University of Alberta

Face Recognition using Local Descriptors and Different Classification Schemas

by

Ting Liu

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Science
in
Software Engineering and Intelligent Systems

Department of Electrical and Computer Engineering

Spring 2013
Edmonton, Alberta

Abstract

There are two main activities in a face recognition practice: representation and classification. The main focus of this work is an analysis of image representation methods leading to better image classification scores. This study applies different feature descriptors and image segmentation techniques of image depiction, and investigates their influence on the classification results. We have proposed a number of single and ensemble classification approaches. For single classification approaches, we have considered different segmentation-based techniques of image processing, with weight-based strategies showing the most promising outcomes. In the case of ensemble-based classification algorithms we have investigated multiple criteria of importance focusing on ranking of candidates, as well as on segments and features sorted based on their prominence. We assessed and compared experimental results obtained for the FERET database. The most significant experimental results have been obtained for weighted-based strategy for single classification.

Acknowledgements

Foremost, I would like to express my deepest gratitude and thanks towards my supervisors, Dr. Marek Reformat and Dr. Witold Pedrycz, for their essential and invaluable supervision and support. Thanks to Dr. Marek Reformat's help, this thesis would not have been possible without his valuable advice and help. He provides me constant support and encouragement during my research process, and brings me a spirit of exploring science to real-world applications. I also want to give my sincere thanks to Dr. Witold Pedrycz for his continuous guidance of my study, research and life. He points me toward effective ways and directions to do research in my own specific manner. It would have been impossible to finish this thesis without their genuine help and guidance. Their patience, innovations, enthusiasm, and prestigious contributions to Software Engineering and Intelligent Systems realm greatly impact me, and encourage me to follow their way.

I wish to thank Dr. Bereta in our research group. I deeply appreciate his insightful ideas and his many hours spent with me discussing my research. I am also thankful for Dr. Karczmarek's detailed help and instruction. During my research, they gave me a lot of valuable advice to help me improve my thesis.

I also wish to thank my friends in Edmonton who made my life enjoyable. During my more than two years' study and work, I feel supported when I am talking to my friends.

Last but not least, I want to give my sincere thanks to my parents and my sister for their endless support. Thanks for their support, trust and encouragement all along, which help me to complete my master study overseas.

Table of Contents

List of Figures

List of Tables

List of Symbols

Symbol	Definition
M	Number of images in the training set
S	Number of elements in one vector of the reduced subspace
N	Number of descriptions
$u = (u_1, u_2, \dots, u_s)$	Orthogonal system with s vectors
a_i	Linear combination parameters
S_w	Within-class of Linear Discriminant Analysis
S_b	Between-class of Linear Discriminant Analysis
σ	Defines the ratio of Gaussian window to the wavelength
M	Orientation of Gabor filters
V	Scale of Gabor filters
p	Number of neighboring
p_i	Gray value of one neighbor pixel
p_c	Gray value of the center pixel
NL	Number of different labels produced by the LBP operator
LBP^{u2}	LBP uniform patterns
$Lable$	The person label of one image
x_t	Unknown face without label
x_s	Known face with label
C	Covariance matrix
u_k	Eigenvector
λ_k	Eigen value
$\bar{V}$	The average image of a training dataset
X	Difference image
des^{EF}	Eigenface description
des^{Gabor}	Gabor Filter description
des^{Gabor}	LBP description
$Length$	Length of bins
L	Number of segments
K	Number of sub-segments in one segment
w_j	Weight for segment j.
Num_{can}	Number of total candidates
$P(x)$	Precision of classifier *x*
$R(x)$	Recall of classifier *x*

1. Introduction

1.1. Motivations

For a number of years, facial recognition has been an important research topic because of its wide range of applications. There are various challenges from the perspectives of computer vision and image understanding. A face, as a visual object, can be recognized and classified using both image processing and classification techniques. A general overview of the face recognition methods can be found in [1]. The face recognition process can be divided into two stages:

- the first stage is construction of a face description, where the face can be represented via vectors with a number of discriminable features;
- the second stage is an identification process, in which some classification approaches are used to assign a face image to one of several categories representing individuals.

The first stage of the process has been establishing a dominant and visible position in a face recognition process [2] [3] [4]. However, it is quite apparent that reaching the ultimate goal of a reliable face recognition system; that is, a system recognizing faces under different illumination conditions, of different expressions, requires not only suitable feature description techniques, but also some classification methods that make such a system reliable and adaptive to different levels of image quality.

The goal of an automatic face recognition system is to match an unknown face to the already labeled faces. Though there have been various studies dedicated to this topic, some problems still remain unsolved. The first challenge is the ability to identify images taken under different conditions, such as with or without glasses, with different hairstyles, or separated over long periods of time. The second problem is related to a classification process. In the already-labeled image database, there is often a limited number, maybe just one or two, of images for each individual. It would be difficult to guarantee the recognition results in such a scenario. The third problem is related to the size of the face database: how to make the classification process accurate and fast is quite challenging. Though there are many solutions to these problems, they are focused mainly on the influence and improvements of different feature descriptors, and much less on classification processes.

According to recent studies [5] [6], some facial parts play the most important roles in the process of face recognition. When applying image descriptors to face images, we notice that some descriptors first divide images into several parts and then use the neighborhood information to construct feature descriptions.

Motivated by these observations, we conduct a comprehensive investigation of a face image classification process. The selection of a classification algorithm is important to solve the above-mentioned problems. It is crucial to determine easy and effective classification algorithms that can solve the problems. As for the first problem, it is possible to address only important parts of face images that can overwhelm any noisy information. We introduce a few weight strategies in the process of classification. When it comes to the second challenge, we treat face image segments as individual parts, and construct individual classifiers for these segments. In this case a group of classifiers should make the recognition process more robust and effective. Finally, we select some solutions, which offer good discriminatory abilities, and are relatively independent to the training database, and lead to reduction of computational complexity.

1.2. Objectives and Main Theme

There are several objectives of this thesis. They are:

1) To deliver a comprehensive and comparative analysis of two approaches to a classification process: single classification and ensemble classification. Experiments are conducted with the same descriptors used for image representation. We compare single classification and ensemble classification against each other.

2) To improve current classification algorithms with different descriptors.

Given the special characters of face recognition, the objective is not only to study how a different classification algorithm improves the accuracy of recognition, but also to customize classification algorithms to face recognition processes.

3) To conduct a reliable analysis of different face descriptors. Since different face descriptors represent face images in a different way, our objective is to study their characteristics and their impact on classification performance.

As stated in Section 1.1, face recognition process work in two steps. Based on different face description techniques, the main activities described in the thesis can be illustrated and outlined in Figure 1.

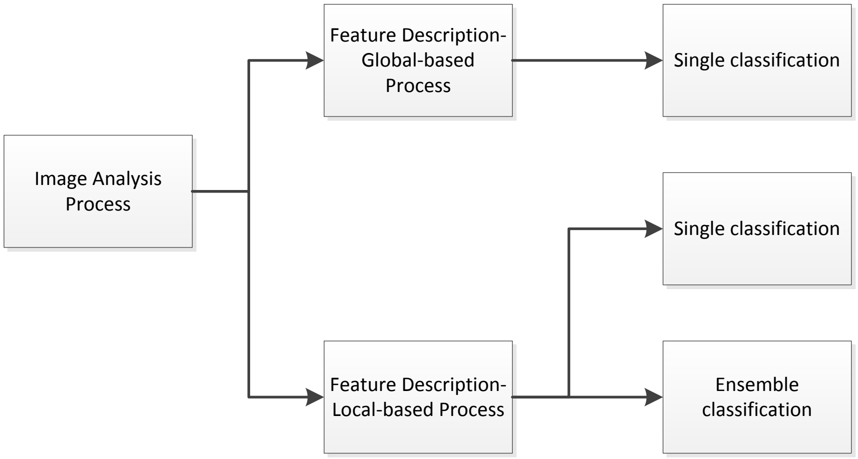


Figure 1. The flowchart of methods and approaches addressed in the thesis

There are two categories of descriptors used in our experiments: global-based descriptors, and local-based descriptors. The global-based descriptors operate on whole images. These descriptors form a new subspace from the original face image space. A single classifier is constructed on this subspace. The local-based descriptors comprise an alternative solution to representing faces. In this case, a face is represented with extracted special characteristics calculated for some selected points or partitioned segments. When applying local-based descriptors, the descriptions can be sent as a whole to a single classifier, or sent to multiple classifiers. When multiple classifiers are used, their outputs are aggregated using ensemble classification approaches.

1.3. Thesis Outline

The thesis is organized as follows.

In Section 2, we review the necessary background material relating to face descriptors and classification. Two categories of face descriptors are introduced: global-based descriptors, and local-based descriptors. Simple classification algorithms and ensemble classification algorithms are also described in this section.

Section 3 talks about our proposed method. We use a PCA-base process as an example of a global-based descriptor, and we examine the process and functionality of this method. We use Gabor filters and Local Binary Patters (LBP) to illustrate how the face recognition process works when local-based descriptors are utilized. Improvements of single classification and ensemble classification approaches are proposed to increase face recognition accuracy.

Section 4 is concerned with the experimental results for the method proposed above. Different feature descriptors and different classification techniques are applied. The obtained results are compared when different processing of images is performed.

We conclude the thesis in Section 5 by summarizing the contribution of our work. We also include some possible directions for future research.

2. Background of Face Recognition

2.1. Texture Analysis Approaches

For humans, face recognition is a simple and quick daily life skill which is achieved through the understanding and interpreting of faces. It seems very easy and takes virtually no time for humans to recognize various different faces; however, it is not easy to implement such a process into software. When following the process of how humans perceive faces, we see that this process is not gained intuitively. Some neurons in the brain are responsible for recognizing faces. When infants are born, these neurons are invoked. Over several days or months of training, these neurons develop the ability to recognize people.

To mimic the process of face recognition, people conduct research on face recognition methodology and try to construct algorithms and systems for the automatic recognition of faces. The challenge of automatic face recognition is that it is a complex system of the human brain, and an unrealistic training process which could take days or even months. Modern face recognition methods apply strong texture analysis approaches to represent original face images by a chain of simpler and more distinctive features, and then use some classification methods to select the most similar candidate as the final step of the recognition process.

How to transform face images into easier and more distinctive features is the key point of texture analysis approaches. Generally speaking, feature-representing approaches can be divided into two categories: global descriptors and local descriptors. Global descriptors use the data of whole images as input to the face recognition system, and the system rebuilds a subspace to represent whole images. Conversely, local descriptors use the neighboring pixels' information around the fixed pixel to construct new features, and these new features are aggregated together to build the global information of faces.

2.1.1. Global Approaches for Texture Analysis

Global approaches are the earliest successful approaches to an automatic face recognition system. Global approaches use the vector that represents the whole face as the input into the system, and pass it on to a data processing method to produce the final classification results. Some of the best-known global approaches include Eigenface with principal component analysis [7], fisher's faces by linear discriminant analysis [8], [9], independent component analysis [10] [11], and neural networks [12]. The following section introduces discusses each of these approaches in more detail.

2.1.1.1. Eigenface by Principal Component Analysis

In 1991[[7]], M. Turk and A. Pentland proposed the first global approach, Eigenfaces. This idea was motivated by physiology and information theory, and assumed that not all information was important for the process of face recognition. In their research, they built one system that projected the whole face onto a smaller subspace that spans the whole set of face images. This subspace, called Eigenfaces, was constructed by a set of significant features which were eigenvectors (principal components) of the gray face spaces. This system was proven to be easily implemented and could achieve near-real-time performance and accuracy.

The process of producing Eigenfaces mainly adopted the method of Principal Component Analysis (PCA). In the theory of orthogonal transformation, the original data space can be represented by the transformation of a smaller data space. In a training set of images, faces are firstly represented by an M-dimensional vector. The PCA method tends to transform the original space into a smaller s-dimensional orthonormal subspace (s<<M) whose basis vectors are orthogonal to each other. This orthonormal subspace can use its orthonormal basis to represent every vector of the original face spaces. In this way, the original face image space is rebuilt by this subspace. This method helps reduce the dimension

of original face space and thus make it handy manipulation and improve computation complexity.

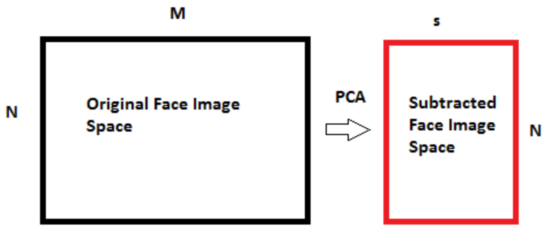


Figure 2. Example of PCA process

The subtracted face image space is supposed to represent the original space with less important or relevant information lost, while creating a smaller data space for computation. In other words, the original M-dimensional vector is reduced to s-dimensional vector, and the first s vectors in this space display as much variance information of the original face image information.

In the training state, each 2-D image in the training set is represented by an M-dimensional vector. Every element v_{ij} in the vector $\mathrm{V_i}$ is represented by concatenating all the pixel values of the image:

$$V_i = \{v_{i1}, \dots, v_{ij}, \dots, v_{iN}\} \quad j = 1,2, \dots, N \tag{2.1}$$

in which

N is the number of pixels,

v_{ij} is the gray value of the jth pixel of the ith image.

For every individual image in the training set, the corresponding variance image from the average image can be represented by subtracting the original image V_i from the average image$\bar{V}$. Let X_i be variance image to V_i

$$X_i = V_i - \bar{V}, i = 1,2, \dots, M \tag{2.2}$$

where

$$\bar{V} = \frac{1}{M}\sum_{i=1}^{M} V_i\,, \quad i = 1,2, \dots M \tag{2.3}$$

The goal of PCA is trying to build an orthogonal linear transformation that transforms the original pixel value space into a new coordinate system with the least information lost. Let the orthogonal system be:

$$OthSystem\, u = \; (u_1, u_2, \dots, u_s) \tag{2.4}$$

where s<<M and

$$\forall\ \mu_i, \mu_j \in OthSystem \text{ is orthonormal to each other}$$

i.e.

$$V_i = \bar{V} + a_i u \tag{2.5}$$

where a_i is the corresponding linear combination parameters

To find out what the system is, we have:

$$InformationLoss = \textstyle\sum_{i=1}^{N} \|V_i + a_i u - \bar{V}\|^2 \tag{2.6}$$

$$= \sum_{i=1}^{N} a_i^2 - 2\sum_{i=1}^{N} a_i u^T (V_i - \bar{V}) + \sum_{i=1}^{N} \|V_i - \bar{V}\|^2$$

Since e is a unit orthogonal system, we can get:

$$a_i = u^T (V_i - \bar{V}) \tag{2.7}$$

Put this expression (2.7) into the original expression (2.6), and we can get:

$$Informationloss = \textstyle\sum_{i=1}^{N} a_i^2 - 2\sum_{i=1}^{N} a_i u^T (V_i - \bar{V}) + \sum_{i=1}^{N} \|V_i - \bar{V}\|^2 \tag{2.8}$$

$$= \sum_{i=1}^{N} a_i u^T (x_i - x_0) - 2\sum_{i=1}^{N} a_i u^T (V_i - \bar{V}) + \sum_{i=1}^{N} \|V_i - \bar{V}\|^2$$

$$= -\sum_{i=1}^{N} u^T (V_i - \bar{V})(V_i - \bar{V})^T u + \sum_{i=1}^{N} \|V_i - \bar{V}\|^2$$

$$= -u^T S u + \sum_{i=1}^{N} \|V_i - \bar{V}\|^2$$

where:

$$S = \sum_{i=1}^{N}(V_i - \bar{V})(V_i - \bar{V})^T$$

The goal of least information lost can be achieved by maximizing the value of μ$^T S$μ.

while $u^T u = 1$, the fomula can be transformed to:

$$u^T S u - \lambda(u^T u - 1) \quad (2.9)$$

The maximum value can be gained while:

$$S_u = \lambda u \quad (2.10)$$

This means that u is the eigenvector of S corresponding to the largest eigenvalue.

The eigenvectors of nonzero eigenvalues produce the orthonormal basis for the subspace that represents the original face images; the eigenvalues reflect the variance degree of corresponding eigenvector. To produce the subspace, the eigenvectors are sorted by their corresponding eigenvalues from high to low. The eigenvectors at higher position mean more variations among training faces. Thus the smallest eigenvector represents the least variance component.

The orthogonal space u is called Eigenfaces which is used to construct the basis coordinate for different images in the training set.

After producing the Eigenface, this face would be transferred to the system. The entire set of testing images would then be projected to this face space and classified to obtain the final ranking.

2.1.1.2. Other Global Approaches

Besides Eigenfaces, there are other long-run global approaches. In 1997, K. Etemad and R. Chellappa proposed the concepts of fisher faces by linear discriminant analysis (LDA) [8]. This method defines the within-class S_w and between-class S_b scatter matrices through eigenvector analysis:

- Within-class S_w is used to show the average scatter of the sample vectors of different classes around their corresponding mean vectors.

- Between-class S_b shows the scatter of conditional mean vectors around the overall mean vector.

The aim of LDA is to maximize S_b while minimizing S_w so that the most relevant features will be extracted for classification.

Besides LDA, C. Liu and H. Wechsler [11] proposed the idea of Independent Component Analysis (ICA) for Face Recognition. Their research involves analysis of such elements as sensitivity to the dimension of the space and performance. During several comparative studies between PCA, LDA and ICA, ICA was proven to produce better results in some conditions. Later, M.S. Bartlett, J.R. Movellan, and T.J. Sejnowski [10] emphasised using ICA to find the feature subspace which was sensitive to the high-order relationships among pixels. Compared to PCA, which focuses more on pairwise relationships between pixels, ICA considers the information of high-order relationships among pixels more important. What is more, two architectures of ICA were also provided for face recognition: one is to conduct the local basis images by using the images as random variables and the pixels as output; one is to produce a factorial face code by using the pixels as input variables and images as output.

These global approaches worked quite well for classifying frontal images; however, the accuracy rate drops sharply when the pose is changed. The reason for this problem is that the global features are sensitive to translation and rotation of the face, so that the base faces would not be accurate while the pose is changed.

Another limitation of global approaches is that sometimes it is hard to find a proper training set for producing the base face for classifying. Normally, the global approaches utilize the images in the training sets to build a base face to help rebuild a subspace for face recognition. But the training sets would be difficult to choose if there are numerous candidates in the database or a limited number of images for every candidate. So later, scholars tried to find other solutions to face recognition which are not restricted to training sets.

2.1.2. Local Approaches for Texture Analysis

Local approaches have been recently proposed and have shown promising results in face recognition. Unlike global methods which utilize the whole image to produce features, local descriptors are applied to the partitioned sub-segments or points, evaluate or compare the neighboring pixels to produce a description. The local descriptions are then aggregated to form the final description for face recognition. According to different selections of feature representation, there are mainly two types of local descriptors:

a. The first descriptors are based on some nominated features over the whole face image, such as EBMG which focuses on a limited number of fiducial points, or Gabor wavelets on some fixed positions. These approaches are widely used and proved to improve recognition accuracy.

b. The second descriptors are based on some sub-segments of the face image. For every pixel in the sub-segments, the neighbourhood information around each pixel is evaluated to construct the feature vectors. The feature vectors are then combined to form the final descriptions. The famous Local Binary Patterns (LBP) and some other LBP-like approaches adopted this method. These are known as segment-descriptor approaches.

The application of local descriptors is produced by comparing between neighbors, so it is not as sensitive to pose changes as global approaches, representing a flexible geometrical relation between the components in the classification. The original images are represented by the local features, and there is no need to build a base image from training sets. These unsupervised approaches are thus more adoptable and more easily implemented in variance circumstances.

Recently, more and more scholars have focused on local approaches, and find them more promising in face recognition. The following section discusses these approaches, mainly Local Binary Patterns and Gabor filters, in more detail.

2.1.2.1. Gabor Filter

Gabor filters, which are spatially localized and selective to spatial orientations and scales, are comparable to the receptive fields of simple cells in the mammalian visual cortex. Because of their biological relevance and computational properties, Gabor filters have been adopted in face recognition. Since Gabor filters detect amplitude-invariant spatial frequencies of pixel gray values, they are known to be robust to illumination changes and achieve good performance.

Gabor wavelets were first introduced in the Elastic Bunch Graph Matching (EBGM) algorithm of face recognition [13]. After the successful application of Gabor in EBGM, Gabor wavelets were considered a reasonable texture analysis approaches to construct descriptions of images.

In EBGM, several important fiducial points, such as eyes and mouth, were selected to be described by set of Gabor wavelet components (jets). Each Gabor kernel is a product of a Gaussian envelope and a complex plan wave. The kernels display information of spatial locality and orientation selectivity. The general Gabor kernels are calculated as:

$$\psi_{\mu,v}(x,y) = \frac{\left\|k_{\mu,v}\right\|^2}{\sigma^2} e^{(-\frac{\left\|k_{\mu,v}\right\|^2 \|z\|^2}{2\sigma^2})}[e^{ik_{\mu,v}} - e^{-\frac{\sigma^2}{2}}] \qquad (2.11)$$

where

$z = (x, y);$

$$k_{\mu,v} = \begin{pmatrix} k_{jx} \\ k_{jy} \end{pmatrix} = \begin{pmatrix} k_v cos\phi_\mu \\ k_v sin\phi_\mu \end{pmatrix}, \qquad k_v = \frac{f_{max}}{2^{v/2}}$$

in which μ and v respectively represent the orientation and scale (frequency) of the Gabor filters.

At most times, parameters are set as follows:

$\mu = 0, \dots, \mu_{max} - 1\ , \mu_{max} = 8\ and\ \ \phi_\mu = \frac{\mu\pi}{8}$

$v = 0, \dots, v_{max} - 1\ , v_{max} = 5;$

σ defines the ratio of Gaussian window to the wavelength;

f_{max} is the maximum frequency to be specified which is related with the actual image size.

The Gabor description of image I(x,y) is defined as a convolution of the image with the Gabor kernels:

$$G_{\mu,v}(x,y) = I(x,y) * \psi_{\mu,v}(x,y) \tag{2.12}$$

where

*is defined as the convolution operator;

$\psi_{\mu,v}(x,Y)$ is defined as a Gabor kernel with a given orientation and scale.

The Gabor wavelet coefficient which is the result of convolution can be written in another form:

$$G_{\mu,v}(x,y) = A_{\mu,v}(x,y) \cdot e^{i\theta_{\mu,v}(x,y)} \tag{2.13}$$

with the magnitude $A_{\mu,v}(x,y)$ and phase $\theta_{\mu,v}(x,y)$.

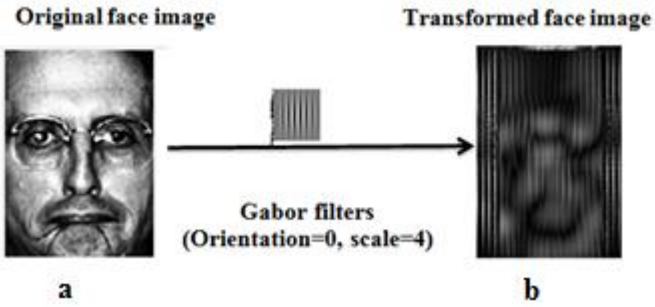


Figure 3. Example of Gabor filters. a is the original face images; b is the transformed image by the Gabor filter with orientation 0 ,scale4

For a long time, magnitude has been considered more useful for facial discrimination. Recent studies [14] [15] also show that the phase is useful in capturing significant features.

2.1.2.2. Local Binary Patterns (LBP)

Local binary patterns (LBP) is a powerful texture analysis descriptor. The idea of LBP was first proposed in 1996 [16]. It has since been found to greatly enhance face recognition accuracy.

The basic idea of LBP is to compare the difference between the central pixel and its nearest 8 neighbors, and translate the difference to a binary number. The binary code is calculated as:

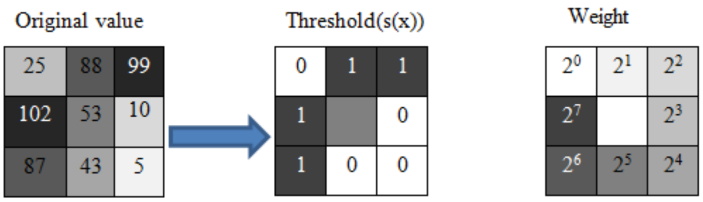
$$LBP(p_c) = \sum_{i=0}^{8} s(p_i - p_c) \cdot 2^i \tag{2.14}$$

where

$$s(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases}$$

p_c and p_i are the grey level values of the center pixel and neighbor pixels.

The final label for a given pixel is translated into a decimal value by multiplying the threshold values and its corresponding weights, and summarizing these. A simple example is presented in Figure 4.



Original value

25	88	99
102	53	10
87	43	5

Threshold(s(x))

0	1	1
1		0
1	0	0

Weight

2^0	2^1	2^2
2^7		2^3
2^6	2^5	2^4

$$LBP(p_c) = 0 \cdot 2^0 + 1 \cdot 2^1 + 1 \cdot 2^2 + 0 \cdot 2^3 + 0 \cdot 2^4 + 0 \cdot 2^5 + 1 \cdot 2^6 + 1 \cdot 2^7 = 198$$

Figure 4. Simple example for LBP

After the LBP value has been calculated for every pixel, the LBP histogram can be defined as:

$$H_i = \sum_{x,y} I\{LBP(x,y) = i\}, i = 0, \dots, n-1, \tag{2.15}$$

where

$$I\{A\} = \begin{cases} 1, & A\ is\ true \\ 0, & A\ is\ false \end{cases}$$

and n is the number of different labels produced by the LBP operator.

In this method, LBP compares the difference between the given pixel and its neighbors; the binary LBP value represents its surrounding gray-scale invariance, and some special patterns such as spots, flat areas, edges, edge ends, curves can be detected within this descriptor. The process of LBP can be treated as a sort of gradient-like operator, since it utilizes the information about the local gray-scale invariance to form the basic description. However, it works differently than other gradient methods. The common gradient methods, such as Histogram of Oriented Gradients (HOG) [17], mainly focused on the gradient directions or edge orientations; LBP, however, is calculated by its surrounding information and form another LBP label for each pixel.

To describe face images, an LBP descriptor constructs histograms for the given images. The LBP label is a binary number, and all the pixels in a given image sub-segment can form a histogram which serves as the description. However, simply one global histogram is not enough to build a whole face description for recognition. One global histogram can only collect special texture pattern information, but loose spatial characteristics of the image. The normal solution is to divide the face image into several non-overlapping segments, and build local histograms for each segment. These local histograms are then concatenated together to build a single global histogram which includes both the texture pattern information and special distribution information. More details about constructing histograms will be described in Section 3.

Recent study has proved that the basic LBP works well in face authentication, detection and facial expression recognition. This method captures the special texture pattern information by simply comparing the local pixel with its neighbors. This simple operation makes this descriptor easy to implement, efficient, robust to different facial expressions, and tolerant to illumination changes.

However, there are some limitations of LBP to be mentioned here. This descriptor is based on comparing with the neighboring pixels, so it mainly captures local variance. The final LBP label is produced by first multiplying the threshold value with the weights and then summing up together. Some noisy pixels at a high weight position can easily affect the values of the label. What is more, when producing LBP histograms, the scales are always fixed, and no multi-scale analysis can be conducted on face images. To solve these problems, further study of LBP has been proposed.

2.1.2.3. Local Binary Patterns Extensions

The earliest extension of the LBP operator was to use different sizes of neighborhoods instead of the 3×3 square neighborhoods [18]. Ojala et al. proposed the method of $\mathrm{LBP}_{\mathrm{P,R}}$ that performs the LBP calculation in a circular neighborhood with the radius R from the central pixel and consisting of P neighbors. Points that do not fall exactly on existing pixels are obtained by bilinear interpolation. Examples about circular LBP are illustrated in Figure 5:

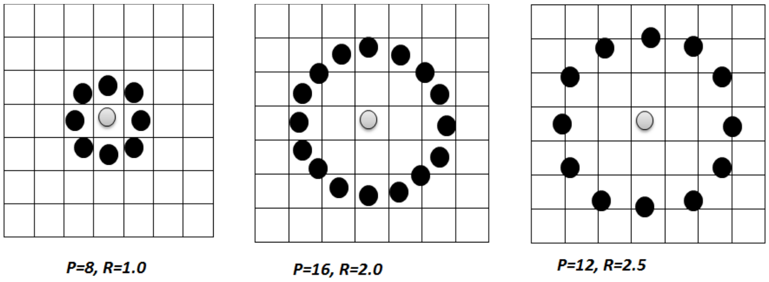


Figure 5. Examples of circular LBP

The formula for producing the $LBP_{P,R}$label is the almost the same as that of LBP. If we make p_c fix at coordinate (0,0), then:

$$LBP_{P,R}(p_c) = \sum_{i=0}^{P-1} s(p_i - p_c) \cdot 2^p \tag{2.16}$$

where

$$p_i = -Rsin\left(\frac{2\pi i}{P}\right), Rcos\left(\frac{2\pi i}{P}\right), i = 0,1,\dots,P-1$$

and

$$s(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases}$$

The circular LBP operator can freely choose different neighborhood space and the size of the neighborhood. It can thus describe texture patterns of different sizes, and make the multi-scale analysis easier to carry out.

Another extension of LBP [18] is the uniform patterns, which contain at most two bitwise transitions from 0 to 1 or 1 to 0 in binary notation. For example, 00010000 (2 transitions) , 11111110 (1 transition) are considered to be uniform patterns and 11011011(4 transitions) 11110100 (3 transitions) are not; more examples are listed in Figure 6. This extension was motivated because they represent the fundamental features such as bright spots, flat area, dark spots and edges. Ojala et al. pointed out that the uniform patterns take up the majority of real world patterns, around 90% when using an (8,1) neighborhood and around 70% in a (16,2) neighborhood. This applies to facial images as well. It is estimated that 90.6% of the $LBP_{8,1}$ patterns and 85.2% of $LBP_{8,2}$ patterns could be described by uniform patterns in FERET databases [19] [20].

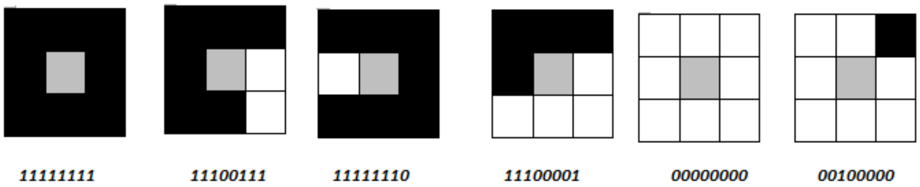


Figure 6. Examples of some uniform patterns

When we use 8 neighborhoods, the LBP operator can produce 256 binary labels, which include 58 binary patterns. The widely accepted idea is to count the histogram of 58 binary patterns to be separate single bins and sum up the remaining patterns to the 59th bin. The notation for this pattern is LBP^{u2}, which refers to at most 2 bit-wise 0-1 or 1-0 transitions. By using uniform patterns, the 256 histogram bins can drop to 59 bins without a severe drop in the recognition accuracy.

2.1.2.4. Relation of Gabor filters and LBP

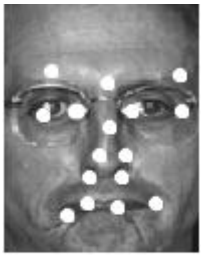
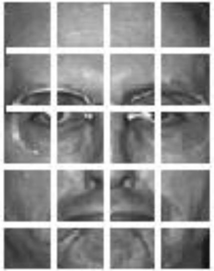
Though the two kinds of descriptors produce different types of descriptions, the procedures of applying different local approaches to face recognition systems is quite similar. The main process consists of three steps (shown in Table 1):

- Decide which local descriptor to be used, and produce the description of the selected fiducial points or the segments of the images;

- Aggregate the local descriptions to form the final global description of the image;
- Compare the unknown image description to the already known candidate face images, and select the best matched images as the classification result.

The comparison is shown in Table 1. The first row lists the attributes of Gabor filters, and the second row lists the attributes of LBP.

Table 1. Comparison between Gabor filters and LBP

Feature selection	Feature description	Operation	Final image description
	$des_i^{point} = [G_{i1}, \dots, G_{in}], i = 1,2,\dots,m$ where G_i is the description of the *i-th* fiducial point, n is the length of local description; m is the number of fiducial points	Concatenation of the description from all the fiducial points	$des^{point} = [\mathrm{des}_1^{point}, \mathrm{des}_2^{point}, \dots, \mathrm{des}_m^{point}]$ where des_i^{Gabor} is the description of the *i-th* fiducial point, m is the number of fiducial points
	$des_i^{segment} = [L_{i1}, \dots, L_{in}], i = 1,2,\dots,m$ where L_i is the description of the *i-th* segment, n is the length of local description; m is the number of sub segments	Concatenation of the description from all the sub segments	$des^{segment} = [\mathrm{des}_1^{segement}, \mathrm{des}_2^{segment}, \dots, \mathrm{des}_m^{segment}]$ where u_i is the description of the *i-th* segment, m is the number of sub segments

What is more, recent studies also indicate that Gabor filters and LBP-like descriptors should not be treated as two different or alternative approaches for texture analysis [21] [22] [23]. Instead they should be treated as complementary ones [15]. One face image can be described as a combination of numerous Gabor filters' described images. LBP-like descriptors can be applied to those Gabor filter described images, and produce the LBP features. Then the LBP features from all the images can be concatenated together to build the whole description of this selected image. This approach has proven successful in face recognition studies.

2.2. Classification

After texture analysis, the original face is described in a set of features. Later, a classification method is applied to compare the unknown face image to the already-labeled face images, and classify the unknown face image to the mostly similar candidate.

In face recognition, most of the design effort has been focused on the development of sound description which is often followed by a dimensionality reduction technique such as, for example, PCA or LDA. The final image matching or/and comparison is usually realized by using a simple k-nearest neighbor with the Euclidean metric or other distance measures, such as inner product, city block distance, or chi square statistics [24] [25] .

2.2.1. K-nearest neighbor (kNN)

K-nearest neighbor [26] is one of the simplest and most straightforward classification methods. This method simply classifies the examples to their nearest neighbours in their feature space. kNN is an example of "lazy learning" or instance-based learning that assumes the feature space contains the correct class, and the searching space is limited in the specified feature space.

The basic process of kNN is based on calculating and comparing the Euclidean distance between the test sample and the training candidates. If k=1, the test sample is classified to the class of the nearest neighbor. If k is greater than 1, the majority vote is applied, and the test sample is assigned to the class with the highest votes.

Suppose there are a lot of training data x_i with label$label_i$, all of which are stored in training set *S*. For an unknown element x_t, the task of nearest-neighbor classification is to find the closet example in the training set, and return the corresponding label, label(x):

$$\mathrm{label(x)} = \mathrm{label_i}, \text{where } \mathrm{i} = \arg\min_{\mathrm{i}\in\mathrm{S}} \mathrm{dist}(\mathrm{x}_t, \mathrm{x_s}) \tag{2.17}$$

One simple kNN classification example is shown below:

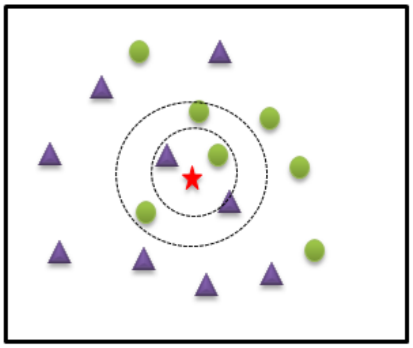


Figure 7. Example of Nearest Neighbor

Figure 7 shows one simple example with only 2 classes in a two-dimensional feature space. In this example, the decision for the test sample (red star) is different according to different parameters. If k=3 (the smaller circle), there are two purple triangles and one green circle in this circle, so the test sample is assigned to the triangle; if k=5 (the larger circle), there are two purple triangles and three green circles in this circle, so this time the test sample is assigned to the circle.

The advantages of using Nearest Neighbor classification are obvious. It is measured by the distance of the source element and target element, and it is easy to implement and analyze. It also uses the information specified in the training set, which can reduce highly adaptive behavior. Finally, this method is simply based on information of the source and target data, with no other data involved, so it is very easy to do parallel implementation.

However, Nearest Neighbor classification also has its limitations. The main problem is that the training data should be stored before doing classification. If the training data set is large, it will use a large amount of memory and can take a long time to find the nearest neighbor. The distance function is also unknown when implementing this classification method; we have to specify the distance function. This method also does not calculate the probability that the unknown element will be classified.

2.3. Ensemble Classification

The method of ensemble classification is the closest analogue to human thought: humans always consider several available options from friends or simply thought by themselves; then put all of the options together, make a comprehensive analysis of these options, and finally choose the best solution. During the process of ensemble classification, the program first evaluates each classifier and then combines these classifiers to arrive at its final decision [27].

The idea of ensemble learning emerged in the late 1970s. In 1977, Tukey [28] first proposed the idea of combining two linear regression models. In his work, he fit the first linear regression model to the original data and the second linear model to the residuals. Later, Dasarathy and Sheela [29] suggested partitioning the input space with two or more classifiers. More significant improvements occurred in the 1990s. Hansen and Salamon [30] suggested that an ensemble of a network on the same database would improve the performance and training of neural networks. Cho and Kim [31] generated a more accurate classification by combining the results from multiple neural networks. Kuncheva and Jain [32] designed two methods to design a multiple-classifier system with a genetic algorithm (GA). They concluded that the GA design can avoid overtraining due to the penalty terms in the fitness function [33].

When we apply the ensemble method to a classification task, it primarily includes these four parts [34]:

1. Training set—a dataset with labeled examples which is used for ensemble training. For most cases, the instances are described as attribute-value vectors. Let's use A to denote the input space containing n attributes: A = {a1, ..., ai , ..., an} and a class label y to represent the class variable or the target attribute.

2. Base Inducer—an induction algorithm that obtains the training set and generates the classification model of the classifier, which associates the input attributes with the target attribute. Let I represent an inducer. We use the formula M = I (S) for representing a classifier M which was induced by inducer I on a training set S.

3. Diversity Generator—the name is self-explanatory.

4. Combiner—combines the various classifiers.

With the above four components, we can simulate the human decision-making process: learn from past experience (the training set), produce some general ideas (base inducer and diversity generator), and finally put them together (combiner).

2.3.1. Evaluation of Classifiers

For evaluation of classifiers, the most widely used criterion is recall. Other measurements related to recall are listed below:

- F-Measure: the weighted harmonic mean of precision and recall. The formula of F-Measure of classifier x is as follows:

$$F(x) = \frac{2\times P(x)\times R(x)}{P(x)+R(x)} \tag{2.18}$$

where P(*x*) stands for precision, R(*x*) stands for recall.

- Receiver Operating Characteristic (ROC) Curves: show the trade-off between true positive to false positive rates. From an ROC curve, the user can select possibly optimal models and discard suboptimal ones from the cost context or the class distribution. ROC analysis is related in a direct and natural way to the cost/benefit analysis of diagnostic decision making.

2.3.2. Aggregation Method

During the combination phase, we will combine the results from ensemble classifiers. There are two main methods for combining the multi-classifiers' outputs: weighting methods and meta-learning methods. In this project we will use weighting methods, which are useful if the base-classifiers perform the same task and are comparably successful.

As for combination methods, we don't know what our multi-classifiers will be like, and we don't know what methods will be better. There are many combination methods, such as majority voting [35], Bayesian Combination [36], Entropy Weighting [37], Boosting [38] and Borda count [39]. These aggregation methods tend to produce better results from the multi-classifiers. They seek to promote diversity among the models they combine [40] [41]. In this thesis, we employ the Borda count algorithm as the aggregation method.

2.3.2.1. Borda count

The Borda count is a single winner method that gives points to each candidate according to their positions. At each classifier, the candidate in last place receives point 0, the next-to-last candidate receives a point of 1, and the candidate in first place receives Num_{can}-1 points (where Num_{can} is the number of candidates). For each candidate, the point at each classifier is added together, and the candidate with the largest point total is the winner.

Let's take an example of 5 candidates with 3 classifiers to illustrate this algorithm. Suppose we have 5 candidates: a, b, c, d, e; and 3 classifiers: C1, C2, C3. The outputs of each classifier are:

C1: a, b, d, c, e

C2: b, c, d, e, a

C3: b, d, a, c, e

Let’s first take a look at the output of C1: a is at the first place of the classification result, so it is assigned a point of 4=5-1; b is at the second place of the classification result, so it is assigned a point of 3=5-2; and then c is assigned a point of 1, d is assigned a point of 2, and e is 0.

If we use the algorithm for other classifiers we can determine the point each candidate receives from the three classifiers. We summarize the final score, and the results are listed in Table 2.

Table 2. Results of applying Borda count

	a	b	c	d	e
C1 Borda score	4	3	1	2	0
C2 Borda score	0	4	3	2	1
C3 Borda score	2	4	1	3	0
total Borda score	6	11	5	7	1

Thus, a got 6 points; b got 11 points; c got 5 points; d got 7 points; and e got 1 points. The final ranking is sorted according to their final point by descending order. So the final result is b, d, a, c, e.

Because this algorithm is easy to use and effective, we will select it to apply to our face recognition process.

3. Proposed Approach

3.1. General Overview of Face Recognition

The face recognition processes used in this thesis can be divided into two main categories according to different texture analysis approaches. One of them contains approaches which focus on the global analysis of an image, while the other contains approaches that analyse an image at the level of local neighbourhoods of selected image points. Both categories, together with their subcategories, are shown in Figure 8.

The image analysis method from the category “Global-based Process” utilized here is PCA. When PCA is used as a texture analysis approach, the full-face space is used as an input. After dimension reduction, a smaller feature space is constructed. To aggregate the information of the sub feature space, we simply concatenate them and pass the data to a classifier. A kNN classifier is used, different similarity measures are investigated, and the obtained results are compared.

Another alternative for face recognition is to use local descriptors for texture analysis in the category “Local-based Process”. In general, face detection using local approaches is divided into three steps: a face image is first extracted and normalized into a rectangle area, and then it is divided into *m* segments or *m* points uniformly; for each segment or point, local descriptors are calculated to extract the features; after feature extraction, some aggregation and classification combination methods are applied to obtain the final result. The flow of the process is shown in Figure 9.

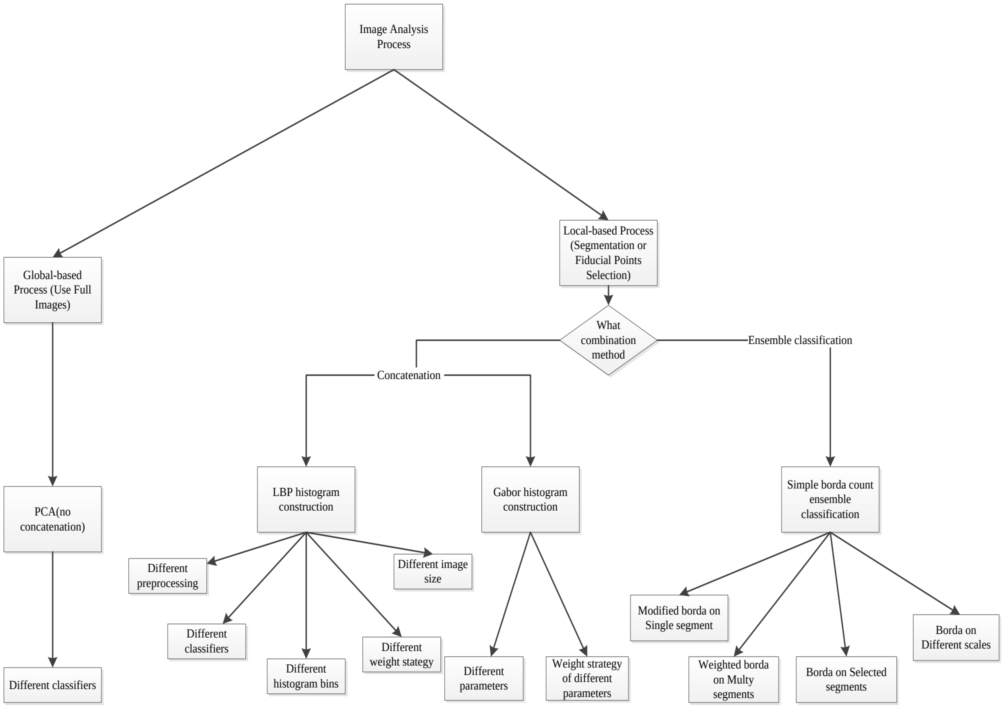


Image Analysis Process
Global-based Process (Use Full Images)
Local-based Process (Segmentation or Fiducial Points Selection)
What combination method
Concatenation
Ensemble classification
PCA(no concatenation)
Different classifiers
LBP histogram construction
Gabor histogram construction
Simple borda count ensemble classification
Different preprocessing
Different image size
Different classifiers
Different histogram bins
Different weight stategy
Different parameters
Weight strategy of different parameters
Modified borda on Single segment
Weighted borda on Multy segments
Borda on Selected segments
Borda on Different scales

Figure 8. Categories of image analysis methods used in the thesis

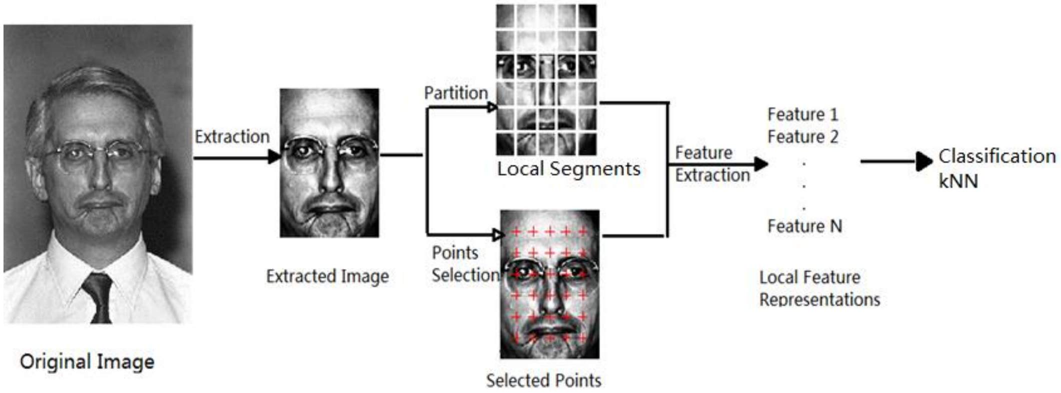


Figure 9. Processing flow of local-based approaches for image analysis

In most face recognition applications, there are many classes of individuals, but very few training samples per individual. It is also common for some classes to have only gallery (reference) samples and no training samples at all. In view of this situation, estimating the parameters of sophisticated classifiers is difficult. Therefore, the simple nearest neighbor classifier is usually adopted. The key to a classification process is a similarity of distance measures determined between image descriptors. In this paper, different similarity measures, such as Euclidean distance, inner product, and chi-square statistics, are applied as the distance function of kNN.

Since in local matching approaches, faces are partitioned into local components or local segments, an unavoidable question is how to combine these local features to reach the final classification. Nearly all of the existing local matching methods choose to combine local features before classification. The local features are either simply concatenated into a longer global feature vector or combined linearly by assigning weights to them. We propose another approach for combining local features: let them act as individual classifiers and combine the results from all these classifiers into the final classification. In this project, we adopt the Borda count method. To further improve accuracy, a modified Borda count and Weighted Borda count algorithm is proposed.

The project proceeds as follows: the face images are first partitioned into similar segments, and descriptors (such as LBP, Gabor) are used to build descriptions for each segment or point. These segments are combined in two different ways:

1. Concatenation of all descriptions: In this stage, descriptions of each segment or point produced by descriptors are simply concatenated. For Local Binary Patterns, many techniques including different preprocessing, different classification, different histograms, different weight strategy, and different image sizes are used and compared. For Gabor filters, different parameters and different weights are compared and studied.

2. Production of multiple Classifiers: descriptions of single segments are used to build single classifiers, and descriptions of segments build multiple classifiers. These classifiers are combined using the static aggregation method - Borda count algorithms. Some further improvements of the Borda count are also applied to Borda, such as enhancing the importance of the higher-ranked candidates, or emphasizing more important segments or features.

3.2. Generation of Features

3.2.1. PCA-based Feature Descriptors

PCA is one of the global approaches which can extract face basis (Eigenfaces) for a whole face image. Once the Eigenfaces have been computed, face recognition can be transformed and classified with the Eigenfaces. The process of applying PCA to obtain image description is performed on selected training face images. Based on these training face images, a universal sub face space is produced. All the remaining test images are transformed to this sub space, and classified afterwards. In section 2, we describe the mathematical view of applying PCA, providing more details about generating face descriptions with PCA.

The first step is to select suitable M face images to represent all the face images in the database. These M face images are used to build up the training space S. Each image is transformed by its gray value of N pixels. So the training space can be described as:

$$S = \{V_1, \dots, V_i, \dots, V_M\}\ i = 1,2, \dots, M \tag{3.1}$$

where

M is the number of images,

V_i is the feature description of gray value of the ith image; it is represented by

$$V_i = \{v_{i1}, \dots, v_{ij}, \dots, v_{iN}\}\ \ j = 1,2, \dots, N \tag{3.2}$$

in which

N is the number of pixels,

v_{ij} is the gray value of the jth pixel of the ith image;

After selection and production of training space, we generate the average image of the M face images:

$$\bar{V} = \frac{1}{M}\sum_{i=1}^{M} V_i\,,\ \ i = 1,2, \dots M \tag{3.3}$$

The following figure is the mean image $\bar{V}$ of our training set.

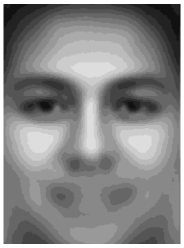


Figure 10. An example of a mean image

For every image in the training set, there would be differences between the selected images and the mean image:

$$X_i = V_i - \bar{V}, i = 1,2, \ldots, M \tag{3.4}$$

After this step, the feature space of variance is constructed, and we can build the covariance matrix C to find the Eigenfaces. It is defined as follows:

$$C = \sum_{i=1}^{M} X_i X_i^T \tag{3.5}$$

The following task is performed to find the orthonormal subspace, which is built by the eigenvectors (also called the Eigenfaces) u_k of covariance matrix C. The corresponding eigenvalues of the eigenvectors of u_k are calledλ_k. The eigenvectors are selected when the eigenvalue

$$\lambda_k = \frac{1}{M} \sum_{i=1}^{M} (u_k^T X_i)^2, k = 1,2, \ldots, M \tag{3.6}$$

is a maximum of the following eigenvalues, i.e, $\lambda_1 > \lambda_2 > \cdots \lambda_s$

where s is the desired number of diminutions to be reduced

and:

$$u_i u_j = \begin{cases} 1, & if\ i = j \\ 0, & \text{otherwise} \end{cases}$$

With the eigenvectors, we can construct the eigenvectors (also called Eigenfaces):

$$u_i = \sum_{k=1}^{M} v_{ik} X_k \quad i = 1, \ldots, s \tag{3.7}$$

where s is the number of dimension of the reduced space.

One example of eigenvectors is shown below:

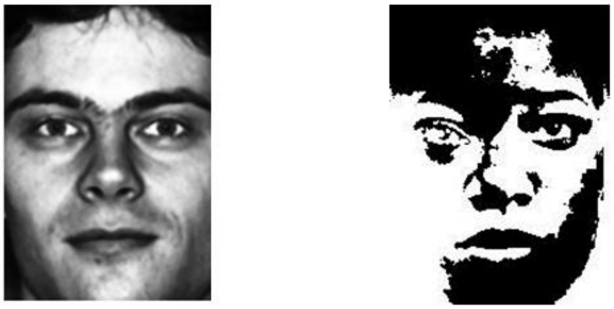


Figure 11. An example eigenvector. The original face images on the left; the corresponding Eigenface on the right

After finding the eigenvectors from the training sets, a new face image *NewI* in the testing datasets is represented by the eigenvectors. It is the product of difference between the mean image and each eigenvector. Since there are M eigenvectors after training, the new image would be described by a vector of M elements:

$$des_k^{EF} = u_k^T(NewI - \bar{V})\ k = 1,2, \dots s \tag{3.8}$$

These descriptions can simply be concatenated for further classification.

$$Des^{EF} = [des_1^{EF}, des_2^{EF}, \dots, des_s^{EF}] \tag{3.9}$$

The above described process reduces dimensionality of image feature space, and nominate features are extracted for classification. More details about classification will be described later.

3.2.2. Gabor-based Feature Descriptors

As discussed in Section 2.1.2, there are two parameters in Gabor filters: orientation and scale. When Gabor filters are used to describe faces, usually a number of 8(orientation)×5(scale)=40 features are treated as one Gabor jet (short for 5×8 Gabor filters) on one pixel. If apply this set of Gabor jet to the whole image; for example, a 128×128 pixels image, the description would consist of 40×128×128 =655360 features. It seems unrealistic to deal with such a large number of features. The widely adopted approach is to set the points evenly across the whole image, and apply the Gabor filters on those points. One example of this method is shown below:

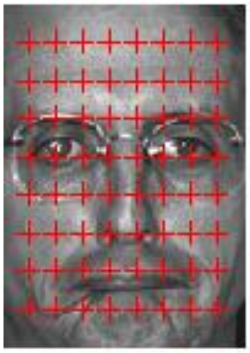


Figure 12. Gabor filters applied on 8×8 points of face images

The 5×8 Gabor filters are listed in Figure 13:

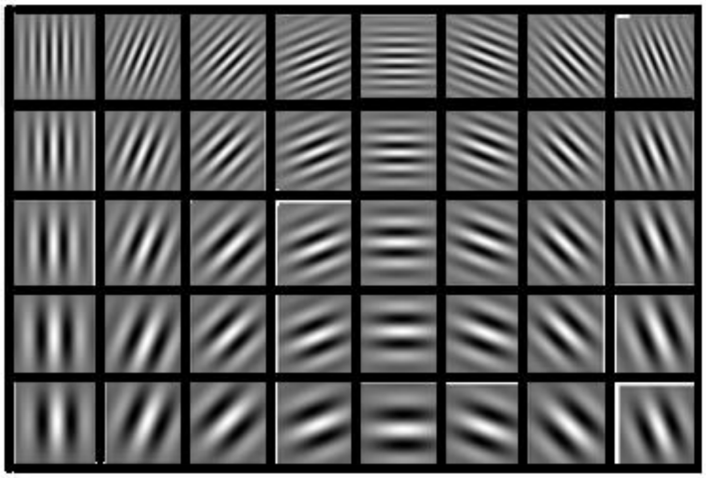


Figure 13. Gabor filters with 8 orientation and 5 scales

Figure 12 is one example of this process. In this method, 8×8 points are placed evenly across the image. For every point, we apply the Gabor filters with five scales and eight orientations to these points. In this example, we use the five scales$v' \in \{4,6,8,10,12\}$, and eight orientations $\boldsymbol{\mu}' \in \{\mathbf{0}, \frac{\boldsymbol{\pi}}{\mathbf{8}}, \frac{\mathbf{2}\boldsymbol{\pi}}{\mathbf{8}}, \frac{\mathbf{3}\boldsymbol{\pi}}{\mathbf{8}}, \frac{\mathbf{4}\boldsymbol{\pi}}{\mathbf{8}}, \frac{\mathbf{5}\boldsymbol{\pi}}{\mathbf{8}}, \frac{\mathbf{6}\boldsymbol{\pi}}{\mathbf{8}}, \frac{\mathbf{7}\boldsymbol{\pi}}{\mathbf{8}}\}$. So for each point, there are 40 descriptions. For the ith point, the corresponding feature description can be represented in the following way:

$$des^{Gabor}(i) = [G_{0,0}(x,y), \dots . G_{\mu,v}(x,y), \dots, G_{8,5}(x,y)] \qquad (3.10)$$

where

$$\mu = 0, \ldots, \mu_{\max} - 1, \mu_{\max} = 8 \text{ and } \phi_\mu = \frac{\mu\pi}{8}$$

$$v = 0, \ldots, v_{\max} - 1, v_{\max} = 5;$$

Another option of using Gabor filters is to also to apply the five scales $v' \in \{v'_1, v'_2, v'_3, v'_4, v'_5\}$ in pixels and eight orientations $\boldsymbol{\mu}' \in \{\mathbf{0}, \frac{\pi}{8}, \frac{2\pi}{8}, \frac{3\pi}{8}, \frac{4\pi}{8}, \frac{5\pi}{8}, \frac{6\pi}{8}, \frac{7\pi}{8}\}$ of Gabor filters are applied to face recognition, but with different jets. In this method, the eight Gabor filters with the same wavelength ($\mathbf{v}'$) and position(x,y), but different orientations $\boldsymbol{\mu}'$ are considered as one Gabor jet(short for 1×8 Gabor filters). We place these Gabor jets uniformly one corresponding wavelength apart, i.e. when the wavelength is 4, the corresponding distance between two jets is 4; when the wavelength 6, then 6 is the distance between two jets; and so on(see Figure 14). So, for different wavelength, there would be different dimensions of feature description.

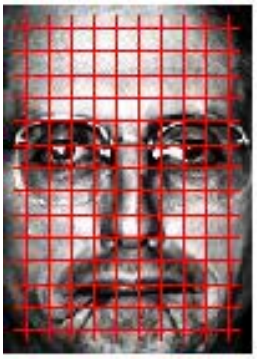


Figure 14. An example of one image with a grid of wavelength 10

The Gabor filters and corresponding transformed image are shown in Figure 15:

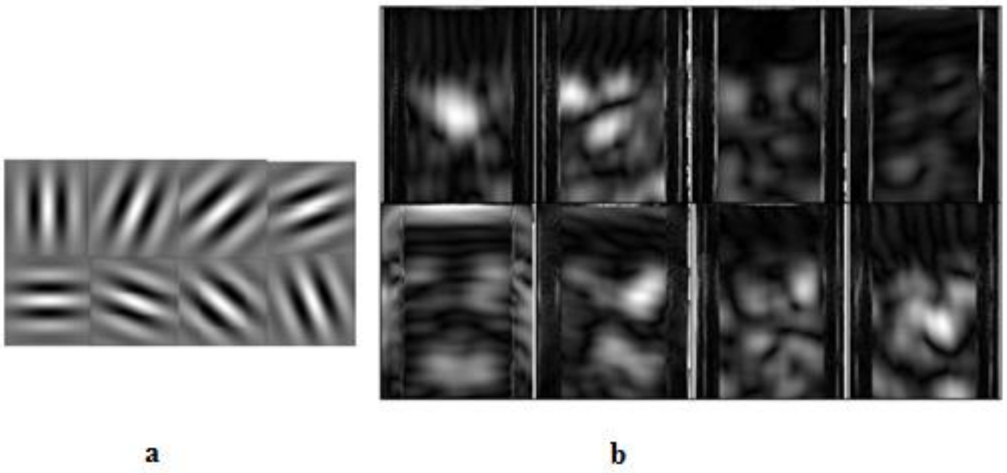


Figure 15. Example of using 8 Gabor filters in one jet. a is the 8 Gabor filters in one Gabor jet, b is the corresponding transformed image

In this example, the wavelength is 10, and the points are selected by every 10 pixels (see Figure 14), for every points, there are 8 descriptions which is from 8 orientations (Figure 15). The eight Gabor filters are the left figures of Figure15, and the right are the transformed images. The eight Gabor filters count as one Gabor jet.

3.2.3. LBP-based Feature Descriptors

The LBP descriptor is applied to all segments, from which the feature histograms are constructed. LBP is proved to be an efficient texture descriptor; however, merely one global histogram of LBP is not enough for proper face recognition. Just one histogram of LBP of the whole image can retain only the information between pixels while losing spatial information. A good solution is to partition the image into segments, and form the global histogram from these segments.

3.2.3.1. Segmentation

The first step of face recognition is to align face images into some easily partitioned shape, and then partition them into several segments. In this project, the alignment and partition process is shown in Figure 16:

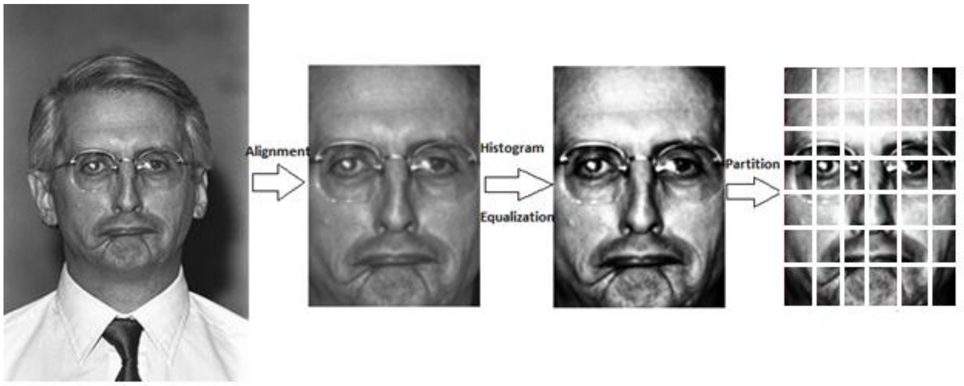


Figure 16. Alignment and partition of face images

The face is aligned into a rectangle area by some transformation (translation, rotation and scaling) based on some detected fiducial points (such as eyes, nose, and mouth). With the scaled image, the histogram equalization preprocessing method is first conducted, and then an image is divided into local segments uniformly. The recognition is studied on those partitioned segments.

3.2.3.2. Histogram Construction for LBP

After images are partitioned into segments, the LBP descriptor is applied to every segment and the histogram is formed for every segment. Let the number of segments be m, and the corresponding histogram of LBP description is defined as:

$$Des_{i,j}^{LBP} = H_{i,j} = \sum_{x,y} I\{LBP_l(x,y) \in E_i\} I\{(x,y) \in R_j\} \qquad (3.11)$$

$$i = 1, \dots, n; j = 0,1, \dots, m-1$$

where n is the number of histogram bins and

$$\mathrm{I\{A\}} = \begin{cases} 1, & A\ is\ true \\ 0, & \text{A is false} \end{cases}$$

$$E_i = \{(i-1) \times length, (i-1) \times length + 1, \dots, i \times length - 1\},$$
$$i = 1, \dots, n$$

in which *length* is the length of bins

And it is restricted to the following formula:

$$n \times length = NL$$

$$NL = 2^{Num_{neig\square bor}}$$

where NL is the number of different labels produced by the LBP operator; it is related to the number of neighborhoods $Num_{neig\square bor}$ of LBP operators.

In this histogram, the image description is distinguished on several levels:

- The pixel level: create LBP value for every pixel
- The segment level: create sub-segment histograms
- The global level: in this project, the segmental histograms are either concatenated or the classification results are aggregated. With the segmental unit, the global description is constructed.

3.3. Single Classifier with Whole Images

After applying the Eigenface method to obtain the subspace of face images, the face images are represented by lower-dimension vectors. We simply concatenate the description as the input to the classifier. A simple Nearest Neighbor algorithm is an easy and effective method for face recognition. The key point of the Nearest Neighbor method is the distance or similarity measures. In the following section, we will discuss various distance or similarity measures used for Nearest Neighbor classifiers.

a) Cosine Distance:

For the unknown image x_t and the already labeled imagex_s, the distance is one minus the cosine of the angle between points (treated as vectors).

$$d_{st}(x_s x_t) = 1 - \frac{x_s {x'}_t}{|x_s||x_t|} = 1 - \frac{\sum_{j=1}^{n}(x_{sj} \cdot x_{tj})}{\sqrt{\sum_{j=1}^{n} {x_{sj}}^2} \cdot \sqrt{\sum_{j=1}^{n} {x_{tj}}^2}} \tag{3.12}$$

where x_{sj} is the jth feature description of the known face; x_{tj} is the jth feature description of the unknown face.

b) City block metric:

$$d_{st}(x_s x_t) = \sum_{j=1}^{n} \left| x_{sj} - x_{tj} \right| \tag{3.13}$$

where x_{sj} is the jth feature description of the known face; x_{tj} is the jth feature description of the unknown face.

An example of city block metic is shown in Figure 17.

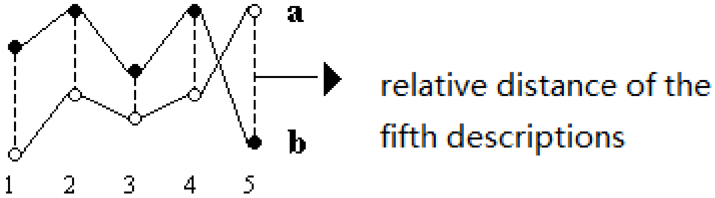


Figure 17. Example of city block

The black points are descriptions of element a, and the white points are descriptions of element b. Element a and element b have 5 descriptions, and their city block distance is the relative distance of each features, i.e. the length of five dashes connected between the black points and its corresponding white points.

c) Euclidean Distance:

$$d_{st}(x_s x_t) = \sqrt{\sum_{j=1}^{n} \left(x_{sj} - x_{tj} \right)^2} \tag{3.14}$$

where x_{sj} is the jth feature description of the known face; x_{tj} is the jth feature description of the unknown face

d) Correlation:

One minus the sample correlation between points (treated as sequences of values).

$$d_{st}(x_s x_t) = 1 - \frac{(x_s - \bar{x}_s)(x_t - \bar{x}_t)'}{\sqrt{(x_s - \bar{x}_s)(x_s - \bar{x}_s)'}\sqrt{(x_t - \bar{x}_t)(x_t - \bar{x}_t)'}} \tag{3.15}$$

where

$$\bar{x}_s = \frac{1}{n} \sum_{j=1}^{n} x_{sj}, \qquad \bar{x}_t = \frac{1}{n} \sum_{j=1}^{n} x_{tj}$$

x_{sj} is the jth feature description of the known face; x_{tj} is the jth feature description of the unknown face.

3.4. Single Classifier with Aggregated Segments

3.4.1. Concatenation of Histograms

The first method used to combine feature descriptions is simple concatenation of all feature description calculated on the sub-segments or selected points. Though Gabor filters are applied on points and LBP is applied on sub-segments, the method of aggregating is similar.

To construct the global description for the ith face, the image feature description is defined as:

$$Feature(i) = [Des(I,1), Desc(I,2), \dots, Des(I,j), \dots Des(I,m-1)] \quad (3.16)$$

$j = 0,1, \dots, m-1$

where m is the number of points or segments.

For Gabor filters, the process is as follows:

a. Select the points evenly across the whole face image;

b. Apply Gabor jets to these points, and obtain the feature description of each points;

c. Concatenate feature description of the points sequentially, and build the global Gabor descriptions.

The LBP feature vector, in its simplest form, is created in the following manner:

- Divide the examined window into cells (e.g. 3x3 pixels for each cell).
- For each pixel in a cell, compare the pixel to each of its 8 neighbors (on its left-top, left-middle, left-bottom, right-top, etc.). Follow the pixels along a circle, i.e. clockwise or counter-clockwise.
- If the center pixel's value is greater than that of the neighbor, write "1"; otherwise, write "0". So this 3×3 pixels window, which has 8 neighbors, gives an 8-digit binary number (which is usually converted to decimal for convenience).

- Compute the histogram for each cell, representing a frequency of occurrence of “number” (i.e., each combination of which pixels are smaller and which are greater than the center).
- Normalize the histogram.
- Concatenate normalized histograms of all cells. This gives the feature vector for the window.

3.4.2. Nearest Neighbor

3.4.2.1. Euclidean Distance

K-nearest-neighbor (kNN) classification is one of the most fundamental classification methods. The classification is commonly based on the Euclidean distance between a test sample and the candidacy image from a training set. When applying Euclidean distance to local based process, features are first described in local segments or points, and later aggregated. In this single classification process, descriptions are concatenated by segments or points. The Euclidean distance is defined as:

$$d_{st}(x_s, x_t) = \sqrt{\sum_{i,j}(x_{sij} - x_{tij})^2}, \qquad (3.17)$$

in whichx_t, x_s represent the feature description of testing and to be compared gallery images independently; i and j refers to ith bin in the histogram of the jth segment or point.

3.4.2.2. Chi-square Statistics Distance

Chi-square is another statistic technique used to measure the dissimilarity between histograms. The value for chi-square statistic distance is given as:

$$d_{st}(x_s, x_t) = \sum_{i,j} \frac{(x_{sij} - x_{tij})^2}{x_{sij} + x_{tij}} \qquad (3.18)$$

in whichx_t, x_s represent the feature description of testing and to be compared gallery images independently; i and j refers to ith bin in the histogram of the jth segment or point.

When the image has been divided into segments, it can be expected that some of the segments contain more useful information to distinguish images from each other than other segments do. For example, in the real world, people can easily use some fiducial points (such as eyes, nose, or mouth) to recognize different people. This process can also be mimicked by a computer. Motivated by this idea, we can set a higher weight for those important segments and a smaller weight to those less important. The corresponding statistics are defined as follows:

$$d_{st}(x_s, x_t) = \sum_{i,j} w_j \frac{(x_{sij} - x_{tij})^2}{x_{sij} + x_{tij}} \qquad (3.19)$$

in which w_j is the weight for segment or point j.

3.5. Ensemble Classification with Segments

An alternative approach for combining local features is to treat each segment as one classifier, and aggregate results from these classifiers to obtain the final result. Many classifier aggregation methods, such as majority vote, sum rule, and Borda count, have been studied. In face recognition, the usual case is that there is just one image of one person in the gallery, and the size of the gallery can reach thousands. It is unrealistic to build a proper training set, and the top ranking images from each sub-segment classification cannot be so accurate. So, we first consider the simple but effective Borda count. In the following section, we discuss ensemble classification with improved Borda count to increase the recognition accuracy.

The general schema for ensemble classification with a single segment is shown below in Figure 18:

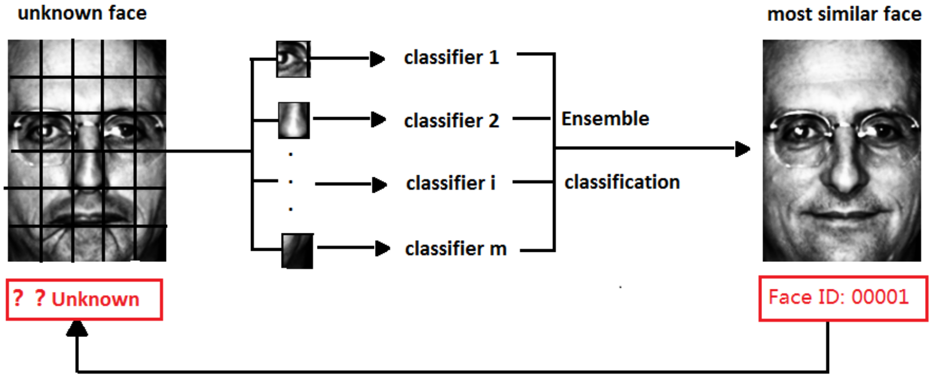


Figure 18. Schema for ensemble classification with single segment

In this schema, the image is divided into 6×5 segments. For each segment, LBP is applied to the sub segment, and produces local feature descriptions. These local feature descriptions are passed to their corresponding classifiers and generate classification results. In this example, there are 6×5 segments, so there are 6×5 classifiers for this face. With these 6×5 classifiers, we apply the aggregation method to obtain the final result. The highest ranking image (the right figures of Figure 18) is selected and its corresponding face id is assigned to the original image (the left figures of Figure 18).

3.5.1. Nearest Neighbor

In ensemble classification, the nearest neighbor is selected to be the classification method. To make a comparison, the Euclidean distance and chi-square statistics are still used as the measure or similarity measures.

3.5.2. Modified Borda Count on Single Segment

In order to investigate behavior of the Borda count, we apply the Borda count algorithm on a training set of 1000 facial images, and 427 candidates (each candidate has more than one image). All face images are divided into 7×7 segments, so there are 49 classifiers altogether. The Borda score of each classifier is calculated and summed together to provide the ranking list according to their Borda score. Motivated by the idea of finding the relation between Borda score and the classification, we simply compare the rank1 Borda score of the successfully classified images and unsuccessfully ranked ones. Before we describe our modifications, we first list one example to understand the Borda count algorithm.

Suppose there are three classifiers: classifier-1, classifier-2, classifier-3; and there are five candidates: a, b, c, d, e. And the real candidate is b. Each classifier produces a rank list:

- The output rank list of classifier-1 is a, b, c, d, e
- The output rank list of classifier-2 is b, d, a, e, c
- The output rank list of classifier-3 is d, b, c, e, a

So the Borda score of each candidate is listed in Table 3:

Table 3. Borda count on three classifiers with five candidates

	a	b	c	d	e
Classifier-1 Borda score	4	3	2	1	0
Classifier-2 Borda score	2	4	0	3	1
Classifier-3 Borda score	0	3	2	4	1
Total Borda score	6	10	4	7	2

According to the Borda count algorithm, the final rank list is b, d, a, c, e. And candidate b obtained the highest Borda score of 10, ranked first in the rank list. Its average Borda score of all the classifiers is 10/3=3.3, which means that its average ranking is 0.7=5-3.3-1. It is also the real candidate, so it is considered a successfully classified candidate.

Let’s take a look at another example, in which are also three classifiers: classifier’-1, classifier’-2, classifier’-3; and the same five candidates: a, b, c, d, e. The real candidate is still b. The new classifiers also produce their own rank list:

- The output rank list of classifier’-1 is a, b, c, d, e
- The output rank list of classifier’-2 is e, c, d, b, a
- The output rank list of classifier’-3 is c, b, d, e, a

Table 4. Borda count on three new classifiers with five candidates

	a	b	c	d	e
Classifier'-1 Borda score	4	3	2	1	0
Classifier'-2 Borda score	0	1	3	2	4
Classifier'-3 Borda score	1	3	4	2	0
Total Borda score	5	7	9	5	4

According to the Borda count algorithm, the final rank list after aggregating the new classifiers is c, b, a, d, e. And candidate c got the highest Borda score of 9, ranked in the first place. Its average Borda score is 3=9/3, which means its average ranking is 5-3-1=1. However it is not the real result, so it is considered an unsuccessfully classified candidate.

In our thesis, the successfully classified images mean that after Borda count aggregation, the correct candidates were recognized in the first place; while the unsuccessfully classified images mean that the incorrect candidates were placed first. We think there are some relations between the Borda score and the results whether the candidates were selected successfully. In order to study these relations, we compared the results of the Borda score of successfully classified candidates and unsuccessfully classified candidates. The result is shown below:

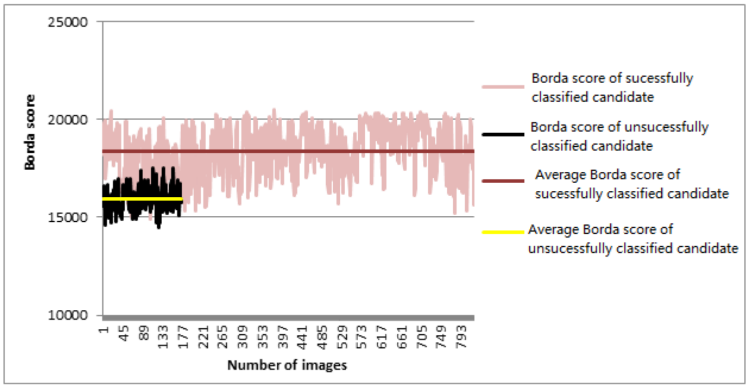


Figure 19. Borda score of successfully classified candidate vs unsuccessfully classified candidate

From this figure, we can see that the Borda score of the successfully ranked list is often higher than that of the unsuccessful ones. For comparison, we list the detail of the Borda scores of the two categories:

The average Borda score of the successful classifications is 18375.58, which means that the average Borda score of a single image is 375, in other words, the average ranking is 51（46=427-1-375）;

The average Borda score of the unsuccessful classification is 15950.83, which means that their average Borda score is 325, and the average ranking is 101（101=427-1-325）;

From this comparison, we can see that when we apply a Borda count, some images with average ranking (rank around 100) but higher total Borda score are selected. To avoid such a problem, the candidates at higher position should be given more points; there can be some modification of the Borda count algorithm. Here are two proposed modifications to the Borda count:

a. The candidate at higher place (for example, ranked before 100) should receive more points than in the case of the original Borda count algorithm;

b. The higher the position, the more points should be assigned to a candidate.

These two possible modification principles motivate us to develop a new version Borda count algorithm.

As for the original Borda count, the corresponding point formula is used:

$$f(x) = Num_{can} - 1 - x \tag{3.20}$$

in which x is the current position, and Num_{can} is the number of total candidates.

To incorporate the first modifications, the new function is proposed:

$$g(x) = \begin{cases} f_{new}(x)\left(f_{new}(x) > f(x)\right), & x < 100; \\ f(x), & x \geq 100; \end{cases} \tag{3.21}$$

As for the second modification, we look for a function such that its value is gradually increasing while the position is increasing (or the position number decreasing), i.e. the derivative of f_{new}(x) is a decreasing function, but above the original f'(x)= -1. Based on this assumption, we select the arcos function, whose derivative is$-\frac{1}{\sqrt{1-x^2}}$. To make the new function more similar to the original function, some transformation is needed. Here we do the following function transformation:

$$f_{new}(x) = \frac{arcos(-1+w\cdot x)}{\frac{\pi}{2}-(-1+w\cdot x)} \cdot f(x) \qquad x \leq 100 \qquad (3.22)$$

A set of experiments has led to the observation that $w \leq 0.01$; i.e., 0.01 is the critical parameter that makes $f_{new}(x) \geq f(x)$ for every $x \leq 100$. Different w results in different function curves; some examples of different function curves according to different w are shown in Figure 20:

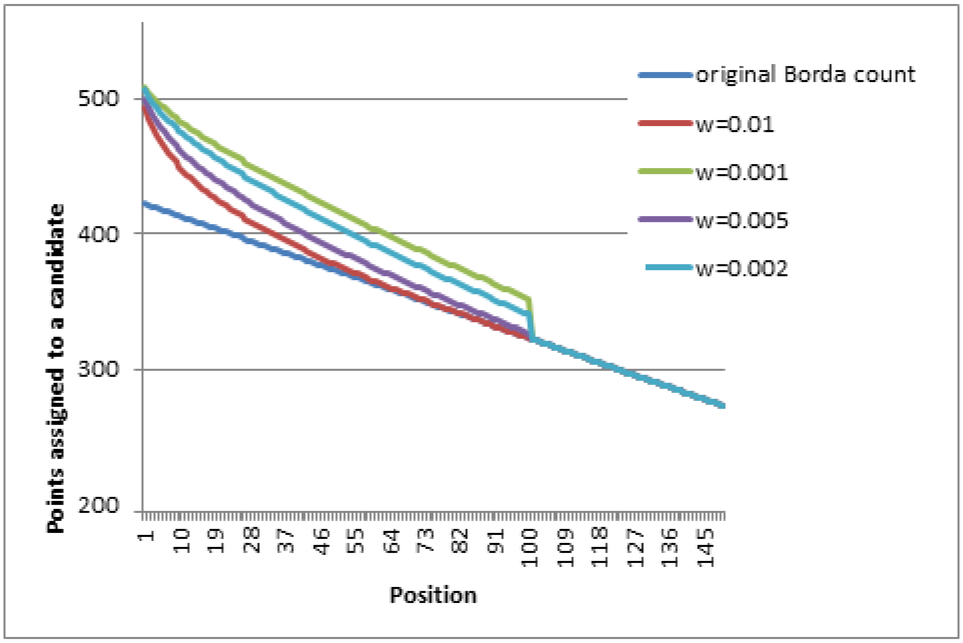


Figure 20. Comparison of original Borda count and modified Borda count

In the experiment reported in Section 4, we selected w of 0.001 and 0.005 to see the modification effects.

3.5.3. Weighted Borda Count on Multi Segments

According to previous assumption, some segments of a face image such as eyes, nose, are more important in the process of face recognition. Therefore, some weight strategies can also be applied to ensemble classification. To study how different weight strategies might influence the results, we proposed two weighted Borda count methods.

To study a weighted Borda count method, suppose we have already had a map of weights for face images. Aimed at applying weights to ensemble classification, we first use a simple "multi-segments" method of ensemble classification. In the first step, the face image is divided into L non-overlapping segmentsR_i. Each Segment R_i is further partitioned into K non-overlapping sub-segments sR_{ij}. Every sub-segment has a weight according to its importance in face recognition. LBP descriptors are first applied on each sub segment sR_{ij} of Segment R_i , and then all the feature descriptions are concatenated, resulting in the feature descriptions of segment R_i. The feature descriptions of segment R_i are passed to the segment's corresponding classifiers for further aggregation. Figure 21 illustrates this method.

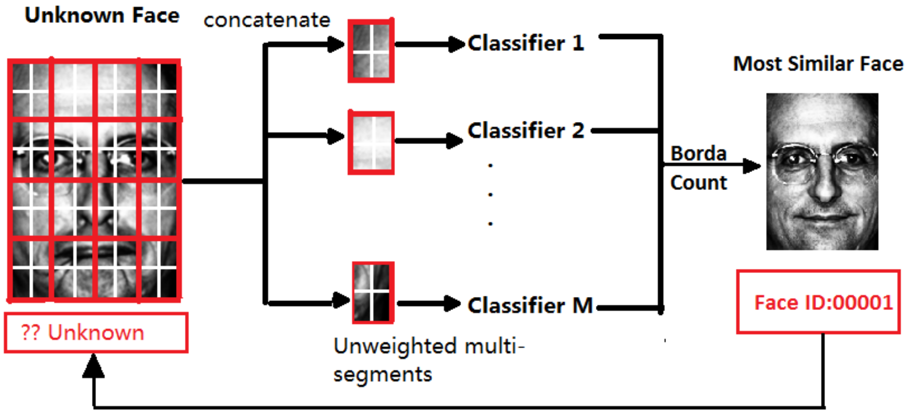


Figure 21. Example of simple "multi-segments" method for ensemble classification

In the thesis we investigate the influence some sub-segments have on the final result. Two approaches are studied: the weights may be applied in the sub-segments before aggregation, or applied during aggregation. In order to illustrate these two approaches more clearly, a face image divided into 8×8 segments is used in the following example. Both approaches are illustrated in Figure 22 and Figure 23, respectively.

The first method is to apply weights before aggregation. We simply adopt the idea of weighted chi-square static distance for each segment. The example of applying weight before aggregation is shown in Figure 22.

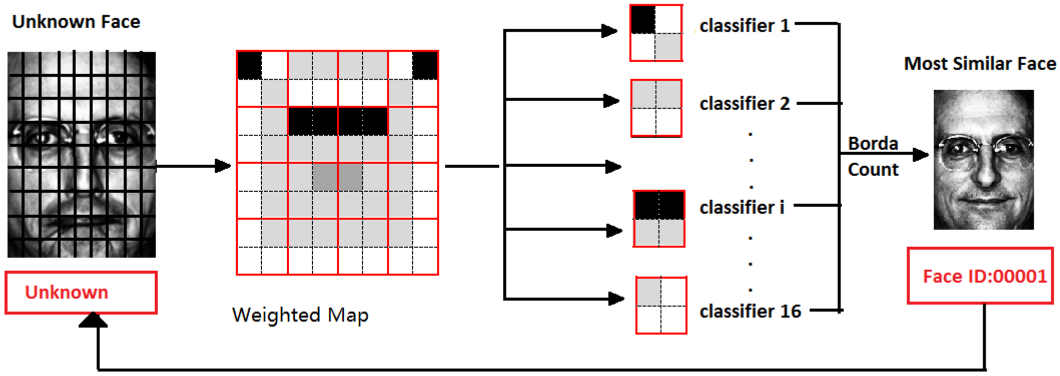


Figure 22. Example of applying weight before aggregation. Different color of weighted map means different weight of the sub segments

The image is first divided into 4×4 segments, and further divided into 2×2 sub segments. For each sub-segment sR_{ij}, weight w_{ij} is assigned, representing its accuracy in face recognition. We concatenate the sub-segments, and use the weighted chi-square static distance as the measure for each classifier.

$$d_{st}\left(x'_{s,i}, x'_{t,i}\right) = \sum_j w_{ij} \frac{(x'_{sij} - {}'x_{tij})^2}{x'_{sij} + x'_{tij}} \qquad (3.23)$$

where

$x'_{s,i}$ is the feature description of the ith is segment$R_{s,i}$ of the training image,

$x'_{t,i}$ is the feature description of the ith segment $R_{t,i}$ of the testing image;

w_{ij} is the weight of the jth subsegment sR_{ij}

x'_{sij} is the feature description of the jth subsegment sR_{sij} of training image

x'_{tij} is the feature description of the jth subsegment sR_{tij} of testing image

The above distance is used as distance measure of the ith classifier. After the results for all segments are obtained, they are aggregated using the Borda count algorithm.

Another alternative to enhance the importance of some segments is to apply weight during aggregation. The process of applying weights during aggregation is divided into the following steps:

- apply LBP to all sub segments, and concatenate the feature descriptions of sub segments sR_{ij} to obtain feature description of segment R_i
- use the concatenated feature descriptions of R_i as input for its classifier, produce the ranking list and generate its corresponding Borda score for every candidate.
- generate the weight for the classifier of Segments R_i; it is simply the mean of its sub segments

$$W_i = \frac{1}{K}\sum_{j=1}^{j=K} w_{ij}\,, \quad i = 1,2,\ldots,K \tag{3.24}$$

  where w_{ij} is the weight of jth sub segments sR_{ij} in the ith segment R_i

- Multiply the Borda score by its corresponding weight. For a person *p*, let’s suppose its rank is xth, then its weighted Borda score is

$$Weightf(P)_i = W_i \times f(x)_i \tag{3.25}$$

  where $f(x)_i$ is the its original Borda score of the ith classifier

- Aggregate all the Borda score to obtain the final score

$$Weightf(P) = \sum_{i=1}^{M} Weightf(P)_i \tag{3.26}$$

- Sort the result according to their final score.

The process of the second approach is illustrated in Figure 23.

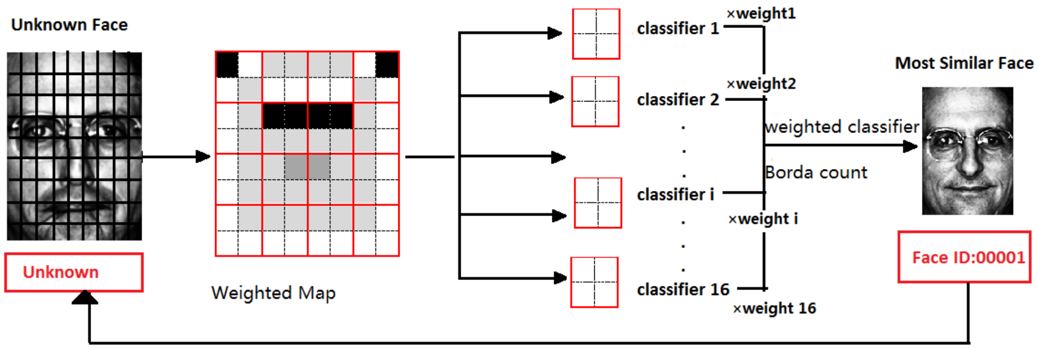


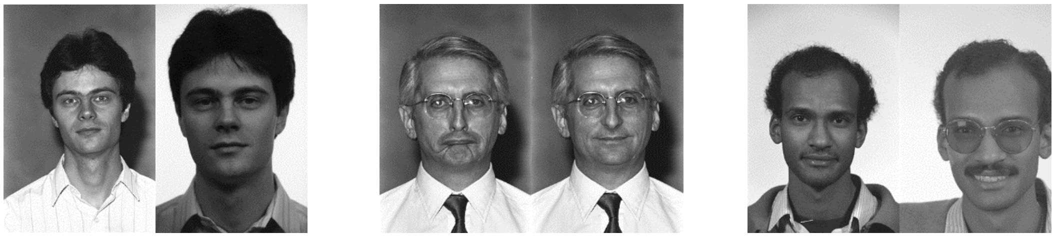
Figure 23. Example of applying weight during aggregation. Different colors of the weighted map mean different weights of the sub segments

4. Experimental Results

4.1. Introduction to FERET Database

This chapter introduces a database that is used in the experiments described in this section. All face recognition methods are run on the FERET database [19] [20].

The FERET database consists of 14,501 facial images from 1,199 individuals. The project to build this database was started in 1993 and finished in 1997. The goal of building this database has been to support face recognition research: testing and evaluating face recognition algorithms on a standardized set of face images. The images are stored as eight-bit grayscale images with the profiles frontal, left and right. These face images were taken under variant conditions, differing in lighting, facial expressions, presence of glasses, and other factors. Some examples are illustrated in Figure 24.



(a) different light condition (b) different expression (c) wearing glasses or not

Figure 24. Comparison of same person images under different conditions

The images are divided into five categories: fa(gallery), fb, fc, dup1 and dup2:

- fa(gallery) contains 1196 subjects with 1 image per subject. It has at most one image per person. It is used as the gallery image dataset for the remaining testing set;
- fb contains 1195 subjects, and 1 image per subject. The images were taken at the same day as fa under the same camera and illumination condition. But images in this set have different facial expressions from that in fa;

- fc contains 194 subjects and 1 image per subject. The images were taken under different lighting conditions;
- dup1 contains 243 subjects and 722 images. The images were taken later than that of fa. The time span ranges from one minute to 1031 days;
- dup2 contains 75subjects and 234 images. It is a subset of dup1 and contains images taken at least 18 months after images from fa.

In most cases, fb, fc, dup1 and dup2 are used as testing datasets; all the images are treated unlabelled. During testing, fa is used as the gallery set with the subject ID, which would be assigned to the unknown subjects after recognition.

In order to capture the most significant variance in face images which can improve face recognition accuracy, a large and representative training set is needed. FERET provides a standard training set of 1002 images from 429 subjects under different conditions, such like lighting, pose and expression.

Since the images in FERET are of various sizes, we crop face images into 150 pixels by 210 pixels. Histogram equalization is applied to the cropped images to help improve recognize faces. Some other preprocessing technologies will be discussed in Section 4.3.1.

4.2. Experimental Results for Single Classifier with Whole Images

As explained in the previous chapter, Eigenface is one of the global-based methods used to extract features and reduce the data’s dimensionality. In this chapter, our goal is to find how different classifiers will affect the face recognition performance.

At first, this experiment is conducted on FERET standard training sets. There are 1000 images and 423 subjects, of which 500 images were selected randomly, with at least one image per subject. The image size is 150×210 pixels. So, the original feature space is 500 vectors, and each vector has 31500 descriptions; i.e., 500×31500 dimensions. This is a very large and highly dimensioned space; a smaller sub space is needed. By applying PCA to this feature space, we reduce the space to 200 descriptions per image.

With the produced Eigenfaces, we use different classifiers. The k Nearest Neighbor with distance measures, including cosine, city block, Euclidean, and correlation are studied here, and the results are listed in Table 5.

Table 5. Results for applying Eigenface with different classifiers

	fc	fb	dup1	dup2
Cosine	13.4%	86.6%	36.84%	12.82%
City-block	**58.24%**	**90.125%**	**41.14%**	**19.65%**
Euclidean	**58.24%**	**90.125%**	**41.14%**	**19.65%**
Correlation	13.4%	86.44%	36.38%	12.39%
Average	35.82%	88.32%	38.88%	16.13%

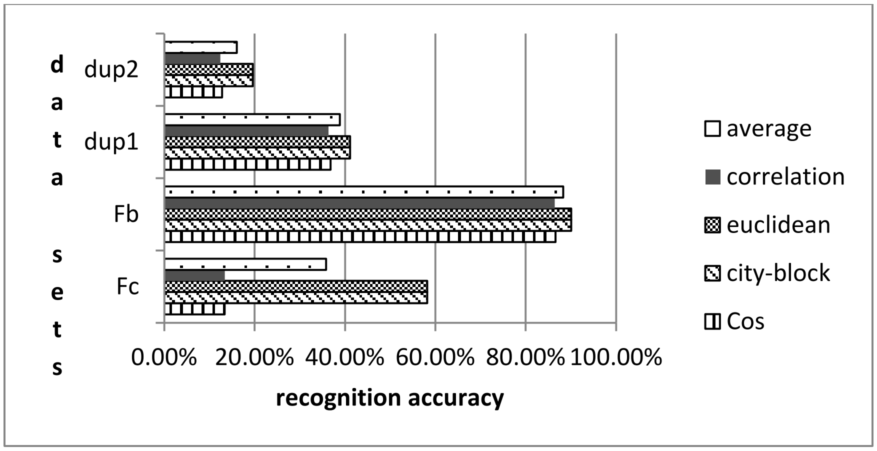


Figure 25. Results for different classifiers of Eigenfaces

From this figure, we can see that fb has the highest average accuracy, as the recognition accuracy of each classifier is higher than 85%; dup2 has the lowest average recognition accuracy, as all accuracy values are lower than 20%. And when four different classifiers are applied to dup1 dataset, all results are very similar: around 40%. But when applying the four different classifiers to fc, the results seemed extremely different among the four classifiers: cosine and correlation have very poor performance with a recognition accuracy of 13.4% while Euclidean and city-block have an accuracy of 58.24%.

Among the results for the 4 classifiers on 4 datasets, we can see that classifiers kNN with Euclidean and city-block distance measures achieve the highest accuracy; while cosine and correlation based kNN achieve relative low recognition accuracy.

4.3. Experimental Result for Single Classifier with Aggregated Segments

4.3.1. Local Binary Patterns

In Section 3.4, we discussed single classification on aggregated segments. Here, we perform an experimental comparison of them. Histogram equalization is first conducted on every facial image. The original face is cropped to 150×210 pixels. According to suggestions for LBP [42], $\boldsymbol{LBP_{8,2}^{u2}}$in 18×21 pixel windows would be a good trade-off between recognition performance and feature vector length. We used the $\boldsymbol{LBP_{8,2}}$ descriptor, and divided the images into 8×10 sub-segments. There are several aspects that will influence the accuracy of the face recognition process. In the following section, we mainly focus on five aspects, including different image sizes, different preprocessing techniques, different classifiers, different histogram bins and different weight strategy.

4.3.1.1. Simple Concatenation with Different Histogram Bins

As described in Section 2.1.2 , the number of histogram bins is a parameter of the face recognition process. In this project, we use $LBP_{8,2}$ as the image descriptors. Besides the normal histogram bins which divide the produced labels evenly, there is another descriptor similar to normal LBP, which extracts the most significant histograms, called uniform patterns.

A $LBP_{8,2}$ can produce $2^8 = 256$ labels; we can evenly divide the 256 labels into a fixed number of bins, focus on the uniform pattern, and gather the remaining patterns as one histogram is another solution. To make a comparison, we compared the results for 32 bins and 59 bins of uniform patterns. What is more, this comparison is applied in different classifiers, kNN with Euclidean and Chi-square static measures. The result is shown in Table 6:

Table 6. Results for different number of histogram bins and classifiers

	fc	fb	dup1	dup2
Euclidean-32 bins	47.40%	94.80%	50.40%	23.90%
Euclidean-59 bins	61.86%	95.06%	49.17%	23.93%
Chi-32 bins	61.30%	97.10%	55.00%	26.90%
Chi-59 bins	**70.62%**	**97.49%**	**57.89%**	**32.48%**

After applying uniform pattern of 59 bins, the recognition accuracy of both classifiers is greatly enhanced when compared with the results obtained for 32 bins. With fewer features; i.e., 59 histograms for each segment, we can achieve better recognition accuracy. This can help reduce algorithm complexity while keeping recognition accuracy.

4.3.1.2. Classification with Different Image Sizes

In many existing face recognition methods, face images are cropped, and only inner parts of the original image, which include the fiducial points, are used. However, a study by [43] suggests that using of all the head information could lead to better performance than just using internal features. Motivated by this idea, we cropped the same images into different sizes and performed tests on the FERET database.

The face images are first resized to 210×150 pixels, which include heads with the boundaries. To study how the shape information contributes to the recognition, we gradually cropped images until all the head boundaries have been cropped, leaving only internal features. The images are first cropped to 150×170 pixels, abandoning the upper and lower boundaries; then cropped to 140×150 pixels, abandoning the left and right boundaries; and finally cropped to 133×146 pixels, leaving only internal features. Taking one image as example, whose corresponding image sizes are listed in Figure 26:

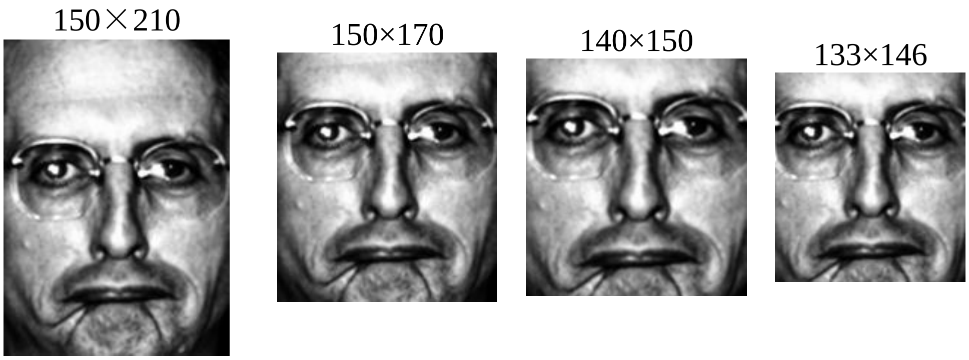


Figure 26. Examples of same image with different sizes

The classification is conducted on a classifier of chi-square kNN, and using simple concatenation of descriptions. In Table 7, the results for images of different sizes are presented.

Table 7. Face Recognition Accuracy of different sizes

size	fc	fb	dup1	dup2
150×210	**70.62%**	**97.49%**	**57.89%**	**32.48%**
150×170	48.97%	92.56%	52.49%	22.22%
140×150	42.27%	91.30%	53.32%	28.21%
133×146	33.51%	90.88%	53.05%	29.49%

Comparing the results obtained for different image sizes, the original image of 150× 210 achieved the highest score, which means the shape information is useful for face recognition. More specifically, the trend of recognition accuracy of fc, fb is similar. As the edges are gradually excluded from the input area, the recognition accuracy is gradually reduced. The testing set dup1 and dup2 shows a similar trend. dup1 and dup2 have the highest accuracy with the overall heads shape information (150×210 pixels), but the lowest recognition accuracy occurs when the size is 150×170. Images with smaller areas such as 140×150 or 133×146 obtain higher recognition accuracy values when compared with that of 150×170. And as for dup2, the smallest area 133×146 almost brought the second highest recognition accuracy. The results illustrate that information on the head shape is important in the face recognition process. The images of dup1 and dup2 were taken during different time intervals, and dup2 is a sub dataset of dup1, and the images were taken during a longer period. From the experiment, we can see that during the longer period, the right and left edges were not as important as internal features and they may even lead to incorrect recognition.

4.3.1.3. Simple Concatenation with Different Preprocessing Techniques

Another perspective that might affect face recognition is the different preprocessing techniques. In this section, we apply three techniques to increase the quality of images and hope it will benefit face recognition process. The two techniques include histogram equalization and adaptive histogram equalization.

a) Histogram Equalization

Histogram equalization can increase the close contrast between images. This method gathers information from all pixels, spreads out the most frequent intensity values and transforms all pixels, producing a higher contrast.

b) Adaptive histogram equalization (AHE)

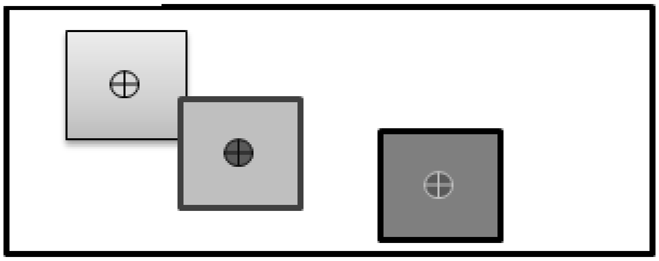


Figure 27. Example of adaptive histogram equalization

Adaptive histogram equalization (AHE) is different from the ordinary histogram equalization, which will improve local contrast. It computes several histograms of a distinct section of the image, and uses them to redistribute the lightness values of the image. Examples of adaptive histogram equalization are shown in Figure 27. For a fixed pixel, the transformation is similar as the regular histogram equalization but based on the neighbourhood segment (the rectangle area). The transformation function derived from the neighbourhood segment (the rectangle area). For this method, the area is a parameter of this method. In the following experiment, we applied the AHE on 50×50, 60×60, 70×70 rectangles.

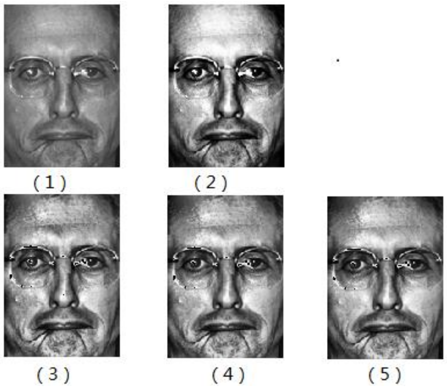


Figure 28. Examples of preprocessing.(1) is the original image, (2) is image by histogram equalization, (3) is image by adaptive histogram equalization 50×50, (4) is image by adaptive histogram equalization 60×60, (5) is image by adaptive histogram equalization 70×70

The influence of different preprocessing on face recognition can be shown in Table 8 and Figure 28.

Table 8. Results for Different preprocessing

	fc	fb	dup1	dup2
adaptive-50-50	59.28%	97.30%	57.89%	**38.46%**
adaptive-60-60	54.64%	97.41%	**58.59%**	38.03%
adaptive-70-70	59.28%	**97.66%**	58.03%	35.90%
hist-eq	**70.62%**	97.49%	57.89%	32.48%

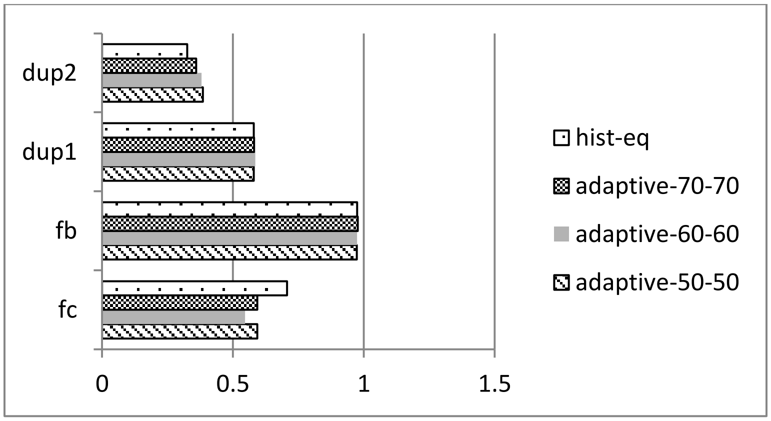


Figure 29. Results for different preprocessing

From the comparison, we can see that for fb and dup1, different preprocessing techniques have similar influence, but for fc and dup2, the results are quite different from each other. As for fc, the highest recognition accuracy is reached when applying histogram equalization; and as for dup2, the highest accuracy is achieved by adaptive histogram equalization with a segment of 50×50 pixels.

4.3.1.4. Simple Concatenation with Different Classifiers

To study the effect of classifiers, we compare different classifiers applied to the same data. In this section, we still adopt Nearest Neighbor as the classification method with different similarity measures. The different similarity measures include Euclidean distance, Chi square distance and cosine distance.

To study the different classifiers, the comparison experiment is conducted on the images preprocessed with histogram equalization, and of a size 150×210. The result is listed in Table 9.

Table 9. Results for different classifier

	fc	fb	dup1	dup2
Euclidean	61.86%	95.06%	49.17%	23.93%
Chi	**70.62%**	**97.49%**	**57.89%**	**32.48%**
cosine similarity	62.89%	95.48%	51.11%	23.08%

From Table 9, different similarity measures affect the results for face recognition severely. Among the three similarity measures, chi-square works better than the other two. The accuracy can be even 9% higher than Euclidean on fc. So, it is beneficial to use chi-square for a later classification process.

4.3.1.5. Weighted Concatenation with Different Weight Strategy

When looking for the weights, research in [42] provides suggestions how to obtain weights for each sub segment. We followed their idea and make some further improvement to calculate the weights for each sub-segment. The procedure to determine the weights as follows:

a. Use a feature histogram for one segment at a time.

b. Rank the segments on the left half and right half face according to their recognition rate.

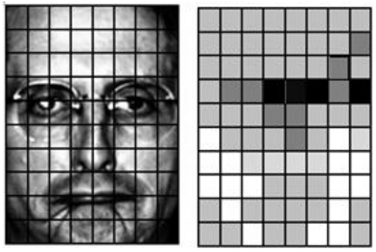
c. Rank the recognition accuracy of different segments, and classify them into four classes.

- The first class is a highly contribution-segments class, which makes the highest contributions to face recognition. It is assigned a weight of 4;
- The second class is a contribution-segments class; it is ranked the second in contributions to face recognition. It is assigned a weight of 2;
- The third class is a normal-segments class; it makes normal contributions to face recognition. It was assigned a weight of 1;
- The fourth class is a bad-contribution-segments class; it makes the least contributions to face recognition, and sometime may even mislead the recognition process. It is assigned a weight of 0.

With the above definitions of segments, we defined three weight strategies to determine the weight of each segment:

a) Overall weight strategy. Based on the training results, classify the segments whose training recognition ranked 5% of all the segments as the high-contribution-segment class; classify the segments whose training recognition accuracy ranked between 5%- 15% of all the segments as the contribution-segment class; classify the segments whose training recognition accuracy ranked between 15%-80% as the normal-segment class; and the remaining segments are classified as low-contribution-segment class.

One example of this weight strategy is shown in Figure 30. The darkest segments are high-contribution segments; the dark gray segments are contribution segments; the light gray segments are normal segments; the white segments are low-contribution-segments.



.

Figure 30. Example of overall weight strategy

b) Symmetrical weight strategy. This strategy is similar to the overall weight strategy, but the segments classification is processed separately in right and left face images. The similar process of determining the weights of segments is defined on each half of the face image, and gathered together in the end. An example of symmetrical weight strategy is shown in Figure 31.

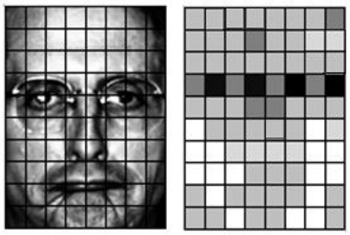


Figure 31. Example of symmetrical weight strategy

c) Standard deviation weight strategy. In the training stage, each segment was treated as input of the independent classifier. After training, each segment obtains its recognition accuracy of the training datasets. We would like to define the weight in such a manner: suppose the average segment face recognition accuracy of all segments is $\overline{\text{accu}}$, and the standard deviation of the segment face recognition accuracy of all segments is accu_{std}. The segments with accuracy higher than $\overline{\text{accu}} + \text{accu}_{std}$ are defined as highly-contribution and contribution segments. Among them, those segments that ranked in the top $\frac{1}{3}$ of this category are classified as high-contribution segments; and the remaining segments in this category are classified as contribution segments. Those segments whose accuracy lies in $[\overline{\text{accu}} - \text{accu}_{std}, \overline{\text{accu}} + \text{accu}_{std}]$ are classified as normal segments. The remaining segments are classified as low-contribution segments. One example of this weight strategy is listed in Figure 32.

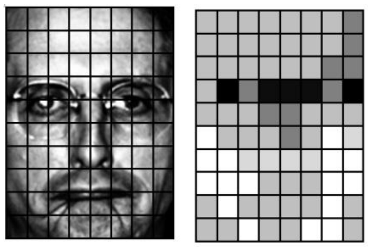


Figure 32. Example of standard deviation weight strategy.

After defining the three weight strategies, we apply them to FERET datasets. We used LBP uniform pattern $LBP_{8,2}^{u2}$ as feature descriptors, and kNN with weighted-chi square distance formula as classifiers. The results are shown in Table 10.

Table 10. Results for different weight strategies on FERET

	fc	fb	dup1	dup2
Overall weight	73.71%	98.91%	**64.40%**	**53.42%**
Symmetrical weight	**79.90%**	**99.16%**	62.19%	47.86%
Standard deviation weight	74.23%	98.91%	63.57%	52.14%
Unweighted	70.62%	97.49%	57.89%	32.48%

From this comparison, the highest accuracy for fc and fb was achieved using symmetric weight strategy; and the highest accuracy for dup1 and dup2 was achieved applying weight of overall weight strategy. The reason for different results might be that some segments play different roles of importance in different datasets.

4.3.2. Gabor Filters

4.3.2.1. Different Parameters of Gabor Filters

There are two ways of using Gabor filters as feature descriptors. The first method is to select some fixed points, and use 40 different Gabor filters (5×8 Gabor filters): 5 different wavelengths and 8 orientations which give us 40 different filters. Each filter is applied to each point. For example, we can chose $\{0,\frac{1}{8}\,\pi,\frac{2}{8}\,\pi,\frac{3}{8}\,\pi,\frac{4}{8}\,\pi,\frac{5}{8}\,\pi,\frac{6}{8}\,\pi,\frac{7}{8}\,\pi\}$ (short for (0, PI,8)) as the 8 orientations, which means there will be 8 rotation from 0 to PI radians; and choose 4,16,8 as the 5 different wavelengths, from 4 pixels to 16 pixels.

The normal formula for orientation is (0,PI,8), which gives us 8 orientations. However, the value of the wavelength should be adjusted to the particular size of the image. The smallest possible value should be 2 pixels. And for bigger images, the wavelength should be increased. And the meaning of the wavelength (2,16,5) is defined as: minimum wavelength is 2 pixels, maximum is 8 pixels, and there will be 5 different wavelengths generated from this range.

In order to adjust the wavelengths for better effects, we tried 9 sets of parameters. With the purpose to make a similar comparison to LBP, we selected 8×10 point evenly across images. The 5×8 filters are used for every segment, and the results are shown in Table 11.

Table 11. Results for different parameters of 5×8 Gabor filters

	fc	fb	dup1	dup2
(3,16,5)	57.22%	92.47%	54.02%	33.33%
(4,16,5)	59.28%	92.64%	55.26%	35.04%
(5,17,5)	62.89%	92.13%	**58.45%**	41.03%
(6,18,5)	64.43%	92.47%	57.20%	40.60%
(7,19,5)	65.46%	92.30%	57.34%	39.74%
(8,20,5)	65.46%	91.63%	58.03%	**42.31%**
(9,21,5)	63.92%	90.96%	56.79%	40.60%
(10,22,5)	**66.49%**	91.72%	56.51%	38.03%
(10,24,5)	65.98%	**92.64%**	54.43%	34.62%

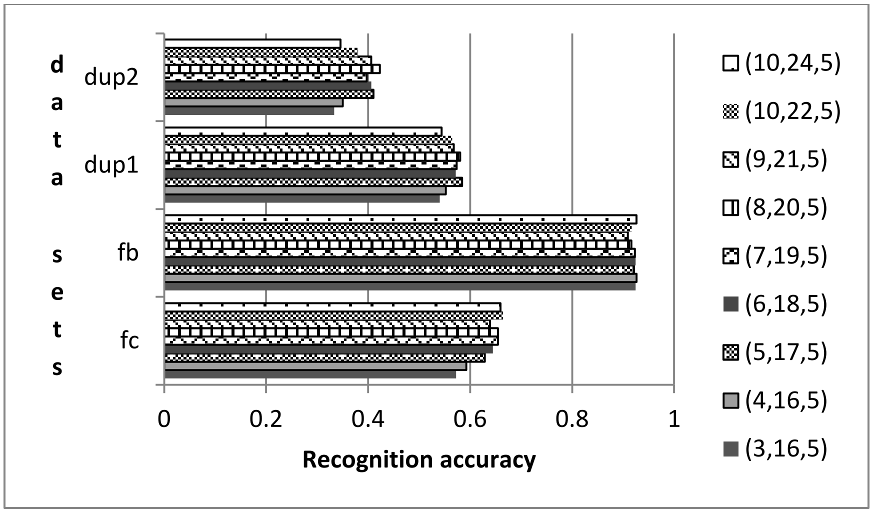


Figure 33. Results for different parameters of 5×8 Gabor filters

From Table 11, we can find the best parameters for different datasets. For fc, the parameter for highest accuracy is (10,22,5); for fb, the parameter for highest accuracy is (10,24,5); for dup1, the parameter for highest accuracy is (5,17,5); for dup2, the parameter for highest accuracy is (8,20,5). Though there are no specific parameters for all datasets, normally speaking, the suitable parameters for our project are around (8,20,5) and (6,18,5). We would use the two sets of parameters for further study.

Another option of using Gabor filters as feature descriptors is to use 8 Gabor filters of 8 orientations (1×8 Gabor filters) as the feature description and choose point evenly one wavelength apart. In the following, we used the wavelengths of (8,20,5); i.e., 8,10.4,12.8, 15.2, 17.6, as the wavelength. The results are shown in Table 12 and Figure 34:

Table 12. Results for different parameters of 1×8 Gabor filters

	fc	fb	dup1	dup2
(17.6,17.6,1)	49.48%	85.36%	46.40%	29.91%
(15.2,15.2,1)	52.58%	85.94%	51.25%	33.76%
(12.8,12.8,1)	56.70%	88.37%	52.63%	33.76%
(10.4,10.4,1)	**57.22%**	90.80%	**54.29%**	**39.74%**
(8,8,1)	54.12%	**93.39%**	50.83%	33.76%

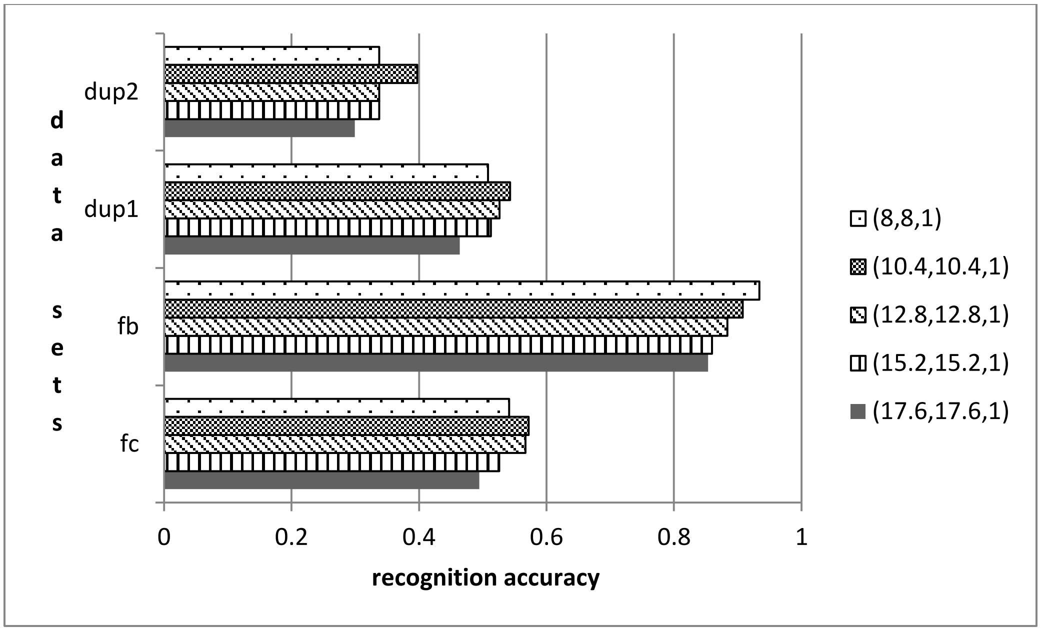


Figure 34. Results for different parameters of 1×8 Gabor filters

From the results above, the highest recognition accuracy is achieved at a wavelength of 10.4. And for most of the wavelength, the smaller wavelength leads to higher recognition accuracy. This is mainly due to the number of feature descriptions. In this method, the points are selected one wavelength apart, so a smaller wavelength means smaller margin and more points.

Another interesting point is that in this case, as wavelength decreased to 8, the recognition accuracy is reduced. This means that the number of descriptions is important in face recognition, but the length is also of importance in this process.

4.3.2.2. Weighted Strategy of Gabor Filters

Based on the interesting result presented in Section 4.3.1.5, we decided to apply the overall weight strategy to the Gabor descriptors with parameters (6,18,5) and (8,20,5) . kNN with Chi square is used as the classifier. The results are shown in Table 13:

Table 13. Comparison of weighted and unweighted method of Gabor filters

	fc	fb	dup1	dup2
(6,18,5)-weighted	61.86%	**96.49%**	54.16%	46.15%
(6,18,5)-unweighted	64.43%	92.47%	57.20%	40.60%
(8,20,5)-weighted	63.92%	95.56%	54.99%	**47.44%**
(8,20,5)-unweighted	**65.46%**	91.63%	**58.03%**	42.31%

At first sight, the weighted method does not provide the best results, but it is noticeable that:

- After using the weighted method, fb and dup2 achieves the highest accuracy among all the parameter sets;
- Though the recognition accuracy is reduced after using weighted classification, there is not as severe decrease as the increase in fb and dup2.

The reason for the failure of the weighted method for fc and dup1 might be improper weight values. As for LBP, the value of the histogram belongs to a range of [0,1], so a weight of 4 or 2 would be suitable for enhancing some segments. For Gabor filters, the values can be more than 5, and when this large value is multiplied by a weight of 4 or 2, some noisy feature information might be exaggerated.

4.4. Experimental Result for Ensemble Classification

4.4.1. Modified Borda and Ranking of Segment

In this method, we postulate that the candidates with higher ranking should have higher point values. To address the importance of higher ranking, we gave more points, when compared with the original Borda count algorithm, to the candidates at higher positions. We use a parameter range from 0.01 to 0.05. The experiments are done on images after histogram equalization, and used $LBP_{8,2}^{u2}$ as feature descriptors. We analyzed this method on kNN with Euclidean distance and Chi-square distance (Table 14 and Figure 35). And the bad-contribution segments obtained from overall weight strategy were not included in this experiment.

Table 14. Results for Ensemble classification with single segments of LBP

	fc	fb	dup1	dup2
euclidean+Borda	48.97%	95.73%	52.63%	38.46%
euclidean+Borda(0.001)	54.64%	96.32%	55.26%	41.88%
euclidean+Borda(0.005)	54.64%	96.40%	55.82%	41.45%
chi+Borda	55.15%	98.24%	59.97%	45.73%
chi+Borda(0.001)	64.95%	98.83%	61.36%	47.01%
chi+Borda(0.005)	**65.98%**	**99.00%**	**61.63%**	**47.01%**

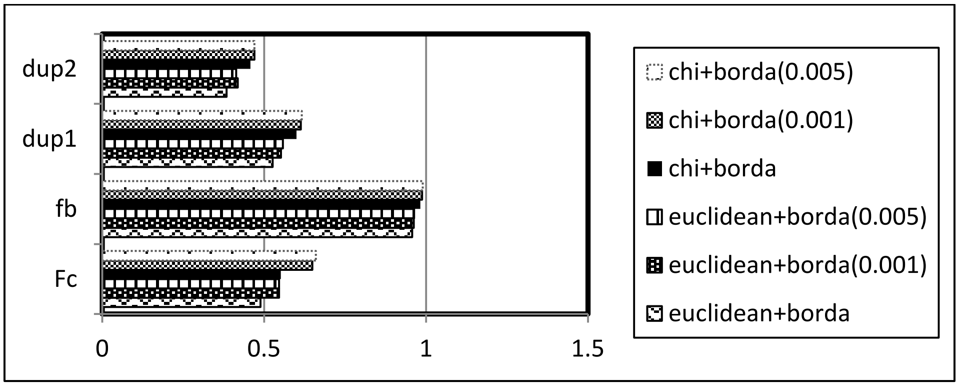


Figure 35. Results for Ensemble classification with Single Segments of LBP

In this case, the highest parameter is kNN with Chi square statistics under 0.005. From the comparison, the modified Borda produces a higher accuracy rate than that of the original Borda.

4.4.2. Weighted Borda on Multi Segments

According to the previous study, some features or some segments may be more important than others. To study the effect of weight strategy, we can combine the weight information before or during the aggregation.

In this experiment, the number of combined neighbourhood segments is set to be 4 or 8 of the 8×10 segments. The combination is shown in Figure 36. The left is combined by a neighborhood of 4 segments, and the right shows the combination of 8 neighborhoods.

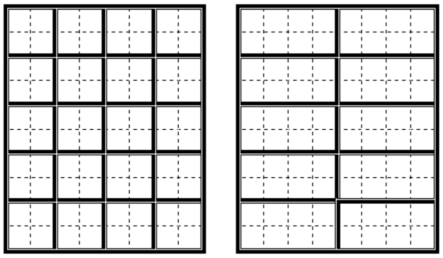


Figure 36. Examples of combination the neighborhood segments. The left is combination of a neighborhood of 4 segments; the right is combination of a neighborhood of 8 segments

The images with histogram equalization were chosen as the dataset, and $\mathrm{LBP}_{8,2}^{u2}$ as feature descriptors. We analyzed this method using kNN with Chi-square distance. Table 15 contains detailed information of the obtained results.

Table 15. Results for weighted Borda on a neighborhood of 4 or 8

	fc	fb	dup1	dup2
Weight-before-4	56.19%	97.91%	**64.68%**	47.44%
weight-during-4	61.54%	**98.74%**	60.11%	**47.86%**
nonweight-4	55.46%	96.57%	59.56%	39.74%
weight-before-8	61.34%	97.91%	61.50%	47.01%
weight-during-8	55.67%	97.74%	57.89%	41.03%
noweight -8	**62.89%**	95.48%	57.89%	35.04%
Original Borda	57.22%	96.49%	57.76%	36.75%

From this table, it can be extracted that the results obtained with the weighted Borda count are better than the results obtained with the original Borda count algorithm. The recognition accuracy is higher than accuracy with the original no-weight and no-combination Borda algorithm. The results for fb, dup1 and dup2 show that the highest accuracy is achieved when combining 4 neighborhood segments. As for fc, though the weighted Borda count method improved the recognition accuracy while combining 4 neighborhood segments, the result is still not higher than that of simply combining 8 neighborhood segments.

4.4.3. Borda Count on Selected Classifiers

In this section, the target is to study how the Borda count on selected classifiers can affects the classification process. In our experiment, we continue to use the images with histogram equalization, partitioned each image into 8×10 segments, and used$\boldsymbol{LBP_{8,2}^{u2}}$ as feature descriptors. To meet the objective, a number of classifiers in an ensemble classification approached are being increased by 10 on each step. The sequence to add the every 10 classifiers were defined according to their face recognition accuracy obtained in the training stage. That means the top 10 accurate classifiers were added the first time; and later the top 20 accurate classifiers were used for ensemble classification; the process went on until all 80 classifiers were added. Table 16 and Figure 37 show the results for gradually added 10 classifiers to the classification process. We also added the modified Borda with parameter of 0.05 as a comparison.

Table 16. Results for selected classifiers with Borda and modified Borda

	fc	modified-fc	fb	modified-fb
10 classifiers	26.80%	39.18%	93.81%	95.98%
20 classifiers	32.99%	48.45%	98.24%	98.41%
30 classifiers	48.97%	56.70%	98.24%	98.74%
40 classifiers	52.58%	59.28%	98.33%	98.66%
50 classifiers	54.64%	63.92%	98.33%	98.74%
60 classifiers	56.19%	64.95%	**98.41%**	**99.16%**
70 classifiers	**59.79%**	**68.56%**	97.57%	98.74%
80 classifiers	57.22%	63.40%	96.49%	97.82%
	dup1	modified-dup1	dup2	modified-dup2
10 classifiers	32.69%	38.78%	29.49%	33.76%
20 classifiers	52.63%	57.76%	40.17%	44.87%
30 classifiers	52.08%	54.99%	36.75%	39.74%
40 classifiers	52.91%	56.37%	35.90%	41.03%
50 classifiers	58.59%	60.25%	44.87%	**47.86%**
60 classifiers	60.53%	61.63%	**46.58%**	47.44%
70 classifiers	**60.80%**	**62.60%**	45.30%	**47.86%**
80 classifiers	57.76%	59.42%	36.75%	38.03%

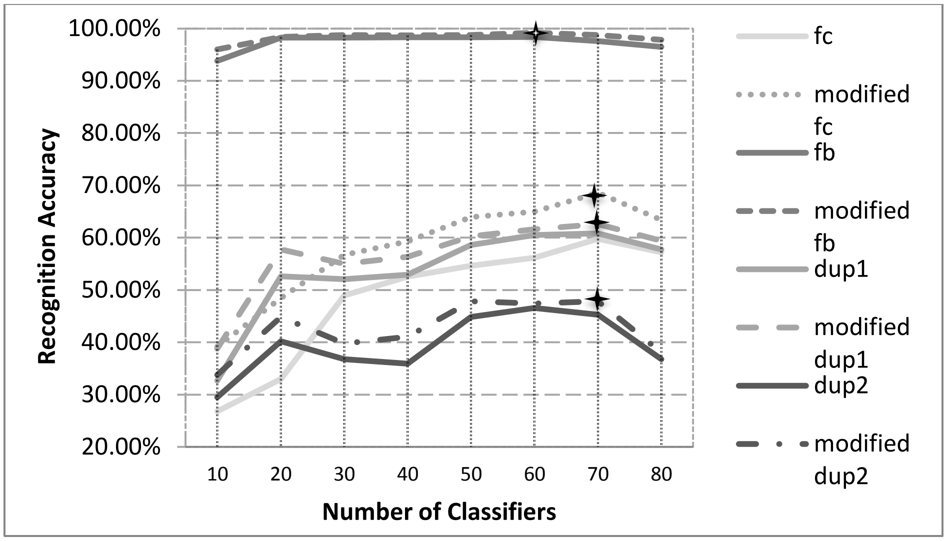


Figure 37. Results for selected classifiers with Borda and modified Borda

According to Table 16 and Figure 32, the highest accuracy is achieved with around 60-70 classifiers. At the beginning, the recognition accuracy gradually increased as the number of classifiers increased; and when it reached around 60 or 70 classifiers, the accuracy rate reached its peak and began to drop a little. These results support our previous assumption, that the information brought by some segments might introduce noise to the classification process.

Another obvious conclusion is that the modified Borda does improve the classification accuracy. For every dataset of any number of classifiers, the modified Borda count algorithm works better than that of the original Borda count algorithm.

The sequence of adding classifiers is very important in the process, so we would like to focus on the sequence. By studying the component of training set S, we find that there are 1000 images, but the set does not contain images of fc and dup2. Of all 1000 images, S includes 270 images from fb, 190 images from dup1, 270 images from gallery, and the remaining images were not included in either fc or dup2 data sets. So, the provided training sets might not be so representative of testing images. Since dup2 is a subset of dup1, it is reasonable to use the information derived from this training dataset. However, the information derived from this training dataset might be improper for fc. In order to add classifiers in a reasonable and proper sequence, we built another training set *S'* which includes ¼ of the images from fc, and used the remaining ¾ images for testing. With this training set S', we can gain another rank list $\boldsymbol{l'}$ of segments according to their recognition accuracy. Besides this list, we also have another rank list *l* derived from the previous training set *S*. In order to obtain a reasonable rank list, we used the Borda count to aggregate rank lists *l* and *l'* to generate a final rank list *L*. With this rank list, we gradually added 10 classifiers each step. The result is shown in Table 17 and Figure 38.

Table 17. Results for selected classifiers under new sequence with Borda and modified Borda

	fc	modified fc	fb	modified fb
10 classifiers	**68.56%**	**76.80%**	95.23%	96.74%
20 classifiers	63.40%	72.68%	97.66%	97.91%
30 classifiers	61.34%	72.16%	97.99%	98.41%
40 classifiers	65.98%	**76.80%**	97.99%	98.58%
50 classifiers	67.01%	76.29%	98.08%	98.58%
60 classifiers	59.28%	72.68%	**98.33%**	**98.91%**
70 classifiers	60.31%	68.56%	97.66%	98.41%
80 classifiers	57.22%	63.40%	96.49%	97.82%
	dup1	modified dup1	dup2	modified dup2
10 classifiers	43.63%	48.75%	39.32%	42.31%
20 classifiers	46.68%	51.80%	30.77%	36.75%
30 classifiers	49.31%	51.39%	32.48%	33.33%
40 classifiers	53.19%	55.12%	36.32%	38.46%
50 classifiers	57.62%	59.28%	40.17%	41.88%
60 classifiers	**59.14%**	**61.22%**	**42.31%**	**44.44%**
70 classifiers	59.00%	61.08%	40.17%	44.02%
80 classifiers	57.76%	59.42%	36.75%	38.03%

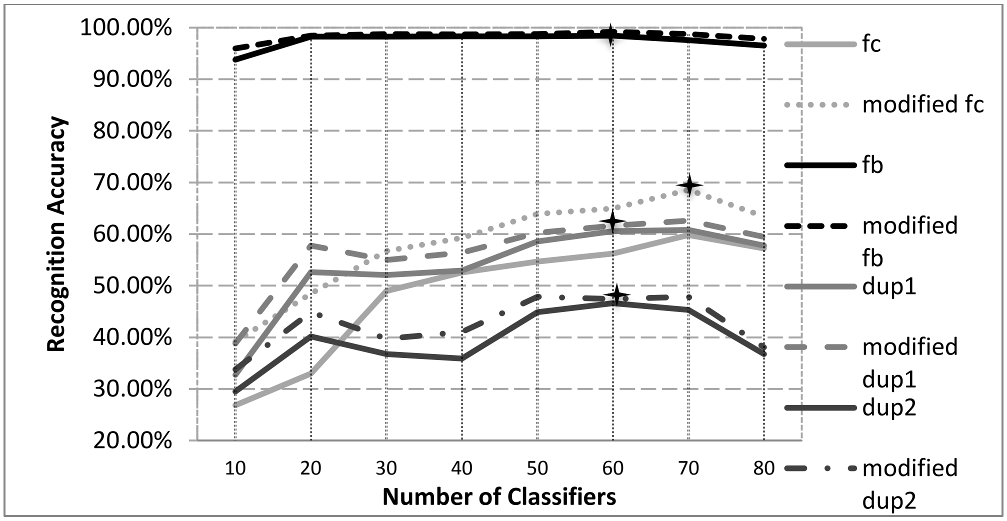


Figure 38. Results for selected classifiers under new sequence with Borda and modified Borda

Similar conclusions can also be gained from the results above. But one significant difference is that after applying the new sequence, adding only a few classifiers provides good recognition accuracy. This is especially true for fc, which achieves its highest accuracy with only 10 classifiers.

4.5. Experimental Results for Ensemble Classification with Different Number of Partitions

In this section, we would like to perform an ensemble classification of images partitioned into different number of segments. The aim of this experiment was to see how Borda works under different scales. In this experiment, we first partitioned the image into 4×5, 8×8, 10×10, 12×15 segments. $LBP_{8,2}^{u2}$ is used as the feature descriptor, overall weight strategy is also applied, and kNN with chi square as the classifier for each partition is used. After obtaining the rank lists of the 4 partitions, we aggregated their results with the previously obtained rank list for 8×10 partitions. The aggregation was performed using Borda count algorithm. The results are shown in Table 18 and Figure 39.

Table 18. Results for different scale and ensemble classification

	fc	fb	dup1	dup2
8×10	73.71%	98.91%	64.40%	**53.42%**
4×5	35.05%	96.90%	46.81%	21.79%
8×8	58.25%	**99.00%**	57.06%	36.75%
10×10	75.77%	**99.00%**	64.82%	50.85%
12×15	**78.87%**	**99.00%**	**65.79%**	51.71%
Ensemble of all	67.53%	98.83%	60.11%	32.05%

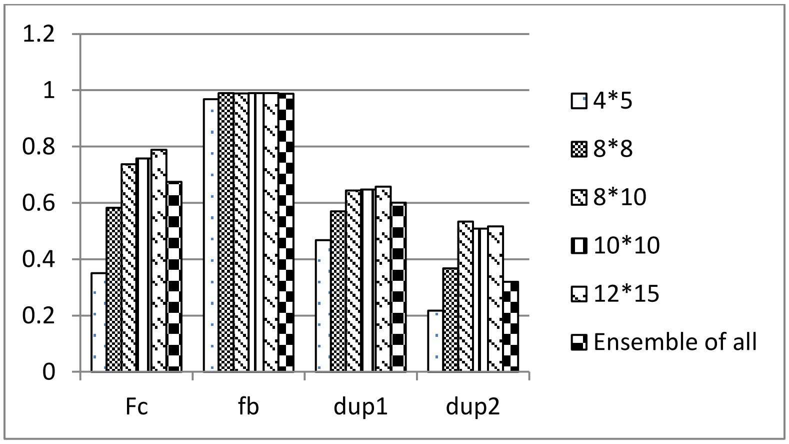


Figure 39. Results for different scale and ensemble classification

From the results above, we can see that when ensemble of rank lists obtained for different partitions is used, the results are comparable with the accuracy values obtained for individual partitions. Possible improvement can be obtained when different descriptors are used to represent image with different partitions.

5. Conclusion

5.1. Contribution

This thesis was a comparative study of main steps in a face recognition process. We start with the construction of an overall schema, go through different face description techniques and various classification processes for different descriptors, and end with modifications of classification algorithms in order to seek the best approaches to face recognition. We also present a comprehensive comparison how different image processing technologies influence the final classification results.

The first and main contribution is a systematic investigation of multiple methods based on different descriptors to improve accuracy of a classification process. Given the incoherency between global feature descriptors and local feature descriptors, we applied different strategies:

1) Global feature descriptors reduce dimensionality of an original feature space and construct a more representative subspace. We studied and compared kNN as a classification method used with different distance measures. This process shows the influence of distance measures on the final results.

2) Local feature descriptors generate several sub-segment or point descriptions, which can be combined in multiple ways to build a description of the whole image. There are two combination approaches used to transform an image representation from a sub-segment or point level to a global level. With different approaches, we obtain different improvements in classification. The two main approaches used in the thesis are:

   a. Concatenation of sub-segment or point descriptions. The concatenated local descriptions are passed to a single classifier. In a classification process, we consider different distance measures and different weight strategies. The experimental results show that these modifications provide an improvement in a classification process.

b. Construction of an ensemble classifier that contains multiple classifiers. Each of them is “tied” with a single sub-segment description. The Borda count algorithm is used to aggregate the results obtained from individual classifiers. Based on our careful observation of the Borda count algorithm, we propose a modified Borda count algorithm. Another improvement is inspired by weight strategies used in the single classification approach (point a above). We apply the weighted Borda count approaches to focus only on more prominent parts of images, and eliminate the effect of possible noise.

The second contribution of this thesis is to provide a reference to study how different face descriptors influence a classification process. In this thesis, basic concepts and general schemes of local descriptors and global descriptors are studied and illustrated with examples and experiments. Further study of applying various face feature descriptors to the classification process can be conducted based on the results presented here.

An investigation of the usefulness of an ensemble classification approach is the third contribution. The conducted experiments show that some face image information may have a negative impact on the recognition process. Based on the experiments with a number of selected classifiers in the ensemble classification, we observe that the best results are obtained when only some image segments are used.

5.2. Future Work

Based on the experimental work conducted here, we can identify a few open research problems and possible directions for future research.

Because the main emphasis of the thesis is on the classification stage, the local descriptors used in this work are the basic ones. Thus, improving face recognition accuracy through application of other descriptors is an interesting direction that can be investigated.

In the case of an ensemble classification, we conducted experiments with local descriptors with a limited variation in LBP parameters' values. It would be interesting to investigate a classification process with different parameter values of Gabor filters used to construct local descriptions. Also, an application of different classifiers would be another interesting research direction. Selection of the best classifiers for different descriptors and then aggregating their classification results could also lead to improvement in the classification process.

In this thesis, we mainly used one weighted strategy in both single classification and ensemble classification approaches. It shows great improvements when compared with the results obtained with the original recognition process. Therefore, another interesting research direction could be the exploration of different weighting schemas for local descriptors.

The Borda count algorithm is a noteworthy approach to aggregation and shows some improvements when applied to a face recognition process. However, more studies and investigations of other aggregation methods should be conducted.

Bibliography

[1] Zhao, W.; Chellappa, R.; Phillips, P.J.; Rosenfeld, A., "Face recognition: A literature survey," *ACM Computing Surveys,* vol. 35, no. 4, pp. 399 - 458, 2003.

[2] Sirovich, M. Kirby and L., "Application of the Karhunen–Loeve procedure," *IEEE Trans. Pattern Anal. Mach. Intell.,* vol. 12, no. 1, p. 103–108, Jan. 1990.

[3] Phillips, P.J., Moon, H., Rizvi, S.A., Rauss, P.J.:, "The FERET evaluation methodology for face recognition algorithms," *IEEE Trans.Pattern Anal. Mach. Intell.,* vol. 22, no. 10, pp. 1090 - 1104, 2000.

[4] Xie, S., Shan, S., Chen, X., and Chen, J., "Fusing Local Patterns of Gabor Magnitude and Phase for Face Recognition," *IEEE Transactions on Image Processing,* vol. 19, no. 5, pp. 1349 - 1361, 2010.

[5] Jie Zou,Qiang Ji, "A Comparative Study of Local Matching Approach for Face Recognition," *IEEE TRANSACTIONS ON IMAGE PROCESSING,* vol. 16, no. 10, pp. 2617-2627, 2007.

[6] I. Craw, N. Costen, Y. Kato, G. Robertson, and S. Akamatsu, "Automatic face recognition: Combining configuration and texture," *Proc. Int. Workshop Automatic Face and Gesture Recognition,* p. 53–58, 1995.

[7] M. Turk, A. Pentland, "Eigenfaces for Recognition," *Journal of Cognitive Neurosicence,* vol. 3, no. 1, pp. 71-86, 1991.

[8] K. Etemad, R. Chellappa, "Discriminant Analysis for Recognition of Human Face Images," *Journal of the Optical Society of America A,* vol. 14, no. 8, pp. 1724-1733, August 1997.

[9] P.N. Belhumeur, J.P. Hespanha, D.J. Kriegman, "Eigenfaces vs. Fisherfaces:

Recognition using Class Specific Linear Projection," *IEEE transations on pattern analysis and machine intelligence,* vol. 19, no. 7, pp. 336-341, July 1997.

[10] M.S. Bartlett, J.R. Movellan, T.J. Sejnowski, "Face Recognition by Independent Component Analysis," *IEEE Trans. on Neural Networks,* vol. 13, no. 6, pp. 1450-1464, November 2002.

[11] C. Liu, H. Wechsler, "Comparative Assessment of Independent Component Analysis (ICA) for Face Recognition," *Proc. of the Second International Conference on Audio- and Video-based Biometric Person Authentication, AVBPA'99, 22-24,* pp. 211-216, March 1999.

[12] M. Fleming, G. Cottrell, "Categorization of faces using unsupervised feature extraction," *Proc. IEEE IJCNN Int. Joint Conf. on Neural Networks,* vol. 2, p. 65–70., 1990.

[13] J.-M. F. N. K. C. v. d. M. L. Wiskott, "Face Recognition by Elastic Bunch Graph Matching," *Chapter 11 in Intelligent Biometric Techniques in Fingerprint and Face Recognition, eds. L.C. Jain et al., CRC Press,* pp. 355-396, 1999.

[14] Zhang, B., Shang, S., Chen, X., Gao, W.:, "Histogram of Gabor Phase Pattern (HGPP): A novel object representation approach for face recognition," *IEEE Trans.on Image Processing,* vol. 16, no. 1, pp. 57 - 68, 2007.

[15] Xie, S., Shan, S., Chen, X., and Chen, J.:, "Fusing Local Patterns of Gabor Magnitude and Phase for Face Recognition," *IEEE Transactions on Image Processing,* vol. 19, no. 5, pp. 1349 - 1361, 2010.

[16] Ojala T, Pietikäinen M & Harwood D, "A comparative study of texture measures with classification based on featured distribution.," *Pattern Recognition,* vol. 29, no. 1, pp. 51-59, 1996.

[17] Albiol, A., Monzo, D., Martin, A., Sastre, J., Albiol, A., "Face recognition using HOG–EBGM," *Pattern Recognition Letters 29*, pp. 1537 - 1543, 2006.

[18] Ojala, T., Pietikainen, M., Maenpaa, T., "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 971-977, 2002.

[19] P. J. Phillips, H. Wechsler, J. Huang, and P. Rauss, "The FERET database and evaluation procedure for face recognition algorithms," *Image Vis. Comput. Journal*, vol. 16, no. 5, p. 295–306, 1998.

[20] P. J. Phillips, H. Moon, S. A. Rizvi, and P. J. Rauss, "The FERET evaluation methodology for face-recognition algorithms," *IEEE trans Pattern Anal Machine Intelligence*, vol. 22, no. 10, p. 1090–1104, Oct 2000.

[21] Zhang, B., Shang, S., Chen, X., Gao, W.:, "Histogram of Gabor Phase Pattern (HGPP): A novel object representation approach for face recognition," *IEEE Trans.on Image Processing*, vol. 16, no. 1, pp. 57 - 68, 2007.

[22] Guo, Y., Xu, Z., "Local Gabor Phase Difference Pattern for Face Recognition," *Pattern Recognition, 2008. ICPR 2008. 19th International Conference*, pp. 1-4, 2008.

[23] Zhang, W., Shan, S., Zhang, H., Gao, W., Chen, X., "Multi-resolution histogram of local variation patterns (MHLVP) for Robust Face Recognition," *AVBPA 2005. LNCS*, vol. 3546, pp. 937-944, 2005.

[24] R. Beveridge, D. Bolme, M. Teixeira, and B. Draper, "The CSU Face Identification Evaluation System User’s Guide: Version 5.0," CO: Comput. Sci. Dept., Colorado State University, Denver, 2003.

[25] Y. Rubner, J. Puzicha, C. Tomasi, and J. M. Buhmann, "Empirical evaluation of dissimilarity measures for color and texture," *Comput. Vis. Image*

Understand., vol. 84, no. 1, p. 25–43, 2001.

[26] Cover TM, Hart PE, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21-27, 1967.

[27] R., Polikar, "Ensemble Based System in Decision Making," *IEEECircuits and System Magazine*, vol. 6, no. 3, pp. 21-45, 2006.

[28] J.W., Tukey, Exploratory data analysis, Mass: Addison-Wesley, Reading, 1977.

[29] B.V., Dasarathy, "Composite classifier system design: Concepts and methodology," *Proceedings of the IEEE*, vol. 67, no. 5, pp. 708-713, 1979.

[30] J., Hansen, "Combining Predictors. Mata Machine Learning Methods and Bias/Variance & Ambiguity Decompositions," *PhD dissertation. Aurhus University*, 2000.

[31] CHO, Sung-Bae, and Jin H. KIM, "Multiple Network Fusion Using Fuzzy Logic.," *IEEE Transactions on Neural Networks*, vol. 6, no. 2, p. 497–501, 1995.

[32] KUNCHEVA, Ludmila I., "Diversity in Multiple Classifier Systems," *Information Fusion*, vol. 6, no. 1, p. 3–4, 2005.

[33] KUNCHEVA, Ludmila I., and Lakhmi C., "Designing Classifier Fusion Systems by Genetic Algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 4, p. 327–336, 2000.

[34] Opitz, D., "Feature Selection for Ensembles," *In Proc. 16th National Conf. On Artificial Research*, pp. 11:169-98, 1999.

[35] Penrose, LS, "The elementary statistics of majority voting," *Journal of the Royal Statistical Society*, 1946.

[36] W, Buntine, A theory of learning classification rules, Sydney. Australia:

Doctoral Dissertation. School of Computing Science, University of Technology, 1990.

[37] DL Mon, CH Cheng, Evaluating weapon system using fuzzy analytic hierarchy process based on entropy weight- Fuzzy sets and systems, Elsevier, 1994.

[38] J. Friedman, T. Hastie, and R. Tibshirani, "Additive logistic regression:A statistical view of boosting," *Ann. Statist.28,* vol. 28, no. 2, p. 337–374, 2000.

[39] J. J. H. a. S. N. S. T. K. Ho, "Decision combination in multiple classifier systems," *IEEE Trans. Pattern Anal. Mach. Intelligence*, vol. 16, no. 1, p. 66–75, Jan. 1994.

[40] Brown, G. and Wyatt, J. and Harris, R. and Yao, X, "Diversity creation methods: a survey and categorisation," *Information Fusion,* vol. 6, no. 1, pp. 5-20, 2005.

[41] J. J. García Adeva, Ulises Cerviño, and R. Calvo, "Accuracy and Diversity in Ensembles of Text Categorisers," *CLEI Journal,* vol. 8, no. 2, pp. 1 - 12, December 2005.

[42] Ahonen, T., Hadid, A. and Pietikainen, M., "Face recognition with local binary patterns," *Proc. 8th European Conference on Computer Vision, ser. Lecture Notes in Computer Science, Springer,* vol. 3021, pp. 469 - 481, 2004.

[43] Poggio, P. Sinha and T., "I think I know that face…," *Nature,* vol. 384, p. 831–836, Aug. 1996.