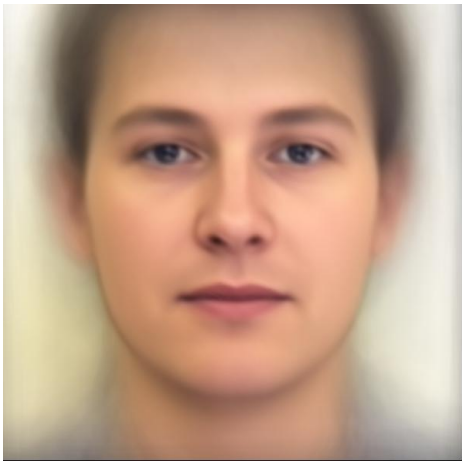
學號：R06942128 系級： 電信碩一 姓名：許祐銘

1. PCA of colored faces

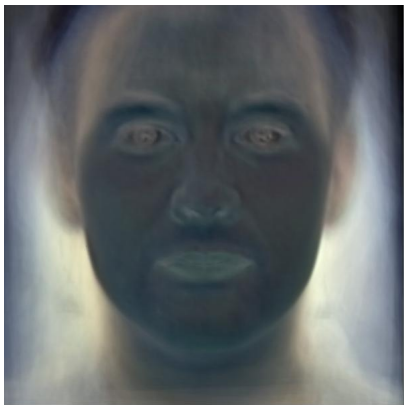
1. (.5%) 請畫出所有臉的平均。



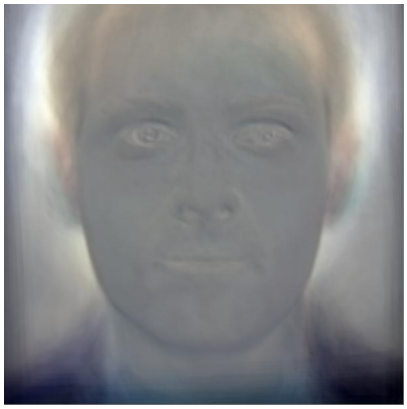
2. (.5%) 請畫出前四個 Eigenfaces，也就是對應到前四大 Eigenvalues 的 Eigenvectors。

eigenface_1

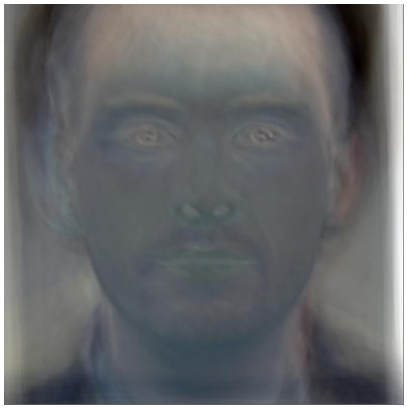
eigenface_2



eigenface_3

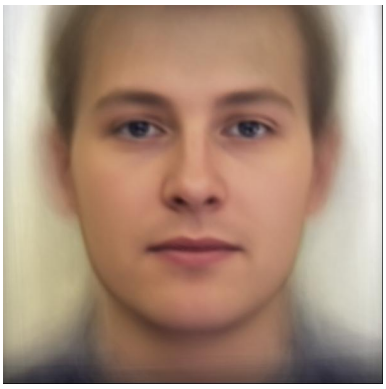


eigenface_4

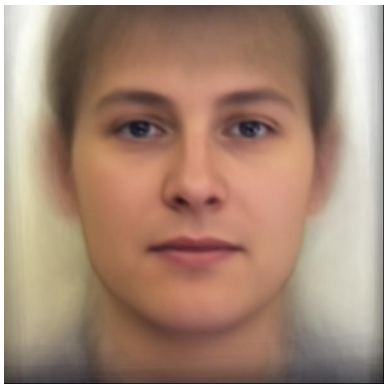


3. (.5%) 請從數據集中挑出任意四個圖片，並用前四大 Eigenfaces 進行 reconstruction，並畫出結果。

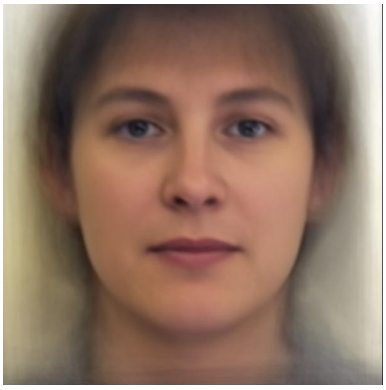
3.jpg



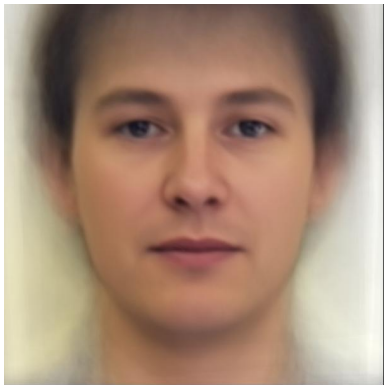
139.jpg



219.jpg



414.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。

	eigenface_1	eigenface_2	eigenface_3	eigenface_4
weight	4.1%	2.9%	2.4%	2.2%

2. Visualization of Chinese word embedding

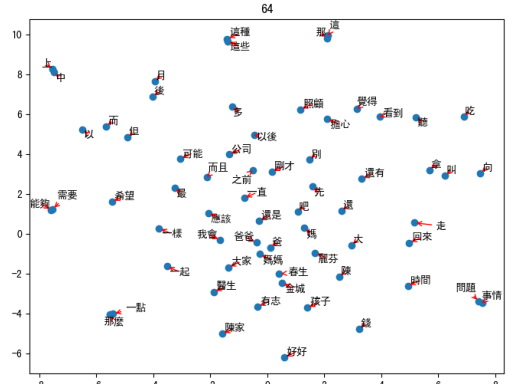
1. (.5%) 請說明你用哪一個 word2vec 套件，並針對你有調整的參數說明那個參數的意義。

   我先使用 jieba 來切割句子，接著使用 gensim 來做 word embedding，再來使用TSNE做降維。

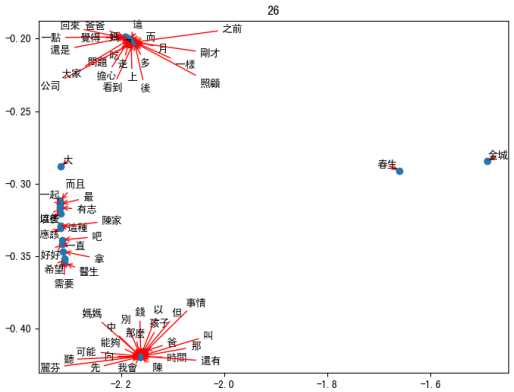
   model = Word2Vec(sent2word_index, size=128, window=5, min_count=5)

   調整的參數： size 是 embedding 的大小， window 則是會訓練時會看前後幾個單詞， min_count 是訓練的最小次數，也就是在 train data 出現小於 5 次的單詞則不會訓練。

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



TSNE參數：
perplexity = 10
n_components=2
learning_rate=10



TSNE參數：
perplexity = 80
n_components=2
learning_rate=10

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

   很明顯的我們可以觀察到perplexity=10的分群效果比80要好的多，80幾乎將許多詞聚集在一起了，而perplexity=10可以將『爸』、『媽』以及人物姓名分在一起，還有『拿』、『叫』的動詞分在一起，看得出來分群效果還不錯。

3. 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)

   我使用的是DNN和CNN的auto encoder，DNN的架構如下：

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 784)               0
_________________________________________________________________
dense_1 (Dense)              (None, 256)               200960
_________________________________________________________________
dense_2 (Dense)              (None, 128)               32896
_________________________________________________________________
dense_3 (Dense)              (None, 64)                8256
_________________________________________________________________
dense_4 (Dense)              (None, 32)                2080
_________________________________________________________________
dense_5 (Dense)              (None, 64)                2112
_________________________________________________________________
dense_6 (Dense)              (None, 128)               8320
_________________________________________________________________
dense_7 (Dense)              (None, 256)               33024
_________________________________________________________________
dense_8 (Dense)              (None, 784)               201488
=================================================================
Total params: 489,136
Trainable params: 489,136
Non-trainable params: 0
```

   而CNN的架構與DNN大同小異，以下是兩種的kaggle分數：

FScore	Public	Private
DNN	0.89285	0.89406
CNN	0.80752	0.80882

   我原本預想的是 CNN 在 image 的結果會比較好，但發現居然是DNN的效果比較好，有可能是參數調整的不好，也有可能是DNN剛好很適合這個data set

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。