Module EE5907: Pattern Recognition
Project Report - CA2

Hanaé Camille CARRIÉ
ID: A0214205X

National University of Singapore - Faculty of Engineering

Introduction

This project has been coded using the Python programming language.
The Scikit-Learn package has been used to perform the following analysis and classification: Principal Component Analysis, Linear Discriminant Analysis, Nearest Neighbors classification, Gaussian Mixture Model classification, Support Vector Machine classification.
The Pytorch package has been used to perform the Convolutional Neural Network.

The 25 CMUPIE subject numbers randomly chosen for this project out of the 68 available subjects are: 2, 3, 4, 6, 7, 9, 13, 14, 15, 28, 29, 32, 33, 35, 36, 38, 39, 44, 48, 52, 56, 58, 62, 66, 68.
My 10 selfie pictures are labeled as subject 0.
There are about 170 images per CMUPIE subject. The entire selected dataset consists of 4,254 gray scale pictures of dimension 32x32.

We applied a 70%-30% ratio split between train (2,977) and test (1,277) set.

1 Question 1: Principal Component Analysis (PCA) for Feature Extraction, Visualization and Classification

1.1 PCA based data distribution visualization

In this subsection, only 500 training images were selected for visualization purpose. We first normalize this subset of images before applying PCA.

Figure 1 shows those 500 images projected on space (2-dim plane) formed by the 2 first Principal Components (PCs) (top sub-figure) and on the space (3d-space) formed by the 3 first PCs (bottom sub-figure). Images coming from the same subject are represented by the same marker as specified in the legend. Projected points corresponding to my selfies are displayed as larger blue squares. Those plots do not exhibit clear clusters nor outliers. The first PC accounts for 43.03% of the variance while PC2 accounts for 25.61% and PC3 for 5.92% of the dataset variance.

Figure 2 shows the first 3 eigen faces learned from PCA.

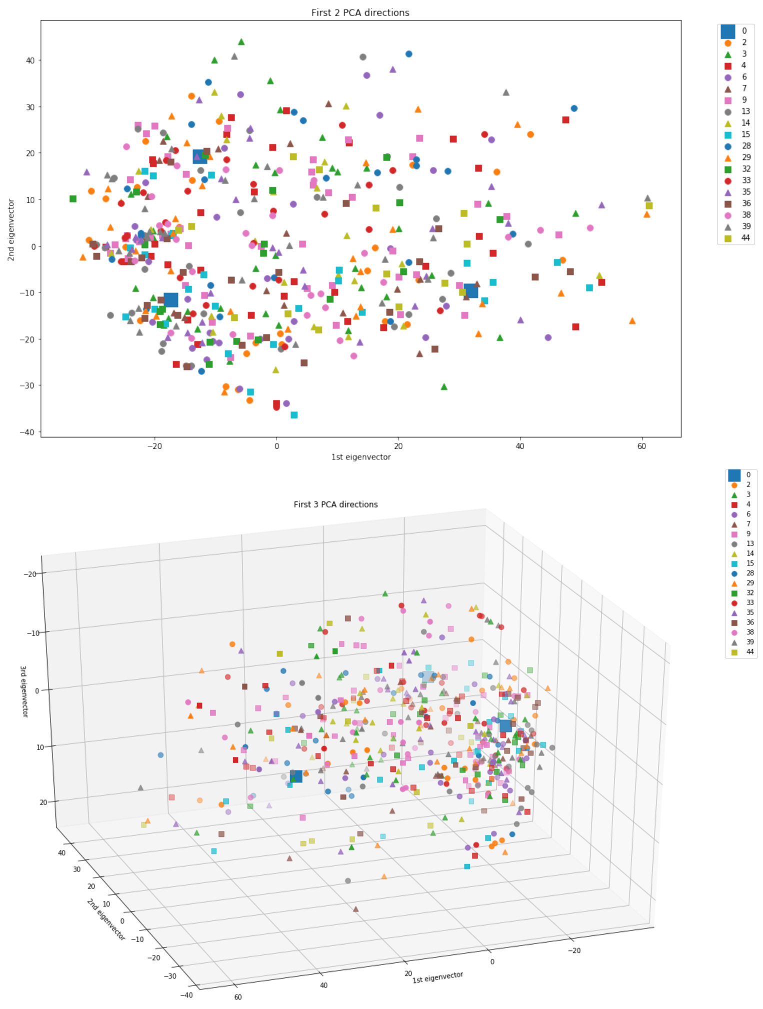


Fig. 1. Dataset projected on the 2-dim or 3-dim PCA hyperplan (PC1, PC2)

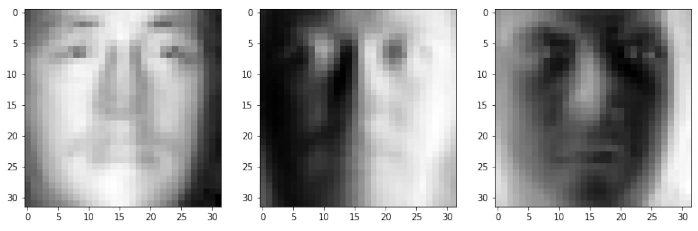


Fig. 2. Eigenfaces from PCA

1.2 PCA plus nearest neighbor classification results

In this subsection, the whole training and testing set are used. The following table shows the different classification accuracy results on the testing set using a 3-Nearest Neigboor classifier on different numbers of PCA features. We notice that the accuracy increases when the number of kept PCA features increases. Plot k. However, the accuracy on the selfies was perfect for 40 and 80 PCA features, there is one misclassification using the 200 PCA features. Let's remind that there are only 3 images in this testing set.

Table 1. Nearest Neighboor classification results using PCA projection as input.

# of first PCA dimensions kept	accuracy on CMUPIE	accuracy on selfies
40	83.75	100
80	87.21	100
200	88.46	66.67

2 Question 2: Linear Discriminant Analysis (LDA) for Feature Extraction, Visualization and Classification

2.1 LDA based data distribution visualization

In this subsection, only 500 images from the training set were selected for visualization purpose. We first normalize this subset of images before applying LDA.

Figure 3 shows those 500 images projected on space (2-dim plane) formed by the 2 first LDA Components. Figure 3 shows the projection on the space (3d-space) formed by the 3 first LDA Components. Images coming from the same subject are represented by the same marker as specified in the legend. Projected points corresponding to my selfies are displayed as larger blue squares. Those plots exhibit clearer clusters: pictures for the a same subject are quite well grouped together.

2.2 LDA plus nearest neighbor classification results

In this subsection, the whole training and testing set are used. The table 2 shows the different classification accuracy results on the testing set using a 3-Nearest Neighbor classifier on different numbers of LDA features. We notice that the accuracy increases when the number of kept LDA features increases. Plot k. However, the accuracy on the selfies is 0.

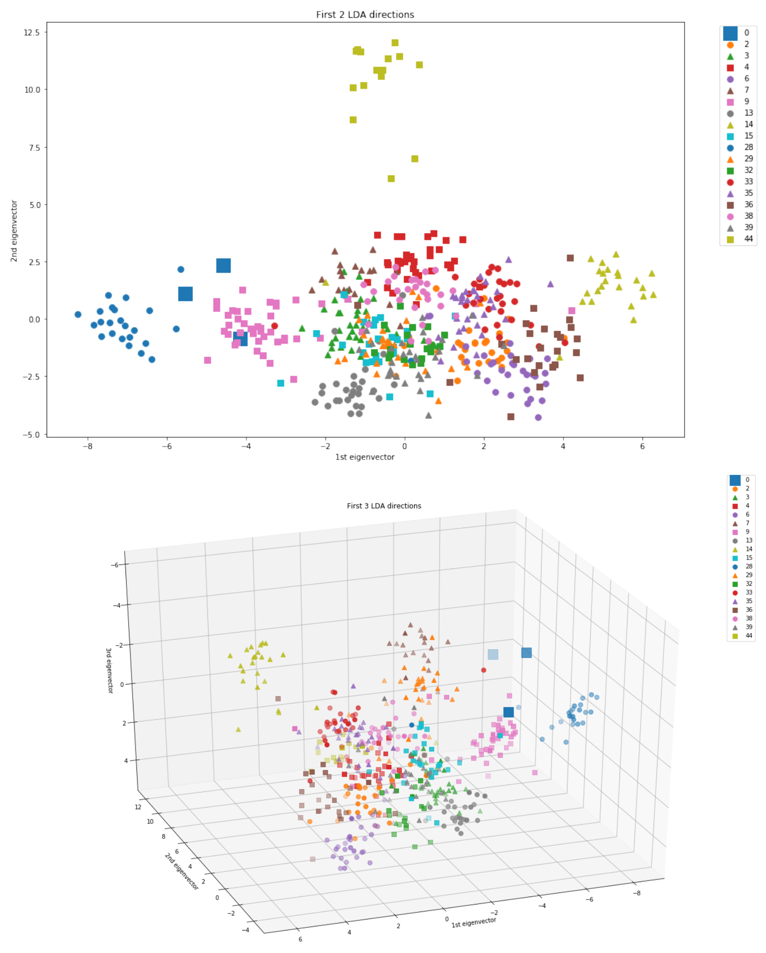


Fig. 3. Dataset projected on the 2-dimension and 3-dimension LDA hyperplan (PC1, PC2)

Table 2. Nearest Neighbor classification results using LDA projection as input.

# of first LDA dimensions kept	accuracy on CMUPIE	accuracy on selfies
2	44.74	66.67
3	64.68	100
9	92.78	33.33

3 Question 3: Gaussian Mixture Model (GMM) for clustering

To answer this question, we applied a 3-component GMM on different input dataset (raw or PCA reduced datasets) and we visualize those clusters on the

PCA plane. The 3 colors represent the 3 GMM components. The more PCA features we use, the more distinct in the 2d PCA plane the clusters are. Using the raw images, the blue and green clusters (1 and 3) are a bit overlapping. Using 80 PCA features, the orange and green clusters are a bit overlapping. Using the 200 PCA features, the 3 clusters are more distinct.

For the subjects contained in each cluster, Fig 4 show 5 frequent subjects in each GMM component for the case using raw images as input. We notice that the sample of cluster 0 contains Asian looking men . Cluster 1 contains 2 dark pictures, 2 men with mustaches and 1 Asian girl. Finally, cluster 3 contains more Western looking people, one of them wears glasses. The same figure for the other kind of inputs (PCA 80 and 200 features) is not displayed in the report as they look quite similar.

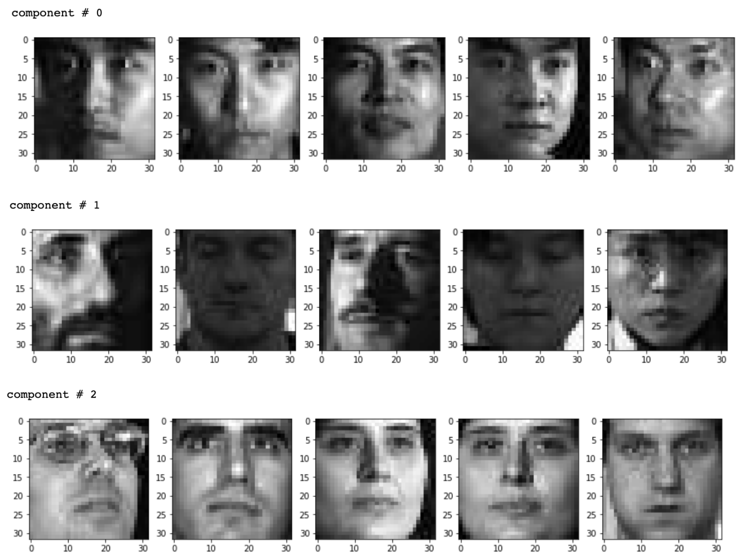


Fig. 4. Faces in each GMM cluster using the 3 different inputs (raw, 80-feature PCA, 200-feature PCA) respectively

4 Question 4: Support Vector Machine (SVM) classification results with different parameter values

In this section, using the same 3 types of input (raw, 80-PCA, 200-PCA), we train a linear SVM with different penality parameter C. We notice that the best result is given when C = 0.1 in all cases as seen in Fig 6. A surprising result is that the linear SVM performs better on the raw vectorised images.

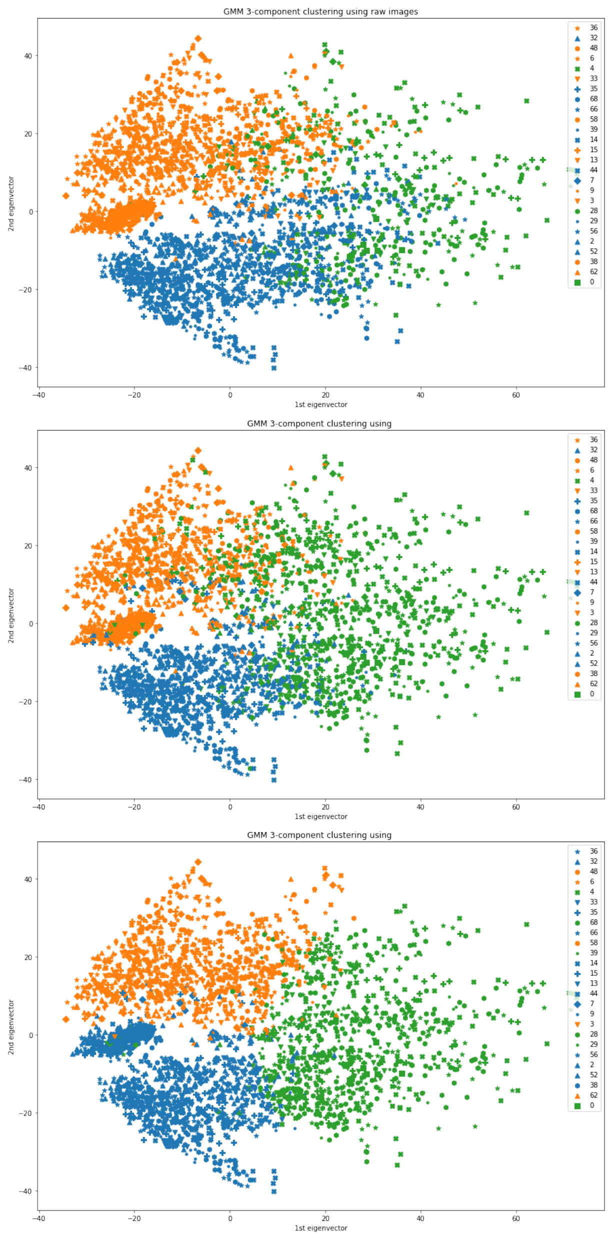


Fig. 5. GMM 3-component clustering using raw images (1st figure), using PCA 80-component feature extraction (2nd figure), using PCA 200-component feature extraction (3rd figure). Color legend: blue = GMM comp. 1, orange = GMM comp. 2, green = GMM comp. 3

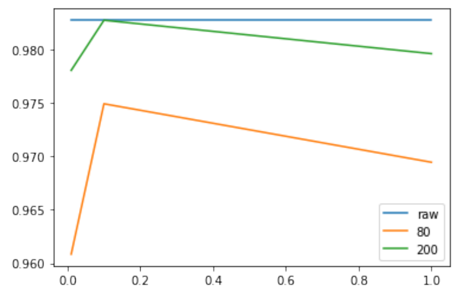


Fig. 6. SVM classification results with respect to C parameter and input nature

Table 3. SVM classification results.

input nature	accuracy with $C = 0.01$	accuracy with $C = 0.1$	accuracy with $C = 1$
raw images	98.28	98.28	98.28
80-dim PCA	96.01	97.81	97.10
200-dim PCA	97.49	98.20	97.73

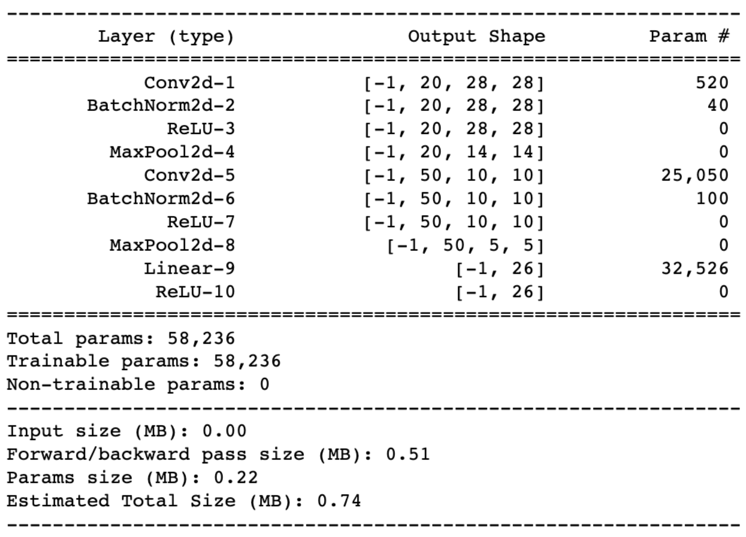
5 Question 5: Convolutional Neural Network (CNN) classification results with different network architectures

The CNN architecture is displayed in Fig 7. This architecture follows the guidelines specified in the CA2 questions except for the last linear layer which has 1,250 input parameters due to the shape of the last convolutional layer after pooling (50x5x5). I did not find a way to set this number of nodes to 500 as asked in the questions.

The following parameters are used for the training:

Table 4. CNN parameters.

parameter	value
optimizer	Stochastic Gradient Descent
learning rate	0.0001
momentum	0.90
loss	Cross-Entropy
data augmentation	No
batch size	4
train-val-test	49%-21%-30%
# epochs	25



```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1           [-1, 20, 28, 28]             520
       BatchNorm2d-2           [-1, 20, 28, 28]              40
              ReLU-3           [-1, 20, 28, 28]               0
         MaxPool2d-4           [-1, 20, 14, 14]               0
            Conv2d-5           [-1, 50, 10, 10]          25,050
       BatchNorm2d-6           [-1, 50, 10, 10]             100
              ReLU-7           [-1, 50, 10, 10]               0
         MaxPool2d-8             [-1, 50, 5, 5]               0
            Linear-9                   [-1, 26]          32,526
             ReLU-10                   [-1, 26]               0
================================================================
Total params: 58,236
Trainable params: 58,236
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 0.00
Forward/backward pass size (MB): 0.51
Params size (MB): 0.22
Estimated Total Size (MB): 0.74
----------------------------------------------------------------
```

Fig. 7. CNN model architecture and number of parameters

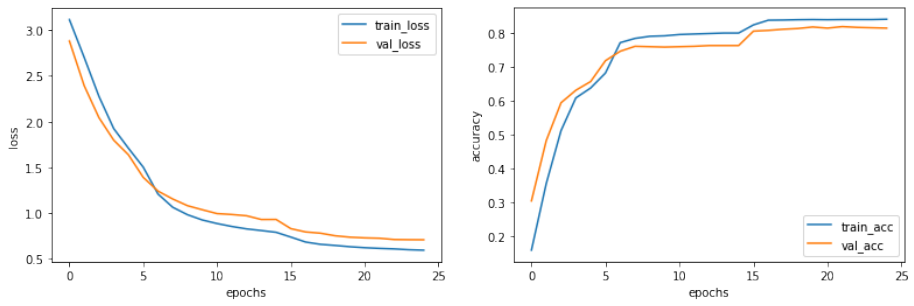


Fig. 8. CNN loss and accuracy learning curves

Table 5. CNN classification results.

input	accuracy
training	84.01
validation	81.88
testing	83.09

The results are satisfactory with only 25 epochs. The training and validation learning curves are close. The validation accuracy keeps decreasing suggesting that the model is not overfitting. The results on either sets of the data are close.

The results could be further improved by considering other hyperparameters (learning rate), data augmentation (flip, rotation, Gaussian noise) and other architectures.

Conclusion

Finally, the table below summarizes the best testing accuracy obtained using different classifiers:

Table 6. Summary of classification results on the full test set.

method	testing accuracy
PCA	88.72
LDA	93.03
SVM	98.28
CNN	83.09