Assignment 2

ID: R06945011, NAME: Shih-Jou Chung(鍾詩柔), Department: BEBI

1. ROC curve

a.

A: TN, B:TN, C:FP, D:TP,E:FN

在 x 為正值時會沿著 $f_1(x)$行走，而 T 的沿線切開後也正確的被劃分為正值(TP)的就是 D 領域；而 E 領域亦為正值，卻被劃分成負值(FN)，為錯的負值；而 x 為負值時會沿著 $f_2(x)$行走，而 T 的沿線切開後也正確的被劃分為負值(TN)的就是 A 和 B 領域；而 C 領域亦為負值，卻被劃分成正值(FP)，為錯的正值。

b.

ROC curve 是由 TP rate、FP rate 所形成的曲線圖，由這兩種 parameter 就可以知道當曲線越接近 a 領域則代表在一開始 FP rate 很低就能得到越高的 TP rate，則可以說 classify 很好；而在 b 線條則為 TP 和 FP 在任何情況下都得到一樣的比例，說明 accuracy = 0.5;而在 c 領域則代表 classify 不好且 accuracy 非常差。

當 accuracy = 0.5，classify 有錯的成分跟對的成分依樣多，可以說成是在(a)小題的 T 剖線為兩曲線交集的點。

2.

Code:

Main:

path_train 是 training data 所在的路徑，拿來 train 的 path。

path_test 是 testing data 所在的路徑，拿來 test 的 path。

有兩個 path_test 是因為第一個是 person8_image6，第二個是 120 張 testing data。

Func:

imagelist_meanface 是會畫出 meanface 和 data load 近來順序的 function。(如果不需要畫出 meanface 和順序可註解掉)

difference_and_eigenface 是找出 eigenvect 和 eigenface 的函數。(如果不需要畫出 eigenface 可註解掉)

compare 是畫出 reconstruct image 和 MSE

利用 folder 讀取圖片，所以第一個 class 是第 10 個人，第二的 class 是第 11 個人，第三個 class 是第 12 個人，直到第二十個 class 為第 1 個人(10-19,1 ,20-29,2 ,30-39,3 ,40-49 ,4-9)

PCA

Meanface: Eigenface: Reconstruct image: <5,50,150,279>

Meanface 是將全部 280 張 images(280*1*2576)相加之後除以 280 後得到的影像。

Eigenface 是將 training data – meanface 得到的 difference(280*2576) 做 covariance(280*280)後，
得到 eigenvect (280*280)，將 eigenvalue 從大到小得到 eigenIndex，
將 eigenVect 依照 value 的 index 在乘上原本的 difference (1*2576)得到的影像(280*2576)。

Reconstruct image 是利用$p = V^T(x - m)$ 得到 p (我命名為 weight ，x 為 person_8_image_6, m 為 meanface)，再利用$\sum_{i=0}^{n} p_i v_i + meanface$, n=<5,50,150,279>

MSE:[[693.70217443]] eigenvetor:5

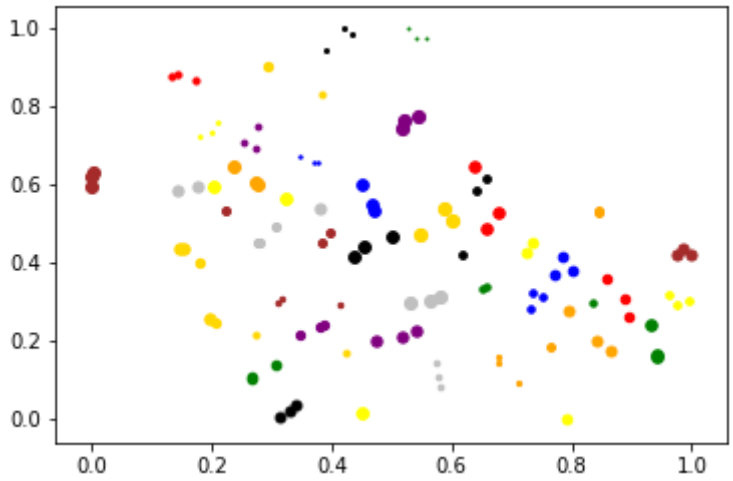
MSE:[[119.20025848]] eigenvetor:50

MSE:[[75.56487217]] eigenvetor:100

MSE:[[8.44333996e-27]] eigenvetor:279

我有思考過我的 Eigenface 跟別人不一樣的問題，因為方法都是一樣，所以不太對甚麼跟別人做出來的圖像差距有點大，但是特徵都有顯現出來。

t-SNE: 使用 10 個 colors，但同樣顏色不同大小的圓圈為不同分群
利用 eigenVect (2576*2576)投影到 testing data (120*2576)得到的矩陣(120*2576)取前 100 維 (120*100)之後利用 t-SNE 畫出來。



在經過 PCA 降維成 100 後(eigenvect project on test images)，120 張 testing data 的 classify 分布區塊還是不太明顯，只有同一張臉的分布會聚集在一起

LDA

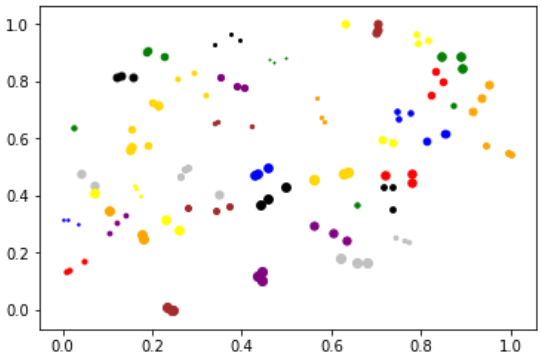
Code:

Sb: fisherface 的更改可在此 function 裡面更改

Fisherface:

Fisherface: 利用講義上的公式得到 Sb 和 Sw 得 eigenVect(2576*2576)，再利用 PCA 得到的 eigenface 和 Sw 和 Sb 矩陣相乘得到新的 Sww 和 Sbb，拿 Sww、Sbb 的 eigenvect 和 eigenface 得到 fisherface。

t-SNE: 使用 10 個 colors，但同樣顏色不同大小的圓圈為不同分群，與上面 PCA 相同，換成 LDA 降維。



在經過 LDA 降維成 30 後(eigenvect project on test images)，120 張 testing data 的 classify 分布區塊較 PCA 明顯一些且某些特徵類似的會聚集在一起

3.

Code:

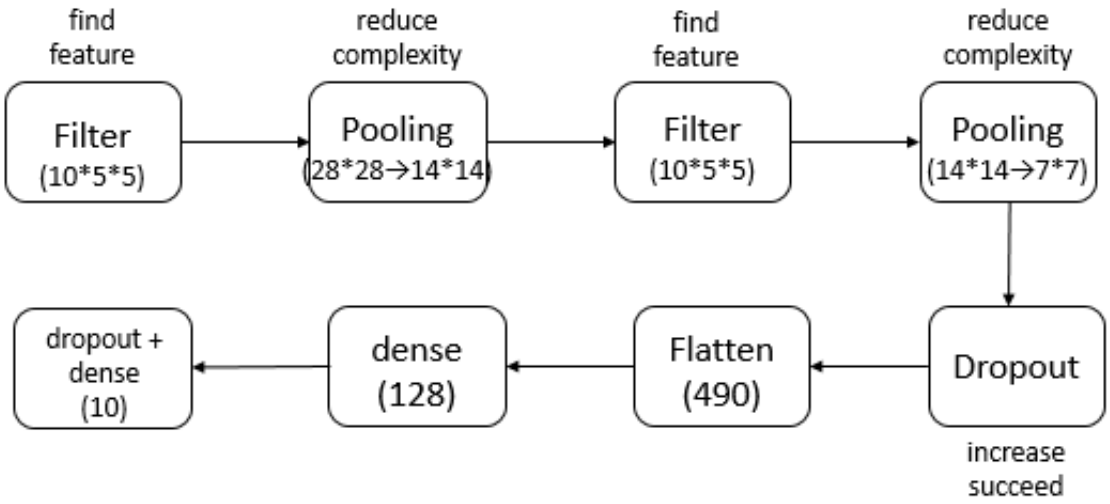
Train.py

load_image: 如果 flag =0 則 image list 會是拿來 train，如果是其他則會拿來當 valid

最後會將 model 存成 CNN_model.h5 的檔案。

Test.py 會將.h5 的 model load 進來後，將 x_test、y_test (依照助教給的為多少)，prediction 為 model 算出來的 label，算完之後會變成'scores.csv'

流程圖:

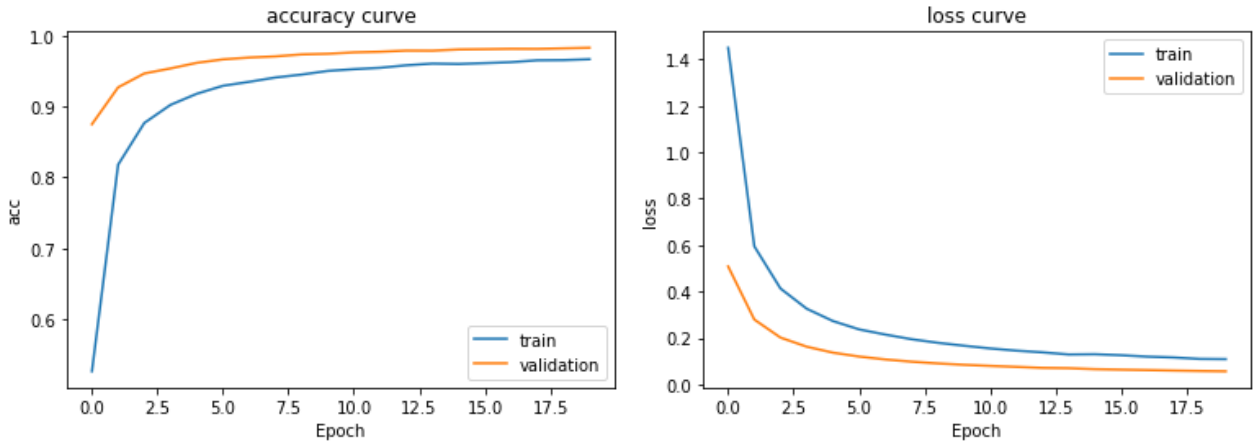


Model:

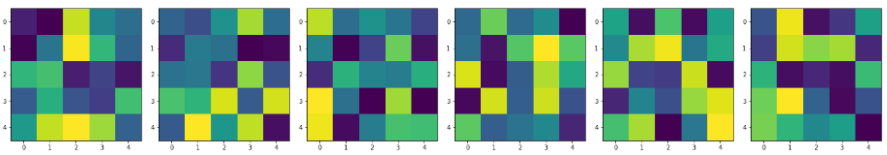
Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 28, 28, 10)	260
max_pooling2d_1 (MaxPooling2	(None, 14, 14, 10)	0
conv2d_2 (Conv2D)	(None, 14, 14, 10)	2510
max_pooling2d_2 (MaxPooling2	(None, 7, 7, 10)	0
dropout_1 (Dropout)	(None, 7, 7, 10)	0
flatten_1 (Flatten)	(None, 490)	0
dense_1 (Dense)	(None, 128)	62848
dropout_2 (Dropout)	(None, 128)	0
dense_2 (Dense)	(None, 10)	1290

Total params: 66,908
Trainable params: 66,908
Non-trainable params: 0

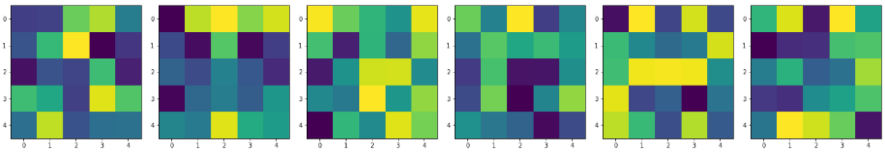
Result of training curve:



Filter:

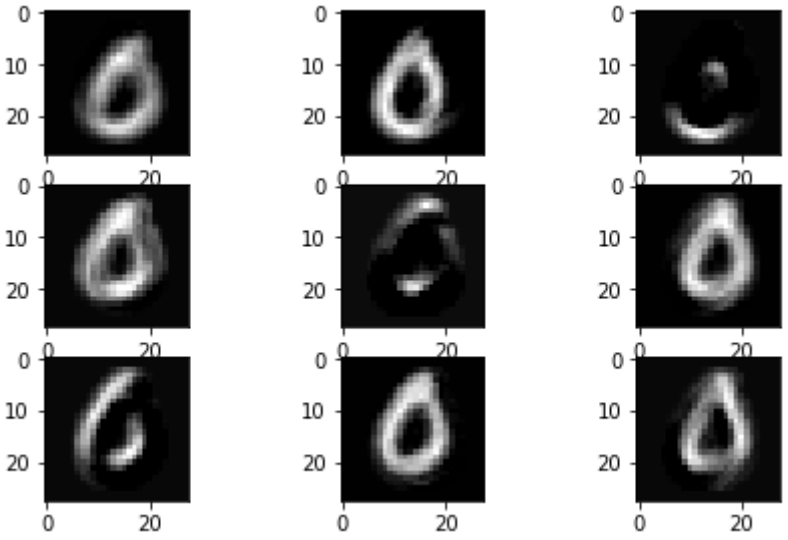


1st conv_layer [0]

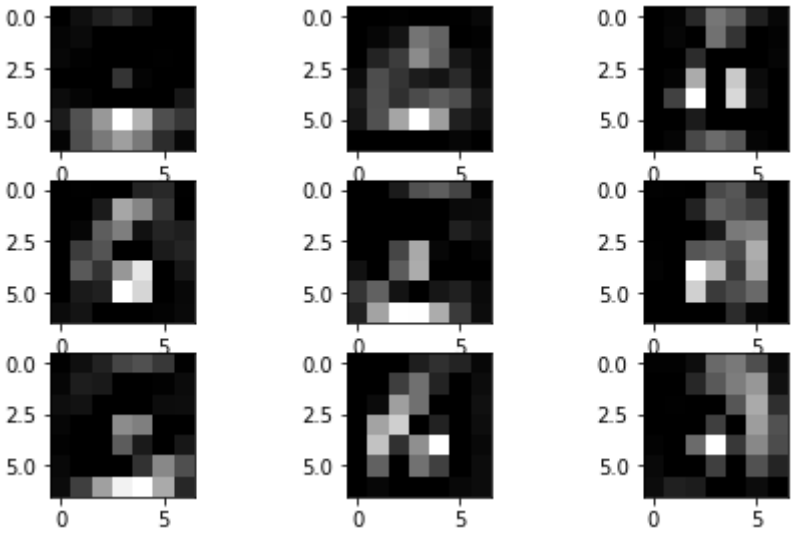


2nd conv_layer[3]

Extracted in different layers: 利用 class0 的第十個 training data 來 construct 經過不同 filter 之後的 image



1st conv_layer [0]



2nd conv_layer[3]