王　兆東

マルチメディア信号解析

2020年6月19日

Report of PCA with KNN

In this report I will discuss how I design my program to complete the assigned task, it include how to read set of images by program and use PCA to shrink the feature, and check the accuracy of classification by using KNN.

1. Environment Introduction

· Programming language: Python3.6

· library would be used: Numpy

· text book：jupyter

2. Implementation

To read a set of images, a simple method I come up with is establishing a list, which would contains the paths of images. Traversing the list then we can get every images' vector in a huge matrix.

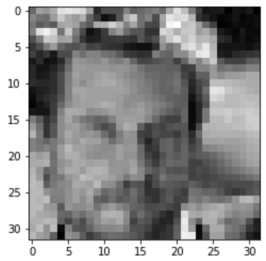
Image matrix is a 1024 one dimension vector which is reshaped from 32x32 image.

Face Cube contains 5993 images of face.

Nonface Cube contains 6001 images of nonface object.

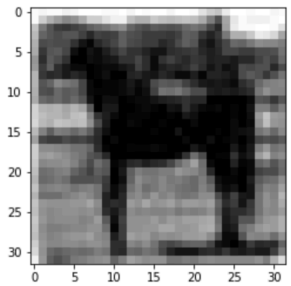
```
import matplotlib.pyplot as plt
def face_show(index):
    img = face_cube[index,:].reshape(32,32)
    plt.imshow(img,cmap = 'gray')

def nonface_show(index):
    img = nonface_cube[index,:].reshape(32,32)
    plt.imshow(img,cmap = 'gray')
```

```
face_show(300)
```



```
nonface_cube = readimg(path2+nonface_list[0])
for line in nonface_list:
    tmp_vector = readimg(path2+line)
    nonface_cube = np.append(nonface_cube, tmp_vector, axis = 0)
```

```
nonface_show(200)
```



The Piece the two matrixes together to form the dataset. From now we can run our PCA method to reduce the dimensions.

1.decentralize the dataset from every features

2. transpose and calculate the covariance matrix.

3. Calculate the eigenvalue and eigenvector from covariance matrix.

4. Sort the eigenvalue with eigenvector from max to min.

5. Compute the cumsum number and get the 90% cumsum of eigenvalue (here is 72 eigenvalues be choosed like the screenshot below).

6. piece the eigenvectors together to become the transfer matrix.

7. Dot the transfer matrix and dataset to complete dimension reducing(PCA job) .

```
In [59]: cond = (eig_val/eig_val.sum()).cumsum()

In [28]: cond = cond >=0.90

In [61]: cond[:100]

Out[61]: array([0.29826921, 0.40190871, 0.47637474, 0.53831225, 0.58882512,
                0.6210394 , 0.6492939 , 0.67078777, 0.68720066, 0.70186039,
                0.71379577, 0.72473342, 0.73459641, 0.74399218, 0.75208562,
                0.75969042, 0.76698008, 0.77382196, 0.77965965, 0.78540512,
                0.79084195, 0.79578197, 0.80045267, 0.80470558, 0.80884292,
                0.81263952, 0.81636292, 0.81992623, 0.82341233, 0.82671967,
                0.82996753, 0.83296561, 0.83591704, 0.83866334, 0.84132136,
                0.84388463, 0.84637886, 0.8487997 , 0.8510966 , 0.85331245,
                0.85542724, 0.85751347, 0.85951795, 0.86144468, 0.86332941,
                0.86514906, 0.86688009, 0.86857243, 0.87022621, 0.87185539,
                0.87345662, 0.87501807, 0.87652676, 0.87798417, 0.87943195,
                0.88081028, 0.88216126, 0.88349076, 0.88480788, 0.88607299,
                0.88731478, 0.88852315, 0.88969079, 0.89083843, 0.89194874,
                0.89304295, 0.89413592, 0.89520678, 0.89627097, 0.89730745,
                0.89833822, 0.89933302, 0.9002925 , 0.90123433, 0.90216207,
                0.90307625, 0.90397349, 0.90486079, 0.90573111, 0.90658843,
                0.90742712, 0.90825602, 0.9090704 , 0.90988034, 0.91068567,
                0.91147833, 0.91225943, 0.9130147 , 0.91375608, 0.91446441,
                0.91516949, 0.91586898, 0.91655522, 0.91723974, 0.91791087,
                0.91857225, 0.91922807, 0.91987125, 0.92051031, 0.92114283])
```

For comparing the accuracy between PCA method and normal method, we shuffle the whole images and randomly pick 1000 images as test set,

The result for PCA ’s accuracy is 0.942, and the raw vector’s accuracy is 0.927

```
In [48]: KNN(PCA_pairs, PCATest_set)
-16
label successfully predicted! 934, 991
12
label successfully predicted! 935, 992
-18
label successfully predicted! 936, 993
-18
label successfully predicted! 937, 994
-10
label successfully predicted! 938, 995
-20
label successfully predicted! 939, 996
-20
label successfully predicted! 940, 997
20
label successfully predicted! 941, 998
-14
label successfully predicted! 942, 999
the accurancy of test data is 0.942
```

```
In [58]: KNN(Normal_pairs, NormalTest_set)
-16
label successfully predicted! 919, 991
20
label successfully predicted! 920, 992
20
label successfully predicted! 921, 993
-20
label successfully predicted! 922, 994
-12
label successfully predicted! 923, 995
20
label successfully predicted! 924, 996
20
label successfully predicted! 925, 997
20
label successfully predicted! 926, 998
20
label successfully predicted! 927, 999
the accurancy of test data is 0.927
```

The conclusion is PCA method can reduce the dimension and keep the principal components, Which is a good method for avoiding the explosion of dimension.