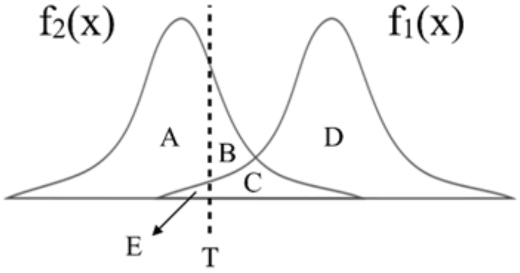
# Computer Vision: from Recognition to Geometry
# HW2

Name: 蔡昕宇 Department: 電機四 Student ID: B04901096

Problem 1

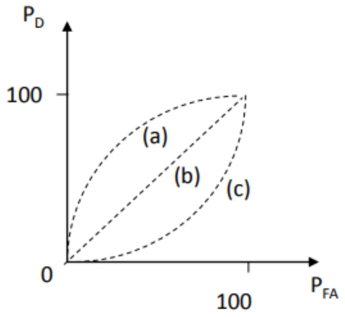
(a) Assume $X$ is a continuous random variable that denotes the estimated probability of a binary classifier. The instance is classified as positive if $X > T$ and negative otherwise. When the instance is positive, $X$ follows a PDF $f_1(x)$. When the instance is negative, $X$ follows a PDF $f_2(x)$. Please specify which regions (A ~ E) represent the cases of *False Positive* and *False Negative*, respectively. Clearly explain why.



$T$ as a threshold for classification. *False Positive (False Alarm)* is an error where it indicates presence of a condition when the condition does not present actually. On the other hand, *False Negative (False Reject)* is a situation that it fails to detect the condition where in reality it does exists. Take the figure above as example, when $X$ follows $f_2(x)$, the area $B + C$ is misclassification. Similarly, when $X$ follows $f_1(x)$, the area $E$ has not been classified correctly. Hence, according to the discussion,

$$False\ Positive: B + C$$
$$False\ Negative: E$$

(b) There are three ROC curves in the plot below. Please specify which ROC curves are considered to have reasonable discriminating ability, and which are not. Also, please answer that under what circumstances will the ROC curve fall on curve (b)?



Two distributions are randomly distributed. If distributions are not separated but not totally same, the ROC curve will be (a). If two distributions are identical, the ROC curve will (b). There is no possibility that the ROC curve will be (c). Hence,

$$Reasonsable: (a) 、 (b)$$

$$Not\ reasonable: (c)$$

If distributions are identical, ROC curve follows (b).

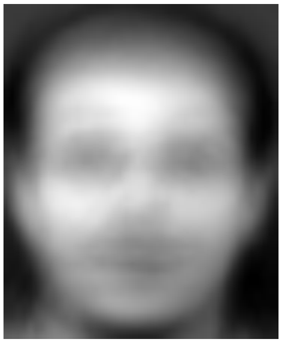
## Problem 2

(a) PCA

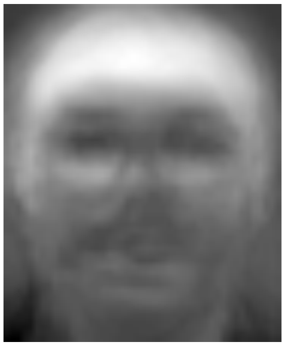
In this task, you need to implement PCA from scratch, which means you cannot call PCA function directly from existing packages.

1. Perform PCA on the training data. Plot the mean face and the first five eigenfaces and show them in the report.

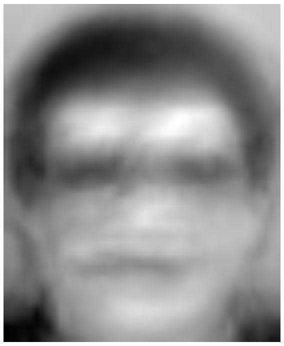
mean face



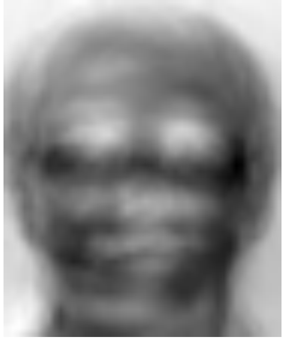
eigenface_1



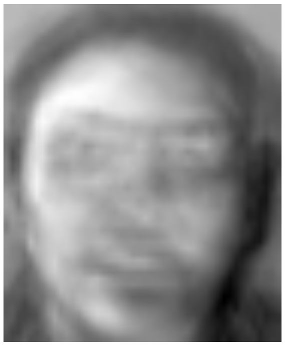
eigenface_2



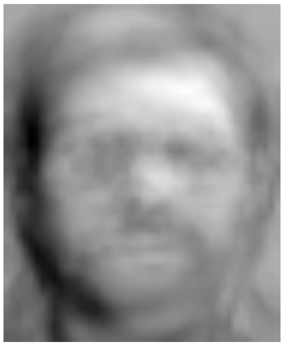
eigenface_3



eigenface_4

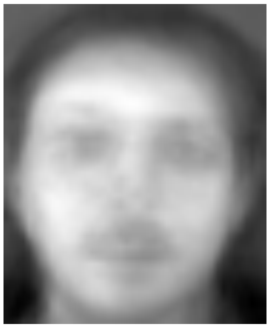


eigenface_5



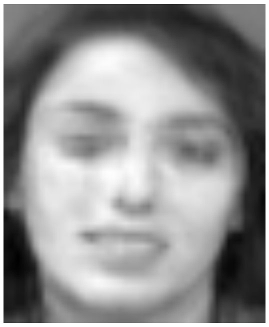
2. Take *person*$_8$*_image*$_6$, and project it onto the above PCA eigenspace. Reconstruct this image using the first n = { 5, 50, 150, all } eigenfaces. For each n, compute the mean square error (MSE) between the reconstructed face image and the original *person*$_8$*_image*$_6$. Plot these reconstructed images with the corresponding MSE values in the report.

n = 5



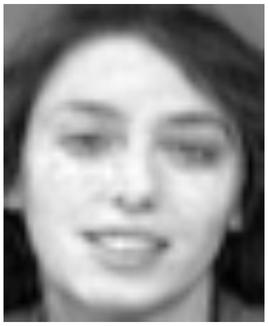
MSE ≈ 693.702

n = 50



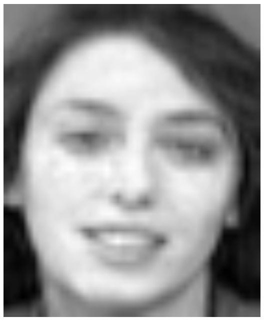
MSE ≈ 119.2

n = 150



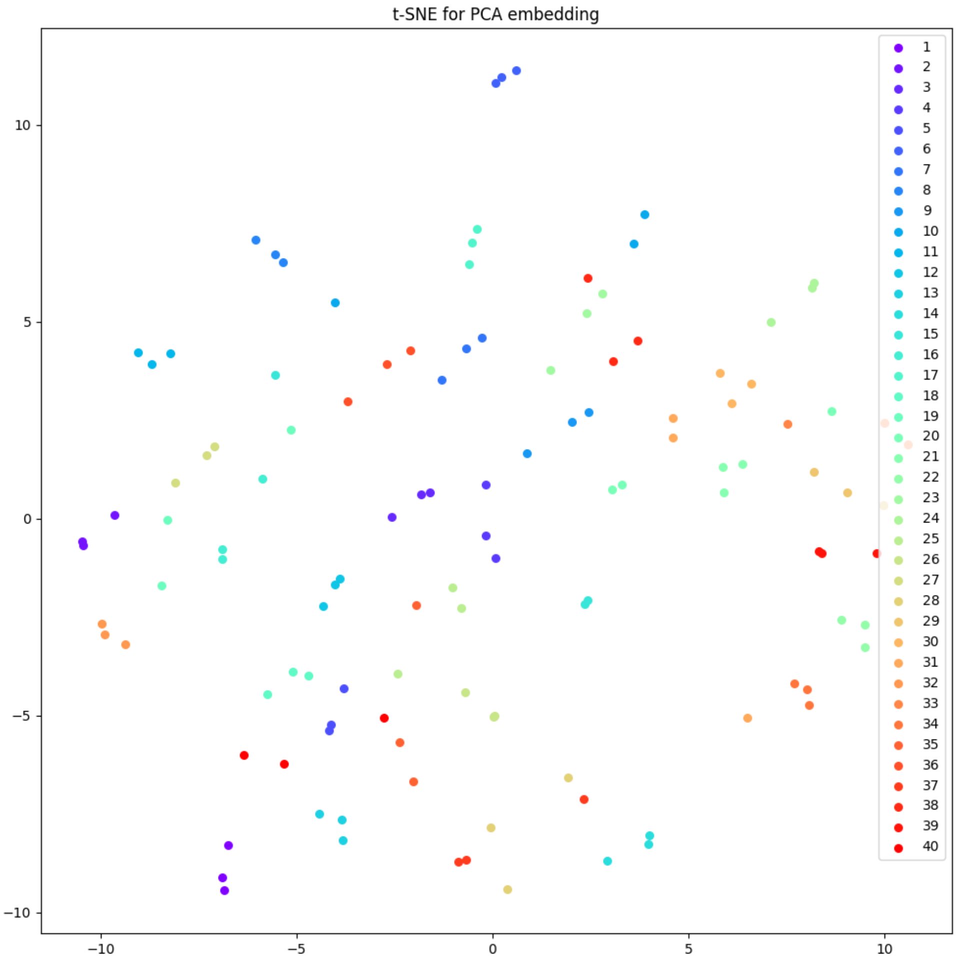
MSE ≈ 40.397

n = all



MSE ≈ 5.306e-27 (~0)

3. Reduce the dimension of the image in testing set to dim = 100. Use t-SNE to visualize the distribution of test images.
   40 classes, 3 images in each classes

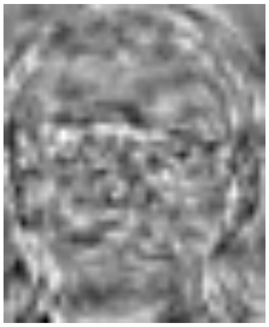


(b) LDA

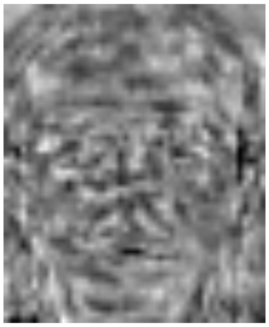
In this task, you need to implement LDA from scratch, which means you cannot call LDA function directly from existing packages.

1. Implement LDA and plot first 5 Fisherfaces.

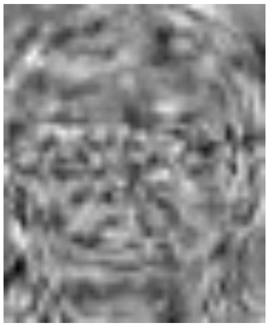
Fisherface_1



Fisherface_2



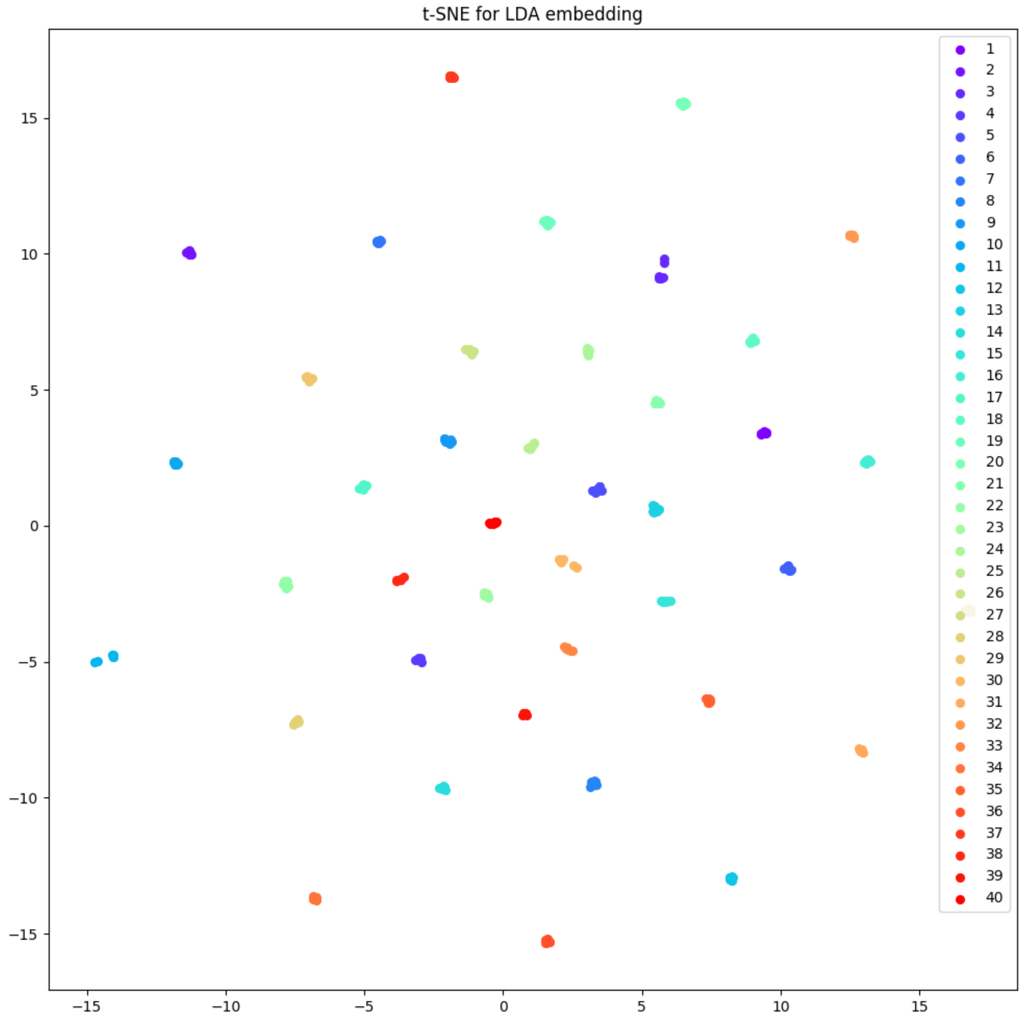
Fisherface_3



Fisherface_4

Fisherface_5

2. Use t-SNE to visualize the distribution of the projected testing data, which has the dimension of 30.



We can observe that LDA has better classification ability on training set, since each embedded vectors between classes are widely separated, and the vectors belonging to same class are really close. This is the result that we expect LDA will do, but it may overfit on training set, where we discuss in next question.

(c) To apply the k-nearest neighbors (k-NN) classifier to recognize the testing set images, please determine the best k and n values by 3-fold cross-validation.

For simplicity, the choices for such hyper-parameters are:

k = {1, 3, 5} and n = {3, 10, 39}.

Please show the cross-validation results and explain your choice for (k, n). Also, show the recognition rate on the testing set using your hyper-parameter choice. Please apply the above comparing method on both PCA and LDA.

Do you observe an improved recognition rate using fisherfaces (compared to eigenfaces obtained by PCA)? If so (or if not), what might be the possible explanation?

Apply k-NN on embedded vectors of training set after PCA

| k / n | 3 | 10 | 39 |
|---|---|---|---|
| 1-nearest | 0.700/0.663/0.688 | 0.867/0.863/0.825 | 0.933/0.938/0.900 |
| 3-nearest | 0.542/0.625/0.613 | 0.725/0.788/0.725 | 0.808/0.888/0.850 |
| 5-nearest | 0.433/0.513/0.600 | 0.683/0.688/0.663 | 0.742/0.800/0.788 |

The hyperparameters (k, n) = (1, 10), (1, 39) have good results in average. Take either of them into consideration and apply on the testing set.
Classifying on testing set with (k, n) = (1, 10), recognition rate is 94.17% (113/120).
Classifying on testing set with (k, n) = (1, 39), recognition rate is 95.83% (115/120).
Two pairs of hyperparameters' results are close, (k, n) = (1, 39) leads to a better result.

Apply k-NN on embedded vectors after LDA

| k / n | 3 | 10 | 39 |
|---|---|---|---|
| 1-nearest | 1.000/1.000/1.000 | 1.000/1.000/1.000 | 1.000/1.000/1.000 |
| 3-nearest | 1.000/1.000/1.000 | 1.000/1.000/1.000 | 1.000/1.000/1.000 |
| 5-nearest | 1.000/1.000/1.000 | 1.000/1.000/1.000 | 1.000/1.000/1.000 |

Cross validation of each hyperparameter (k, n) reaches accuracy = 100% result. We then take compare all of them in testing set.
Testing Accuracy (LDA):

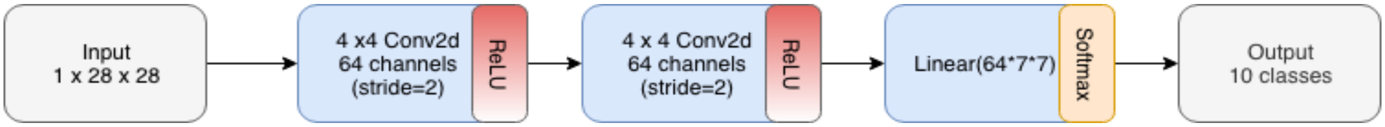
| k / n | 3 | 10 | 39 |
|---|---|---|---|
| 1-nearest | 0.38333 | 0.80833 | 0.91667 |
| 3-nearest | 0.39167 | 0.81677 | 0.91677 |
| 5-nearest | 0.39167 | 0.81677 | 0.91677 |

Classifying on testing set with (k, n) = (1, 39), (3, 39), (5, 39) has same result with accuracy 0.91677 (110/120)

Comparing the results of PCA and LDA, although LDA has good results in training set (i.e. better classification ability, which can be seen in t-SNE result), it does not imply the results in the testing set will be better, since the validation set has been used for the dimension reduction process, it may have some overfitting issues during cross validation, which may result in bad outcomes on testing set. We can obtain that the result of LDA is worse than that of PCA in testing set.

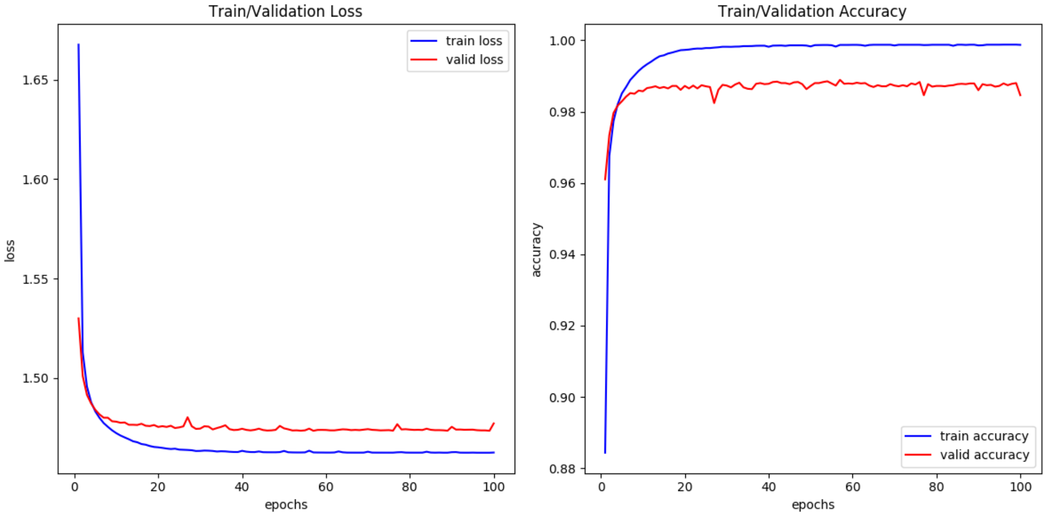
Problem 3

(a) Build a CNN model and train it on the given dataset. Show the architecture of your model in the report.



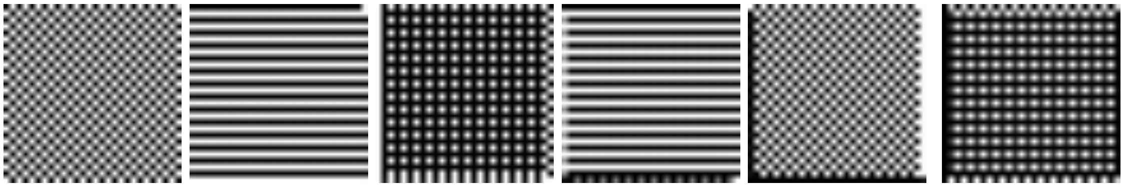
2 Convolution layers and 1 fully connected layer for this task.

(b) Report your training / validation accuracy, and plot the learning curve (loss, accuracy) of the training process.
Training Accuracy: 0.998701
Validation Accuracy: 0.988900

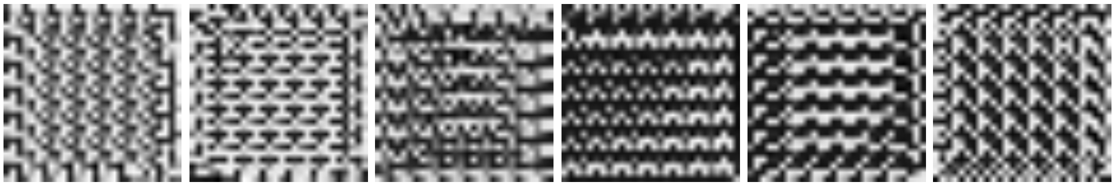


Training accuracy reaches 0.999 and validation accuracy is about 0.985.

(c) Visualize at least 6 filters on both the first and the last convolutional layers.
First convolution layer:



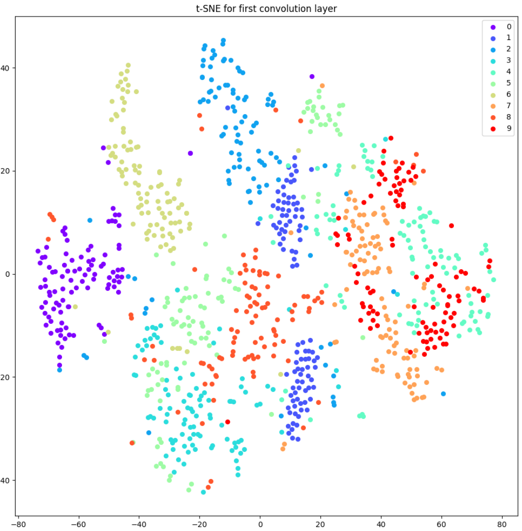
Last convolution layer



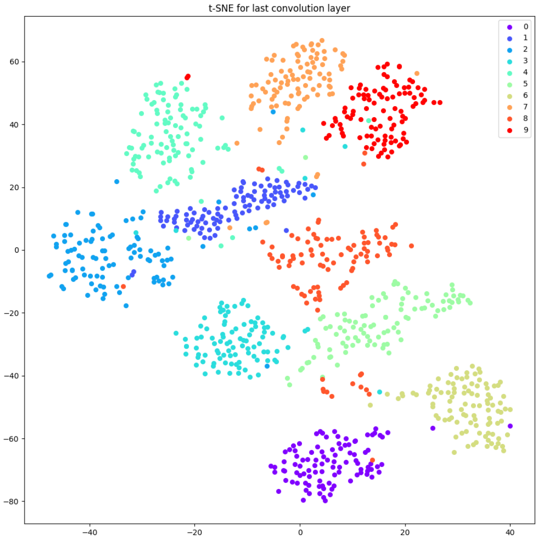
We implement gradient ascent for visualizing filters. The first convolution layer seems to represent straight lines or some simple shapes, while the last convolution layer seems to represent local features and some specific points.

(d) Visualize high-level and low-level features of 1000 validation data (100 for each class) extracted from different layers, and explain what you have observed from the two t-SNE plots.

Low-level features



High-level features



From two plots above, we can observe that the features in the first convolution layer does not separate widely, some are mixed together. However, in the last convolution layer, it seems to make those features separate widely. That is why it can provide fully connected layer to classify those features more easily. Higher level features may represent characteristics we expected that makes the model classify the image. As the result, we got the accuracy over 98% on validation set.