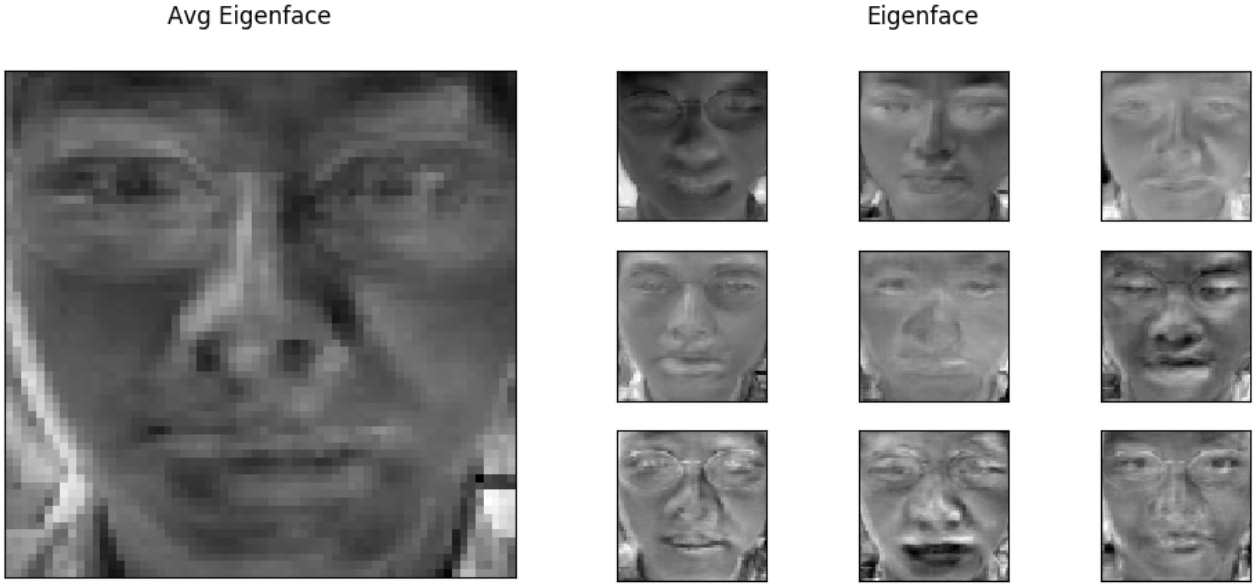
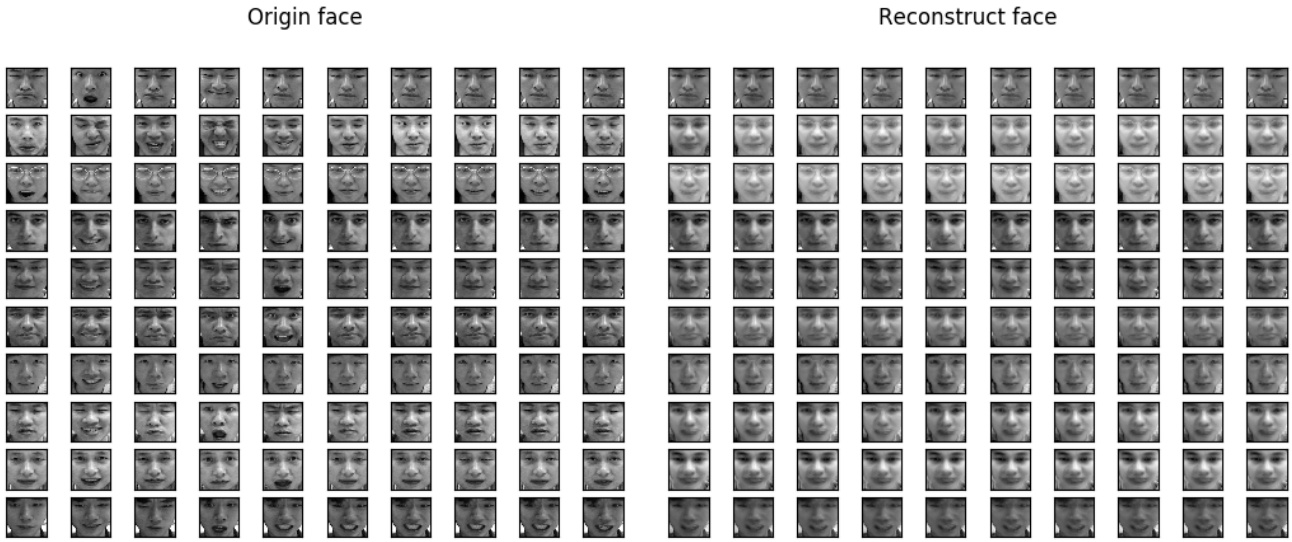
1.1. Dataset 中前 10 個人的前 10 張照片的平均臉和 PCA 得到的前 9 個 eigenfaces:
答： (左圖平均臉，右圖為 3x3 格狀 eigenfaces, 順序為 左到右再上到下)

- 使用numpy.linalg.eigh來求face - face.mean前9大的eigenvector



1.2. Dataset 中前 10 個人的前 10 張照片的原始圖片和 reconstruct 圖 (用前 5 個 eigenfaces):
答：(左右各為 10x10 格狀的圖, 順序一樣是左到右再上到下)

- 使用前5大的eigenvector來降維，並轉換回原始大小　　arr.dot(U[:5].T).dot(U[:5])
- 由下圖可以發現reconstruct後，每個人臉部的表情都變得一樣，張口的部份都變成閉口



1.3. Dataset 中前 10 個人的前 10 張照片投影到 top k eigenfaces 時就可以達到 < 1% 的 reconstruction error.
答： (回答 k 是多少)

- 59
- 由小到大枚舉k值

2.1. 使用 word2vec toolkit 的各個參數的值與其意義:

答：

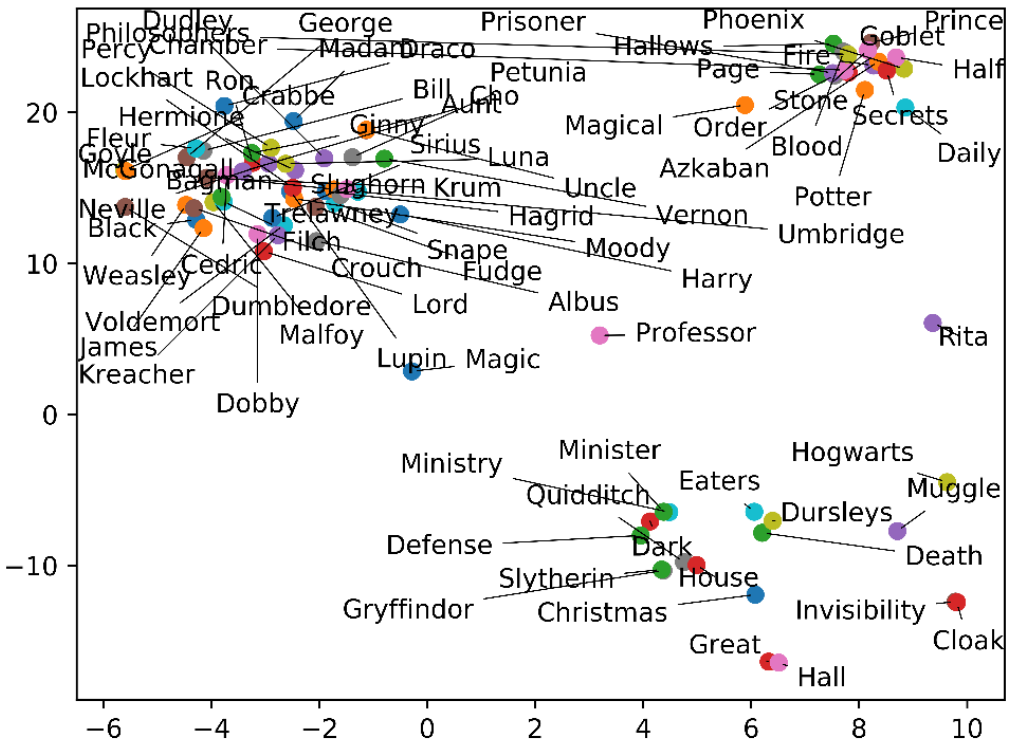
- ❖ 我有用到的
  - train: args　　training txt 位置
  - output: args　　model 儲存位置
  - size: 87　　降為後的維度
  - verbose: True　　是否印出過程

```
word2vec.word2vec(
    train=args.corpus_path,
    output=args.model_path,
    size=WORDVEC_DIM,
    verbose=True)
```

- ❖ 我沒用到的(default)
  - iter_: 5　　iteration
  - alpha: 0.025　　learning rate
  - min-count: 5　　低頻詞threshold
  - window: 5　　上下文窗口大小介於5~10
  - cbow: True　　是否使用CBOW model

2.2. 將 word2vec 的結果投影到 2 維的圖:

答：(圖)



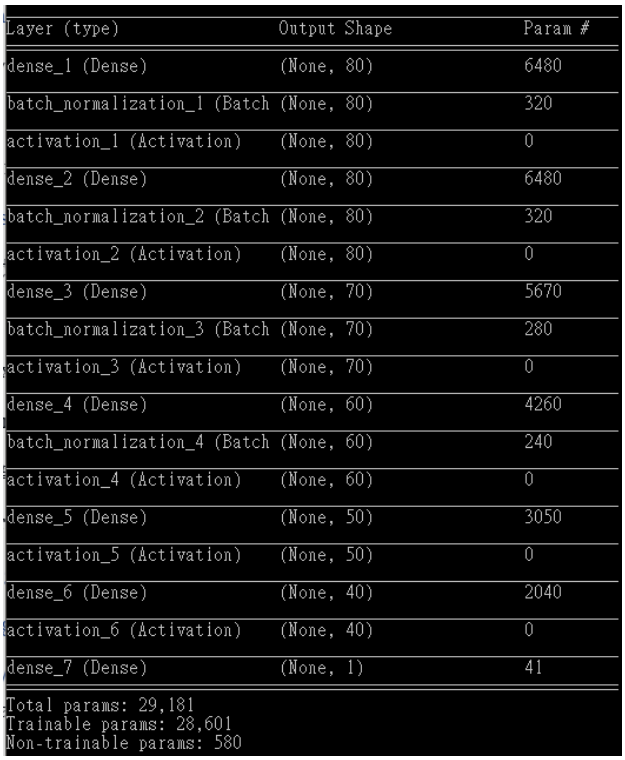
2.3. 從上題視覺化的圖中觀察到了什麼？

答：

- ❖ 左上角主要是一些人名(Voldemort, Dumbledore, Dobby)，以及他的稱謂(Lord, Madam, Uncle)
- ❖ 右上角是和主題相關的(Secret, Stone, Prince, Blood, Fire)
- ❖ 右下角是一些活動、原創、專有名詞(Gryffindor, Quidditch, Chritmas, Muggle)

3.1. 請詳加解釋你估計原始維度的原理、合理性，這方法的通用性如何？
答：

- DNN & merge model, public test: 0.054
  - 原理：考慮gen給出的狀況、原始dim和eigenvalue有很大的關係，我透過助教的程式生成12000比training data，將前80大的eigenvalue當作feature，原始的dimension當作label進行DNN training (200 epochs)，並將不同參數training的結果拿來觀察他再哪些dimension可以比較準確的預測，然後把他們merge起來，DNN的結構如右圖。



Layer (type)	Output Shape	Param #
dense_1 (Dense)	(None, 80)	6480
batch_normalization_1 (Batch	(None, 80)	320
activation_1 (Activation)	(None, 80)	0
dense_2 (Dense)	(None, 80)	6480
batch_normalization_2 (Batch	(None, 80)	320
activation_2 (Activation)	(None, 80)	0
dense_3 (Dense)	(None, 70)	5670
batch_normalization_3 (Batch	(None, 70)	280
activation_3 (Activation)	(None, 70)	0
dense_4 (Dense)	(None, 60)	4260
batch_normalization_4 (Batch	(None, 60)	240
activation_4 (Activation)	(None, 60)	0
dense_5 (Dense)	(None, 50)	3050
activation_5 (Activation)	(None, 50)	0
dense_6 (Dense)	(None, 40)	2040
activation_6 (Activation)	(None, 40)	0
dense_7 (Dense)	(None, 1)	41

Total params: 29,181
Trainable params: 28,601
Non-trainable params: 580

  - 合理性：我們可以知道eigenvalue愈大的可以將數值拉的愈開，所以由大到小取80做training還蠻合理的，然後我們又知道資料量愈大可以使training的準確度愈高，因此選擇從dim1~60各生成了2000比training data來訓練。
  - 通用性：不怎麼通用，因為一般來說你不會知道中間是怎麼轉換的，所以沒有辦法像這題可以生成很多測資並且觀察不同model得結果

- 相較起這個方法，我一開始嘗試過用threshold的方式，只取大於總和某一個百分比的，但是在kaggle上只拿到0.3，無法通過baseline，但是這個方法可以通用在所有的data set上

3.2. 將你的方法做在 hand rotation sequence datatset 上得到什麼結果？合理嗎？請討論之。
答：

- 將所有的照片讀進來，先將照片壓縮成48*50，避免求eigenvalue的時候MLE，接者取出前80大的eigenvalue當作feature餵給model
- 結果：6維
- 本來以為測出來的數字會很可怕，因為這兩個dataset顯然他中間的轉換很不一樣，然後答案是6讓我感到很意外。這些照片非常相似，所以我覺得為度本來就不會太高，結果還算合理。