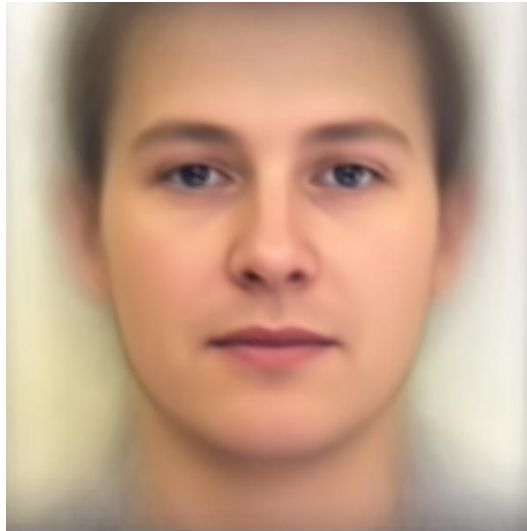


Machine Learning HW7 Report






學號：B05502145 系級：電機三 姓名：林禹丞

1. PCA of color faces:



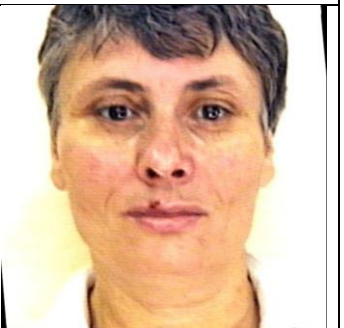





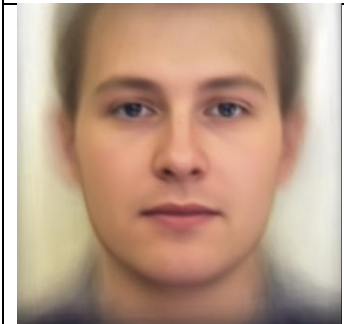

a. 請畫出所有臉的平均。



b. 請畫出前五個 Eigenfaces，也就是對應到前五大 Eigenvalues 的 Eigenvectors。

Eigenface 1	Eigenface 2	Eigenface 3
		
Eigenface 4	Eigenface 5	
		

- c. 請從數據集中挑出任意五張圖片，並用前五大 **Eigenfaces** 進行 **reconstruction**，並畫出結果。

第 10 張原圖	第 30 張原圖	第 50 張原圖
		
第 10 張重建	第 30 張重建	第 50 張重建
		
第 70 張原圖	第 90 張原圖	
		
第 70 張重建	第 90 張重建	
		

- d. 請寫出前五大 **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。

	第一大	第二大	第三大	第四大	第五大
比率	4.1	2.9	2.4	2.2	2.1

2. Image clustering:

a. 請實作兩種不同的方法，並比較其結果(reconstruction loss, accuracy)。

(不同的降維方法或不同的 cluster 方法都可以算是不同的方法)

方法	說明	Reconstruction loss	Accuracy(public/private)
PCA + Kmeans	利用 PCA 把原圖降到 400 維，再用 Kmeans 分成 2 個 cluster	0.21385(MSE)	0.95605/0.95563
Autoencoder(VAE) + Kmeans	利用 autoencoder 把原圖降到 400 維，再用 Kmeans 分成 2 個 cluster	0.01295(MSE)	0.90075/0.90070

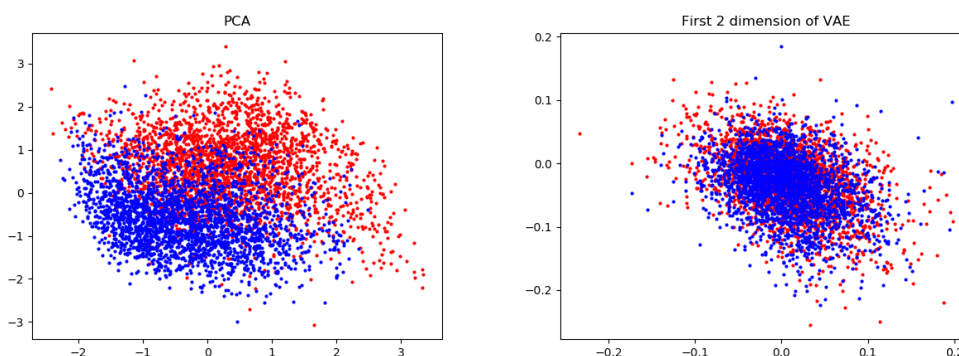
其中 PCA 使用 sklearn 的套件 PCA，其中 whiten=True

使用 4 層 conv(3,60,120,120,120) + flatten + dense(400)的 autoencoder

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

(用 PCA, t-SNE 等工具把你抽出來的 feature 投影到二維，或簡單的取前兩維 2 的 feature)

其中 visualization.npy 中前 2500 個 images 來自 dataset A，後 2500 個 images 來自 dataset B，比較和自己預測的 label 之間有何不同。



由圖可知 PCA 可以很有效的用 2 維就把資料分的很清楚。而 VAE 的部分則需要再用 PCA 再做降維才會有更好的效果。

- c. 請介紹你的 model 架構(encoder, decoder, loss function...)，並選出任意 32 張圖片，比較原圖片以及用 decoder reconstruct 的結果。

Encoder 架構：

Layer (type)	Output Shape	Param #
=====		
input_1 (InputLayer)	(None, 32, 32, 3)	0
conv2d_1 (Conv2D)	(None, 32, 32, 3)	39
conv2d_2 (Conv2D)	(None, 16, 16, 60)	780
conv2d_3 (Conv2D)	(None, 16, 16, 120)	64920
conv2d_4 (Conv2D)	(None, 16, 16, 120)	129720
conv2d_5 (Conv2D)	(None, 16, 16, 120)	129720
flatten_1 (Flatten)	(None, 30720)	0
dense_1 (Dense)	(None, 400)	12288400
=====		
Total params: 12,613,579		
Trainable params: 12,613,579		
Non-trainable params: 0		

Decoder 架構無法和 encoder 部分分開，因為我們的 vae autoencoder 在中間有一層接到兩層 layer，最後再併起來，而 encoder 只是其中一邊。所以我們列出整個 vae 的架構(其中紅色的就是上一小題的 encoder 部分)：

Layer (type)	Output Shape	Param #	Connected to
=====			
input_1 (InputLayer)	(None, 32, 32, 3)	0	
conv2d_1 (Conv2D)	(None, 32, 32, 3)	39	input_1[0][0]
conv2d_2 (Conv2D)	(None, 16, 16, 60)	780	conv2d_1[0][0]
conv2d_3 (Conv2D)	(None, 16, 16, 120)	64920	conv2d_2[0][0]

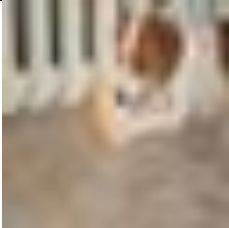

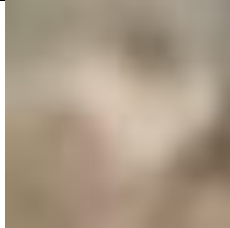
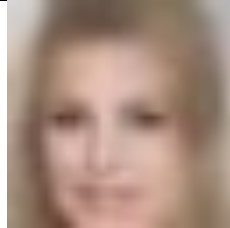
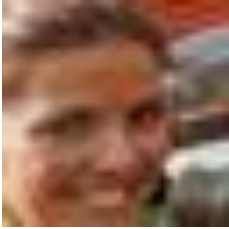
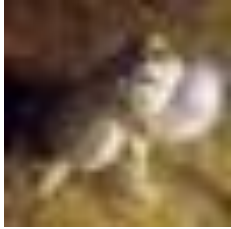
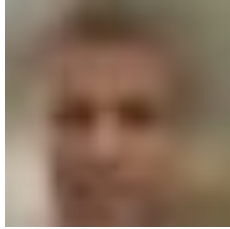
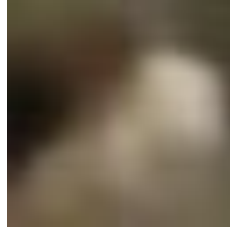

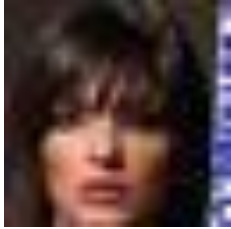

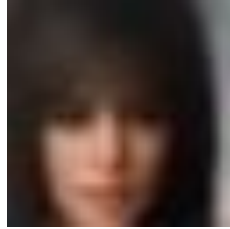
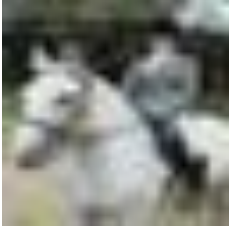

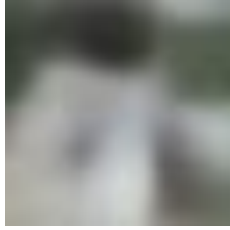
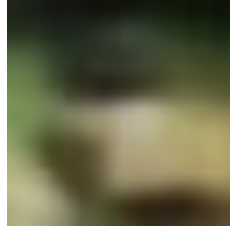
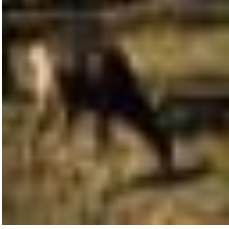
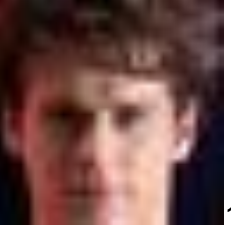
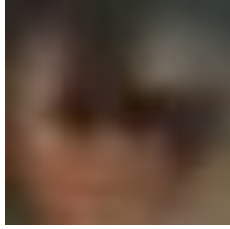
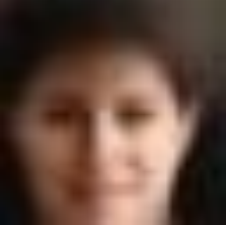
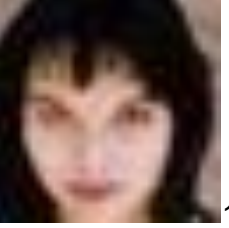

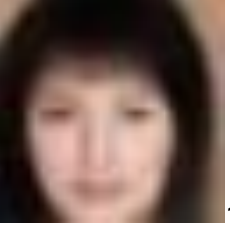
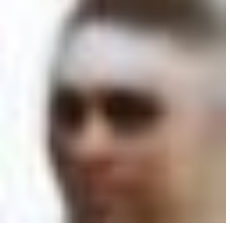


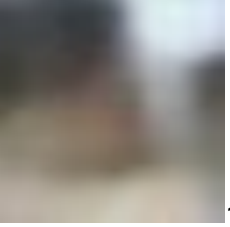
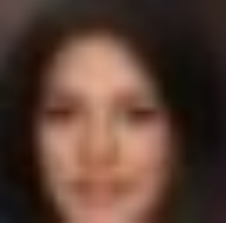
conv2d_4 (Conv2D)	(None, 16, 16, 120)	129720	conv2d_3[0][0]
conv2d_5 (Conv2D)	(None, 16, 16, 120)	129720	conv2d_4[0][0]
flatten_1 (Flatten)	(None, 30720)	0	conv2d_5[0][0]
dense_1 (Dense)	(None, 400)	12288400	flatten_1[0][0]
dense_2 (Dense)	(None, 400)	12288400	flatten_1[0][0]
lambda_1 (Lambda)	(None, 400)	0	dense_1[0][0] dense_2[0][0]
dense_3 (Dense)	(None, 30720)	12318720	lambda_1[0][0]
reshape_1 (Reshape)	(None, 16, 16, 120)	0	dense_3[0][0]
conv2d_transpose_1 (Conv2DTrans	(None, 16, 16, 120)	129720	reshape_1[0][0]
conv2d_transpose_2 (Conv2DTrans	(None, 16, 16, 120)	129720	conv2d_transpose_1[0][0]
conv2d_transpose_3 (Conv2DTrans	(None, 33, 33, 60)	64860	conv2d_transpose_2[0][0]
conv2d_6 (Conv2D)	(None, 32, 32, 3)	723	conv2d_transpose_3[0][0]
=====			
Total params: 37,545,722			
Trainable params: 37,545,722			
Non-trainable params: 0			



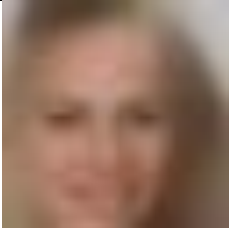
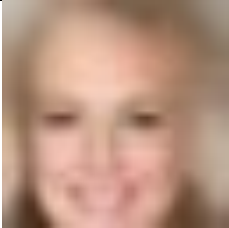
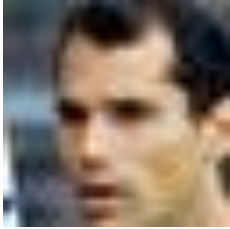
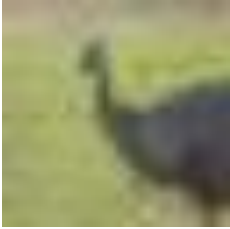
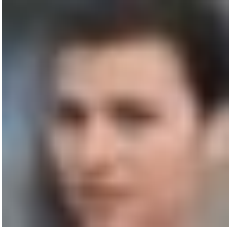
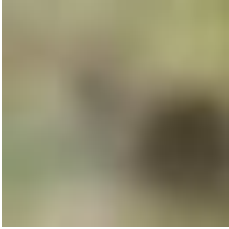


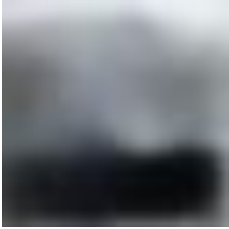
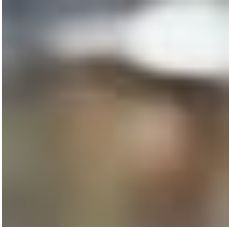
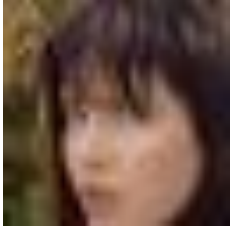
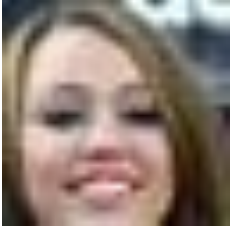
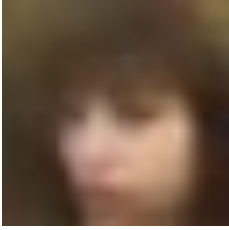
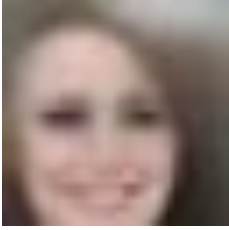
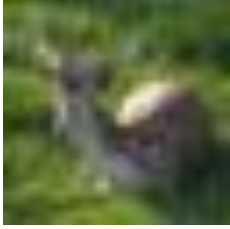

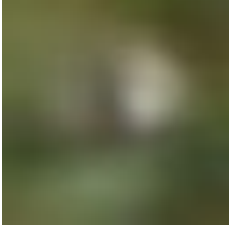
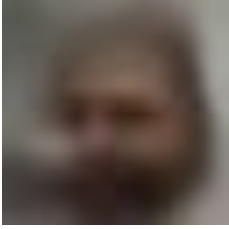
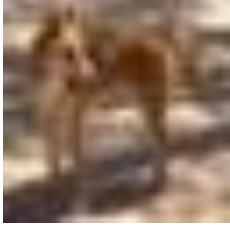
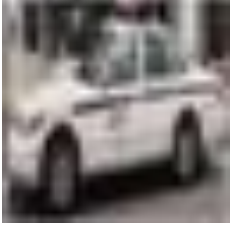
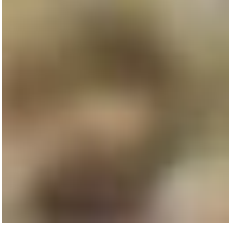
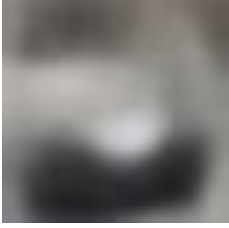

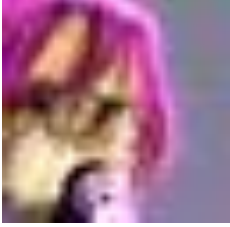
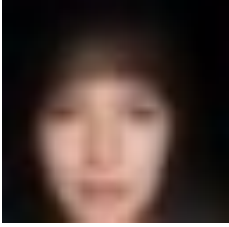
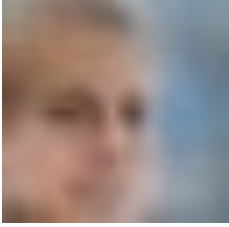
Loss function :


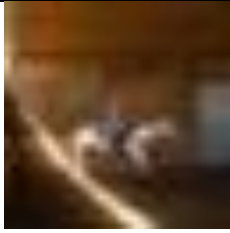
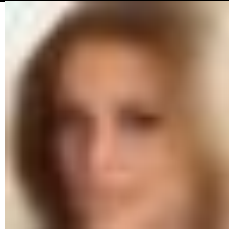
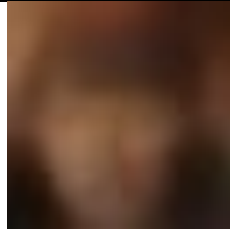
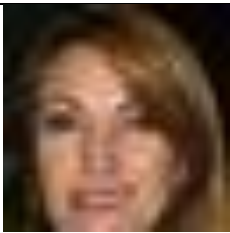

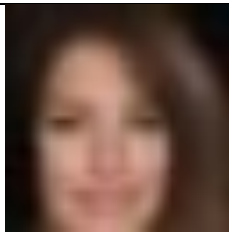
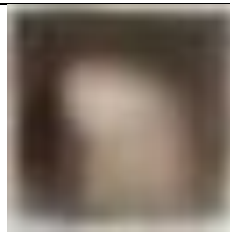
我們的 vae 中 dense1 之後分成一邊平均層(z_mean)和變異層(z_log_var)。除了原圖和復原圖的 binary cross entropy 外還有這兩層的 loss。

```
xent_loss = img_rows * img_cols * metrics.binary_crossentropy(
    K.flatten(x),
    K.flatten(x_decoded_mean_squash))
kl_loss = - 0.5 * K.sum(1 + z_log_var - K.square(z_mean) - K.exp(z_log_var),
axis=-1)
vae_loss = K.mean(xent_loss + kl_loss)
```

左側 32 張為原圖(標號 1 ~ 32)，右側 32 張為 reconstruct 圖。

 1	 2	 1	 2
 3	 4	 3	 4
 5	 6	 5	 6
 7	 8	 7	 8
 9	 10	 9	 10
 11	 12	 11	 12
 13	 14	 13	 14

 15	 16	 15	 16
 17	 18	 17	 18
 19	 20	 19	 20
 21	 22	 21	 22
 23	 24	 23	 24
 25	 26	 25	 26
 27	 28	 27	 28

 29	 30	 29	 30
 31	 32	 31	 32