學號：r05229014 系級：大氣碩二 姓名：鄒適文

A. PCA of colored faces

collaborate(:r05229016 羅章碩)

因為在筆電上跑不動原圖，我把圖全部都壓縮成100*100*3的大小

A.1. (.5%) 請畫出所有臉的平均。

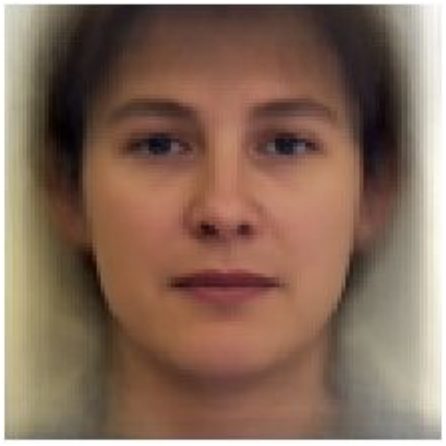
A.2. (.5%) 請畫出前四個 Eigenfaces，也就是對應到前四大 Eigenvalues 的 Eigenvectors。

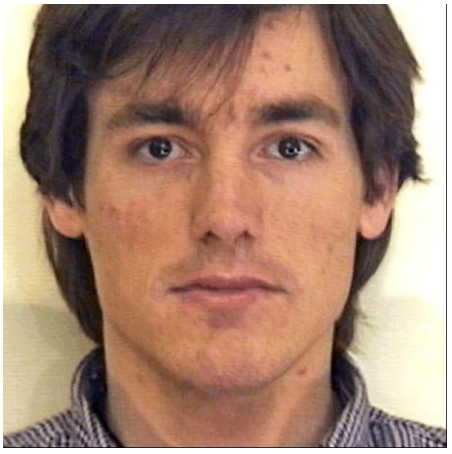
1 *2*

3 *4*

A.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。

reconstruction(4 Eigenfaces) *原圖*





A.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。

1. 4.2%
2. 3.0%
3. 2.4%
4. 2.2%

B. Visualization of Chinese word embedding

collaborate(:r05229016 羅章碩)

B.1. (.5%) 請說明你用哪一個 word2vec 套件，並針對你有調整的參數說明那個參數的意義。

我用gensim做word embedding,並用TSNE做降維。

```
In [139]: model = Word2Vec(train_lines,size=512 , min_count=5, workers=4)

In [140]: len(model.wv.vocab)
Out[140]: 31444

In [141]: model.wv.similarity('爸','爸爸')
Out[141]: 0.79937390623838644

In [142]: model.wv.similarity('媽','媽媽')
Out[142]: 0.56636574903789771

In [143]: model.wv.similarity('還有','還是')
Out[143]: 0.25243919528277053
```

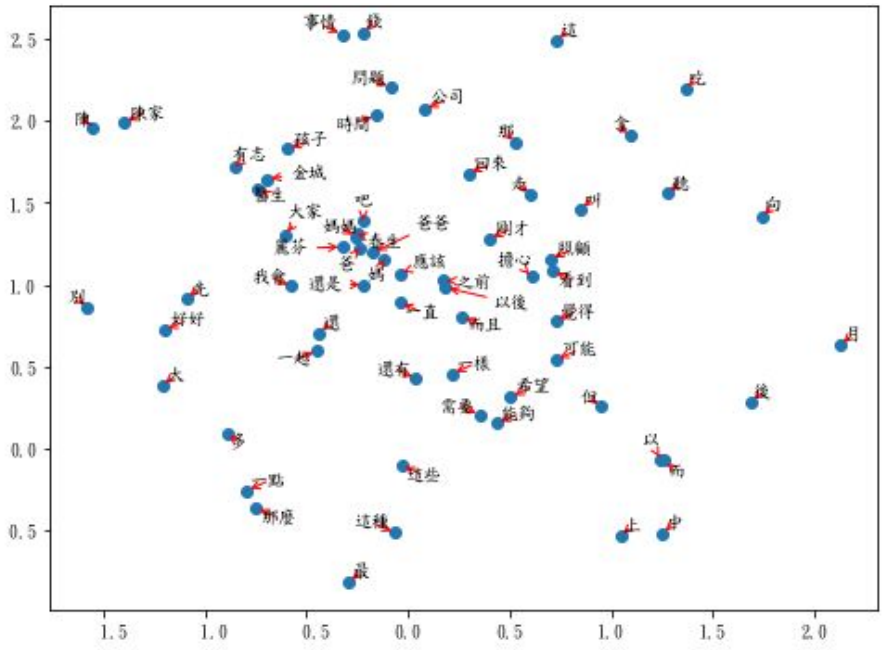
附上我的部分程式碼，其中 size是我embedding開的大小，min_count是最小看到次數，workers是cpu的使用數目。

而可以看到這個case在比較similarity是蠻有意義的。

X_tsne = TSNE(learning_rate=10, n_components=2, perplexity=40)

而另外一個有調整的參數是learning_rate跟困惑度(**perplexity**)，困惑度代表在降維過程中一個點與全部資料點的平衡程度，高的困惑度一個點會把鄰近更比較多的點拉近，而比較低的困惑度就會把兩著距離拉大。而這邊我用40

B.2. (.5%) 請在 Report 上放上你 visualization 的結果。



B.3. (.5%) 請討論你從 visualization 的結果觀察到什麼。

其中一眼就可以看到，陳、陳家兩字的位置非常接近，顯然這分類八成是有意義的，而中間區域很多爸、爸爸、媽、媽媽都擠在一塊，這些詞對這次的分類來說都很接近，但除了這些發現(可能比較相似的東西有被分在一塊)，其實沒有觀察到人類比較好理解的分群，顯然是可以再做得更好的。

C. Image clustering

collaborate(:r05229016 羅章碩)

C.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)

我嘗試了DNN與CNN的auto-encoder

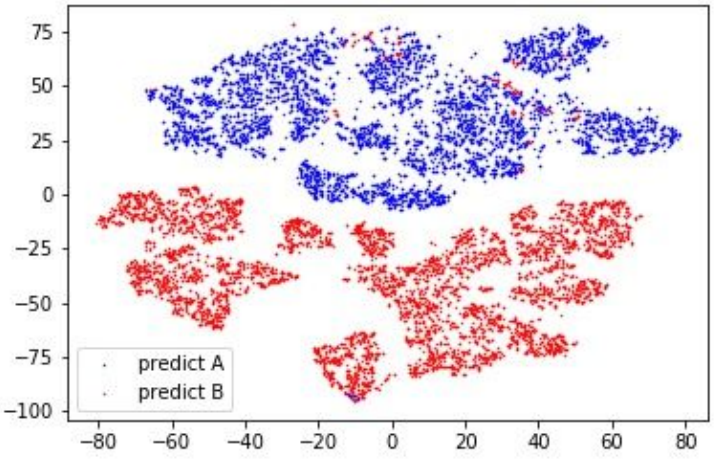
以下兩張圖為dnn與cnn之模式架構

```
input_img = Input(shape=(784,))

encoded = Dense(128, activation='relu')(input_img)
encoded = Dense(64, activation='relu')(encoded)
encoded = Dense(32, activation='relu')(encoded)

decoded = Dense(64, activation='relu')(encoded)
decoded = Dense(128, activation='relu')(decoded)
decoded = Dense(784, activation='relu')(decoded)

# build encoder
encoder = Model(input=input_img, output=encoded)
```

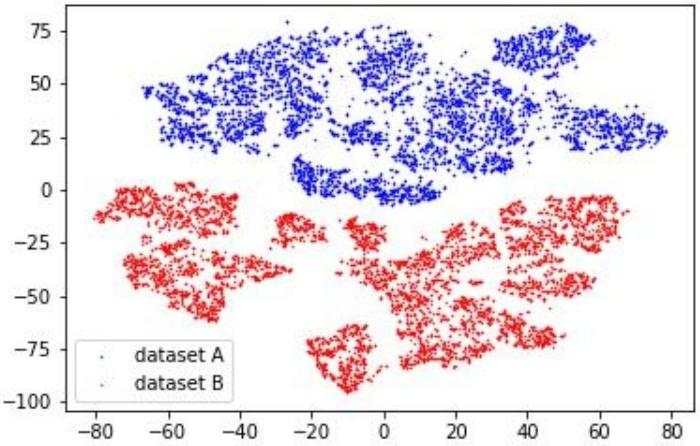
```
input_img = Input(shape=(28, 28, 1))  # adapt this if using `channels_first` image data format

x = Conv2D(16, (3, 3), activation='relu', padding='same')(input_img)
x = MaxPooling2D((2, 2), padding='same')(x)
x = Conv2D(8, (3, 3), activation='relu', padding='same')(x)
x = MaxPooling2D((2, 2), padding='same')(x)
x = Conv2D(8, (3, 3), activation='relu', padding='same')(x)
encoded = MaxPooling2D((2, 2), padding='same')(x)

# at this point the representation is (4, 4, 8) i.e. 128-dimensional

x = Conv2D(8, (3, 3), activation='relu', padding='same')(encoded)
x = UpSampling2D((2, 2))(x)
x = Conv2D(8, (3, 3), activation='relu', padding='same')(x)
x = UpSampling2D((2, 2))(x)
x = Conv2D(16, (3, 3), activation='relu')(x)
x = UpSampling2D((2, 2))(x)
decoded = Conv2D(1, (3, 3), activation='sigmoid', padding='same')(x)
```

發現使用cnn表現通常會比較好，調好參數可以隨意到達strong base line以上，但最後發現直接把activation換成'selu'分數在kaggle上可以得到1分。

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



C.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。



我用TSNE降維，在預測的圖裡面，藍色團中有紅點，紅色團中有藍點，顯然這些是被預測錯誤的地方。