

Assignment 2

Computer Vision (CSI4116-01)

Spring, 2019

Due date : 1st May, 23:55

Overview

[Recognize faces using Eigenfaces]

You are given two face datasets, “train” and “test”.

You have to run PCA to recognize the faces in the “test”.

The assignment consists of 3 steps.

- 1) Select the number of principal components you use for this data.
- 2) Reconstruct all images in the training set using the number of PCs.
- 3) Recognize images in the test dataset using a simple nearest neighbor algorithm.

Dataset

Two datasets are provided for this problem

- `faces_training` : a face database of 39 people
- `faces_test` : 5 faces to be recognized

Filename : `faceXX.pgm` / `testXX.pgm`

- XX indicates the identity number of the image.



data example

Step1: Principal Components Selection

1. Using SVD algorithm, compute principal components of the 'train' dataset.
2. Given a percentage of the variance as an input, select the number of principal components you use for this data.
 - e.g., given 0.80, the smallest d such that $\frac{\sum_{i=1}^d \lambda_i}{\sum_i \lambda_i} \geq 80\%$,

Step1: Principal Components Selection

[Implementation Detail]

1. The percentage of the variance is given as a command line option
2. Output the selected dimension to the 'output.txt' file.

Input Example

```
$ python 2019123456.py 0.85
```

The percentage of the variance is the float value (0,1)
You don't have to consider the case coming from inappropriate inputs

Output Example

```
##### STEP 1 #####  
Input Percentage: 0.95  
Selected Dimension: 27
```

The percentage given as an input
Selected PCs number

Step2: Image Reconstruction

1. Reconstruct all images in the training set using the selected number of principal components(step1). Then save the reconstructed images.
2. Compute the reconstruction error between original images and reconstructed images.

Reconstruction error : Mean Squared Error

$$\frac{1}{n} \sum_{i=1}^n (y_i - t_i)^2$$

Step2: Image Reconstruction

[Implementation Detail]

1. Save the reconstructed image as a format of '2019XXXXXX/faceXX.pgm' (same with the original file).
2. Output the reconstruction loss of each train image to the 'output.txt' file.

Output Sample (face02.pgm)



Original



Reconstructed

Reconstructed image size is equal to the original image size

Output Example

```
##### STEP 2 #####
Reconstruction error
average : 174.9085
01: 71.4251
02: 197.6192
03: 194.7130
04: 134.5422
05: 210.5284
06: 225.0911
07: 171.8459
08: 198.8762
09: 201.7597
10: 217.5175
( omitted.. )
```

Step3: Face Recognition

1. Recognize images in the **test** dataset using a simple nearest neighbor algorithm. We'll use 'l2 distance'.

* Make use of **the projection results** using the selected number of PCs(step1) when you compare two images.

l2 distance(Euclidean distance) :

$$\begin{aligned}d(\mathbf{p}, \mathbf{q}) &= d(\mathbf{q}, \mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2} \\&= \sqrt{\sum_{i=1}^n (q_i - p_i)^2}.\end{aligned}$$

2. Find the closest identity of each image in the **test dataset** among the identities in the **train dataset**.

Step3: Face Recognition

[Implementation Detail]

1. Output the identity number for each image in the test dataset to the 'output.txt' file.

Output Example

```
##### STEP 3 #####  
test01.pgm ==> face25.pgm  
test02.pgm ==> face07.pgm  
test03.pgm ==> face19.pgm  
test04.pgm ==> face23.pgm  
test05.pgm ==> face33.pgm
```

[test image name] ➔ [corresponding train image name]

We have to find **the most nearest face** among data in train datasets. You don't have to consider the case where there are several faces which have same l2 distance.

Submission

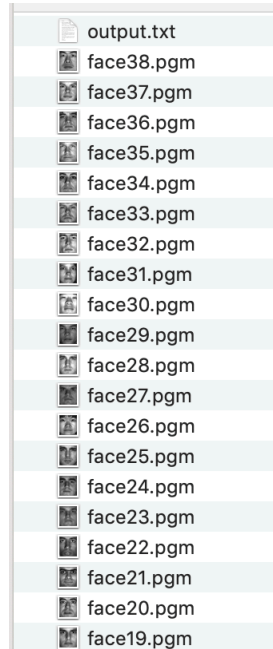
- Deliverables : 2019XXXXXX.py (Your student ID)
 - You don't have to include your result files.
 - But your python code must output the required files in the '2019XXXXXX' directory after running your code.

Directions

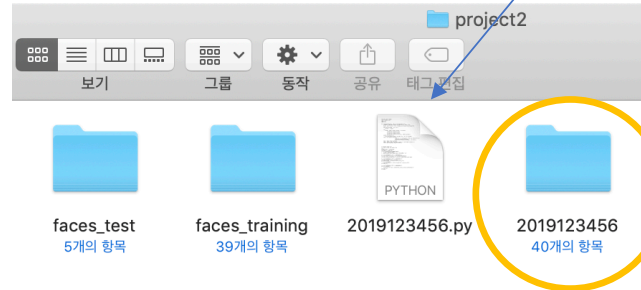
- You cannot use other external libraries except
 - `numpy`
 - `opencv` for image reading and writing
- After running your code, your code must output the 'output.txt' file and 'faceXX.pgm' image files in the '2019XXXXXX'(your student ID) directory.

```
STUDNET_CODE = '2019123456'  
FILE_NAME = 'output.txt'  
if not os.path.exists(STUDNET_CODE) :  
    os.mkdir(STUDNET_CODE)  
f = open(os.path.join(STUDNET_CODE, FILE_NAME), 'w')
```

Output



...



your code is here!

your output!

When you run your python code, 2019XXXXXX(your student ID) directory is made and in the directory, 'output.txt', 'faceXX.pgm' (1~39) must be included.

The overall format of 'output.txt' is on the right side.

The output values change according to the input percentage values. So just use the right capture for formatting reference.

We will grade scores based on the 'output.txt' file and images. So comply with the given output format.

```
output.txt
##### STEP 1 #####
Input Percentage: 0.95
Selected Dimension: 27

##### STEP 2 #####
Reconstruction error
average : 43.5323
01: 8.4699
02: 126.6539
03: 73.1771
04: 41.3908
05: 25.0779
06: 92.6671
07: 73.9484
08: 54.9646
09: 80.6603
10: 66.0986
11: 114.0821
12: 19.7372
13: 82.9884
14: 109.5454
15: 108.9769
16: 34.2934
17: 4.6159
18: 69.4423
19: 75.5550
20: 55.8048
21: 23.6670
22: 29.1555
23: 21.7715
24: 110.9801
25: 22.8054
26: 76.2487
27: 30.4852
28: 114.2096
29: 81.8791
30: 110.1715
31: 4.8081
32: 40.4579
33: 26.7145
34: 77.4484
35: 20.2452
36: 93.9190
37: 26.2279
38: 21.7750
39: 43.5323

##### STEP 3 #####
test01.pgm ==> face03.pgm
test02.pgm ==> face07.pgm
test03.pgm ==> face19.pgm
test04.pgm ==> face23.pgm
test05.pgm ==> face33.pgm
```

Grading Environment & Directions

- Language : [Python](#)
- We grade your score in Linux(Ubuntu 16.04)
- Python3 ($\geq 3.5.2$)
- This is an individual project
- You should follow the input/output format
- Never copy code
- You will get 0 points if you cheat
- If you have a question, upload on the Q&A board of YSCEC.

Grading Policy

- Total 100 pts
- Details
 - Implementation
 - step1 : 30 pts
 - step2 : 30 pts (output.txt 15 pts + output images 15 pts)
 - step3 : 30 pts
 - get input & generate output(format) : 10 pts
 - We will grade implementation scores based on the 'output.txt' file and images.
So comply with the given output format.

Due Date

- 1st May, 23:55
- We'll receive late submission for 2 weeks.
- However, 20% reduction of total score is applied.