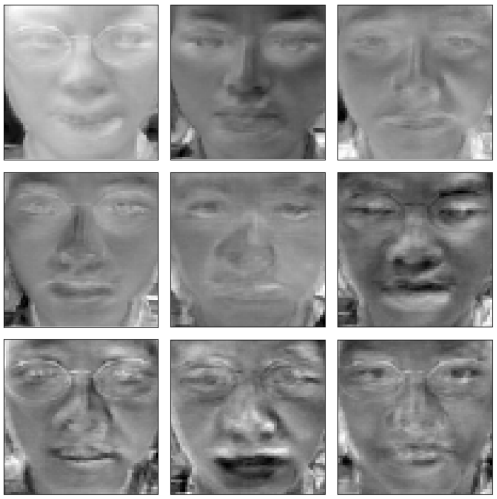
學號：B03902042　　系級：資工三　　姓名：宋子維

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)

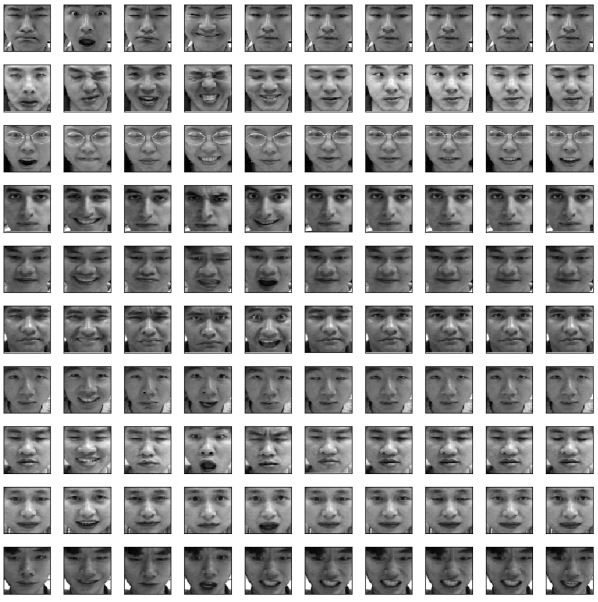
(a) 平均臉。



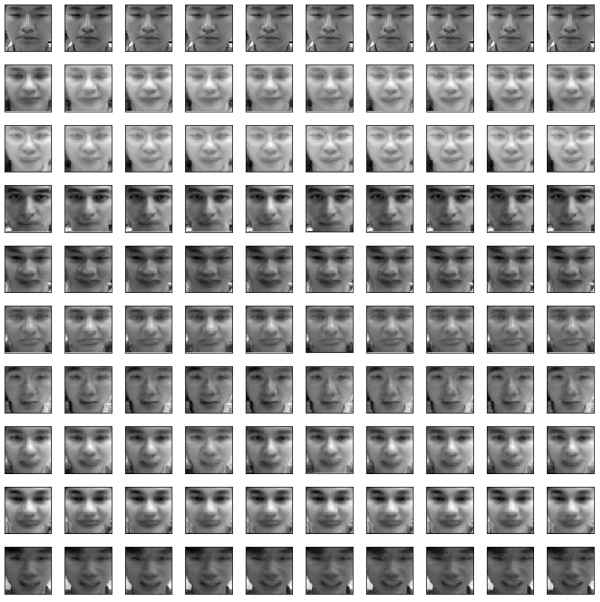
(b) Eigenfaces。

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)



(a) 原始圖片。



(b) Reconstruct 圖片。

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：k=59。

2.1 使用 word2vec toolkit 的各個參數的值與其意義:

答：

- `size`: 128，產生的 word vector 的大小。
- `window`: 5，skip gram (cbow) model 的 window 大小，固定中間的字，往左右會最多看多少個字。
- `hs`: 1，使用 hierarchical softmax。原先 skip gram 或 cbow model 中，做 gradient descent 更新參數時所需要的計算量很多 (softmax layer 需要遍歷所有單字)，而 hierarchical softmax 在訓練前預先建好 binary huffman tree，將短的 code 分給高頻字，亦即將單字先分類好，因此，在更新參數時，能階層式的算出 softmax 的近似值，大幅減少 softmax 的計算量，加快訓練過程。
- `cbow`: 0，使用 skip gram model (1 為使用 cbow)。skip gram 為給定中間的字 w，預測在 window 內的字 c 的機率 $p(c|w)$。而 cbow 則相反，給定 window 內的字 c，預測中間被挖空的字 w 的機率 $p(w|c)$。
- `negative`: 5，使用 negative sampling，且 negative sample 的數量為 5 (0 為不使用 negative sampling)。以 skip gram model 為例，原先的目標是給定 w_I，對 target w_O 做 multiclass classification，而 negative sampling 透過採樣的方式，將目標改為用 logistic regression 來分辨 w_O 和 noise，其中 w_O label 為 1，而其他被採樣出的 negative sample label 為 0。
- `sample`: 1e-3，對高頻字做 subsampling，出現頻率大於 sample 的字在訓練時會有「機率」被忽略，不僅能加速訓練過程，在論文中也提到能使產生的 embedding 更好一些。
- `min_count`: 3，對低頻字做 purning，小於 `min_count` 的字會被丟棄。
- `alpha`: 0.025，learning rate 的初始值，在訓練過程中，learning rate 會一直衰減，因此需要一個初始值。

2.2 將 word2vec 的結果投影到 2 維的圖:

答：

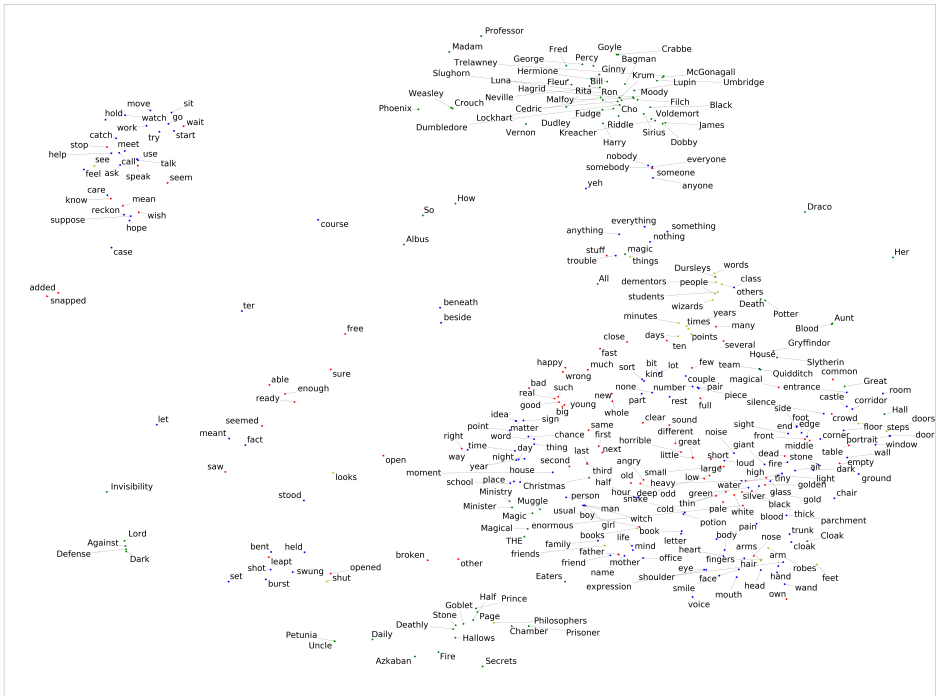


Figure 3: Vector 投影到 2 維。

2.3 從上題視覺化的圖中觀察到了什麼？

答：可以發現左上角的群落 (see, help, stop...)，都是原型動詞；正上方的綠色點群落 (Fred, Harry, Rita...)，都是名字；正下方綠色點群落 (Prince, Philosophers, Prisoner...)，都是出現在書名裡的單字，而在此群落的左方，有個都是過去式動詞的群落 (bent, held, opened...)；而右下角的小群落 (arms, fingers, nose...)，都是身體的器官。

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：**原理為透過統計的方式，算出各個 intrinsic dimension(ID) 的統計量 (我是用最近鄰居的距離)，在測試時也對輸入資料量算出相對應的統計量，看和哪個 ID 的該統計量最為相近即選哪個 ID 作為預測值。**詳細過程如下：

首先，我用 `gen.py` 生成 ID 從 1 到 60 的 data (shape 為 $N \times 100$)，然後對 data 做以下處理：

- X: data 的前 5000 個點，其中每個點都是 100 維，故 X 的 shape 為 5000×100。
- NN: X 裡面每個點和最近鄰居的距離，故 NN 為 5000 維的向量，每個維度的數值都是相對應點的最近鄰居距離。
- μ: NN 的平均，即為每個點和最近鄰居的距離之平均。

為了增加準確度，我對每個 ID 都做 100 次採樣，將其算出的 μ 做平均。最後，能得到各個 ID d 中的 "點和最近鄰居距離之平均" μ_d。而在測試時，只須依照上述方法對輸入資料算出 μ，接著和已經算好各個 ID d 的 μ_d 相比，找與 μ 最相近 μ_d 的 ID d 當作預測值。

此方法若做在用 `gen.py` 產生的資料相當合理，因為用 `gen.py` 產生的資料，在經過多次採樣後，其 "最近鄰居的距離之平均" 會隨著 ID 上升而上升，且固定 ID 之下，該統計量的標準差很小，代表該統計量確實和 ID 有著顯著的關係，且相當穩定；但若測試資料不是由 `gen.py` 產生，則在測試資料上的統計量未必和 `gen.py` 產生資料的統計量有著相似的分佈，因此，此方法並不是個通用的方法。

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：我認為 Ground Truth 應該是 3，因為每張圖片皆為二維，且圖片依序對著一維的曲線旋轉，因此，實際 ID 為 3 感覺是很合理的數值。然而，用上述的方法，在此 datatset 上預測 ID 為 1，顯然非常不合理，畢竟單一張圖片就有二維了，預測為 1 顯然是少考慮了許多資訊，我覺得可能是因為圖片的 orcale network 和 `gen.py` 不同，導致最近鄰居距離和預先建好的表有著不一樣的分佈。我因此實作了 Minimum Neighbor Distance Estimators of Intrinsic Dimension[1] 這篇文章所提到的 `MiND_ML` 和 `MiND_KL` 這兩個方法，在 hand 這個 dataset 上都預測為 3，然而，這兩個方法在這次的 kaggle 比賽中並無法取得比較好的成績。

References

[1] Gabriele Lombardi et al. "Minimum Neighbor Distance Estimators of Intrinsic Dimension". In: *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2011, Athens, Greece, September 5-9, 2011, Proceedings, Part II*. Ed. by Dimitrios Gunopulos et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 374–389. ISBN: 978-3-642-23783-6. DOI: 10.1007/978-3-642-23783-6_24. URL: http://dx.doi.org/10.1007/978-3-642-23783-6_24.