report.

STEP 8. Truncating the SVD at the point that you think is best by looking at the plot of singular values. Project the data matrix onto the truncated basis vectors using the code below

```
UKEEP=U(:,1:NK);
PROJD=(D'*UKEEP)';
```

where NK is the number of basis vectors we want to keep and D is the normalized train data matrix.

STEP 9. Project the normalized test data matrix using the code below:

```
PROJT=(T'*UKEEP)';
```

where T is the notmalized test data matrix.

STEP 10. Find the 2-norm difference between the vectors in PROJT and PROJD as described in METHOD-2. How many faces are not recognized correctly (recognition errors)?

STEP 11. If your results are not as good as METHOD-1 then you can try to find the best number of basis vectors (NK) by trying all of them. repeat steps 5,8 9 and 10 for all NK=1,2,....,200. Plot the number of recognition errors versus NK. What is the best NK for your test data set? Do you get better results with this NK than METHOD-1? How much memory do you save with this NK compared to METHOD-1?

4 What to turn in

You should turn in a CD which includes (1) A report on your project (2) a Matlab .m file of your program. Your report should include your results for each STEP. You should clearly divide your report into STEPs. Failing to do so will get you a zero grade! You Matlab program should have a comment for each line and each segment of code explaining what that line or segment of code does. If you fail to comment your code, you will get a zero grade! I should be able to run your code directly from the CD and see the plots and images it makes. If your code has errors when I run it or does not work when I run it from the CD, you will get a zero grade!

5 CHEATING :-(

Copying a portion of someones report or Matlab code is considered cheating. If even a portion of your report or Matlab code is similar to someone else you will both get a zero on the project, be introduced to the university disciplinary committee (Komite Enzebaty) and possibly fail the course. SO DO NOT EVEN SHOW YOUR CODE OR REPORT TO ANYONE ELSE!