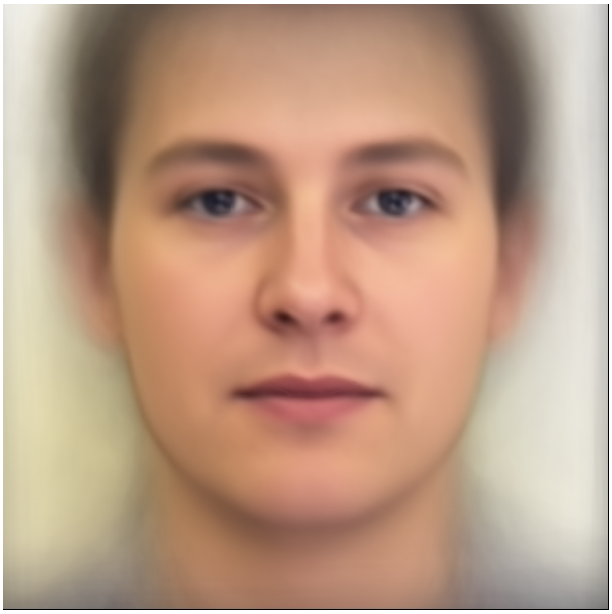
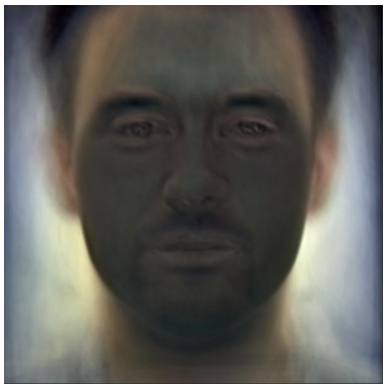
學號：B04902011 系級：資工三 姓名：張立暐

A. **PCA of colored faces** (Collaborator: b04902008 曾千育)

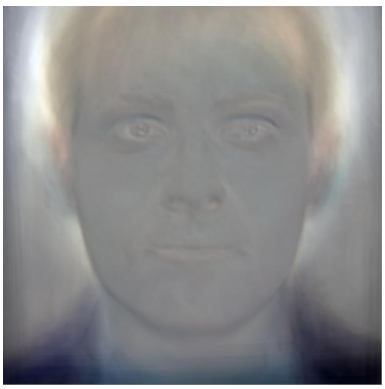
A.1. 請畫出所有臉的平均。



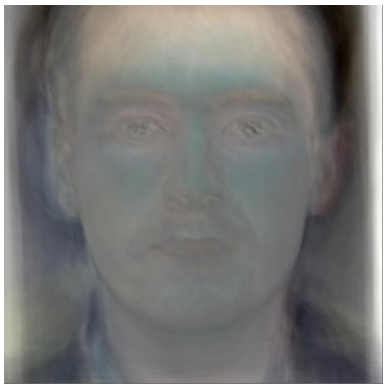
A.2. 請畫出前四個 Eigenfaces，也就是對應到前四大 Eigenvalues 的 Eigenvectors。



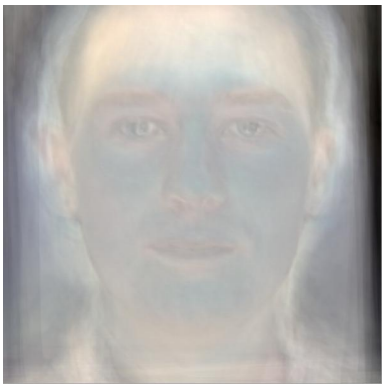
Eigenface 1



Eigenface 2

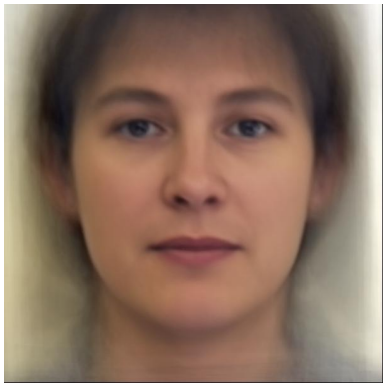


Eigenface 3

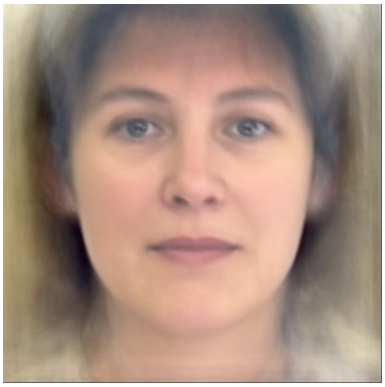


Eigenface 4

A.3. 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。

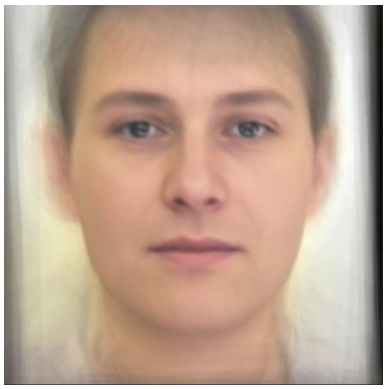


Reconstruct 0.jpg



Reconstruct 4.jpg

Reconstruct 9.jpg



Reconstruct 15.jpg

A.4. 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	4.1%
2	2.9%
3	2.4%
4	2.2%

B. Visualization of Chinese word embedding

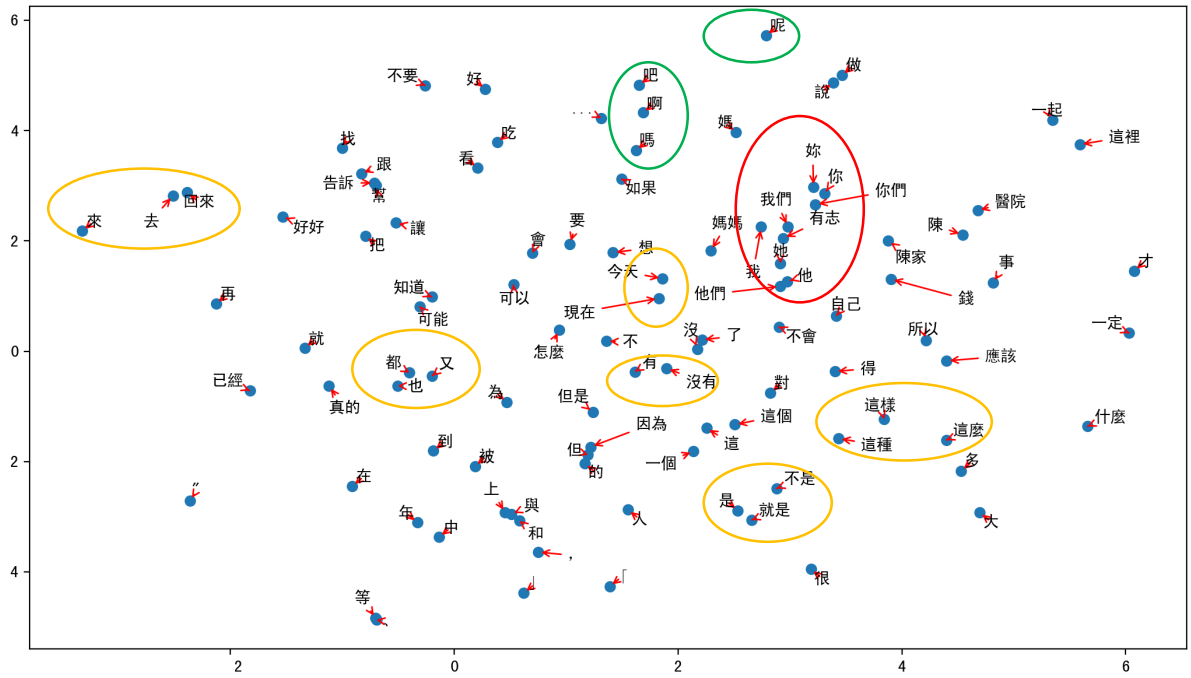
B.1. 請說明你用哪一個 word2vec 套件，並針對你有調整的參數說明那個參數的意義。

我使用 gensim.models.word2vec.Word2Vec 來訓練 word embedding，其中有調整的參數為：

- size=128：將訓練出來的詞向量維度設為 128 維。
- min_count=4500：設定出現次數超過 4500 次的詞才輸出其詞向量。

備註：在使用 jieba 斷詞前有自訂詞典：*jieba.set_dictionary('dict.txt.big')*

B.2. 請在 Report 上放上你 visualization 的結果。



B.3. 請討論你從 visualization 的結果觀察到什麼。

可以從詞在二維平面上的分佈看出一些詞之間的關聯性是有成功地被 word embedding 表現出來的。如「呢、吧、啊、嗎」等會出現在結尾的語助詞分佈在上方偏中的位置；「你、我、他、他們」等人稱代名詞也群聚在上圖紅色圓圈中，甚至有「有志」這個看起來像是名字的詞也跟他們放在一起，充分表現這類詞的相似度；另外像是「是、不是、就是」、「有、沒有」、「現在、今天」、「來、去、回來」等（如上圖中橘色圈圈），各類相似（或相反）詞由於出現在句子中的模式類似而有較靠近的分佈。

C. Image clustering

C.1. 請比較至少兩種不同的 feature extration 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）

Method	Public score	Private score
PCA + KMeans	0.03024	0.03051
TSNE + KMeans	0.06142	0.06135
AutoEncoder + KMeans	**0.99940**	**0.99919**

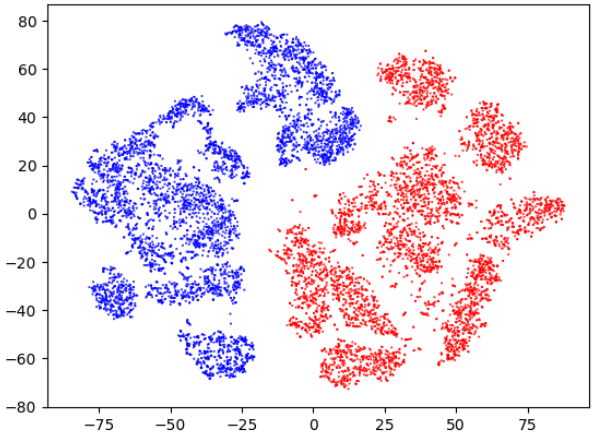
其中 PCA, TSNE, KMeans 皆是使用 sklearn 套件中含有的物件，在參數上，PCA 將原圖降成 64 維、TSNE 將原圖降成 2 維；而 AutoEncoder 是使用 Keras Dense Layer 疊成，過程以 batch_size=256 訓練 128 個 epochs，其參數構造如下：

- unit 依序為 512, 256, 128, 64, 128, 256, 512, 784，並取 64 維那層當作 Encoder 輸出，故此方法將原圖降成 64 維。

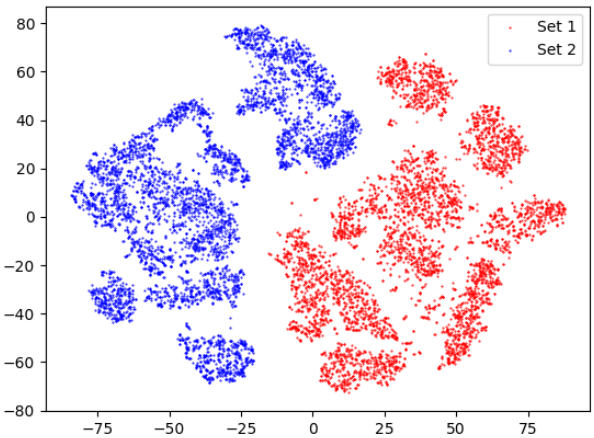
- activation 除了最後一層使用 tanh，其餘皆為 relu 。
- optimizer 為 Adam (lr=0.0001)。
- loss function 為 mse。

可以看出使用 AutoEncoder 來達成降維有最好的分類表現，而另外兩者差強人意，推論原因在於使用多層 Dense Layer 疊成的 NN 可以表現出更多元的轉換效果，是單使用 TSNE 或 PCA 的算法所達不到的。另外，在運算時間上，PCA 雖然結果極差但執行非常快；而 TSNE 執行時間非常久（約八小時），但其結果也不是很起色；AutoEncoder 約需半小時到一小時（用 CPU），其成效相對於時間成本來說也是最好的。

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



C.3. 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。



可以看出與 C.2. 的圖幾乎完全一樣，可想而知是由於用來預測的 model（C.1. 中在 kaggle private score 有高達 0.99919 的 model）在原本的 140000 張圖片中有非常好的分類效果，所以在看到另外 10000 張圖片時雖然有極些微的錯誤，但仍能夠將其幾乎無錯誤的分類正確。