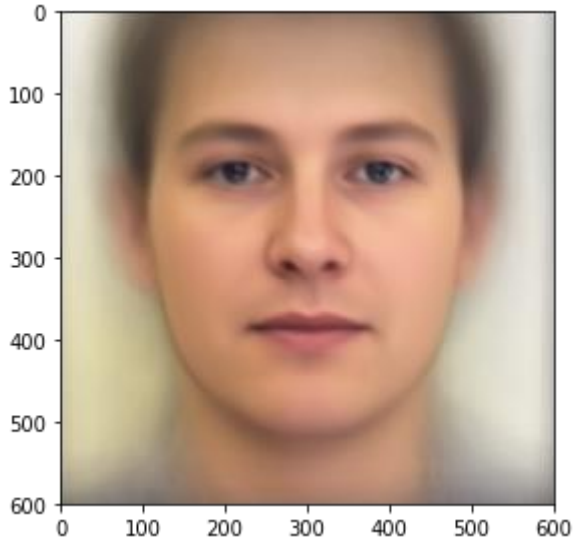
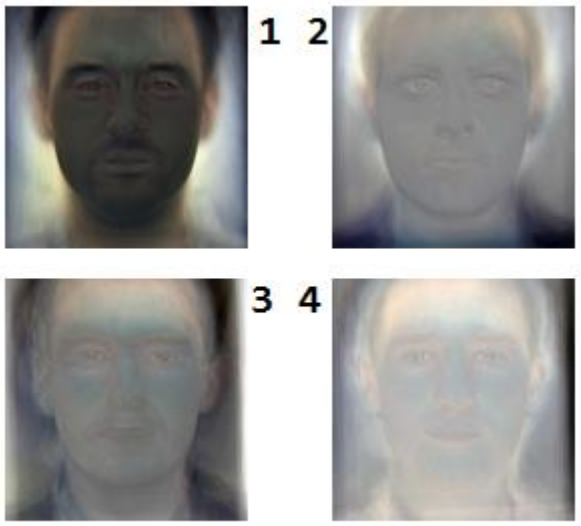
A. PCA of colored faces

A.1. (.5%) 請畫出所有臉的平均。



圖一、平均臉

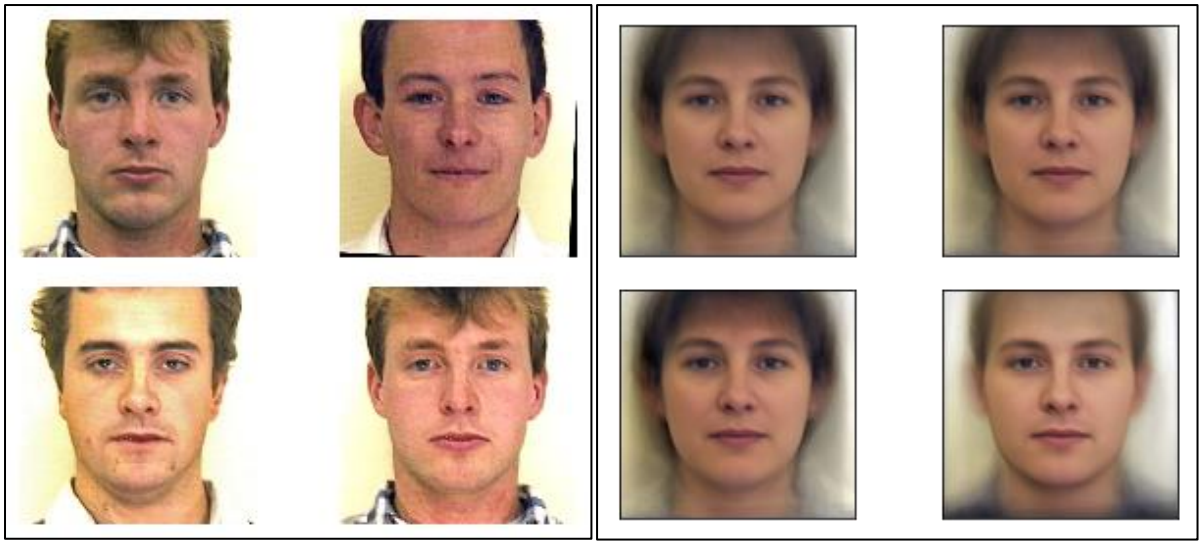
A.2. (.5%) 請畫出前四個 Eigenfaces，也就是對應到前四大 Eigenvalues 的 Eigenvectors。



圖二、top 4 Eigenfaces

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。

以下是我任意挑選的四張圖片原圖，以及利用前四大 Eigenfaces 作的 reconstruction。



圖五、原圖 (左) 和 重建圖 (右)

A.4. (.5%) 請寫出前四大 Eigenfaces 各自所佔的比重，請四捨五入到小數點後一位。

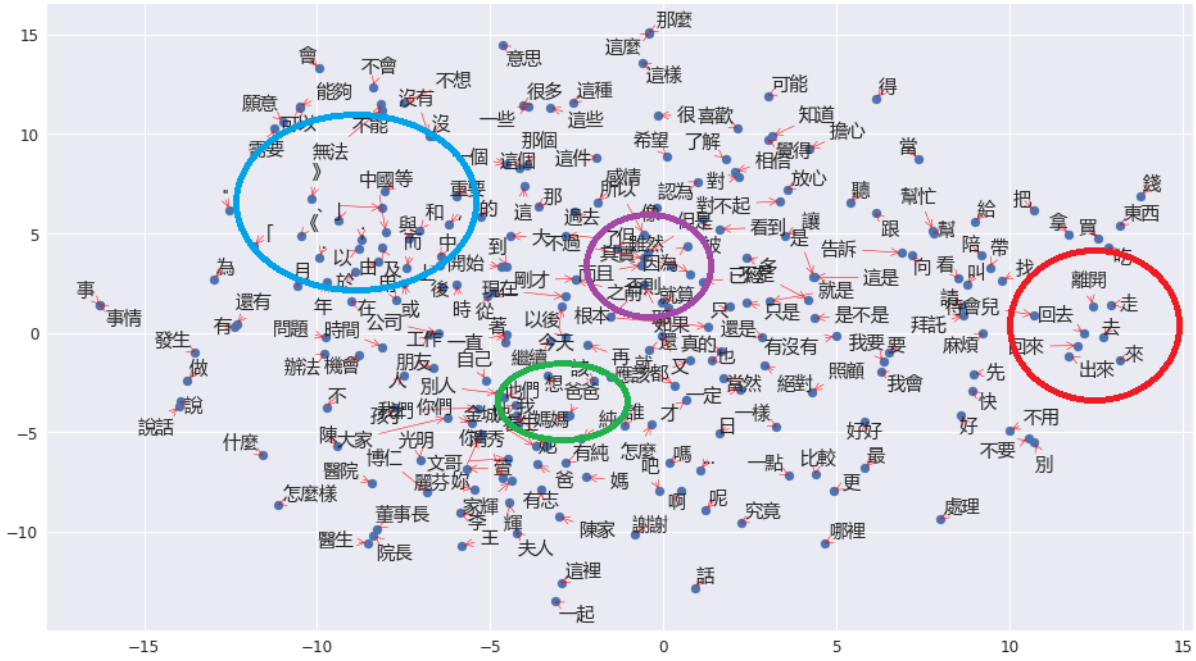
1st Eigenfaces proportion	2nd Eigenfaces proportion	3rd Eigenfaces proportion	4th Eigenfaces proportion
4.1%	2.9%	2.4%	2.2%

B. Visualization of Chinese word embedding

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。

我使用 gensim word2vec 套件，並且調整 size = 100 和 min_count = 5。其中，size 控制 embedding 的 latent dimension，這邊就是將一個詞用 100 維描述。min_count 則是用來篩選提供給模型訓練的詞彙，若是詞彙出現於 corpus 中的次數小於 min_count，就拿掉，不予以計算。

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



圖六、Embedding visualization

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

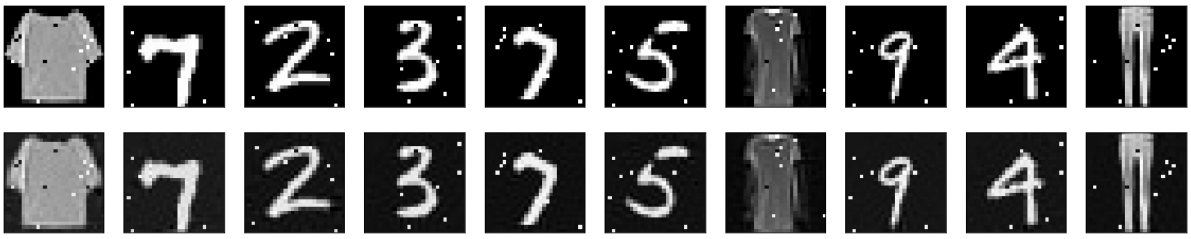
首先可以看到圖六中，藍圈處有許多標點符號集中在一起。而紫圈處都是屬於一些轉折語、介係詞等。紅圈處都來來去去的動作詞。綠圈處則可看到爸爸媽媽兩個詞很近。因此，不難發現 word2vec 能考量到語意上的分布。

C. Image clustering

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)

這邊我比較兩種降維方法，PCA 和 Autoencoder:

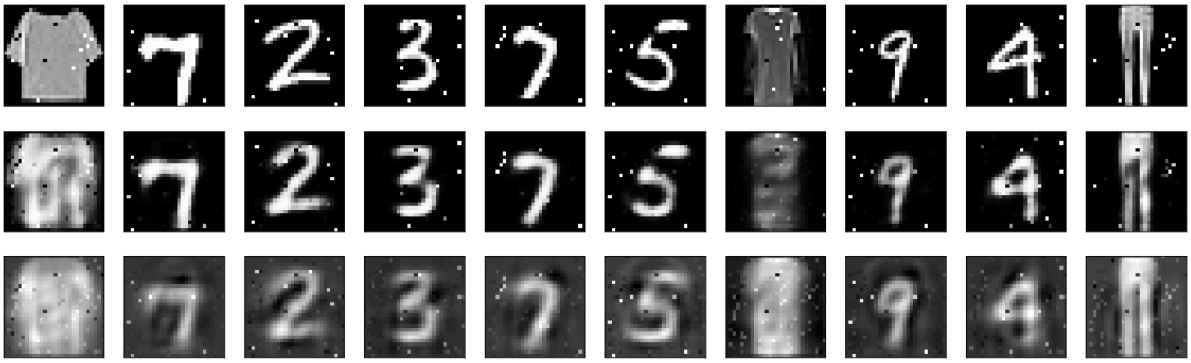
圖七、原圖(上) 和 Autoencoder 重建(下)



圖八、原圖(上) 和 PCA 重建

比較實驗的設定是先將 784 維的向量降成 400 維，再用這 400 維向量重建原

圖。從圖七和圖八中可以看到用不同降維方法並且重建的結果。可以發現兩者重建效果差不多，可能是因為 400 維保留的資訊還算豐富。在 400 維的狀態下利用 Kmeans 分群的 Kaggle 結果是 PCA 比較好，F1 score 是 1.0，而 Autoencoder 則約為 0.90055。若降到 32 維，並重複上述實驗:



圖九、原圖(上)、Autoencoder 重建(中) 和 PCA 重建(下)

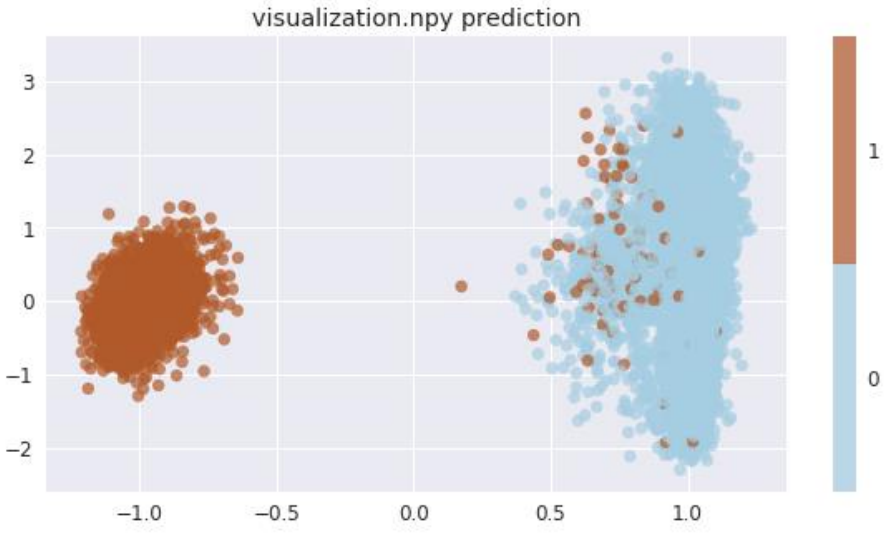
F1 score	Autoencoder	PCA
32 dims	0.03706	0.01985
400 dims	0.90055	1.0

表一、 kaggle 分數比較

從圖九和上表可以看出，Autoencoder 在大幅度壓縮下可以有較好的表現，即圖案比較清楚，並且分群後的 F1 也略高，整體而言保留較多資訊。

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

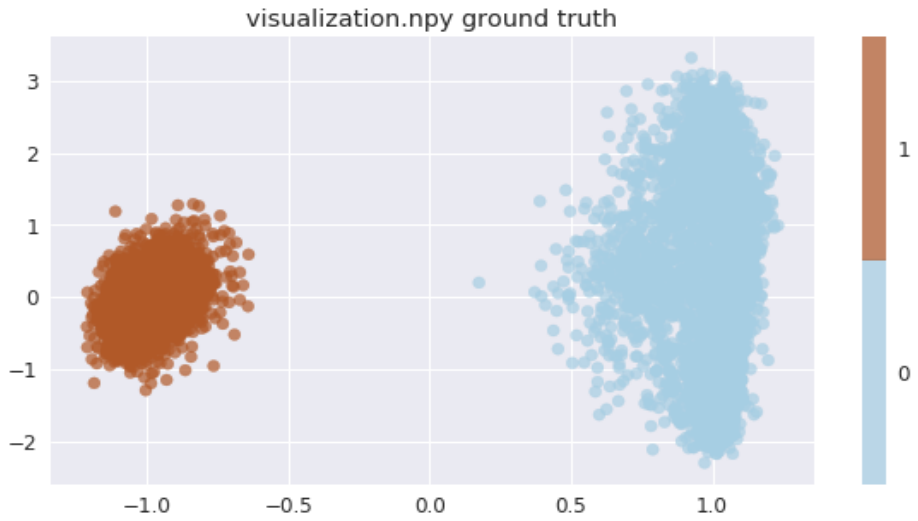
用 image.npy train 出來的模型去預測 visualization.npy。並用 PCA 將維投影到二維平面作圖。



圖十、二維平面上視覺化 visualization.npy 的 label

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。

在比較 visualization.npy 的 prediction label 與 ground truth 差異時有發現有 96 個 ground truth label 為 0 的圖片被分為 1。接著將圖片用 PCA 降成 2 維作圖。比較圖十和圖十一，可以看到在圖十一中應該是藍點(0)的 sample，在圖十中被標為咖啡色(1)的點混雜在一群藍色點中。這些點應該就是那些被 Kmeans 分錯的點。



圖十一、二維平面上視覺化 visualization.npy 的 ground truth label