EE5907R Pattern Recognition
Face Recognition Project I

Swee King Phang, `A0033585A`

NUS Graduate School for Integrative Sciences and Engineering

National University of Singapore (NUS)

Singapore

Abstract

In this project, an interesting pattern recognition problem, human face recognition is going to be investigated. The simplified version of the CMU Pose, Illumination, and Expression (PIE) database, which contains only five near frontal poses (C05, C07, C09, C27, C29) will be utilized. Two main classification methods, namely the *k-Nearest-Neighbor* and *Bayes classifier* will be used to classify each human face expressions. Both the methods will then be improved by incorporating some feature extraction methods, such as *Principal Component Analysis* and *Linear Discriminant Analysis*. Their performance will be compared and discussed in detail.

1 Introduction

Pattern recognition, or more specifically in this project, face recognition has greatly attracted interests from many research institutes, due to its usefulness and fanciness. Recently in 2010, the famous social network service — Facebook, has incorporate an efficient face recognition algorithm in tagging photos to replace the traditional manual tagging [1]. Also, affordable laptop such as Alienware from Dell Inc., has also embedded the face recognition feature in determining the correct user of the laptop [2]. These examples show the importance of face recognition in the current technology world.

In this project, CMU Pose, Illumination, and Expression (PIE) database will be utilized. The CMU PIE database consists of 41,368 images of 68 people. Each person is under 13 different poses, 43 different illumination conditions, and with 4 different expressions [3]. For this project, a simplified version of the database which only contains five near frontal poses (C05, C07, C09, C27, C29) is used. There will be about 170 images for each individual. In addition, all the images have been resized to 32×32 pixels. The dataset in MATLAB format was provided in this course, and it contains variables **fea** and **gnd** as face vector and label. Some example face images are shown below.

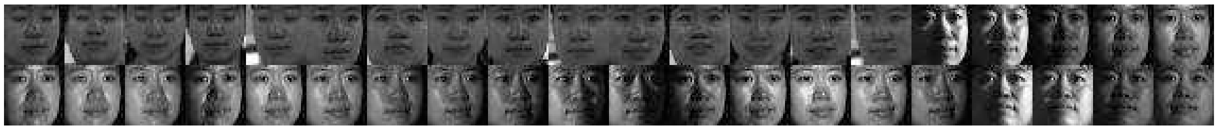

Figure 1: Sample face images from the simplified CMU PIE

In this project, a random subset of 100 images per subject was taken with labels to form the training set, and the rest of the database was considered to be the test set. There are 50 random splits, specified by variables **testIdx** and **trainIdx**. The resulting splits are available in the course's IVLE workbin.

The following face recognition analysis are required in this project:

1. Pre-process the dataset (PIE_32x32.mat) by normalizing each face image vector to unit (i.e., dividing each vector by its magnitude).
2. Design a k-Nearest-Neighbor (k-NN) classifier using the training set and evaluate its performance on the test set. Since there are 50 different splits between the training set and test set, 50 sets of results will be obtained. Report the average error rate and the standard deviation over the 50 splits. Note that different distance measures and different values of k can be experimented.
3. Design a Bayes classifier using the training set and evaluate its performance on the test set. Report the average error rate and the standard deviation over the 50 splits. State clearly all the assumptions made.
4. Improve the accuracy of the above two classifiers (i.e., k-NN and Bayes) by using at least one of the following: illumination normalization, pose alignment, and feature selection (e.g., PCA, LDA, and their variants). Report the average error rate and the standard deviation.

As mentioned in the requirement above, there are 2 types of classification methods to be performed — the k-NN and the Bayes classification. The project is organized as follows: In Chapter 2, different value of k for k-NN classification will first be presented, follows by two other ways of calculate distance, i.e., 1-norm and ∞-norm. In Chapter 3, Bayes classifier will be investigated and the result will be compared to the k-NN classifier result obtained earlier. In the next two chapters, two feature extraction method, namely the Principal Component Analysis and the Linear Discriminant Analysis will be performs. They represented the unsupervised and supervised feature extraction respectively. Finally, conclusion remarks will be made in Chapter 6.

2 Normalization of Face Vector

Note that before any classification procedures are carried out, the face images will be normalized so that they can be fairly compared. This is a common practice for all image processing and pattern recognition related problems. In this project, every row vector of **fea** will be scaled down such that the resulting row vectors will have a 2-norm of 1.

3 Design and Implementation of k-Nearest-Neighbor Classifier

The k-Nearest-Neighbor (k-NN) classifier is a very intuitive method that classifies unlabeled examples based on their similarity with examples in the training set. Its working principle is such that for a given unlabeled data, k number of closest labeled examples in the training data set will be identified. This unlabeled data will then be assigned to the class that appears most frequently within these k training data.

The advantages of using k-NN classifier are obvious:

1. It is analytically tractable;
2. It is very simple to be implemented;
3. It uses local information, as such it has very high adaptive behavior;
4. It gives near optimal result in the event of large sample size;
5. It is easy to perform parallel implementations to increase efficiency.

It has, however, a few disadvantages due to its nature, such as large storage requirements, computational intensive, and very highly susceptible to the curse of dimentionality.

For the ease of implementation, a guideline to implement the classifier in any programming language is listed below [4]:

1. Calculate the Euclidean distances between the testing data and each of the training data;
2. Select the k training data that gives shortest distance to the testing data;
3. Find the class that repeated most in the k selected training data;
4. The testing data is belong to the identified class.

3.1 Results: 1-NN vs k-NN

In the simulation, several number of k value is evaluated. The source code for MATLAB simulation can be found in the later chapter. Error rate of each run (of a total 50 sets) is obtained, and tabulated. The mean and standard deviation of the error rate for each k value is calculated as shown in the table below:

k	Mean	Standard Deviation
1	0.0607	0.003922
3	0.1203	0.00516
5	0.1985	0.005533
10	0.2693	0.008123
25	0.3269	0.009052
50	0.3639	0.011379
75	0.3738	0.011302
82	0.3750	0.011351
83	0.3820	0.011183
100	0.3933	0.01136

In order to observe the trend more closely, the plot of each parameters are shown below:

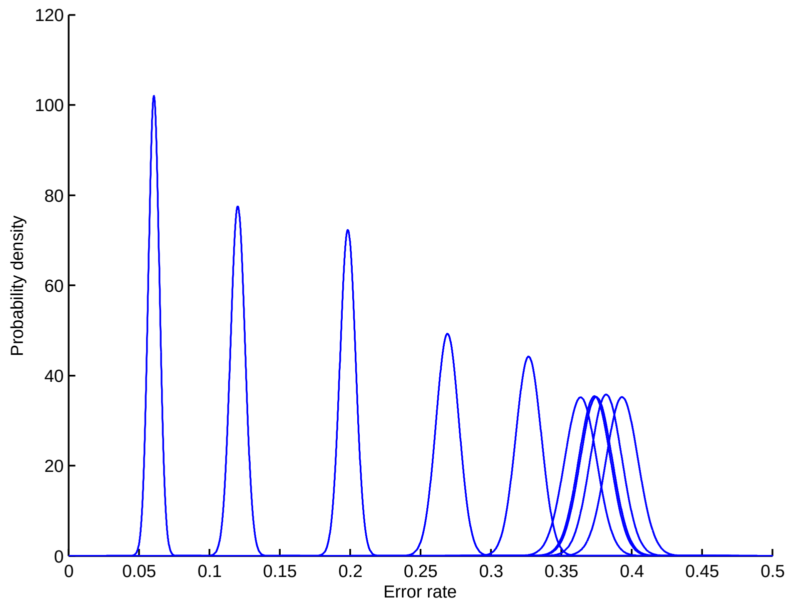

Figure 2: Probability density of error rate for various k value

Note that the error rates are assumed to follow Gaussian distribution in this plot, and the Gaussian curve from left to right are respectively correspond to $k = \{1, 3, 5, 10, 25, 50, 75, 82,\ 83, 100\}$. We see that $k = 1$, also the so called 1-NN has the best performance with mean error rate of 6.07%, and the error rate increases with the increase of value of k. This face recognition problem has shown that although large values of k is having advantages of yielding smoother decision regions and provides more intuitive probabilistic information, the large values of k is, however, prone to destroys the locality of the estimation. Since the training data and the testing data are all taken from the frontal view of human face, smaller value of k seems to be a better choice to preserve the locality of the estimation. Also, it is suspected that the down-sampled image resolution (to 32×32 pixels) has greatly reduced the feature of the images, and hence higher value of k is not suitable in this case.

3.2 Results: 1-norm vs 2-norm vs ∞-norm

The above analysis was done by measuring the distance between the training data and the testing data using Euclidean 2-norm distance. In this section, two other type of distance calculation is proposed, namely the 1-norm and the ∞-norm distance.

The k-NN algorithm with $k = 1$ is implemented with different distance type, and the mean and standard deviation of the error rate is tabulated below:

Norm	Mean	Standard Deviation
1	0.0586	0.004357
2	0.0607	0.003922
∞	0.1868	0.005089

From the results obtained, one can notice that by using 1-norm method, the error rate of the face recognition has improved slightly. However, ∞-norm has shown a very undesired result of approximately 18% error rate. It is understand that ∞-norm method takes the highest value among all the individual pixel distance, and thus it is very susceptible to noises. For example, a noise in a single pixel of the image would produce a large variation in calculating its ∞-norm with other images. Therefore we can conclude that ∞-norm is not a good measurement here for face recognition.

4 Design and Implementation of Bayes Classifier

A Bayes classifier is a probabilistically optimal classifier based on the Bayes' theorem [4]. For example in this project, there are 68 states of nature (68 individual human subjects) assigned as $\{\omega_1, \omega_2, ..., \omega_{68}\}$, hence there will be 68 possible actions $\{\alpha_1, \alpha_2, ..., \alpha_{68}\}$ for a testing image with feature x. Now, define $\lambda(\alpha_i|\omega_j)$ as the loss incurred for taking action α_i when the state of nature is ω_j, then the conditional risk of taking action α_i provided that x occurs is

$$R(\alpha_i|x) = \sum_{j=1}^{j=68} \lambda(\alpha_i|\omega_j)P(\omega_j|x). \tag{1}$$

Obviously, the main task is now to reduce the overall risk. To minimize the risk

$$R = \int R(\alpha(x)|x)p(x)dx, \tag{2}$$

one needs to minimize each individual risk $R(\alpha(x)|x)$ for every x. In other words, among all possible actions $\{\alpha_1, \alpha_2, ..., \alpha_{68}\}$, The action α_i is selected subject to $R(\alpha_i|x)$ is minimized. If a zero-one loss function is assumed,

$$\lambda(\alpha_i, \omega_j) = \begin{cases} 0 & i = j \\ 1 & i \neq j \end{cases}, \tag{3}$$

then the conditional risk can be written as

$$\begin{aligned} R(\alpha_i|x) &= \sum_{j=1}^{j=68} \lambda(\alpha_i|\omega_j)P(\omega_j|x) \\ &= \sum_{j\neq i} P(\omega_j|x) \\ &= 1 - P(\omega_i|x). \end{aligned} \tag{4}$$

We can see that now the problem of minimizing risk has transformed into a problem of maximize *posteriori* $P(\omega_i|x)$. Hence, ω_i will be assigned to the testing data if

$$P(\omega_i|x) > P(\omega_j|x) \quad \forall\ j \neq i. \tag{5}$$

The above is refer to as the Maximum *A Posteriori* (MAP) criterion. To further simplify the problem, we consider the Bayes Rule

$$P(\omega_i|x) = \frac{P(x|\omega_i)P(\omega_i)}{P(x)}. \tag{6}$$

Since the evidence $P(x)$ is same for all classes, and the prior $P(\omega_i)$ for all 68 human subjects is equal given in the project description, the decision strategy is thus transformed to decide ω_i if

$$P(x|\omega_i) > P(x|\omega_j) \quad \forall\ j \neq i. \tag{7}$$

The above criterian is the Maximum Likelyhood (ML) criterion. To calculate $P(x|\omega_i)$, a practical approach is to assume Multivariate Gaussian Distribution for each class ω_i, i.e.,

$$P(x|\omega_i) = \frac{1}{(2\pi)^{d/2}|\Sigma_i|^{1/2}} \exp\left[-\frac{1}{2}(x-\mu_i)^t\Sigma_i^{-1}(x-\mu_i)\right], \tag{8}$$

where d is the dimension of the feature vector (1024 in this project), μ_i is the mean of the i-th class and Σ_i is the covariance matrix of the i-th class.

An easier way to obtain the discriminant function is to take the natural logarithm of $P(x|\omega_i)$ to deal with the exponential term in the Gaussian distribution expression. The discriminant functions can thus be written as

$$\begin{aligned} g_i(x) &= \ln P(x|\omega_i) && (9) \\ &= -\frac{1}{2}(x-\mu_i)^t\Sigma_i^{-1}(x-\mu_i) - \frac{d}{2}\ln 2\pi - \frac{1}{2}\ln|\Sigma_i|. && (10) \end{aligned}$$

Note that in this project, as the sample size for each human subject is much lower than the feature size (100 vs 1024), individual covariance matrices for each class will be near singular. Thus, we assumed the covariance matrices for each classes are identical, and it will be estimated with all training data (6800). Since $\frac{d}{2}\ln 2\pi$ and $\ln|\Sigma_i|$ are constants and common positive ratio does not matter, the discriminant functions can be reduced to

$$g_i(x) = -(x-\mu_i)^t\Sigma^{-1}(x-\mu_i). \tag{11}$$

The remaining problem now is to estimate μ_i and Σ_i for $i = 1, 2, ..., 68$ based on the training data. According to [4], for the Maximum-Likelyhood (ML) Estimation for the Multivariate Gaussian case with both μ and Σ unknown, the estimated mean for the i-th class

$$\hat{\mu}_i = \frac{1}{n_i} \sum_{k=1}^{n_i} x_{ik}, \tag{12}$$

and the estimated covariance matrix is

$$\hat{\Sigma} = \frac{1}{n} \sum_{k=1}^{n} (x_k - \hat{\mu})(x_k - \hat{\mu})^t, \tag{13}$$

where n_i is the sample size of the i-th class, x_{ik} is the k-th training data that belongs to the i-th class, n is the total sample size, and x_k is the k-th training data.

Finally, by utilizing all the training data to estimate the mean μ_i and covariance matrix Σ for each class i and substituting the current testing data x into Eqn. 11, we will obtain different $g_i(x)$ values for different i. The testing data is then classified into the i-th class that has the largest $g_i(x)$ value.

For the ease of implementation, a guideline to implement Bayes classifier is shown below:

1. Divide the training data into 68 groups;
2. For each group, calculate the mean vector μ_i;
3. Calculate the covariance matrix Σ by utilizing all the training data;
4. Use μ_i and Σ to estimate a Gaussian probability distribution for each class;
5. Compute $g_i(x)$ by substituting feature x of testing data into Eqn. 11;
6. Choose the largest $g_i(x)$ as the classified group for testing data x.

4.1 Assumptions

4 main assumptions are made

1. The loss function of taking action α_i while the state of nature is ω_j is a zero-one loss function;
2. The priors of all 68 classes, $P(x)$, are the same;
3. All classes follow Multivariate Gaussian Distributions;
4. All classes has the same covariance matrix.

4.2 Results of Bayes Classifier

The face recognition result by using Bayes classifier is listed in the table below. Note that the mean and standard deviation of the error rate is obtained from 50 different set of experiments. It is also compared to the k-NN classification method discussed in the previous chapter.

Classification Method	Mean	Standard Deviation
Bayes	0.0478	0.002667
1-NN with 1-norm	0.0586	0.004357
1-NN with 2-norm	0.0607	0.003922

It is obvious that Bayes classifier outperforms then k-NN method. In terms of efficiency, Bayes classifier also shown a much better result, while k-NN classifier requires longer time to be computed. In the next two chapters, both the classification method will be improved by performing feature extraction before the classification work.

5 Unsupervised Feature Extraction with Principal Component Analysis

In this chapter, the unsupervised feature extraction method — Principal Component Analysis (PCA) is proposed to increase the efficiency of the k-NN and Bayes classifiers. In general, PCA is utilized to reduce the dimension of the feature, at the same time minimize the information loss, such that computational cost would be minimized.

Most pattern recognition techniques may not be effective for high-dimensional data as there are problems such as the curse of dimentionality [5]. Most of the time, the intrinsic dimension of images might be small, and therefore not every features are important. Also, feature extraction saves storage space especially for k-NN classifier.

PCA reduces the dimension of a data set by finding a new set of variables that is smaller than the original set. The new set of variables are called the principal components (PCs), and they are a series of linear least squares fits to a sample set, each orthogonal to all the previous ones. In this project, to compute the first principal component of the sample, we have the linear transformation

$$z_1 = a_1^T x = \sum_{i=1}^{68} a_i x_i, \tag{14}$$

where the vector $a_1 = (a_{1,1}, a_{2,1}, ..., a_{68,1})^T$ is chosen such that $var[z_1]$ is maximum.

To find a_1, we have

$$\begin{aligned} var[z_1] &= \frac{1}{n}\sum_{i=1}^{n}(a_1^T x_1 - a_1^T \bar{x})^2 \\ &= a_1^T S a_1, \end{aligned} \tag{15}$$

where $S = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})(x_i - \bar{x})^T$ is the covariance matrix, and $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ is the mean. To find a_1 that maximizes $var[z_1]$ subject to $a_1^T a_1 = 1$, let λ be a Lagrange multiplier,

$$\begin{aligned} L &= a_1^T S a_1 - \lambda(a_1^T a_1 - 1) \\ \frac{\delta}{\delta a_1} L &= S a_1 - \lambda a_1 = 0 \\ (S - \lambda I)a_1 &= 0. \end{aligned} \tag{16}$$

We obtain a_1 as an eigenvector of S correspond to the largest eigenvalue $\lambda = \lambda_1$.

Similarly, a_2 will be an eigenvector of S whose eigenvalue $\lambda = \lambda_2$ is the second largest. In general,

$$var[z_k] = a_k^T S a_k = \lambda_k, \tag{17}$$

where λ_k is the k-th largest eigenvalue of S. The transformation matrix to transform high dimensional feature to low dimensional feature will then be

$$\Theta = [a_1, a_2, ..., a_p], \tag{18}$$

for p is the desired dimension for the low-dimension-feature.

To ease the implementation of PCA, the following guideline provides a step by step approach to design PCA:

1. Form the covariance matrix, S;
2. Compute eigenvectors of S;
3. Use the first p eigenvectors, which correspond to p biggest eigenvalue to form the transformation matrix, $\Theta = [a_1, a_2, ..., a_p]$;
4. Perform classification by using the reduced size feature, $y = \Theta^T x$.

5.1 Results from PCA with 1-NN and Bayes Classifiers

PCA algorithm has been implemented and simulated to obtained a low dimension feature. Here the number of feature retained is set as 100. The low-dimension-feature is then classified using the previous discussed classifier — the k-NN ($k = 1$) classifier and the Bayes classifier. The result is tabulated below, with the mean and standard deviation of 50 experiments:

Classification Method	Mean	Standard Deviation
Bayes	0.0478	0.002667
Bayes with PCA	0.8946	0.005876
1-NN	0.0607	0.003922
1-NN with PCA	0.2921	0.007437

As we can observed, the performance of such feature extraction is terrible, for both Bayes and k-NN classifier. It shows that PCA feature extraction has its limitation, as the projection axes chosen by PCA might not provide good discrimination power, since it is unsupervised feature extraction. However, the speed of computing the k-NN and Bayes classifiers by adopting PCA feature extraction has greatly increased, by about 10 times faster. In the next chapter, a supervised feature extraction method, LDA is proposed, as to compare to this PCA method.

6 Supervised Feature Extraction with Linear Discriminant Analysis

As previously shown, the PCA method has a disadvantage of missing class information while extracting feature. In this chapter, the Linear Discriminant Analysis (LDA) will be utilized to solve this problem.

LDA works by finding the most discriminant projection by maximizing between-class distance and minimizing within-class distance. Before deriving the between-class covariance and within-class covariance, we first define the class mean vector of the sample,

$$m_i = \frac{1}{n_i} \sum_{x \in C_i} x, \tag{19}$$

where n_i is the size of class C_i. The class covariance matrix and the total mean vector of the sample will then be

$$\begin{aligned} S_i &= \frac{1}{n_i} \sum_{x \in C_i} (x - m_i)(x - m_i)^T && (20) \\ m &= \frac{1}{n} \sum x, && (21) \end{aligned}$$

where $n = n_1 + n_2 + ... + n_{68}$ is the number of all samples. Then, the within-class scatter matrix will be

$$S_W = \sum_{i=1}^{68} \frac{n_i}{n} S_i, \tag{22}$$

and between-class scatter matrix will be

$$S_B = \sum_{i=1}^{68} \frac{n_i}{n} (m_i - m)(m_i - m)^T. \tag{23}$$

Then, the goal of LDA is to seek a linear transformation Θ that reduces the dimension of a given feature vector by maximizing the discrimination criteria as follows:

$$J(\Theta) = tr(\tilde{S}_W^{-1} \tilde{S}_B), \tag{24}$$

where

$$\begin{aligned} \tilde{m}_i &= \Theta^T m_i \\ \tilde{m} &= \Theta^T m \\ \tilde{S}_B &= \Theta^T S_B \Theta \\ \tilde{S}_W &= \Theta^T S_W \Theta. \end{aligned}$$

By taking the derivative of J with respect to Θ and set to zero, we obtain the following criterion (please refer to [5])

$$J(\Theta) = \lambda_1 + \lambda_2 + ... + \lambda_p, \tag{25}$$

where λ is eigenvalue of $S_W^{-1}S_B$. Thus, it is obvious that we can maximize J by selecting the p largest eigenvalues of $S_W^{-1}S_B$. Note also that $p < \min(n, 68)$ as S_B has a maximum rank of 67.

Finally, the transformation matrix, similar to the PCA method, will then be $\Theta = [a_1, a_2, ..., a_p]$ where $[a_1, a_2, ..., a_p]$ are the eigenvectors set correspond to $\lambda_1, \lambda_2, ...\lambda_p$.

To ease the implementation of LDA method, the following guideline shows the step by step guide to realize it:

1. Calculate m_i and m;
2. By using value obtained from the previous step, calculate S_i, subsequently S_W and S_B;
3. Compute the matrix $S_W^{-1}S_B$ and find its p biggest eigenvalue with the corresponding eigenvector, $a_1, ..., a_p$.;
4. Transform the high dimension feature data with $y = \Theta^T x$ where $\Theta = [a_1, ..., a_p]$;
5. Proceed with k-NN or Bayes classifier using the low-dimension-feature, y.

6.1 Results from LDA with 1-NN and Bayes Classifiers

LDA algorithm has been implemented and simulated to obtained a low dimension feature. The number of feature is set as 67 as it is the largest possible number. The low-dimension-feature is then classified by utilizing the k-NN ($k = 1$) and Bayes classifiers. The result is tabulated below, with the mean and standard deviation of 50 experiments:

Classification Method	Mean	Standard Deviation
Bayes	0.0478	0.002667
Bayes with LDA	0.0478	0.002657
1-NN	0.0607	0.003922
1-NN with LDA	0.0366	0.002638

It can be observed that by extracting the important feature using LDA method, the classification result shows an obvious improvement with k-NN method. The average error rate by using 1-NN method with the LDA approach is at 3.66% which is considerably good. As for the Bayes classifier with LDA, the error rate are almost identical to the one without any feature extraction. It is, however, much faster to compute by utilizing LDA method, as the number of dimension has reduced from 1024 to 67. The computation time are more than 10 times faster during the experiments.

One might think that LDA will always be better than PCA for classification task. The answer is uncertain, as it depends on the testing sample. If the testing sample does not follow the distribution of the training samples, then LDA might be working worse than PCA. Fortunately in this project, all sample data obtained consists of only frontal view of the persons, and thus the distribution follows. In this case, LDA will outperform PCA, as proven by the simulation.

7 Conclusion

In this project, 2 classification method, namely the k-Nearest-Neighbor (k-NN) and the Bayes classifiers are investigated on human face recognition. It is shown in the experiment results that for k-NN classification, the error rate will be lower for low value of k, and it is of lowest at $k = 1$, which is a special case for k-NN classifier called 1-NN. Also, it is noticed that by computing the average distance using 1-norm, it will perform better. As for Bayes classifier, the computation time is shorter, and hence more effective. Also, the classifier result shown is slightly better than the 1-NN classifier.

To further improve the performance of the classifier, in terms of computation cost and accuracy, 2 feature extraction methods has been tried — Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA). Experimental results has shown that PCA able to improve the computational time for the k-NN and Bayes classification, with the expense of classification accuracy. This is very undesired as the error rate of classification is large and unintended. A better feature extraction method, LDA, is then shown to have outperformed PCA, as not only it improves the computational time but also the overall accuracy of the classification methods.

In conclusion, LDA feature extraction with 1-NN classification has given the best result for face recognition project.

Acknowledgement

A simplified version of the CMU Pose, Illumination, and Expression (PIE) is used as the database for this project. Without such valuable resource, the project would be far more tedious to be carried out. I wish to thank to these people who has shared their finding and valuable resources to the public. Also, I wish to thank Dr Sun and Dr Yan for their teaching and guidance in this project, and also in the field of pattern recognition.

References

[1] Facebook Inc., "Facebook," Internet: http://www.facebook.com, 2011.

[2] Alienware, "Alienware AlienSence FAQ," Internet: http://aliendl.alienware.com /Right_Now/5973/AlienSenseFAQ _02-02-09.pdf, 2011.

[3] CMU, "CMU/VASC Image Database," Internet: http://vasc.ri.cmu.edu/idb /html/face/, 2011.

[4] Y. Sun, *Pattern Recognition EE5907R Part I,* Lecture note, National University of Singapore, Singapore, 2011.

[5] S. Yan, *Pattern Recognition EE5907R Part II,* Lecture note, National University of Singapore, Singapore, 2011.