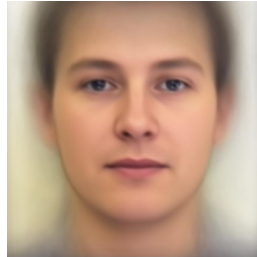


Homework 4 Report Unsupervised Learning

學號：b05902013 系級：資工二 姓名：吳宗翰

Part 1 -- PCA of colored faces

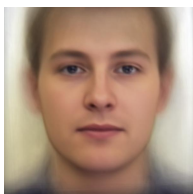
1. (.5%) 請畫出所有臉的平均。



2. (.5%) 請畫出前四個 Eigenfaces，也就是對應到前四大 Eigenvalues 的 Eigenvectors。

Eigface	1	2	3	4
				

3. (.5%) 請從數據集中挑出任意四個圖片，並用前四大 Eigenfaces 進行 reconstruction，並畫出結果。

Reconstruct	0.jpg	1.jpg	2.jpg	3.jpg
				

4. (.5%) 請寫出前四大 Eigenfaces 各自所佔的比重，請用百分比表示並四捨五入到小數點後一位。

實驗結果如下：(4.1%, 3.0%, 2.4%, 2.2%)

Part 2 -- Image clustering

1. (.5%) 請比較至少兩種不同的 feature extraction 及其結果。(不同的降維方法或不同的 cluster 方法都可以算是不同的方法)
- 方法一：Convolution x Dense Autoencoder + KMeans

```
# Autoencoder
Input (784)
Reshape(28, 28, 1)
Conv(16, 5, 5) + selu
Conv(16, 3, 3) + selu
Flatten()
Dense(50) -> encode
Dense(200)
Output(784)

# Kmeans
n_components=2, random_state = 7122 (others setting= default)
```

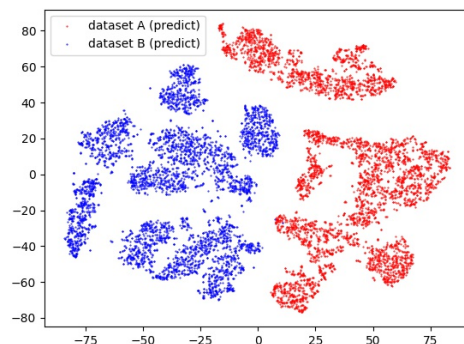
結果如下：Public Score : 0.99575 / Private Score: 0.99567

- 方法二：人工feature extraction

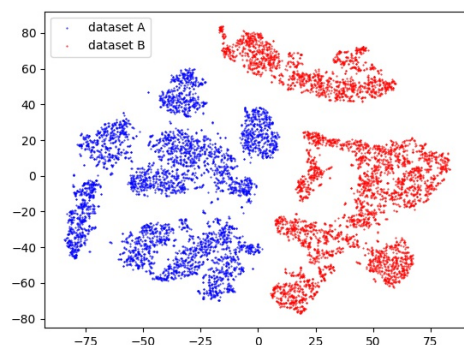
在人工的方法中，由於我認為MNIST和Fashion MNIST一定差異很大(MNIST全黑的比例一定差很多)，因此就直接比較相似性來cluster 判斷的準則是：當兩張(28 × 28)的圖片，全0的差異超過50，我就判斷他是不同的dataset；反之，我則視他們為相同的dataset

結果如下：Public Score : 0.93609 / Private Score: 0.93655

2. (.5%) 預測 visualization.npy 中的 label，在二維平面上視覺化 label 的分佈。



3. (.5%) visualization.npy 中前 5000 個 images 跟後 5000 個 images 來自不同 dataset。請根據這個資訊，在二維平面上視覺化 label 的分佈，接著比較和自己預測的 label 之間有何不同。



上圖是實際上的答案，有實際印出來發現錯了兩筆(在紅色中間邊右的地方)，不過其實做 autoencoder 的效果蠻好的，在很大比例的 dataset 中都答對了，還算滿意的結果

備註：

- 上面兩張照片都是先用自己 `hw4_train.py` 的 autoencoder (模型上面提過了)，降到 50 維，接著

再用TSNE(pca init)降到2維去visualize，另外第3小題自己預測的label就是上面的KMeans來cluster

- 另外由於方便起見，我在 `hw4_train.py` 中已經預先把label存成 `label.npy` 的檔案，在reproduce的時候直接load進來以加速助教執行評測的時間。(如果助教想要看實作方法的話，可以看 `hw4_train.py`，或者是自己把 `model/auto.ckpt` load近來都可以)

Part 3 -- Ensemble learning

1. (1.5%) 請在hw3的task上實作ensemble learning，請比較其與未使用ensemble method的模型在 public/private score 的表現並詳細說明你實作的方法。（所有跟ensemble learning有關的方法都可以，不需要像hw3的要求硬塞到同一個model中）

- 實作ensemble的方法 (做在hw3的task上)

```
### model 1 -> 6 CNN + 2 Dense ###
Input (48, 48, 1)
Conv(16, 3, 3) + BN + LeakyRelu
Conv(32, 3, 3) + Gaussion noise(0.1) + BN + LeakyRelu
Conv(64, 3, 3) + BN + LeakyRelu + Maxpooling(2, 2) + Dropout(0.1)
Conv(128, 3, 3) + BN + LeakyRelu + Maxpooling(2, 2) + Dropout(0.2)
Conv(256, 3, 3) + BN + LeakyRelu + Maxpooling(2, 2) + Dropout(0.2)
Conv(512, 3, 3) + BN + LeakyRelu + Maxpooling(2, 2) + Dropout(0.2)
Flatten()
Dense(512) + BN + LeakyRelu + Dropout(0.5)
Dense(256) + LeakyRelu
Output (7)

### model 2 ###
修改model 1, 拿掉Gaussion noise

### model 3 ###
修改model 1, Dense第二層改成512個neuron

### model 4 ###
修改model 1, activation function全部換成selu，拿掉BN

### Ensemble ###
每個model分別predict一次，average起來再做argmax即得到答案
```

- model 解釋
 - 使用的ensemble method就只是最簡單的average voting模型而已
 - 由於單一模型都會有盲點，因此在每個model的正確率都高於一定的狀況下採取投票會比單一模型來的好
- code 解釋
 - 上傳的code就是我在hw3作出的模型，ensemble的部分寫在 `ensemble.py` (當初很努力的把5個model壓成一個)
 - `kr_model.py` 是keras model、`ensemble_test.py` 則是附上testing部分的程式碼。傳

上去的 `mean.npy`, `dev.npy` 則是testing要使用的。

- 不過沒有辦法重新reproduce他，因為model並沒有上傳在這次作業的資料夾中，如果需要reproduce的話可以到hw3的資料夾下進行～
- 比較有無使用ensemble method的public / private score

Model	Public Score	Private Score
Model 1	0.70381	0.67929
Model 2	0.69378	0.68514
Model 3	0.70353	0.67929
Model 4	0.67483	0.65059
Ensemble	0.71858	0.70855

p.s. 在這裡我是ensemble 2個model 1的predict結果(分開train兩次)和model 2, 3, 4各一個predict結果