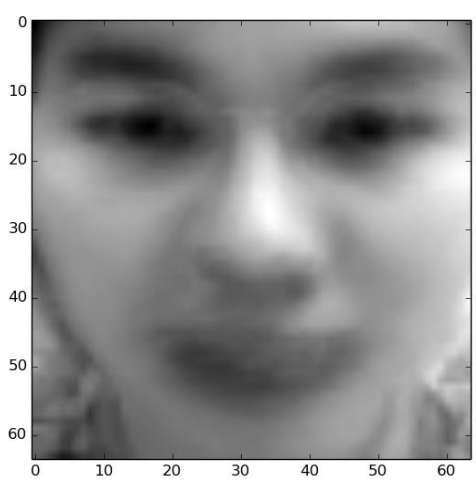
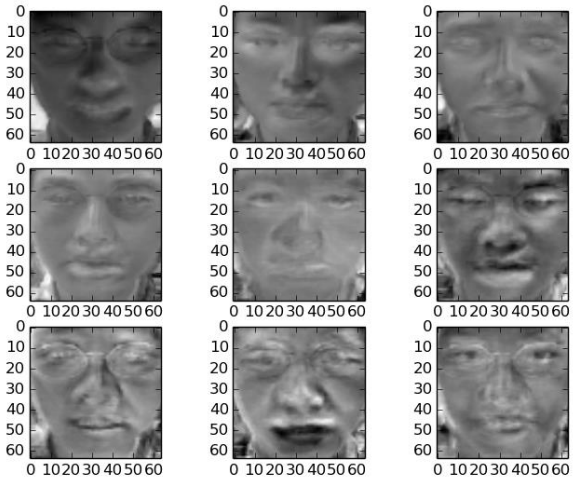
1.1. Dataset 中前 10 個人的前 10 張照片的平均臉和 PCA 得到的前 9 個 eigenfaces:

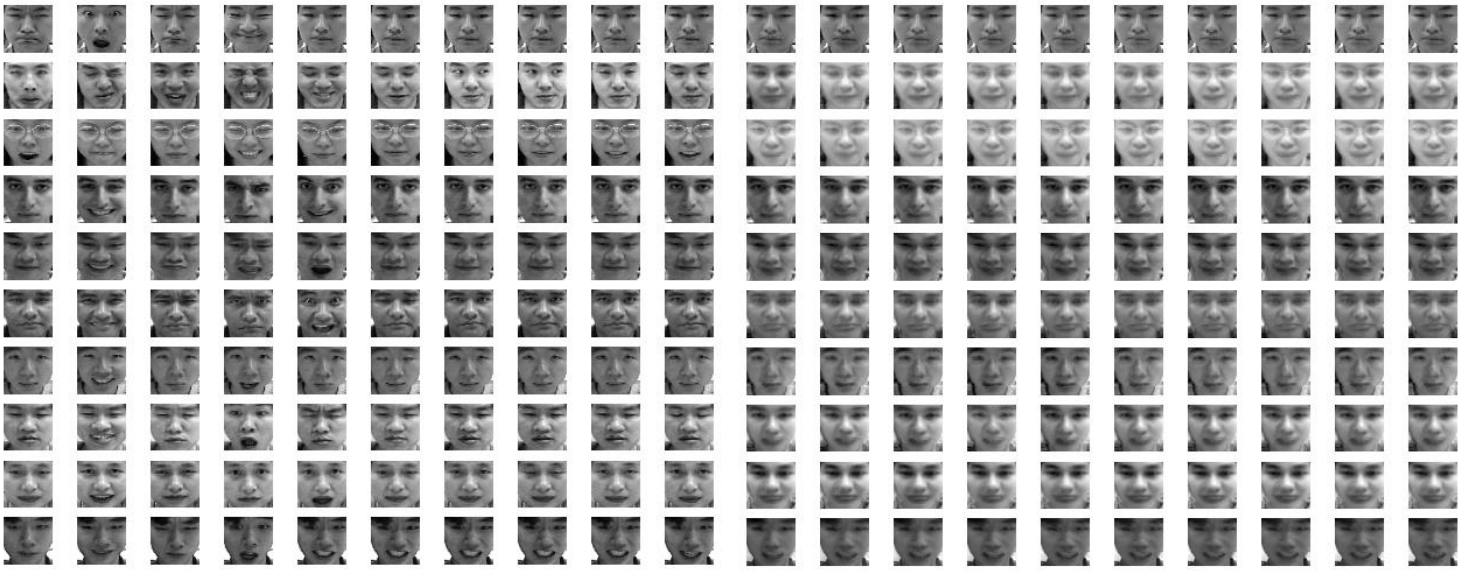
答： (左圖平均臉，右圖為 3x3 格狀 eigenfaces, 順序為 左到右再上到下)





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)



原始圖片　　reconstruct 圖

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)

```
data = data.reshape(100, 4096)
dim = 0
for k in range(100):
    eigen = v[0:k+1]
    x_reduce = np.dot(data_ctr, eigen.T)
    x_rec = np.dot(x_reduce, eigen) + data_mean
    error = data - x_rec
    rmse = ((error**2).mean())**0.5 / 255
    if rmse < 0.01:
        dim = k+1
        break
print(dim)
```

算出來 k = 60 才可達到 < 1% 的 reconstruction error

2.1. 使用 word2vec toolkit 的各個參數的值與其意義:

答：

```
corpus_path = 'all.txt'
output_path = 'model.bin'
MIN_COUNT = 10
WORDVEC_DIM = 1000
WINDOW = 3
SAMPLE = 1e-5
NEGATIVE_SAMPLES = 8
ITERATIONS = 50
MODEL = 1
LEARNING_RATE = 0.025

# train model
word2vec.word2vec(
    train=corpus_path,
    output=output_path,
    cbow=MODEL,
    size=WORDVEC_DIM,
    sample=SAMPLE,
    min_count=MIN_COUNT,
    window=WINDOW,
    negative=NEGATIVE_SAMPLES,
    iter_=ITERATIONS,
    alpha=LEARNING_RATE,
    verbose=True)
```

size(1000)：表示詞(word)的向量維度為 1000

sample(1e-5)：高頻詞的閥值，可隨機降低高頻詞的採樣率

window(3)：訓練一個 word 時，也會看其前後的三個 words

cbow(1)：model 使用 cbow (Continuous Bag of Words)

min_count(10)：word 若出現少於 10 次即被忽略，不看低頻的 word

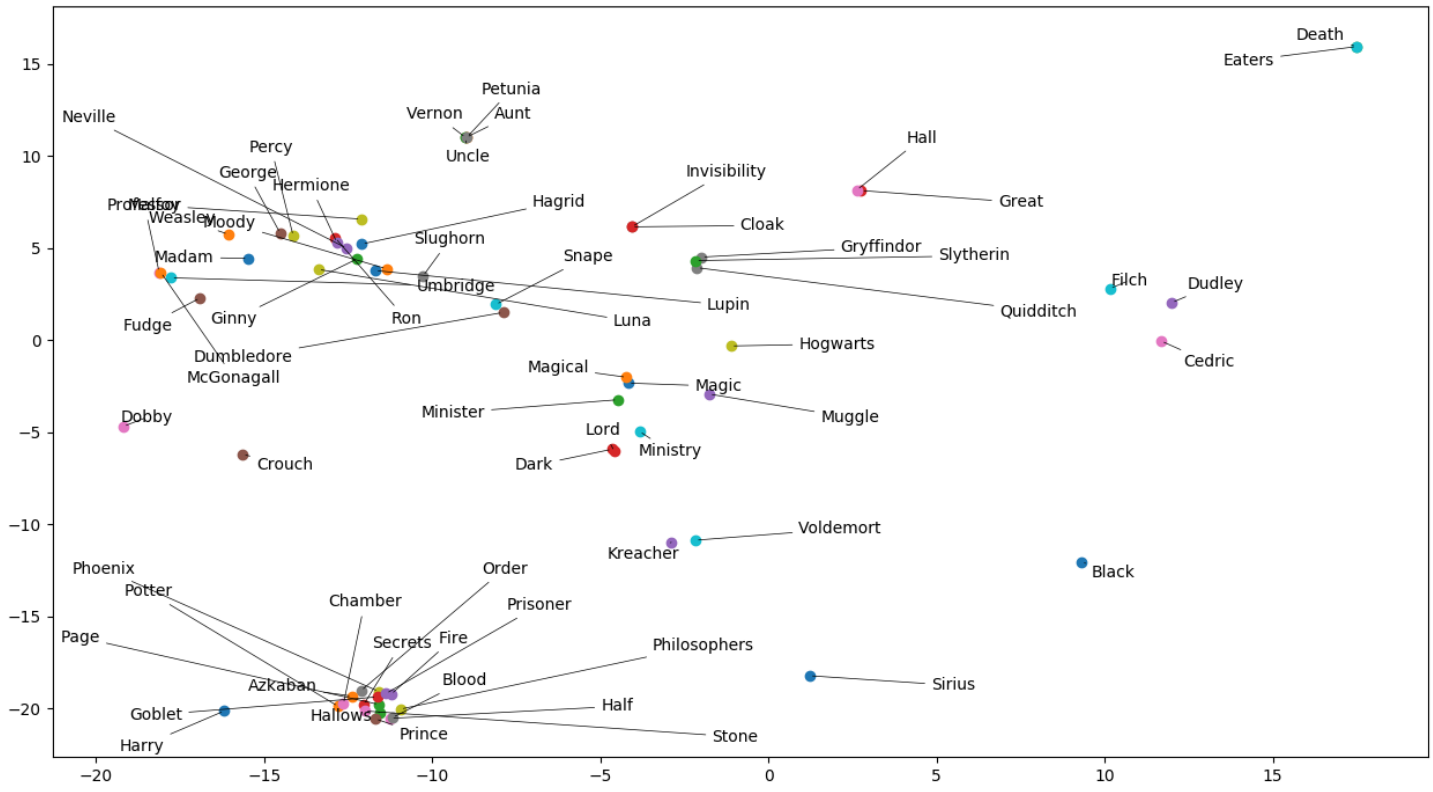
negative(8)：負採樣數目，可提高運算效率

iter_(50)：迭代的次數

alpha(0.025)：開始的 learning rate

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：

經過 TSNE 降至 2-dim 後，較為相關的 word 或者常接連出現的 word 會分布在一起，如 Death、Eaters(Death Eater 是佛地魔的黨羽起初自稱的稱號)；Aunt、Uncle、Petunia、Vernon(之間的關係)；Magical、Magic、Muggle(皆與魔法方面有關)、Invisibility、cloak(隱形斗篷)；Snape、Dumbledore(人物)等等。因此可對 word 做較明顯的分類。

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：

原理為先依序產生 dim 為 1 至 60 的 dataset，再將其擴充為 dim=100，每個 dataset 裡隨機採樣點，再取出各自相鄰的幾個點，形成 subset，最後算出每個平均的 eigenvalue 並存成 train_model；在預估時也是將 test 的 dataset 算出其平均 eigenvalue，再利用 SVR 解出其預估維度。

因為已知道 test 的 dataset 的產生方式，因此用相同的產生方式統計每個維度的平均 eigenvalue，再套用至 test 的 dataset 做比對，所以其預估結果相當合理也有不錯的效果。

其通用性在於若知道 dataset 的產生方式，利用此統計的概念，可得到不錯的結果，但假如事先不知道的話，則無法自己產生有效且合理的 dataset 出來並且統計。

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：

hand rotation sequence datatset 中共有 481 張照片，每張維度為 512*480，我是先利用 PCA 將每張照片降至 100 維(矩陣 size：(481, 512*480) → (481, 100))，再利用 3.1 預估維度的方法來降維，最後跑出來的降為結果大約為 9(8.77086)。

我覺得不太合理，因為我們事先並不知道這個 dataset 的產生方式，直接利用統計的概念去降維，可能較不適合，因為從產生本質就不相同了。

每張照片幾乎只差在角度的問題，因此應該可用一個維度來表示即可，就是旋轉的角度。