2001

A high-accuracy offline handwritten numeral recognition system.

Xinyu. Ye
University of Windsor

A HIGH ACCURACY OFFLINE HANDWRITTEN NUMERAL RECOGNITION SYSTEM

by

Xinyu Ye

A Thesis
Submitted to the Faculty of Graduate Studies and Research
through Electrical and Computer Engineering
in Partial Fulfillment of the Requirements for
the Degree of Master of Applied Science at the
University of Windsor

Windsor, Ontario, Canada

August 2001

0-612-67611-0

945362

ABSTRACT

Handwritten numeral recognition has been confronted with the problems of recognizing infinite varieties of patterns produced from writers with different writing habits, styles, and artistic flavors. As one of the most important topics in pattern recognition, there has been, and still is a significant performance gap between human beings and machines since the late 1960s.

The primary objective of this research is to develop a high accuracy offline handwritten numeral recognition system. This thesis focuses on the architecture and performance improvement of handwritten numeral recognition systems through proper preprocessing, feature extraction, classifier design and combining different classifiers. Hybrid architectures of recognition systems are proved to be a very efficient method in recent research. This thesis proposes a multi-stage and multiexpert classification method integrated with complementary extracted features. It consists of a ruled-based classifier for one feature and neural network classifiers for all features. The final result is made from the fuzzy integral fusion of the outputs from the neural network classifiers. The experiments show that the approach achieves a better result.

To my family

ACKNOWLEDGEMENTS

I wish to sincerely thank my advisor Dr. M. Ahmadi for his valuable guidance and encouragement throughout the progress of the thesis and the period of my master degree program, and also his support. I would like to thank my department reader, Dr. J. J. Soltis, and external reader, Dr. B. Boufama, for their assistance and advice in completing this thesis.

A great deal of thanks also goes to my family, especially my wife, for her enormous understanding and patience over the last two years and will not be forgotten.

TABLE OF CONTENTS

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

1.1 Handwritten Numeral Recognition System

In automated handwritten document analysis and recognition, handwritten numeral recognition still poses a challenge. The problem is compounded by the inherent variations present within occurrences of the same numeral with respect to size, shape, skew and style. Writing style varies to a great extent from person to person, and formulating a generalized approach which is applicable in recognizing all these occurrences has proved to be a difficult task. Over the years numerous approaches have been proposed and implemented to solve this problem, but an ultimate solution continues to elude researchers [1].

Since the 1960s, a significant amount of research from academia and industry has yielded a multitude of algorithms for normalization, feature extraction and classification for OCR [2, 3]. Research on recognition of unconstrained handwritten characters has made impressive progress and many systems have been developed, particularly in machine-printed and online character recognition because both dynamic and static information can be used. However, there is still a significant performance gap between humans and machines in the recognition of offline handwritten character recognition [4].

Offline handwritten numeral recognition has been a difficult problem for machine learning but a surprisingly easy task for human subjects. It is hard to mimic human classification and reasoning ability, and recent surveys have shown that present technology has still a long way to go in terms if robustness and accuracy [5]. Recent computational advances in hardware and software have stimulated more basic research and development of practical systems. Compared with machine-printed numeral recognition, the prime difficulty in the research and development of handwritten numeral recognition system is in the variety of shape deformations.

The principal function of a numeral recognition system is to make a decision concerning the class membership of a numeral in the input image. There are several important processing steps between input numeral image acquisition and the output class membership decision. Generally, a numeral recognition system can be divided into preprocessing, feature extraction, and classification blocks. Figure 1.1 shows the block diagram of a typical numeral recognition system.

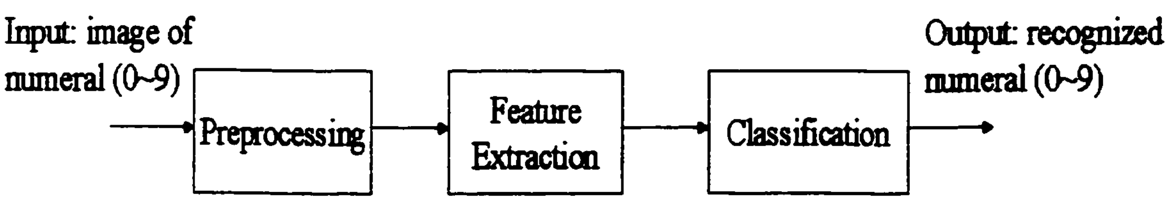


Figure 1.1 Numeral Recognition System

An OCR system often starts with scanning the text image and converting it into electric signal [6]. The text may be corrupted by noise, overlapped by scratches or skewed. Generally, there are two kinds of numerals inside the text: connected and isolated. Various methodologies have been proposed to remove the background noise, deskew the text image [7] and segment the connected characters [8]. The object of this thesis deals with the recognition of handwritten numerals that are isolated.

The preprocessing stage in this thesis includes filtering noise, centering and normalizing, which are necessary for the following stage to recognize handwritten numerals.

The feature extraction stage converts the input numeral image into a vector of numerals. Generally, the input handwritten numerals vary in style and shape, and are subject to different types of distortion, such as scaling, translation, rotation, etc. In feature extraction, vectors of numerals are calculated such that they are similar to classes. The extraction of meaningful features is quite a major task in the design of a recognition system. In practice, none of the feature sets developed so far can perform well under all circumstances. Currently, feature set design is an art rather than a science. Varieties of approaches have been proposed to try to capture the distinctive features of characters. The feature extraction methods generally include Global analysis techniques and points distribution features such as template matching [9-11], moments [12, 13], profile projection [14, 15], border chain-code histograms [16], mathematical transformations (Fourier, Walsh, etc.) [17-20] and structural analysis techniques such as loops, end points, junction arcs, concavities and convexities, etc [21-25].

Structural and topological features use geometrical and structural properties to represent handwritten numerals. They are generally difficult to compute and often sensitive to small distortions. Transformation features can be made generally invariant to various deformations such as translation, rotation and reflection. However, invariant properties can sometimes be disadvantageous in numeral recognition. Transformation features also suffer from their heavy computation burden. Global analysis and points distribution features, on the other hand, are generally less sensitive to distortion and easier to compute. In this thesis, four different point distribution features have been used. Although an individual feature set of this kind is sometimes less separable, the orthogonal property of multiple feature sets can provide a means to overcome this disadvantage [6].

The final step in most character recognition systems is to execute a decision rule. The decision rule can be implemented based on syntactical and statistical techniques. Many researchers have dedicated significant efforts to design classifiers over the past three decades. The classification procedures proposed based on these techniques include nearest neighbor classifiers [26, 27], Bayes classifiers [28], polynomial classifiers [10, 11], Relaxation matching [21], tree classifiers [15, 29], model-based classifiers [30] as well as different kinds of syntactic classifiers [22, 31-33]. However, most commonly used classifiers become optimal only when the amount of training data becomes infinite, and feature statistics are perfectly known. When a finite number of training data is available, classifiers are often not optimal. In this thesis, the artificial neural network approach has been chosen for the major classifier design since it represents a computational paradigm that offers capabilities for solving many difficult problems and has been shown as a universal approximator being able to approximate Bayes decisions [34-36].

1.2 Complementary Feature Extractions

In the task of handwritten numeral recognition, stand-alone experts are often not robust enough to tackle the problem of the huge degree of variation that is present in the samples with respect to size, shape, skew and style. The employment of multiexperts allows more and more features be taken into account, so the overall characteristics of the numeral become clearer. The approach can exploit the strengths of individual experts and can often avoid their weaknesses, making the overall final decision more robust.

The usage of complementary feature extraction methods is due to the requirement of multi-expert recognition algorithm. The advantages of the multi-expert classification method will be described in the next section.

In this thesis, we used the feature extraction method applying four different and complementary point distribution features. Point distribution features are less sensitive to noise, and this method is preferred in pattern recognition application as the differences between different numerals are more significant than the differences between different samples of the same numeral. Three of the four features focus on the different pattern-level characteristics of the image for recognition, so they are complementary. The other extraction method is to draw out the property of the numeral from the view of every pixel in the image. This constructs another complementary form with the pattern-level features; this method is very effective for the recognition algorithm and matches the concept of the whole system design.

Cao [6] used three pattern-level point distribution feature extraction methods with multi-expert neural network classifiers for recognition and obtained decision from the fusion of classifiers.

Favata [37] and Chung [38] also gained better performance by combining complementary features.

1.3 Multi-stage Recognition Algorithm

The multistage classification method focuses on the efficiency of classification. In multistage systems, simple and efficient methods are used to classify well-formed numerals while more complex and costly methods are required for numerals of poorer quality.

Kaynak et al. [39] used a multi-layer perceptron with sigmoidal hidden units as the rule-learner and a k-nearest neighbor classifier as the exception-learner. The cascade algorithm significantly outperforms the individual methods and voting on three optical and pen-based handwritten digit recognition tasks when comparison is based on three criteria: generalization success, learning speed, and number of free parameters. An approximate 95% correct recognition rate was reported.

Duerr et al. [32] reported a remarkable recognition rate achieved with a four-stage classification scheme: a conventional statistical classifier and a fast structural classifier are first applied to all samples, and correctly recognize approximately 95% of them; in the remaining cases, a structural hypothesis reducer and then possible a final heuristic matching stage are invoked. A system performance of 99.5% correctly recognized numerals was reached.

Lam and Suen [21] used convex polygon primitives obtained from the skeletons of numerals; 80% of the data were handled with structural classification based on features extracted from these primitives and the remaining 20% were processed further using relaxation matching. The recognition rate was about 97%.

This thesis proposes an approach for a multi-stage classification method by applying a rule-based classifier and several neural network classifiers. This approach achieves a high recognition rate, and at the same time improves the recognition speed.

1.4 Multi-expert Recognition Algorithm

In recent years, the concept of combining multiple classifiers has been proposed as a new direction for the development of a high accuracy character recognition system [40]. Some preliminary results have indicated that the combination of several complementary classifiers can significantly improve the performance of individual classifiers [21, 32, 40-51]. In a multiple classifier system, the outcomes of many different methods are combined using voting schemes or some more complex rules. The purpose is to combine algorithms to reinforce the strengths and compensate for the weaknesses of the participating classifiers by using more extensive and complementary sources of information.

In multi-expert systems, the outcomes of many methods are combined using majority voting or some more complex rule. Here the rationale is to design and combine algorithms to reinforce each other, thus relying on more extensive, diversified, and complementary sources of information concerning the problem to be solved.

Ho, Hull and Srihari [46] have proposed a multiple classification system, where the decision of each individual classifier is presented as a ranking of classes. The rankings are then combined by reduction and reordering. An intersection method and a union method are used for reduction. Three methods, namely highest rank, Borda count and logistic regression, are used for reordering. Srihari [44] has used the above multiple classification system for postal address recognition. Five neural network classifiers, each using a different type of feature have been combined. A result of 96% correct rate within the top ten choices was reported.

Suen et al. [40] proposed a high accuracy recognition system for unconstrained handwritten numerals that combined the results of four experts. The features used by the four experts are structural features from the skeleton, primitives extracted from the skeleton, database of the frequency of occurrence of features during training and structural features from contours. Different classification methods are applied by the experts including decision tree and relaxation matching processes. The final decision of the multiexpert system is defined by a majority voting rule. A typical performance of 92.95% recognition rate, 2.15% substitution rate and 4.90% rejection rate has been reported.

Xu, Krzyzak and Suen [41] divided the problem of combining classifiers into three categories according to the levels of information available from the various classifiers. Four approaches based on different methodologies including Dempster-Shafer formalism, Bayesian formalism and voting principles have been proposed. Performances on the U.S. zipcode database were reported as 98.9% correct rate with 0.90% substitution and 0.2 rejection as well as 95% recognition, 0% substitution and 5% rejection.

Kimura and Shridhar [49] combined two algorithms to achieve high performance of recognition of handwritten numerals. The first algorithm uses directional chain code histograms in the divided zones as features and a modified quadratic discriminant function as the classifier. The second algorithm is a tree classifier based on structural features extracted from the left and right profiles of the numerals. Different simple schemes of combination including two parallel combinations and four sequential combinations of these algorithms are examined. The reported results were obtained with a parallel combination: recognition rates of 96.23%, 95.08% and 89.55% were obtained with corresponding substitution rates of 0.25%, 0.13% and 0.07%.

Most of the existing approaches of combining multiple classifiers make use of only the abstract level or rank level information that the individual classifiers provided. However, to make full use of the power of individual classifiers, the measurement level information should be considered. In this thesis, an evidence fusion technique based on the notation of fuzzy integral is utilized for combining multiple classifiers. This technique uses measurement level information of the individual classifiers. To use this fusion technique, an algorithm of classifier relevance is employed. This algorithm not only uses the information of overall performance of each classifier for each class, which ensures the best classifier is chosen for each class, but also uses the confusion information among classes to modify the global degree of importance when local conflicting evidence occurs. Thus, the discord misclassification cases that are made by individual classifiers may effectively be left out when fusion is performed.

1.5 The Scope of the Thesis

The primary objective of this research is to develop a robust, efficient and accurate offline handwritten numeral recognition system. The thesis focuses on the architecture and performance improvement of the handwritten numeral recognition system through proper preprocessing, feature extraction, classifier designing and combining different classifiers. These issues are addressed and described in the following chapters. In chapter 2, data collection and preprocessing are presented, respectively. In chapter 3, the various methods of feature extraction used in this thesis are described, along with the classification algorithms for the recognition of the handwritten numerals. Chapter 4 presents a method of combining classifiers. Chapter 5 contains the conclusions and future directions.

The blocks of the designed numeral recognition system are shown in Figure 1.2.

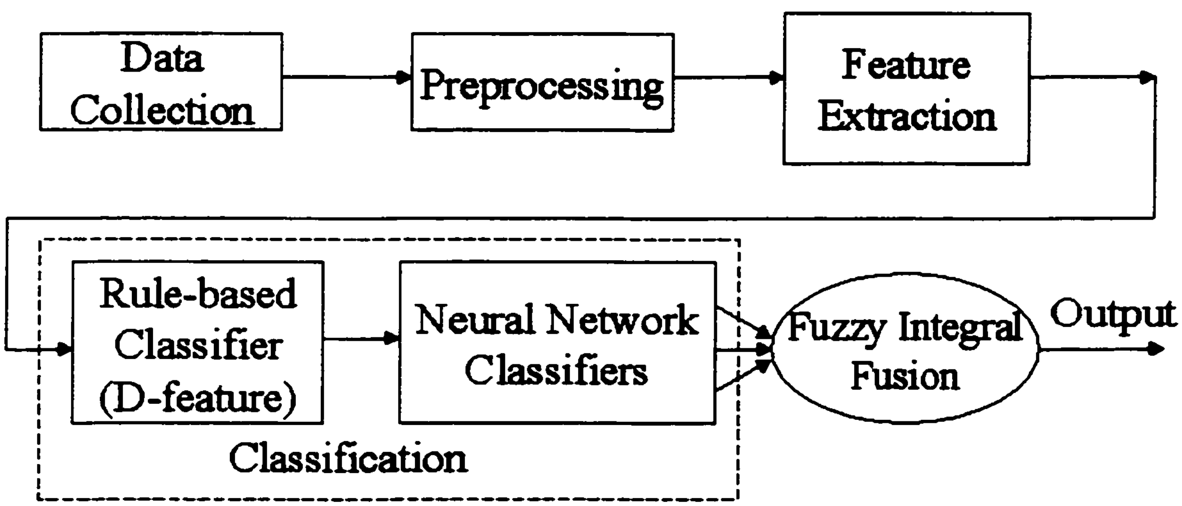


Figure 1.2 Blocks of the Designed Numeral Recognition System

CHAPTER 2

DATA COLLECTION AND PREPROCESSING

2.1 Data Collection

The data collection has an important impact on the handwritten numeral recognition. The data should be natural inputs. The handwritten numeral samples used in this study were collected from different sources.

2.1.1 Alcatel Data

In this data set, there are 15,700 handwritten numeral samples. These samples were supplied by CGA-Alcatel, a French company, and were collected from a large population. In the process of collecting samples, each participating individual was given a sample collection form with spaces provided for writing numerals. The forms were then processed by a scanner at 300 dpi to yield a binary image, which was subsequently stored in a compressed form in a computer. Each numeral occupied a field that measured 64 (width) $\times$ 80 (height) pixels. However, the size of the real numeral inside the field is widely varied. The width of the numerals varied from 6 pixels (for digit '1') to 64 pixels. The height of the numerals varied from 25 pixels to 80 pixels. Figure 2.1 shows examples of the digitized numeral data of Alcatel data set.

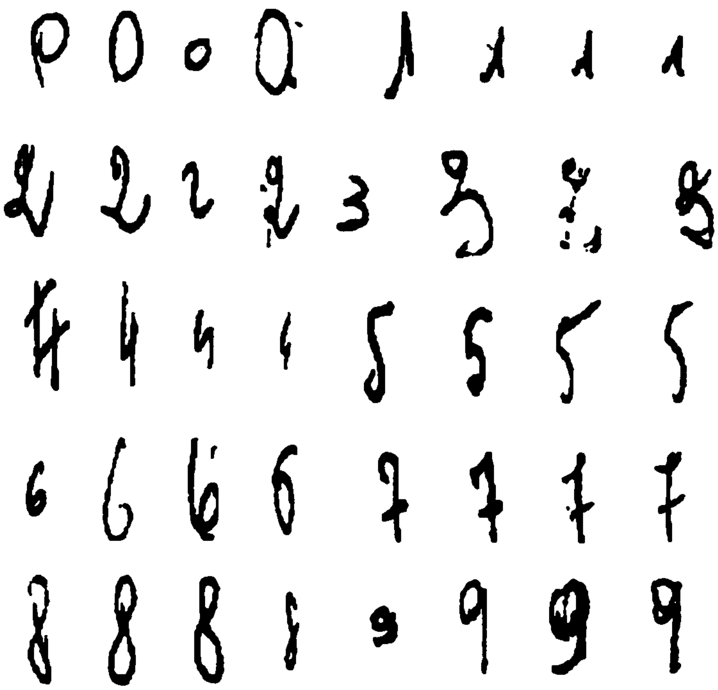


Figure 2.1 Samples of Handwritten Numerals in Alcatel Data Set

2.1.2 Handwritten Address Block Data

The U.S. postal Service provided handwritten address block (HWAB) data extracted from actual mail pieces. 5400 samples were used in this project. The actual mail pieces were converted to gray scale image files by a scanner at a resolution of 300 dpi. The gray scale images were then binarized by using an adaptive threshold algorithm by Otsu [52]. Through a dedicated manual operation of zip code locating and segmentation, the numeral characters in the numeral strings of zip code were isolated and each character was saved in a single file. Samples of HWAB data are shown in Figure 2.2.

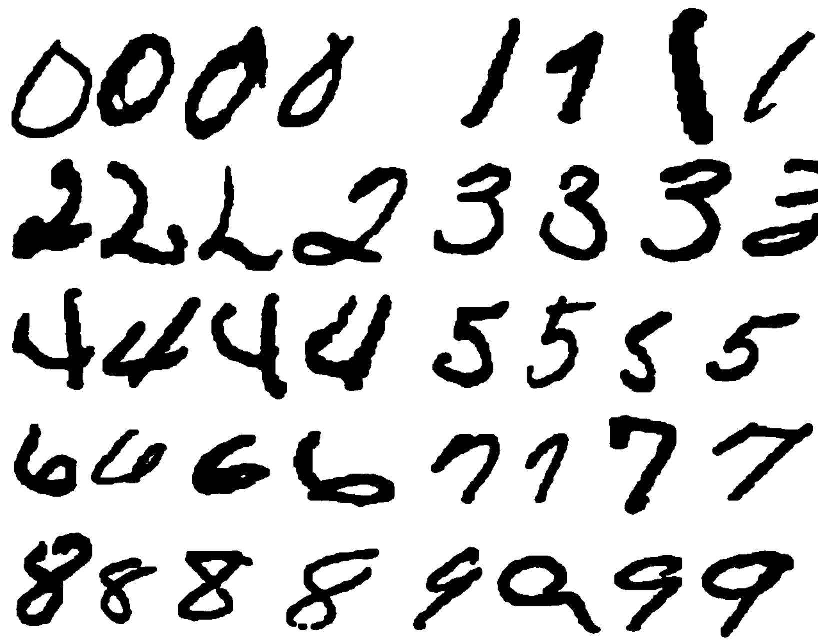


Figure 2.2 Samples of Handwritten Numerals in HWAB Data Set

2.2 Preprocessing

In handwritten recognition systems, a typical problem faced is the variety of handwritten styles that complicate the recognition phase. Variations such as position, size, rotation, inclination and shape distortion are some of the obstacles to be overcome in the preprocessing stage.

Preprocessing is an important phase within a pattern recognition process to alleviate the effect of variation and noise. Basically, a preprocessing stage would reduce these variations. At the same time, preprocessing cannot be overdone since it would destroy some useful information and bring some noise, and the important recognition work should be done by the following stages.

Figure 2.3 is an example about loss of information because of thinning processing. After the processing, the '8' could be mistaken for a '9' [40].

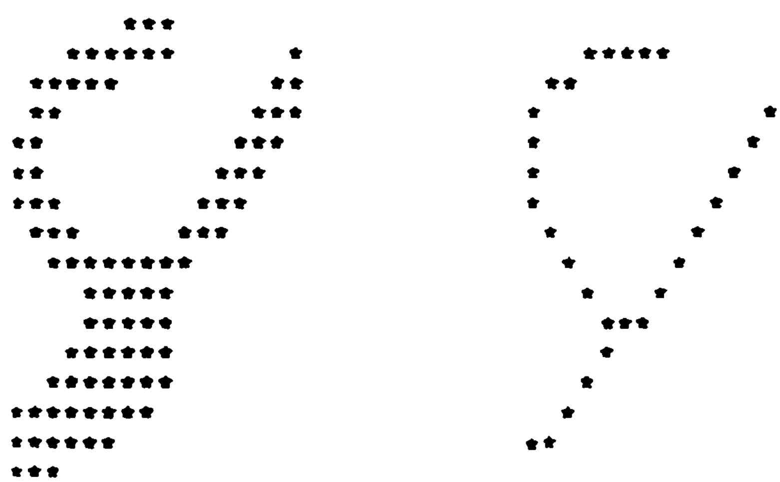


Figure 2.3 Sample of Loss of Information

2.2.1 Filtering Noise

Usually there is much noise in the background of the input image. Median filtering [53] is an effective method to remove the noise. It is similar to using an averaging filter, in that each output pixel is set to an "average" of the pixel values in the neighborhood of the corresponding input pixel. With median filtering, the value of an output pixel is determined by the median of the neighborhood pixels. Because the median is much less sensitive than the mean to extreme values, median filtering is therefore better able to remove the extreme values without reducing the sharpness of the image.

2.2.2 Normalizing

In order to avoid the effect of the size variations for the feature extraction techniques which are not size invariant, the normalization of the numeral data is necessary. In the normalization process a character is scaled linearly to a size needed. An original numeral may be smaller or larger in either of its horizontal or vertical dimensions than the required size. Through linear interpolation, points on the numeral's original image are transposed to the points on the normalized image.

Figure 2.4 shows an example of filtering noise and normalizing.

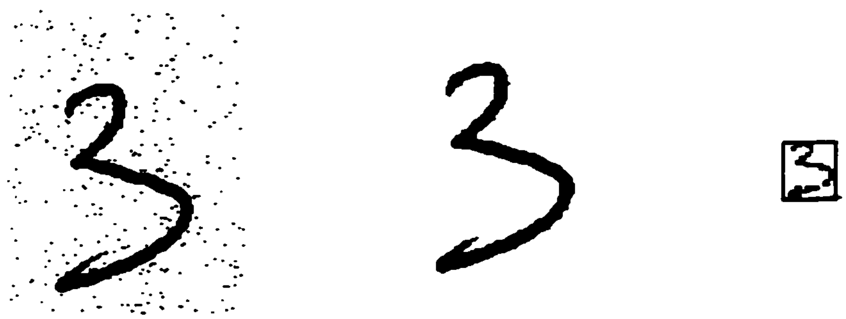


Figure 2.4 Sample of Preprocessing

Chapter 3

FEATURE EXTRACTION AND CLASSIFICATION

3.1 Feature Extraction

3.1.1 Introduction

Given an image I of grid size (m, n), a feature extraction process $T: I \rightarrow F$ is performed on the image I resulting in feature set of F of dimension k. Such feature extraction phase T normally serves the following ideal functionality:

1. Dimension and degree of freedom reduction: $\mathrm{Dim}(I) \geq \mathrm{Dim}(F)$
2. Discrimination enhancement: $P(C_I | F) \geq P(C_I | I)$

It has been indicated that the performance of recognizing handwritten numerals by human beings is much better than by machines. In the past several decades, many different feature extraction methods have been proposed. These methods tried to capture the distinctive features of the characters. Template matching, measurements of density of points, moments, skeletons, contours, and mathematical transforms, such as Fourier, Walsh, Hadamard, etc., are few to mention for feature extraction purpose. These approaches generally fall into three categories:

1. Topological features;
2. Point distribution features;
3. Transformation features.

Topological features use geometrical and structural properties of handwritten numerals, such as loops, arcs, concavities and convexities, cross points, slant lines, and intersection angles. Most topological features are difficult to develop and compute, and sensitive to small distortions. Amit and Geman [54] introduced a simplistic tag feature for binary image that is shape-based and bears a resemblance to the geometric invariants which have been proposed for recognizing rigid shapes and three-dimensional objects.

Point distribution features are often used in conjunction with statistical techniques. They are less sensitive to noise, and require less computation time. Kimura and Shridhar [49] combined a statistical classifier with features derived from the chain codes of character contours, and a tree classifier based on structural features extracted from the left and right profiles to recognize handwritten numerals.

Transformation features transform the image into a series, spectrum, or vector. The transform-based features have the advantages of compressing data significantly and can be made generally invariant to various deformations, such as translation, scale, rotation and reflection. However, in the case of numeral recognition, rotation and reflection invariant may cause ambiguities among some classes such as '2' and '5', '6' and '9', etc. Besides, the computation of transform-based features is heavy. Transformational methods such as Fourier descriptors, moments and wavelets have been investigated by a large number of researchers [55-57]. Belkasim et al. [12] present a detailed study of Zernike and pseudo Zernike moment invariants and conducted a study on handwritten numerals. Wunsch and Laine [58] used wavelets at several resolutions as features into single-layer neural net.

Selection of a feature extraction method is a very important factor in achieving high recognition performance in numeral recognition systems. An ideal feature should contain the information to discriminate between classes. Different feature extraction methods are designed for different representations of the characters. The most important and basic task of feature extraction is to obtain information representing the characteristics of a numeral, which are suitable for classifiers to discriminate in the classification stage. Three criteria must be employed to choose features in a high-accuracy recognition system:

1. The feature can incorporate a generalization of the input map for different representations of the same feature to be properly identified;

2. The feature can incorporate a discriminating power for differentiating between different numerals;

3. The computation time must be taken into account.

Four high efficient features have been used in this thesis; namely, Chain Code Histogram Features (K-features) [16], Directional Distance Distribution Features (D-features) [59], Profiles Projection and Border Transition Features (S-features) [15] and Gray Scale Mapping Features (G-features) [6], which match the three criteria for choosing features. The details of these feature extraction methods are described in the following sections.

3.1.2 Chain Code Histogram Features (K-features)

Chain code generation is the technique used to follow the boundary of an image, and to create a record of the path as the image's outline is defined. In general, a chain code describes the direction to move from the current pixel to the next boundary pixel.

The chain code of a binary image can be generated as follow:

1. Find a pixel on the boundary of the black region;
2. Store the starting pixel's location;

3. Examine each neighbor in a counter-clockwise direction until a black pixel is found;
4. Save the chain link;
5. Let the new pixel of the interest be the neighbor found in step 3. If the pixel of interest is not the starting pixel, go to step 3; otherwise stop.

Figure 3.1 illustrates the processing of the chain code histogram feature extraction. In this process, the normalized pattern can be divided into n×n zones, usually 4×4 zones. In each zone, a local histogram of the contour chain codes is calculated. The feature vector is composed of these local histograms. Since the contour direction is quantized to one of 4 possible values (H, V, L, R), a histogram in each zone has four components.

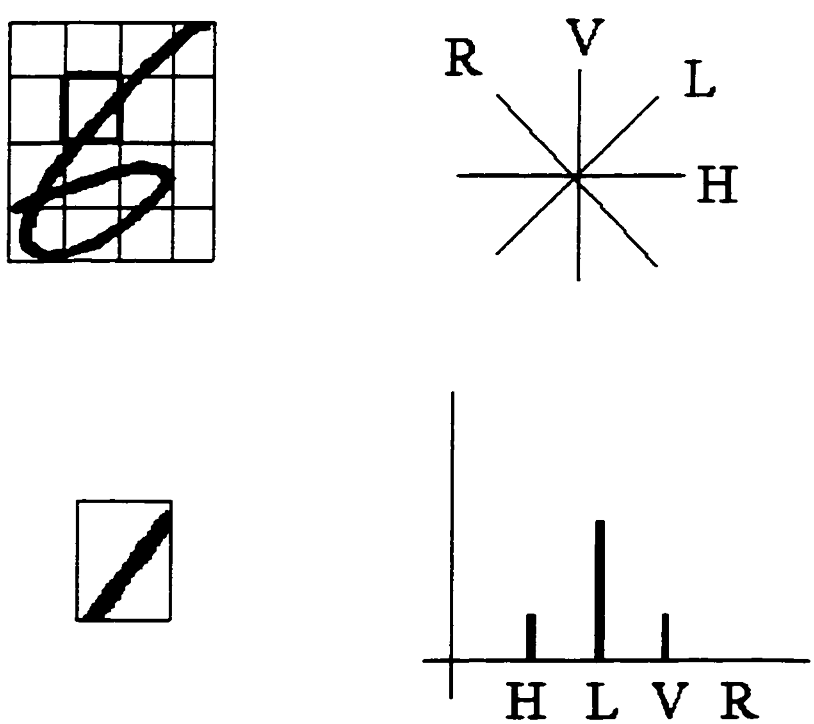


Figure 3.1 Chain Code Histogram Feature Extraction

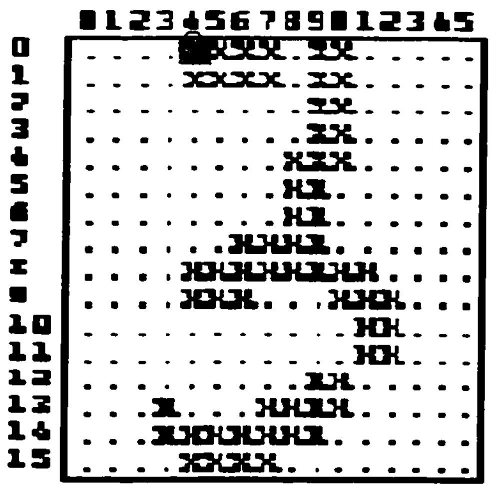
If the image is divided into 16 zones, the feature vectors will have 64 elements.

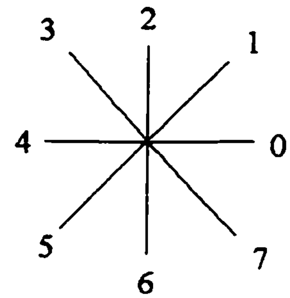
3.1.3 Directional Distance Distribution Features (D-features)

The directional distance distribution feature is based on the distance information computed for every pixel in an image in 8 directions [59]. We can obtain a better discriminating power of the feature by regarding the pixels in an image as being circular when computing the distance. Because of considering both distances from black pixel to white pixel and from white pixel to black pixel, the feature contains both the black/white distribution and the directional distance distribution. Computation time of the extraction algorithm is short because its dominant operations are integer comparison and addition.

To each pixel of the input binary pattern, two sets of 8 bytes, which are called W (White) set and B (Black) set, are allocated. For a white pixel, the set W is used to encode the distances to the nearest black pixels in 8 directions, and the set B is filled with value 0. Likewise, for a black pixel, the set B is used to encode the distance to the nearest white pixels in 8 directions, and the set W is filled with 0. If the goal black/white pixel is not found, the maximum dimension of the array is recorded.

Figure 3.2 explains the computing and encoding of the directional distances for a 16×16 binary pattern array.





for pixel at (0,4), the WB encoding is

	0	1	2	3	4	5	6	7
B	4	4	3	1	1	1	2	2
W	0	0	0	0	0	0	0	0

Figure 3.2 Directional Distance Distribution Feature Extraction

After computing WB codings for all pixels in the pattern, the pattern is divided into 4×4 grids as showed in Figure 3.3.

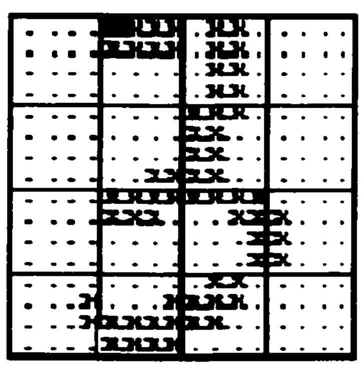


Figure 3.3 4×4 Grids for Computing Directional Distance Distribution Features

In each of 16 grids, an average for each of 16 bytes in WB codings is computed. So there totally are 256 values. The vectors, which are obtained from the computation and normalized to the range from 0 to 1, are then used as inputs to the neural network classifier in the next stage.

It can be seen that there are two major properties of the D-features, which contribute to the discriminating power of the feature. The first one is that the wrap around adjacency property on the numeral image. Some patterns in different classes have a common shape in part of the patterns. D-features can generate different values in these similar areas, and have the ability to discriminate the patterns. The second property is that the feature contains rich information, both of black/white distribution and directional distance distribution over the whole area of the pattern. It can be seen as a natural combination of two different kinds of features.

3.1.4 Profile Projection and Border Transition Features (S-features)

S-features consist of three different kinds of attributes: the zonal profile projection (ZPP), the zonal pixel distribution (ZPD) and the border transition (BT). The three attributes are put into the S-features vector. This is accomplished by dividing the amounts of the ZPP and BT by the maximum value in their respecting fields. ZPD is already presented in the normalized form and therefore left unchanged. The distribution of S-features vector is shown in Figure 3.4.

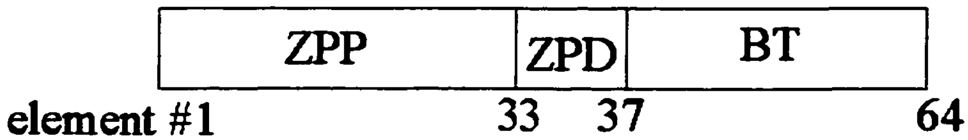


Figure 3.4 Distribution of S-features Vector

3.1.4.1 Zonal Profile Projection (ZPP) Feature

The normalized numeral for recognition is illustrated in Figure 3.5. The frame is divided into four quadrants. The lengths of the projection of the contours on the outer limbs of each quadrant are evaluated through horizontal and vertical scans within each quadrant and used as features.

During a scan in a given quadrant, the number of pixels along the scanning line from the outer limb of the frame to the numeral contour is counted and used as the length of the projection. The direction of the scan depends on the direction of projection and the quadrant in which the projection length is to be evaluated. In each quadrant, four horizontal scanning lines and four vertical scanning lines are used. Since the image is divided into four quadrants, ZPP needs 32 elements.

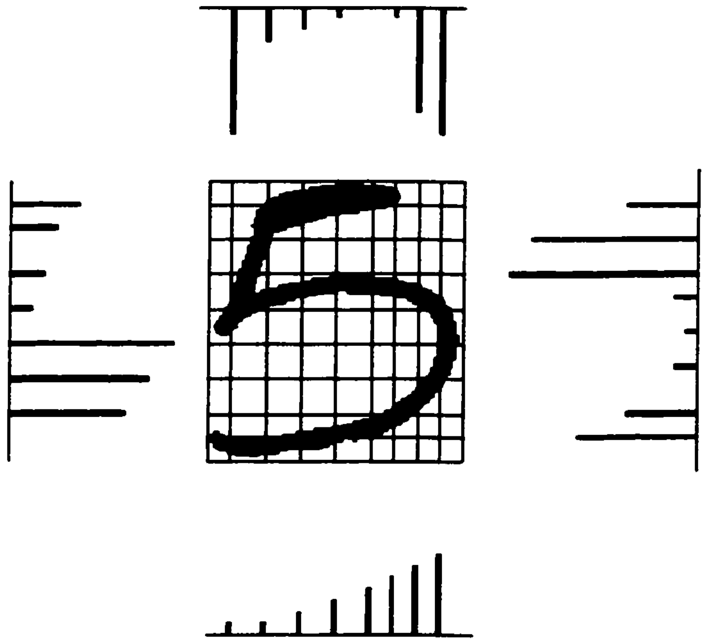


Figure 3.5 Zonal Profile Projection Feature

3.1.4.2 Zonal Pixel Distribution (ZPD) Feature

In this method, the given numeral is also framed and divided into four quadrants as shown in Figure 3.6. In each quadrant, the number of black pixels in the numeral image is counted and used as elements of the feature vector. Then the elements are divided by the total number of black pixels in the whole numeral image. Therefore, ZPD has 4 elements.

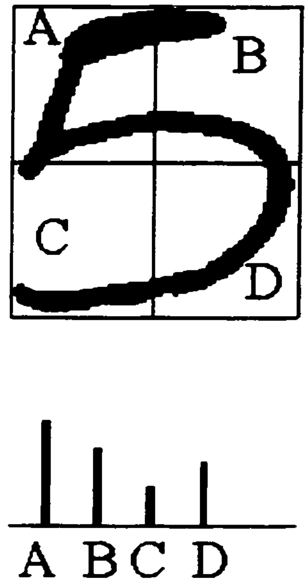


Figure 3.6 Zonal Pixel Distribution Feature

3.1.4.3 Border Transition (BT) Feature

The given numeral image is scanned globally in 0°, 90° and 135° direction respectively. During the scanning, the number of times that the numeral contour is encountered is counted and used as the border transitions. There are eight 0° scanning lines, eight 90°scanning lines and twelve 135° scanning lines. The border transition in each scanning line is used as the features. So there are 28 elements as BT feature. An example is shown in Figure 3.7.

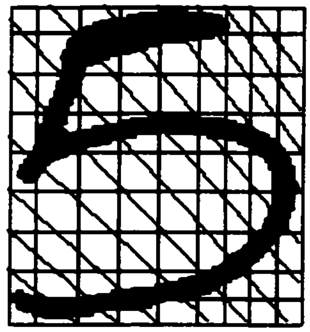


Figure 3.7 Border Transition Feature

3.1.5 Gray Scale Mapping Features (G-features)

The gray scale mapping feature extraction is also applied on normalized numerals. By this extraction method, a neighborhood averaging filter [3×3] is applied on the binary image of the numerals several times (normally 6 times). The entire image is divided into 8×8 regions. In each region, the average value is chosen to be its representative. Therefore, the image is down sampled to a size of 8×8 with gray scale values. Figure 3.8 illustrates the effects of this filtering and down sampling of the numeral image, where T indicates the number of times that the neighborhood averaging filter is applied.

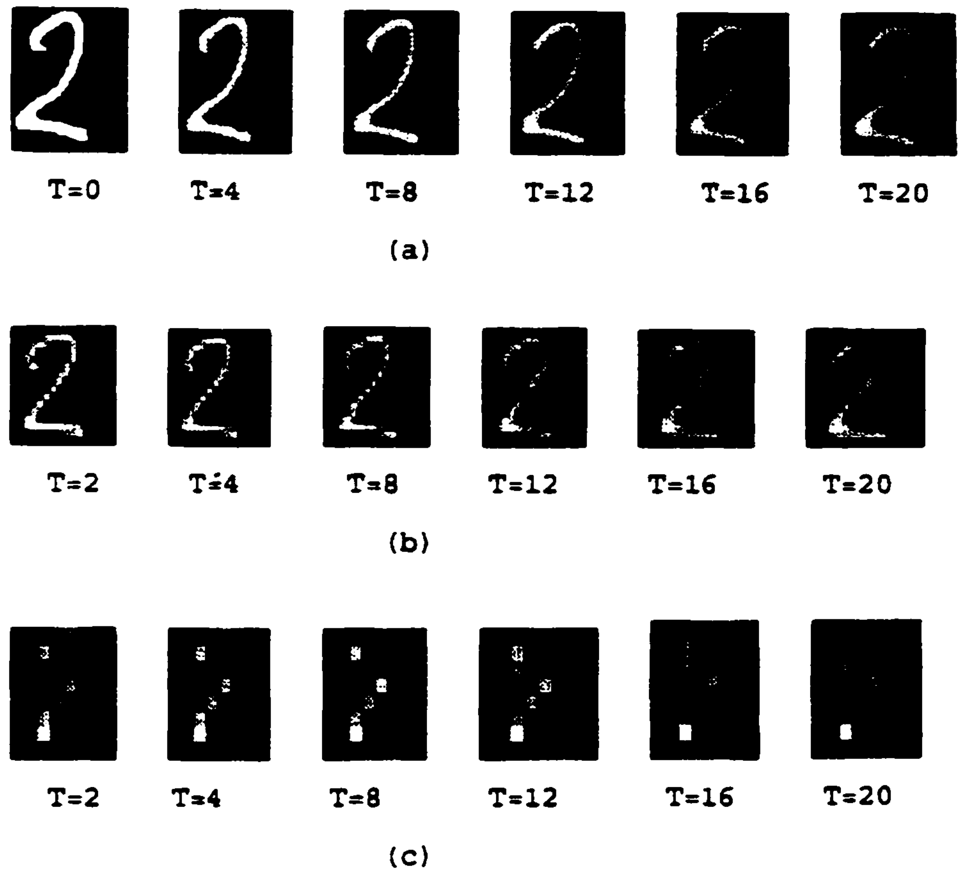


Figure 3.8 Gray Scale Mapping Features

(a) Effects of Filtering

(b) Down-sampling to a Size of 16×16

(c) Down-sampling to a Size of 8×8

3.2 Classification

3.2.1 Introduction

Generally, there are three categories of handwritten character classifiers: statistical approach, structural approach and neural approach [60].

The statistical approach makes a decision from the statistical information derived from the input data. The underlying model is a set of probabilities or probability density functions. The statistical approach is relatively insensitive to noise and distortion, but it is difficult to catch structural information. Cai and Liu [61] used a popular statistical method, the hidden Markov model (HMM), to recognize unconstrained handwritten numerals. Their recognition rate is between 95.5% to 98.0%. Cho [62] also used a HMM classifier for recognition, and the recognition rate is 93.95%.

In the structural approach, each pattern class is defined by structural description or representation, and the recognition is performed according to structural similarities. Primitive elements are extracted from the input data. Structural methods are robust against shape deformations, but they are sensitive to topological distortions or noise caused by image degradations. In the paper of Yamagata et al. [63], a structural classifier with contour analysis and thin-line analysis is used, and the recognition rate is 91.9%.

Neural network is one of the most widely used classifiers. It is well known that artificial neural network (ANN) based classifiers perform particularly well at pattern classification [64-66]. A principle advantage of the neural network is the ability to acquire the knowledge of the given problem directly from sample data. Cho [62] used a neural network classifier to recognize unconstrained handwritten numerals and have produced

95.10% recognition rate. Lee [67] also used a multilayer neural network classifier and obtained 96.69% recognition rate.

The classification in handwritten numeral recognition systems has been proposed and implemented in various ways. However, no scheme with a single feature extractor or classifier can achieve high performance for the recognition of unconstrained handwritten numerals. Thus, recent efforts have focused on multi-stage and multiexpert systems, and it is effective to combine several classification algorithms to improve the performance of a recognition system. In multiexpert systems, the classifiers work in parallel, and they give their decision independently. In multistage systems, the system consists of serial classifiers, and the next classifier is trained only by patterns rejected by the previous classifier. Srihari et al. [44] proposed a high accuracy recognition system that combined the results obtained from three classifiers: template matching, mixed statistical structural classifier based on character contours and structural classifier using features such as strokes, holes, etc. In the paper of Cai and Liu [61], by combining HMM with a structural classifier, the recognition rate has been improved to between 96.2% to 98.4%. In the paper of Yamagata et al. [63], the recognition rate is up to 92.6% by combining the structural classifier with a pattern matching classifier and a decision tree classifier. Cho [62] combined a neural network classifier with a HMM classifier and got a better result, which is 96.55%.

3.2.2 Rule-based Multi-stage Classification

Hybrid system, for pattern recognition which combine neural networks with rule-based methods have been found to be very useful in the areas of speech and natural language processing, and other applications [68]. For classification problems, the operation of the rule-based classifier is traceable, but the set of rules chosen may be more difficult to train and may not generalize as well as a neural network classifier. The neural network, which will be described in the next section, is able to give statistical information about the classification and is easy to train, but it is often not clear how the neural network has arrived at its answer. Hybrid systems combine the advantage of the two methods.

In the architecture of classification in this thesis, the rule-based classifier is designed only for D-features because the D-features is based on every pixel in a binary pattern; while the other three features are based on contour or point distribution at pattern level. The rule-based classifier for D-features is a good complement for the cascaded neural network classifier for this feature. The rules employed are based on the ratios of S-features according to the characteristics of numerals.

3.2.3 Neural Network Multiexpert Classification

Conventional pattern recognition systems employ a single classification procedure to arrive at a decision. However, no scheme with a single feature extractor or classifier can achieve high performance. The concept of combining multiple classifiers has been actively pursued for developing reliable pattern recognition systems [41]. The idea is to combine the decisions of classifiers with complementary performances to improve the classification accuracy. Intrinsically, neural networks are suitable for this idea because they possess the four valuable characteristics. First, they behave as collective systems. Secondly, they can infer subtle unknown relationship from data. Thirdly, they can generalize, meaning they can respond correctly to patterns that are similar to the original training data. Finally, they are nonlinear, so they can solve some complex problems more accurately.

The development of artificial neural networks has been inspired by the characteristics of brain function. The brain consists of a large number (approximately 10^{11}) of highly connected elements (approximately 10^4 connections per element) called neurons [69]. These neurons have three principle components: the dendrites, cell body and the axon [69]. Some of the neural structure is defined at birth. Other parts are developed through learning, as new connections are made. There are two key similarities between biological and artificial neural networks: the building blocks are simple computational devices that are highly interconnected, and the connections between neurons determine the function of the network.

An artificial neural network can be defined as a massively parallel distributed processor made up of simple processing units, which has a propensity for storing experiential knowledge and making it available for use [70-73]. It resembles the brain in two respects. First, knowledge is acquired by the network through a learning process. Secondly, interneuron connection strengths, known as synaptic weights, are used to store the acquired knowledge.

There are three major neural network architectures: single-layer feedforward networks, multilayer feedforward networks and recurrent networks. The architectures of single-layer feedforward networks and recurrent networks are not practical in this application. The multilayer feedforward networks are used in this thesis, which will be described in the next section.

3.2.3.1 Multilayer Perceptron

Typically, multilayer feedforward networks, also referred to as multilayer perceptron (MLP), consist of a set of sensory units (source nodes) that constitute the input layer, one or more hidden layers of computation nodes, and an output layer of computation nodes. The input signal propagates through the network in a forward direction, on a layer by layer basis. The structure of MLP is shown in Figure 3.9.

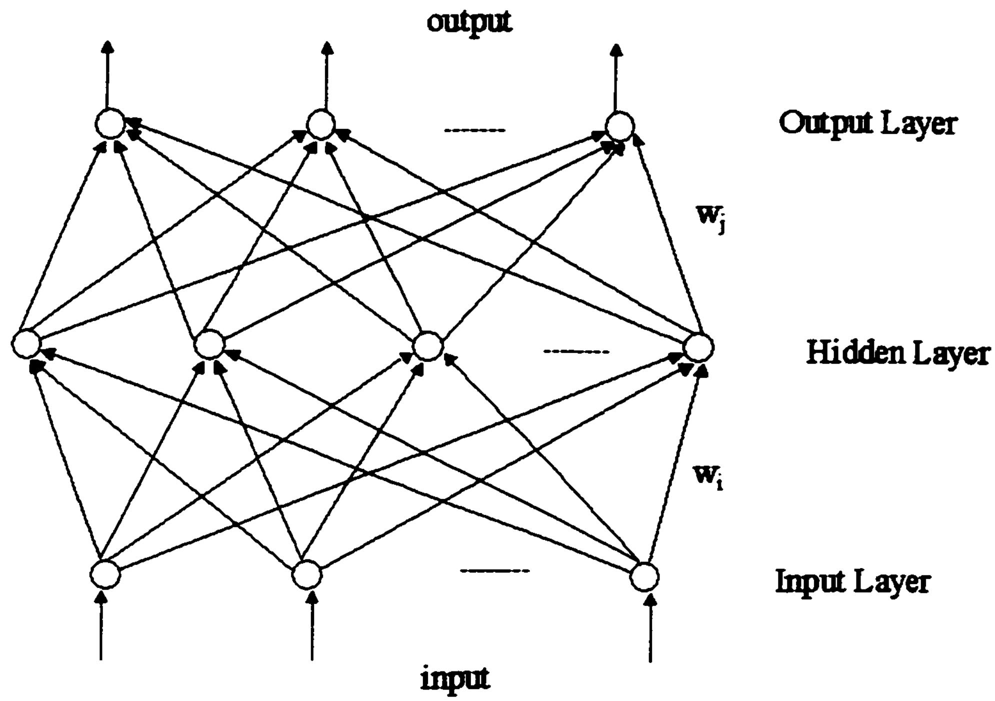


Figure 3.9 Structure of MLP

In this thesis, we have used MLP as the second stage classifiers trained by an error backpropagation algorithm [74], which will be discussed in the next section. Because there are 10 classes in numeral recognition, we applied the modified MLP structure, called modular neural network classifier [75], which consists of 10 networks for the 10 numeral classes, respectively. Each network has one input layer, one hidden layer, and one output layer, and they are fully connected.

There are two advantages of the modular neural network classifiers. First, it is easier to calculate for classification comparing to the conventional structure, which is only one network for the 10 classes. Secondly, the network is much smaller than the conventional one, so it can improve generalizations to get more accurate result [75].

The operation of MLP can be thought of as a nonlinear decision-making process. Given an unknown input set $X=(x_1, x_2, ..., x_T)$ and the output set $\Omega=\{\omega_1, \omega_2, ..., \omega_c\}$, each output node yields the output y_i of belonging to this class by [62]

$$y_i = f\left\{\sum_k \omega_{jk}^{om} f\left(\sum_i \omega_{ki}^{mi} x_i\right)\right\} \tag{3-1}$$

where ω_{ki}^{mi} is a weight between the ith input node and the kth hidden node, ω_{jk}^{om} is a weight from the kth hidden node to the jth class output, and f is a sigmoid function defined as $f(x)=1/(1+e^{-x})$. The node having the maximum value is selected as the corresponding class.

The outputs of the MLP in Equation (3-1) are estimates of Bayesian Aposteriori probabilities [62]. With a squared-error cost function, the network parameters are chosen to minimize the following:

$$E\left[\sum_{i=1}^{c}\left(y_i(X)-d_i\right)^2\right] \tag{3-2}$$

where E[.] is the expectation operator, $\{y_i(X): i=1, ..., c\}$ is the outputs of the network, and $\{d_i: i=1, ..., c\}$ is the desired outputs for all output nodes. Performing several treatments in this formula allows it to be in a form commonly used in statistics, which provides much insight as to the minimizing values for $y_i(X)$ [62]

$$E\left[\sum_{i=1}^{c}\left(y_i(X)-E\left[d_i|X\right]\right)^2\right]+E\left[\sum_{i=1}^{c}\mathrm{var}\left[d_i|X\right]\right] \tag{3-3}$$

where $E[d_i /X]$ is the conditional expectations of d_i and $var[d_i /X]$ is the conditional variance of d_i.

Since the second term in Equation (3-3) is independent of the network outputs, minimization of the squared-error cost function is achieved by choosing network parameters to minimize the first expectation term. This term is simply the mean-squared error between the network outputs $y_i(X)$ and the conditional expectation of the desired outputs. For a "1" of M problem, d_i equals one if the input X belongs to class ω_i and zero otherwise. Thus, the conditional expectations are as the following:

$$E[d_i|X] = \sum_{j=1}^{c} d_i P(\omega_j|X) = P(\omega_i|X) \tag{3-4}$$

which are the Bayesian probabilities. Therefore, for a "1" of M problem, when network parameters are chosen to minimize a squared-error cost function, the outputs estimate the Bayesian probabilities so as to minimize the mean-squared estimation error.

3.2.3.2 Backpropagation Algorithm

MLP has been applied successfully to solve some difficult and diverse problems by training them with a highly popular algorithm known as the error backpropagation algorithm [74, 76]. Standard backpropagation is a gradient descent algorithm. Basically, error backpropagation learning consists of two passes through the different layers of the network: a forward pass and a backward pass. In the forward pass, an input vector is applied to the sensory nodes of the network, and its effect propagates through the network layer by layer to produce a set of outputs. During the backward pass, the synaptic weights are all adjusted in accordance with an error-correction rule. Specifically, the actual response of the network is subtracted from a desired response to produce an error signal. The error signal is propagated backward through the network, against the direction of synaptic connections [70]. Therefore, it is named error backpropagation. By the learning process, the synaptic weights are adjusted to make the actual response of the network move closer to the desired response in a statistical sense.

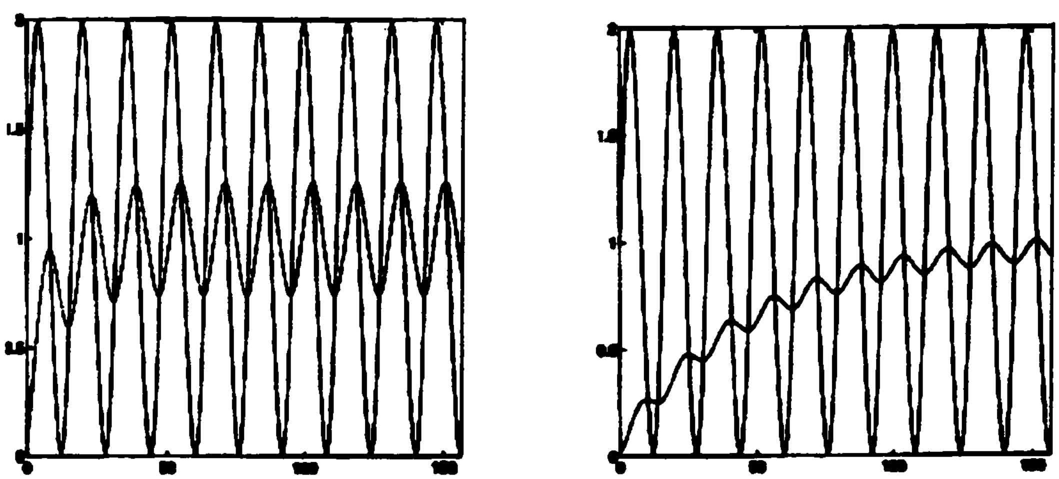
The rule for changing weights following presentation of input/output pair p is given by [74]

$$\Delta_p w_{ji} = \eta(t_{pj} - o_{pj}) i_{pi} = \eta \delta_{pj} i_{pi} \qquad (3\text{-}5)$$

where t_{pj} is the target input for jth component of the output pattern for pattern p, o_{pj} is the jth element of the actual output pattern produced by the presentation of input pattern p, i_{pi} is the value of the ith element of the input pattern, and $\Delta_p w_{ij}$ is the change to be made to the weight from the ith to the jth unit following presentation of pattern p.

There are two different ways in which the gradient descent algorithm can be implemented: incremental mode and batch mode [77]. In the incremental mode, the gradient is computed and the weights are updated after each input is applied to the network. In the batch mode, all of the inputs are applied to the network before the weights are updated.

In the incremental mode, there are two algorithms. One is the basic steepest descent algorithm [69]. For this algorithm, the weights and biases are moved in the direction of the negative gradient of the performance function. The other one is the steepest descent with momentum [69]. This algorithm often provides faster convergence. Momentum allows a network to respond not only to the local gradient, but also to recent trends in the error surface. Acting like a low pass filter, momentum allows the network ignore small features in the error surface. Here is a simple example [69]. Consider the first-order filter: $y(k)=\gamma y(k-1)+(1-\gamma)w(k)$, where $w(k)$ is the input to the filter, $y(k)$ is the output of the filter and γ is the momentum coefficient that must satisfy $0\le\gamma<1$. For this example the input to the filter was taken to be the sine wave: $w(k)=1+\sin(2\pi k/16)$, and the momentum coefficient was set to γ=0.9 and 0.98. We can see from the Figure 3.10 that the oscillation of the filter output is less than the oscillation in the filter input, as we would expect for a low-pass filter. As γ is increased, the oscillation in the filter output is reduced.



a) γ=0.9 b) γ=0.98

Figure 3.10 Smooth Effect of Momentum

Without momentum a network may get stuck in a shallow local minimum. With momentum a network can slide through such a minimum. There are also two similar algorithms in the batch mode: the batch gradient descent and batch gradient descent with momentum [77].

An important consideration in the network computation is the learning rate. The learning rate determines what portion of the calculated weight change will be used for correction. The changes to the weights and biases of the network are obtained by multiplying the learning rate times the negative of the gradient. The best value of the learning rate depends on the characteristics of the error surface. The larger the learning rate, the bigger the step. However, if the learning rate is made too large the algorithm will become unstable; if the learning rate is set too small, the algorithm will take a long time to converge. A general rule is to use the largest learning rate that works and does not cause oscillation by tests.

There are several high performance algorithms, which can converge from ten to one hundred times faster than the two algorithms, gradient descent and gradient descent with momentum. These faster algorithms fall into two main categories. The first category uses heuristic techniques, which were developed from an analysis of the performance of the standard steepest descent algorithm. One heuristic modification is the momentum technique. Two more heuristic techniques are variable learning rate backpropagation and resilient backpropagation [77]. The second category of fast algorithms uses standard numerical optimization techniques. Three types of numerical optimization techniques for neural network training are: conjugate gradient, quasi-Newton, and Levenberg-Marquardt [69].

Practically, it is very difficult to know which training algorithm is the fastest. It depends on many factors, including the complexity of the problem, the number of data points in the training set, number of weights and biases in the network, and the error goal. Generally, on networks which contain up to a few hundred weights the Levenberg-Marquardt algorithm has the fastest convergence, and this advantage is especially noticeable if very accurate training is required. The second choice is the quasi-Newton algorithm, or the conjugate gradient method. The resilient backpropagation algorithm is also very fast, and has relatively small memory requirements.

Another important problem that occurs during neural network training is called overfitting. The error on the training set is driven to a very small value, but when new data is presented to the network the error is large. The network has memorized the training examples, but it has not learned to generalize to new situations.

One method for improving network generalization is to use a network, which is just large enough to provide an adequate fit. The larger a network used, the more complex the functions that the network can create. If we use a small enough network, it will not have enough power to overfit the data, as we did in our system.

3.3 Experiment Results

We used the extracted features as the input data to the classifiers to calculate the outputs. There are 5000 samples used for training and 5800 samples for testing. We used a PC with PII 350 CPU and 64 MB RAM to do the experiments. The program running time is from several minutes to twenty minutes. Computations are integer or float comparison, addition, and multiplication. Table 3.1 to Table 3.4 shows the experiment matrices of neural network classifiers of different extracted features. The elements on the principal diagonal of the performance matrix reflect the overall performance of the classifier on each class. The elements of the principal diagonal reflect the detailed statistics of misclassifications among the classes.

Performance of K-features net

	0	1	2	3	4	5	6	7	8	9	corrects	Errors	%err rate
0	572	0	2	0	2	0	0	2	2	0	572	8	1.38
1	0	549	6	0	1	3	2	2	8	0	549	31	5.34
2	2	1	555	4	2	0	0	10	5	1	555	25	4.31
3	0	0	6	554	0	2	0	4	4	10	554	26	4.48
4	3	6	2	0	548	0	8	7	1	5	548	32	5.52
5	4	0	0	3	0	564	6	1	2	0	564	16	2.76
6	5	0	1	0	9	1	562	0	2	0	562	18	3.1
7	2	1	4	1	3	0	0	563	4	2	563	17	2.93
8	6	5	7	3	3	2	3	1	545	5	545	35	6.03
9	1	4	2	4	4	0	0	5	10	550	550	30	5.17
											5562	238	
											95.90%	4.10%	

Table 3.1 Performance of K-features Net

Performance of D-features net

	0	1	2	3	4	5	6	7	8	9	corrects	Errors	%err rate
0	561	0	6	0	0	0	8	0	4	1	561	19	3.28
1	0	570	0	0	4	0	0	6	0	0	570	10	1.72
2	0	1	559	4	6	4	0	0	6	0	559	21	3.62
3	0	0	6	553	0	6	0	4	5	6	553	27	4.66
4	0	0	8	0	559	0	8	0	1	4	559	21	3.62
5	1	0	4	10	0	557	6	0	2	0	557	23	3.97
6	8	0	0	1	4	4	555	0	6	2	555	25	4.31
7	0	0	4	0	2	0	0	568	0	6	568	12	2.07
8	6	0	1	6	4	2	8	0	547	6	547	33	5.69
9	4	4	6	0	2	0	4	6	1	553	553	27	4.66
											5582	218	
											96.24%	3.76%	

Table 3.2 Performance of D-features Net

Performance of S-features net

	0	1	2	3	4	5	6	7	8	9	corrects	Errors	%err rate
0	570	1	4	0	2	0	1	0	0	2	570	10	1.72
1	2	532	12	1	14	0	0	5	8	6	532	48	8.26
2	2	3	539	9	10	0	0	3	11	3	539	41	7.07
3	0	1	10	546	0	5	1	4	4	9	546	34	5.86
4	0	3	1	0	561	0	2	10	2	1	561	19	3.28
5	0	0	4	7	2	554	3	0	8	2	554	26	4.48
6	0	0	2	0	5	5	562	0	6	0	562	18	3.1
7	0	1	0	5	6	3	0	560	1	4	560	20	3.45
8	10	4	6	7	3	5	4	5	531	5	531	49	8.45
9	2	7	6	6	3	1	0	2	5	548	548	32	5.52
											5503	297	
											94.88%	5.12%	

Table 3.3 Performance of S-features Net

Performance of G-features net

	0	1	2	3	4	5	6	7	8	9	corrects	Errors	%err rate
0	556	2	6	0	0	1	11	0	1	3	556	24	4.14
1	0	552	5	0	4	2	2	9	4	2	552	28	4.83
2	12	8	517	4	2	2	2	6	15	12	517	63	10.86
3	0	2	7	529	0	4	1	4	5	28	529	51	8.79
4	3	7	5	0	552	0	6	3	2	2	552	28	4.83
5	0	2	15	8	2	538	10	0	0	5	538	42	7.24
6	0	0	6	1	4	4	565	0	0	0	565	15	2.59
7	0	8	4	0	1	0	0	566	1	0	566	14	2.41
8	4	6	7	6	4	11	0	8	524	10	524	56	9.66
9	3	6	2	8	4	3	0	4	5	545	545	35	6.03
											5444	356	
											93.86%	6.14%	

Table 3.4 Performance of G-features Net

From the results, we can see that the best performance among the four features is D-features (96.24%), and then K-features (95.90%), S-features (94.88%) and G-features (93.86%), respectively. Table 3.5 shows the best performance of each digit by one of the four feature nets.

Digit	% error rate	Feature
0	1.38	K-features
1	1.72	D-features
2	3.62	D-features
3	4.48	K-features
4	3.28	S-features
5	2.76	K-features
6	2.59	G-features
7	2.07	D-features
8	5.69	D-features
9	4.66	D-features

Table 3.5 Best Performance for Each Digit by a Feature

CHAPTER 4

FUSION OF CLASSIFIERS

4.1 Introduction

The combination of multiple classifiers is a general problem that is interesting not only to the numeral recognition but also to various application areas of pattern recognition.

The fusion of results obtained from multiple classifiers is considered to be important for the design and analysis of complex systems. The fusion process deals with real numerical outputs from different classifiers in order to assign a better quantitative measure of certainty about an object in question. Information from multiple networks may agree or conflict with each other. Therefore, the task of data fusion is to search for a maximum degree of agreement between the conflicting supports of an object. In general, the fusion of multiple information can be stated in the following expression [78]:

$$\phi_i = F(\omega_{i1}, \omega_{i2}, \cdots, \omega_{ij}, \cdots, \omega_{in}), \quad i = 1,2,\cdots,n \qquad \text{(4-1)}$$

where Φ_i is the combined information about an object i, F is a fusion operator, ω_{ij} is the information of an object i given by the classifier j.

There are several different classifier combination algorithms [78], known as majority vote, Borda count, maximum Aposteriori probability, Dempster-Shafer rule, and fuzzy integral.

The final decision of the majority vote method is voted by the majority result of the classifiers.

Borda count method is a generalization of the majority vote. The Borda count for a class is the sum of the number of classes ranked below it by each classifier. In the this method, all classifiers are treated equally, which may not be desired when certain classifiers are more likely to be correct than others [46]. Therefore, more suitable techniques are necessary for decision combination.

Maximum Aposteriori probability estimates the most likely hypothesis given from all possible hypotheses. It is an algorithm for estimating random parameters with prior distributions, and it is used to estimate the most likely information bit that has been transmitted in a coded sequence [78]. This approach needs two specified distributions. The first one is a priori distribution of the unknown vector. The second distribution (which reflects measurement errors) is taken as an uncorrelated normal distribution. The approach then leads to a discrete, strictly convex, nonquadratic optimization problem

In the method of Dempster-Shafer rule, each neighbor of a sample to be classified is considered as an item of evidence that supports certain hypotheses regarding the class membership of that pattern. The degree of support is defined as a function of the distance between the two vectors. The evidence of the nearest neighbors is then pooled by a set of statistical rules of combination [41]. This method is difficult to handle inconsistent pieces of evidence.

Fuzzy integral is based on the concepts of fuzzy sets and fuzzy measure, and will be described in the next section.

Pham and Yan [78] used these combination methods to fuse four neural network classifiers to recognize handwritten numerals. The experiment results showed that the recognition rates were 90.67% for majority vote, 92.24% for Borda count, 90.67% for maximum Aposteriori probability, 92.57% for Dempster-Shafer rule, and 95.00% for fuzzy integral. Thus, it can be seen that fuzzy integral fusion is superior to the others. Cho and Kim [79] applied majority vote, Borda count and fuzzy integral to combine three neural network classifiers to recognize numerals, and found that the recognition rate for fuzzy integral was 88.1%, better than majority vote (84.9%) and Borda count (86.8%).

4.2 Fuzzy Integral

The fuzzy integral is a method for combining the information from multiple sources. It considers both the objective evidence supplied by each information source (called the *h*-function) and the expected worth of each subset of information sources (via a fuzzy measure) in its decision making process.

This process is a very nonlinear combination of information and the worth of these information sources with respect to the decision. This type of information fusion has the ability to deal with the uncertainty associated with the process of extracting and processing information. It has also been used for pattern recognition, object classification, and object history matching. In these types of applications, the fuzzy integral fuses information from multiple sources in order to achieve a final classification. A fuzzy integral is calculated for each classification hypothesis, and then the integral with the largest value usually indicates the class label.

In pattern recognition applications, the fuzzy integral fuses information from the multiple information sources in order to achieve a confidence value for each class hypothesis. If a crisp decision is necessary, the integral with the largest value determines the class label of the object in question.

In this thesis, the fuzzy integral is applied to combine the results given by the neural network classifiers for recognition of handwritten numerals. Figure 4.1 shows the architecture of fuzzy integral fusion of classifiers.

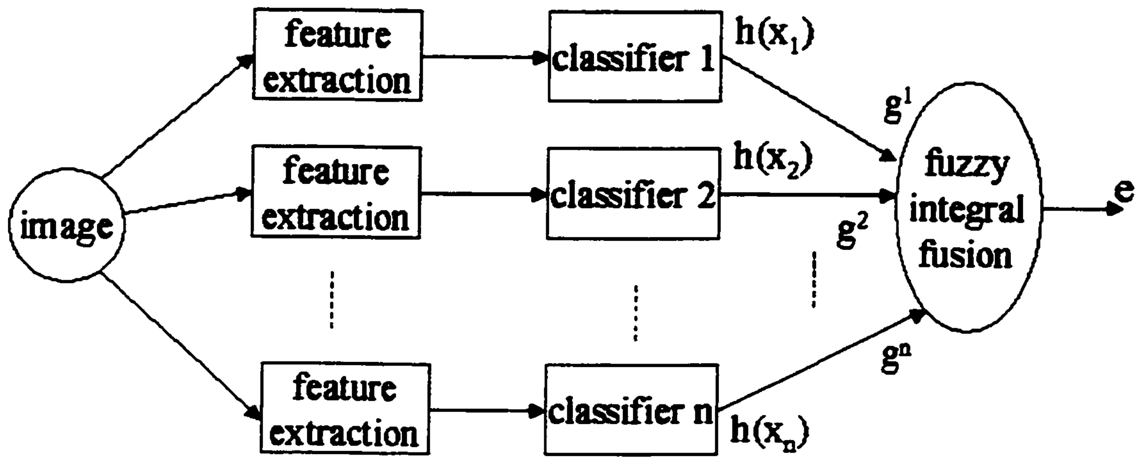


Figure 4.1 Architecture of Fuzzy Integral Fusion

The fuzzy integral is based on the Sugeno fuzzy measure [80], and the calculation of fuzzy density is important and sensitive in the application of the fuzzy integral. They are described as below.

4.2.1 Sugeno Fuzzy Measure and Fuzzy Integral

A fuzzy measure is used to express an evaluation of something which is heavily subject to human perception such as the "grade of importance", or "grade of beauty".

Let $X=\{x_1,x_2,\ldots,x_n\}$ be a finite set which represents a set of n information sources. A fuzzy measure g is a real-valued function defined on the power set of X with range [0,1], satisfying the following properties:

1. $g(\Phi)=0$ and $g(X)=1$, where Φ is an empty set.
2. $g(A)\leq g(B)$ if $A\subseteq B$.
3. If $\{A_i\}$ is an increasing sequence of subsets of X, then

$$\lim_{i\to\infty} g(A_i) = g\left(\bigcup_{i=1}^{\infty} A_i\right) \tag{4-2}$$

A fuzzy measure g is called a Sugeno fuzzy measure (g_λ-fuzzy measure) if it additionally satisfies the following property:

4. For all $A, B\subseteq X$ with $A\cap B=\Phi$,

$$g_\lambda(A\cup B)= g_\lambda(A)+ g_\lambda(B)+ \lambda g_\lambda(A)g_\lambda(B) \quad \text{for some } \lambda>-1. \tag{4-3}$$

Let $g^i=g(\{x_i\})$, the mapping $x_i\rightarrow g^i$ is defined as a fuzzy density function.

The value of λ for any Sugeno measure can be uniquely determined for a finite set X using Equation (4-3) and the facts that

$$X = \bigcup_{i=1}^{n} \{x_i\} \tag{4-4}$$

and $g_\lambda(X)=1$, which leads to solving the following equation for λ [80]:

$$1+\lambda = \prod_{i=1}^{n} (1+\lambda g^i) \tag{4-5}$$

Let (X, B, g) be a fuzzy measure space and h: $X\rightarrow[0,1]$ be a B-measurable function. The fuzzy integral of h with respect to a fuzzy measure g is defined as [80]

$$\int_A h(x)\circ g(\bullet) = \sup_{E\subseteq X}\left[\min\left(\left(\min_{x\in E} h(x), g(A\cap E)\right)\right)\right] \\ = \sup_{\alpha\in[0,1]}\left[\min\left(\alpha, g(A\cap F_\alpha)\right)\right] \qquad (4\text{-}6)$$

where $F_\alpha=\{x/h(x)\geq\alpha\}$.

For the finite case, suppose $h(x_1)\geq h(x_2)\geq\ldots\geq h(x_n)$. (If this is not the case for any object instance, then reorder the set of information sources *X* so that this relation holds.) Then the fuzzy integral, *e*, can be shown to be

$$e = \max_{i=1}^{n}\left[\min\left(h(x_i), g(A_i)\right)\right] \qquad (4\text{-}7)$$

where $A_i=\{x_1,\ldots,x_i\}$.

For a Sugeno measure, the value of $g(A_i)$ can be determined recursively as

$g(A_i)=g(\{x_1\})$,

$g(A_i)=g^i+g(A_{i-1})+\lambda g^i g(A_{i-1})$ for $1<i\leq n$.

4.2.2 Fuzzy Density

The fuzzy density value g^i is interpreted as the importance of the single information source x_i in determining the evaluation of a class hypothesis. A set of fuzzy density values can be constructed for the information sources in the set *X*.

Since $h(x_i)$ can be obtained from the information sources, the determination of the fuzzy density values plays a very important role for the calculation of fuzzy integral, and changes in these values are sensitive to the fuzzy integrals. Fuzzy densities can be given subjectively, or obtained from the training data.

There are three implementation methods to obtain the fuzzy densities: dynamic assignment [6], discrete probability [81], and fuzzy densities with correction factor [78, 82].

In the method of dynamic assignment, for a given neural network classifier (k=0,1,2,..., k-1), the fuzzy densities (g_k^i, for i=0,1,2,...,n), which are used to derive source relevance are constructed as shown below:

$$g_k^i = \frac{p_k^i}{\sum_{m=0}^{k-1} p_m^i} G_T \tag{4-8}$$

where $G_T<1$, and p_k^i is defined as

$$p_k^i = \sum_{j=0, j\neq i}^{n} \sum_{l=0, l\neq k}^{k-1} h_l^i (1 - q_k^{ij}) + \alpha (q_k^{ii})^2 \tag{4-9}$$

where h_l^i is the ith output of classifier l, $\alpha<1$ is a parameter decided experimentally, q_k^{ij} is the element in the performance matrix of classifier k at the ith row and jth column. G_T is chosen to be 0.6.

The performance matrices are shown in Table 3.1 to 3.4. As indicated in Equation (4-9), p_k^i is composed of two components. The first component makes use of the detailed confusion statistics of the performance matrices. For a given input pattern *y*, suppose classifier A classifies it into class *i* and classifier B classifies it to class *j*. One should look at the element at *j*th row and *i*th column of the performance matrix of classifier A, q_A^{ji}, and the *i*th row, *j*th column element of the performance matrix of classifier B, q_B^{ij}. If, for example, q_A^{ji} is quite small and q_B^{ij} is quite large, then we believe that classifier A rarely misclassifies class *j* to class *i*, and classifier B frequently misclassifies class *i* to class *j*. Therefore, the correct classification, most likely, is class *i*. The second component of p_k^i in Equation (4-9) reflects the overall performance of classifier *k* to class *i*. Using the performance matrices, the values of $\alpha(q_k^{ii})^2$ are calculated.

The method of fuzzy densities with correction factor is improved from the discrete probability method, and is the best one among the three methods [82]. Therefore, we used this method in this thesis. Their calculating procedure is described as follows.

Let $H_{ci}=\{h(c_i/r_k): i=1,2,\ldots, n, k=1,2,\ldots, K\}$, or for short notation, $H_i=\{h(i/k): i=1,2,\ldots, n, k=1,2,\ldots, K\}$ be a set of numerical outputs estimated from all classifiers r_k indicating how likely the object in question should belong to class c_i, where $0 \leq h(i/k) \leq 1$. For $h(i/k)=0$, it means that classifier r_k gives no support in that the object in question belongs to class c_i, and $h(i/k)=1$ indicates classifier r_k gives full support in that the object belongs to c_i.

Step 1. Forming confusion matrix

The performance matrix of classifier r_k denoted as P^k is the results of correctly classified and misclassified classes obtained from the training set. It is established for each classifier and expressed in the form:

$$P^k = \left[p^k_{ij}\right] \quad (4\text{-}10)$$

where i and j stand for class c_i and class c_j respectively. For i=j, p^k_{ij} indicates the number of classified objects in class c_i by r_k; whereas for i≠j, p^k_{ij} indicates the number of class c_i being misclassified as class c_j by r_k.

Step 2. Computing the initial fuzzy densities

The initial fuzzy density used herein is in fact calculated as a discrete probability, and interpreted as the degree in which a classifier identifies a certain class correctly. With the assumptions:

$$0 < p_{ij} < \sum_{j=1}^{I} p_{ij} \quad (4\text{-}11)$$

and for i≠j: $p_{ii} > p_{ij}$, which are usually true for most reasonable classifiers, then the initial fuzzy density can be defined as

$$g_{i/k} = \frac{p^k_{ii}}{\sum_{j=1}^{n} p^k_{ij}} \quad (4\text{-}12)$$

where $g_{i/k}$ is the fuzzy density of classifier r_k in the classification of class c_i.

Step 1 and 2 are the method of discrete probability. To improve the accuracy of the calculating results, step 3 and 4 are necessary and described as below.

Step 3. Computing the correction factor

The correction set can be expressed by following equation

$$\delta^{k/m} = \begin{cases} 1 & \text{for } i/k = j/m \\ \dfrac{p^{k}_{i/k,i/k} - p^{k}_{i/k,j/m}}{p^{k}_{i/k,i/k}} & \text{for } i/k \neq j/m \end{cases} \quad (4\text{-}13)$$

where $\delta^{k/m}$ is the correction factor for the fuzzy density g^{k}_{i}, which counts for the same object classified as class c_i by classifier r_k, and as c_j by classifier r_m (m=1,2,..., M=K-1) and m≠k. The symbols i/k and j/k indicate that i (class c_i) is given by k (classifier r_k), and j (class c_j) is given by m (classifier r_m) respectively.

The meaning of this formulation can be further interpreted in that: r_k assigns an object to class i and it checks if another classifier r_m would yield the same classification. If r_m does not, then r_k is assumed to have a relative chance of misclassifying the object. Thus the initial fuzzy density of r_k for class i will be reduced by the proportion of the difference between the number of its correctly classified objects ($p_{i/k,i/k}$) and that of its misclassified objects ($p_{i/k,j/m}$) obtained from the training data set. That means the more mistakes it has made, the more its corresponding fuzzy density is reduced.

Step 4. Updating the initial fuzzy densities

The initial fuzzy densities are updated using the correction factor by the following equation

$$g^{*}_{i/k} = g_{i/k} \times \delta^{k/m} \tag{4-14}$$

where $g^{*}_{i/k}$ is the updated fuzzy densities.

4.3 Fuzzy Integral Fusion for Handwritten Numeral Recognition

The following algorithm demonstrates the calculation of fuzzy integral for the application of handwritten numeral recognition.

```
Classify{
        input handwritten numeral; //the handwritten numeral to be classified
        max=0; //maximum integral value
        maxC=0; //class of maximum integral value

        For class of numeral=1 to 10 {
                For i=1 to 4 {
                        calculate h-function; //for information source i
                }end for
                sort information sources; //with respect to h-function
                calculate Sugeno measures for the appropriate sets
                calculate fuzzy integral e for class C
                if (e>max) {
                        max=e;
                        maxC=c;
                }end if
        }end for
        output maxC; //best classification
        output max; //the classification confidence
}end Classify
```

After the input numeral is processed by the neural network classifiers individually, the outputs of the classifiers are then merged by the fuzzy integral fusion. The flow chart of the process is illustrated in Figure 4.2.

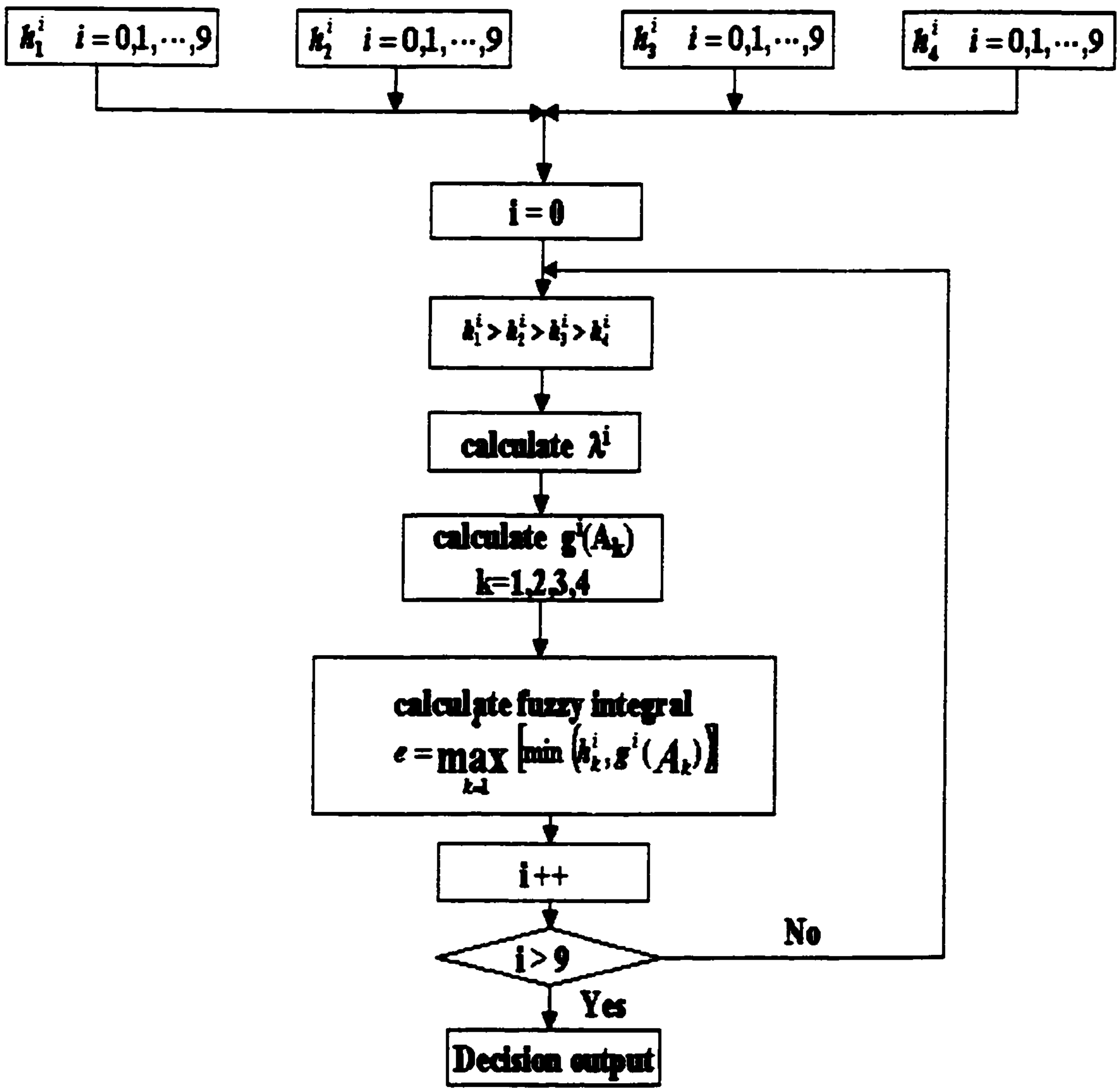


Figure 4.2 Flow Chart of the Fusion Process

4.4 Experiment Results

We used the results of the four neural network classifiers in Chapter 3 to do the fuzzy integral fusion experiments here, which are fusion of K, D, S, and G-net, fusion of K, G and S-net, fusion of K, D, and S-net, and fusion of K, D* (D-net without the rule-based classifier), S, and G-net. The computer we used is the same as the one mentioned in last chapter. Table 4.1 to 4.4 shows the results of the experiments, and Figure 4.3 to 4.6 is the appropriate charts to explain the results more clearly. The comparison of the four fusion results is shown by Table 4.5 and Figure 4.7.

From the results, these can be no doubt that the recognition rate of fusion of classifiers is much better than that of any individual classifier. By the comparison of the fusion results, we can see that KDS fusion with 0.62% error rate has the best performance and KGS fusion with 0.84% error rate is the worst one. KDSG fusion with 0.65% error rate is better than KD*SG fusion with 0.77% error rate, and between KDS and KGS, which is understandable in that, because the performance of G-net is the worst among the four extracted features, the fusion result is affected by it.

Digit	K	D	S	G	Fusion
0	1.38	3.28	1.72	4.14	0.68
1	5.34	1.72	8.26	4.83	0.73
2	4.31	3.62	7.07	10.86	0.86
3	4.48	4.66	5.86	8.79	0.86
4	5.52	3.62	3.28	4.83	0.68
5	2.76	3.97	4.48	7.24	0.73
6	3.1	4.31	3.1	2.59	0.57
7	2.93	2.07	3.45	2.41	0.41
8	6.03	5.69	8.45	9.64	0.61
9	5.17	4.66	5.52	6.03	0.41

Table 4.1 Result of K, D, S, and G-net Fusion

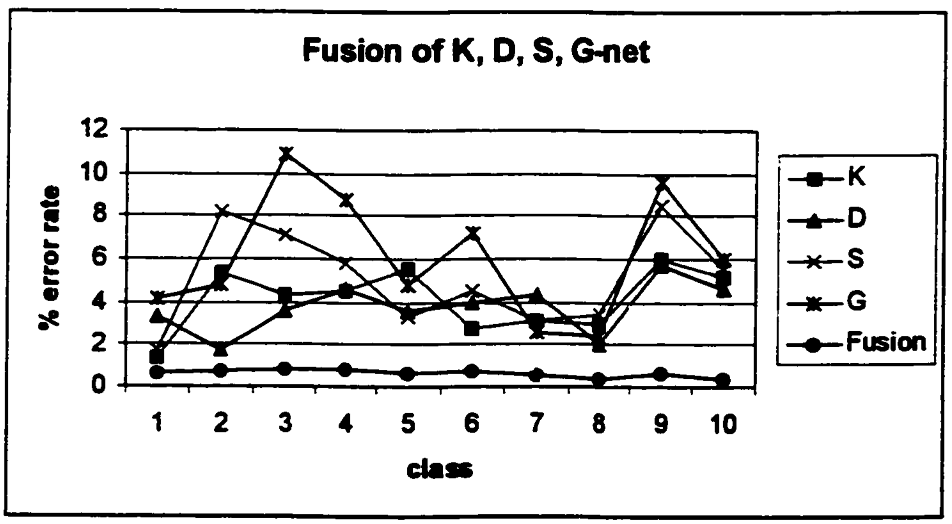


Figure 4.3 Result of K, D, S, and G-net Fusion

Digit	K	G	S	Fusion
0	1.38	4.14	1.72	0.69
1	5.34	4.83	8.26	1.21
2	4.31	10.86	7.07	1.21
3	4.48	8.79	5.86	1.03
4	5.52	4.83	3.28	1.03
5	2.76	7.24	4.48	1.21
6	3.1	2.59	3.1	0.34
7	2.93	2.41	3.45	0.34
8	6.03	9.64	8.45	0.86
9	5.17	6.03	5.52	0.34

Table 4.2 Result of K, G, and S-net Fusion

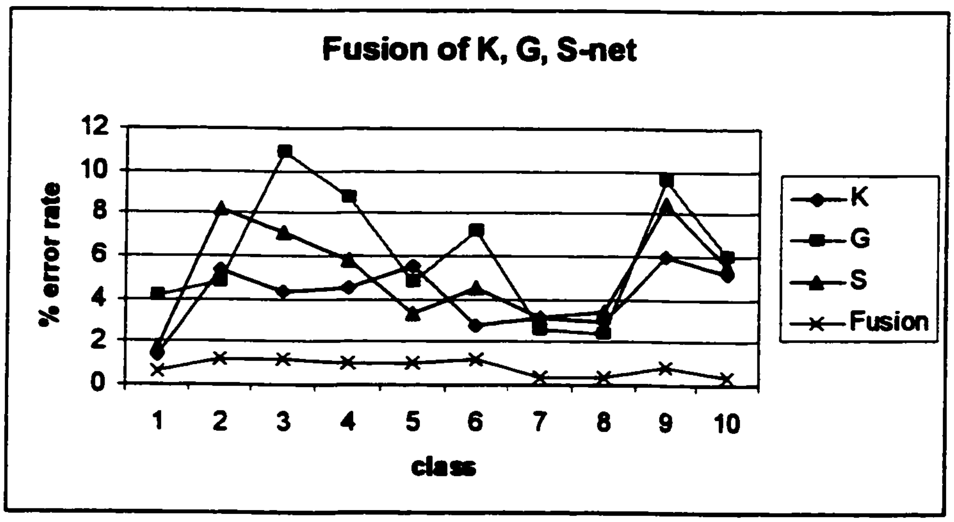


Figure 4.4 Result of K, G, and S-net Fusion

Digit	K	D	S	Fusion
0	1.38	3.28	1.72	0.61
1	5.34	1.72	8.26	0.73
2	4.31	3.62	7.07	0.73
3	4.48	4.66	5.86	0.68
4	5.52	3.62	3.28	0.68
5	2.76	3.97	4.48	0.57
6	3.1	4.31	3.1	0.86
7	2.93	2.07	3.45	0.41
8	6.03	5.69	8.45	0.57
9	5.17	4.66	5.52	0.41

Table 4.3 Result of K, D, and S-net Fusion

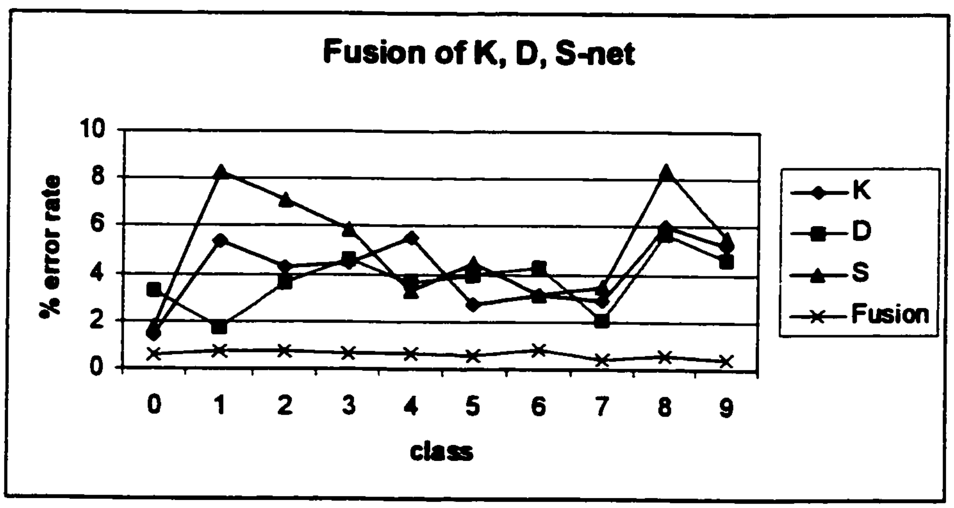


Figure 4.5 Result of K, D, and S-net Fusion

Digit	K	D*	S	G	Fusion
0	1.38	4.14	1.72	4.14	0.67
1	5.34	3.62	8.26	4.83	0.73
2	4.31	4.48	7.07	10.86	0.92
3	4.48	6.38	5.86	8.79	0.85
4	5.52	4.48	3.28	4.83	0.73
5	2.76	5.34	4.48	7.24	0.85
6	3.1	4.14	3.1	2.59	0.85
7	2.93	5.34	3.45	2.41	0.57
8	6.03	6.55	8.45	9.64	0.85
9	5.17	6.38	5.52	6.03	0.46

Table 4.4 Result of K, D*, S, and G-net Fusion

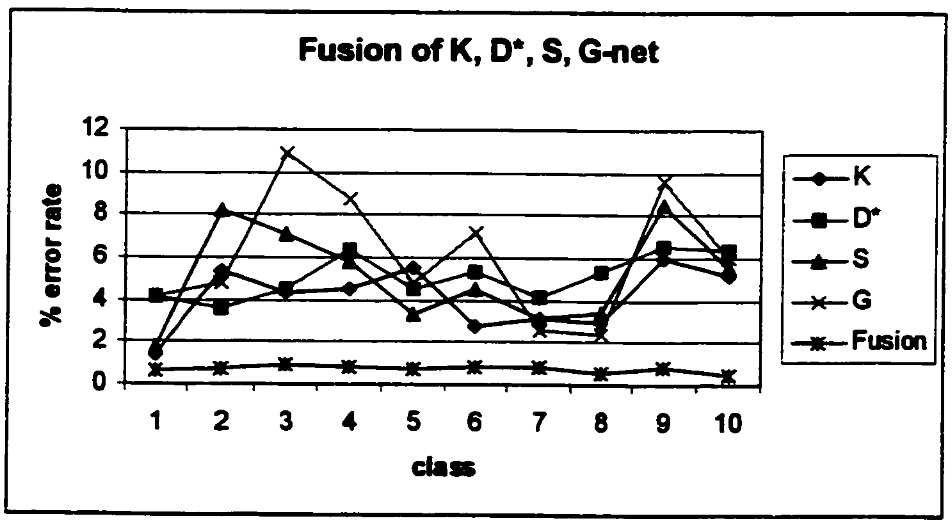


Figure 4.6 Result of K, D*, S, and G-net Fusion

Digit	KDSG	KGS	KDS	KD*SG
0	0.68	0.69	0.61	0.67
1	0.73	1.21	0.73	0.73
2	0.86	1.21	0.73	0.92
3	0.86	1.03	0.68	0.85
4	0.68	1.03	0.68	0.73
5	0.73	1.21	0.57	0.85
6	0.57	0.34	0.86	0.85
7	0.41	0.34	0.41	0.57
8	0.61	0.86	0.57	0.85
9	0.41	0.34	0.41	0.46

Table 4.5 Comparison of Results of KDSG, KGS, KDS, and KD*SG Fusion

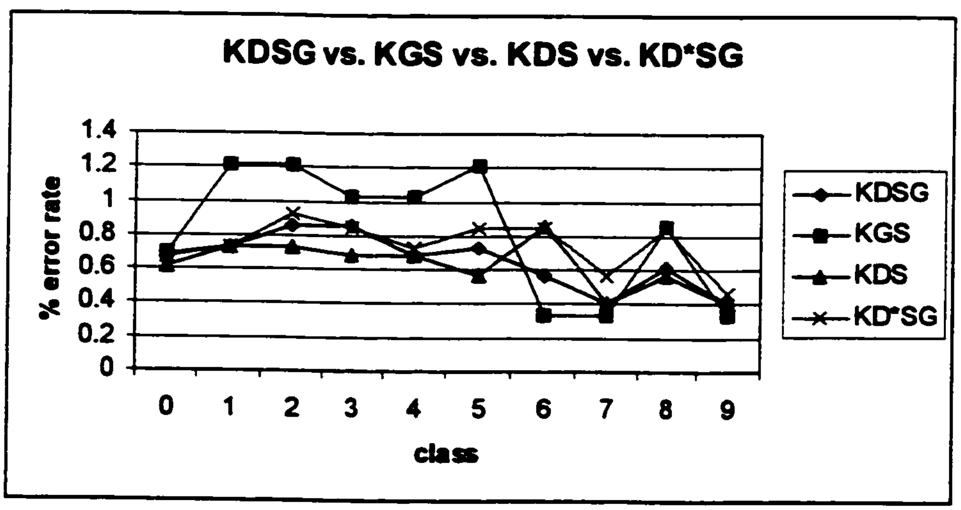


Figure 4.7 Comparison of Results of KDSG, KGS, KDS, and KD*SG Fusion

CHAPTER 5

CONCLUSIONS AND FUTURE WORK

5.1 Conclusions

This thesis addresses the development of a high accuracy offline handwritten numeral recognition system using complementary extracted features and multi-stage and multiexpert classification methods.

It is evident that the overall system must be optimal with respect to all four important stages of the recognition system:

1. Preprocessing of numeral image to remove noise and normalizing for recognition.
2. Features characterizing the numerals and their efficient extraction.
3. The architecture of the multi-stage and multiexpert classification.
4. The combination of multiple classifiers.

Features that uniquely characterize the different numerals are very cruciai in realizing high recognition accuracy. The features should be easy to derive, be robust and be suitable for hardware realization. Four feature sets were used in this study. The first feature set, termed the K-features, consisted of the zonal histogram of the chain code of the contours of the binarized numeral image. The second feature set, termed the D-features, consisted of the distance information computed for every pixel of a pattern for recognition in 8 directions. The third feature set, termed the S-features, utilized the left and right profiles as well as the top and bottom profiles of the numeral image. The fourth feature set, termed the G-features, consisted of the pixel distribution of the numeral image. From the results, we can see that the composition of KDS features achieved the best performance, and the G-features is the worst one among the four features, which also affected the whole performance of the system, so it can be removed to achieve better result.

The architecture of classification reflects the understanding of the computation method for emulating human beings' recognition activity. We used a multi-stage and multiexpert classification method in this thesis. It consists of a ruled-based classifier for one feature, which is complementary to the other extracted features, followed by neural network classifiers for all features. The final result is made from the fuzzy integral fusion of the outputs from the neural network classifiers.

Based on the work of previous students, I designed the high accuracy recognition system using D-features with the other three complementary features. By the comparison of the different fusion conditions of neural network classifiers, the result of KDS fusion achieved the best performance with a contribution of D-features.

5.2 Future Work

We have developed a high accuracy offline handwritten numeral recognition system. However, handwritten numeral recognition is still a challenging task. The following are some possible points that need further research.

1. In applying classification and fuzzy fusion technique, suitable features still need to be investigated such that the orthogonal feature sets can be applied to compensate for one another and proper classifiers are utilized for specific feature sets.

2. For improving the result, appropriate architecture and classifiers for the classification stage are worthy of being examined to elaborate the advantages of multiple-stage and multiexpert classification methods.

3. The methodology of classification and information fusion developed in this thesis may be extended to other pattern recognition problems.

References:

1. A. F. R. Rahman and M. C. Fairhurst, *An Evaluation of Multi-expert Configurations for the Recognition of Handwritten Numerals*, Pattern Recognition, Vol. 31, No. 9, pp. 1255-1273, 1998

2. K. H. Chan, S. S. Erdogan, and G. S. Ng, *A New Feature Selection Criteria for Neural Networks*, Proceedings of The Joint Pacific Asian Conference on Expert Systems/Singapore International Conference on Intelligent Systems, pp. 774-778, 1997

3. J. Mantas, *An Overview of Character Recognition Methodologies*, Pattern Recognition, Vol. 19, No. 6, pp. 425-430, 1986

4. J. Cai and Z. Liu, *Integration of Structural and Statistical Information for Unconstrained Handwritten Numeral Recognition*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 21, No. 3, pp. 263-270, 1999

5. K. H. Chan and S. S. Erdogan, *Contextual Neural Fusion for Multiple handwritten Character Classifiers*, 9th Biennial Conference of the International Graphonomics Society, pp. 131-138, 1999

6. J. Cao, *Handwritten Numeral Recognition with Neural Networks and Fuzzy Fusion*, PhD Dissertation, University of Windsor, 1995

7. J. F. Cullen and K. Ejiri, *Weak Model-Dependent Page Segmentation and Skew Correction for Processing Document Images*, Proceedings of the Second International Conference on Document Analysis and Recognition, Oct. 20-22, pp. 757-760, 1993

8. S. Liang, M. Shridhar, and M. Ahmadi, *Segmentation of Touching Characters in Printed Document Recognition*, Pattern Recognition, Vol. 27, No. 6, pp. 825-840, 1994

9. G. Hannan, *The RCA Multi-font Reading Machine*, in G. L. Fisher, Jr., D. K. Pollock, B. Raddack, and M. E. Stevens, eds., Optical Character Recognition, pp. 3-14, 1962

10. W. Doster, *Contextual Postprocessing System for Cooperation with a Multiple Choice Character Recognition System*, IEEE Transactions on Computers, Vol. C-26, 11, pp. 1090-1101, 1977

11. J. Schuermann, *A Multifont Word Recognition System for Postal Address Reading*, IEEE Transactions on Computers, vol. C-27, 8, pp. 721-732, 1978

12. S. O. Belkasim, M. Shridhar, and M. Ahmadi, *Pattern Recognition with Moment Invariants: A Comparative Study and New Results*, Pattern Recognition, Vol. 24, No. 12, pp. 1117-1138, 1991

13. G. L. Cash and M. Hatamian, *Optical Character Recognition by the Method of Moments*, Computer Vision, Graphics, and Image Processing, Vol. 39, pp. 291-310, 1987

14. M. Shridhar and A. Badreldin, *Recognition of Isolated and Simple Connected Handwritten Numerals*, Pattern Recognition, Vol. 19, No. 1, pp. 1-12, 1986

15. M. Shridhar and A. Badreldin, *Handwritten Numeral Recognition by Tree Classification Methods*, Image and Vision Computing, Vol. 2, No. 2, pp. 143-149, 1984

16. F. Kimura, K. Takashina, S. Tsuruoka, and Y. Miyake, *Modified Quadratic Discrimininant Function and Application to Chinese Character Recognition*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-9, 1, pp. 149-153, 1987

17. M. Shridhar and A. Badreldin, *High Accuracy Character Recognition Algorithm Using Fourier and Topological Descriptors*, Pattern Recognition, Vol. 17, No. 5, pp. 515-524, 1984

18. M. Shridhar and A. Badreldin, *A High-Accuracy Syntactic Recognition Algorithm for handwritten Numerals*, IEEE Transactions on System, Man, and Cybernetics, Vol. SMC-15, No. 1, pp. 152-158, 1985

19. G. H. Granlund, *Fourier Preprocessing for Hand Print Character Recognition*, IEEE Transactions on Computers, Vol. C-21, Vol. 2, pp. 195-201, 1972

20. J. S. Huang and M. L. Chung, *Separating Similar Complex Chinese Characters by Walsh Transform*, Pattern Recognition, Vol. 20, No. 4, pp. 425-428, 1987

21. L. Lam and C. Suen, *Structural Classification and Relaxation Matching of Totally unconstrained Handwritten Zip-code Numbers*, Pattern Recognition, Vol. 21, No. 1, pp. 19-31, 1988

22. F. Ali and T. Pavlidis, *Syntactic Recognition of Handwritten Numerals*, IEEE Transactions on System, Man, and Cybernetics, Vol. CMC-7, pp. 537-541, 1977

23. J. S. Huang and K. Chunag, *Heuristic Approach to Handwritten Numeral Recognition*, Pattern Recognition. Vol. 19, No. 1, pp. 15-19, 1986

24. M. Beun, *A Flexible Method for Automatic Reading of Handwritten Numerals*, Philips Tech. Rev. Vol. 33, pp. 89-101 and 130-137, 1973

25. G. Baptista and K. M. Kulkarni, *A High Accuracy Algorithm for Recognition of Handwritten Numerals*, Pattern Recognition, Vol. 21, No. 4, pp. 287-291, 1988

26. T. M. Cover and P. E. Hart, *Nearest Neighbor Pattern Classification*, IEEE Transactions on Information Theory, Vol. IT-13, No. 1, pp. 21-27, 1967

27. E. A. Patrick and F. P. Fisher, *A Generalized K-Nearest Neighbor Rule*, Information and Control, Vol. 16, No. 2, pp. 128-152, 1970

28. R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*, John Wily & Sons, Inc., 1973

29. B. M. E. Moret, *Decision Trees and Diagrams*, ACM Computer Surveys, Vol. 14, pp. 593-623, 1982

30. B. T. Mitchel and A. M. Gillies, *A Model-Based Computer Vision System for Recognizing Handwritten ZIP Codes*, Machine Vision and Applications, Vol. 2, pp. 231-243, 1989

31. K. S. Fu, *Syntactic Pattern Recognition and Applications*, Prentice-Hall, 1982

32. B. Duerr, W. Haettich, H. Tropf, and G. Winkler, *A Combination of Statistical and Syntactical Pattern Recognition Applied to Classification of Unconstrained Handwritten Numerals*, Pattern Recognition, Vol. 12, No. 3, pp. 189-199, 1980

33. T. Pavlidis and F. Ali, *Computer Recognition of Handwritten Numerals by Polygonal Approximation*, IEEE Transactions on System, Man, and Cybernetics, Vol. CMC-5, No. 6, pp. 610-614, 1975

34. D. W. Ruck, S. K. Rogers, M. Kabrisky, M. E. Oxley, and B. W. Suter, *The Multilayer Perceptron as an Approximation to a Bayes Optimal Discriminant Function*, IEEE Transactions on Neural Networks, Vol. 1, No. 4, pp. 296-298, 1990

35. D. S. Lee, S. N. Srihari, and R. Gaborski, *Bayesian and Neural Network Pattern Recognition: A Theoretical Connection and Empirical Results with Handwritten Characters*, in Artificial Neural Networks and Statistical Pattern Recognition: Old and New Connections. I. K. Sethi and A. K. Jain, eds., Elsevier Science Publishers, pp. 89-108, 1991

36. J. Cao, F. Kimura, M. Shridhar, and M. Ahmadi, *Statistical and Neural Network Classification of Handwritten Numerals: A Comparative Study*, Pro. 11th IAPR International Conference on Pattern Recognition, Vol. 2, pp. 643-646, 1992

37. J. T. Favata, G. Srikantan, and S. N. Srihari, *Handprinted Character/digit Recognition Using a Multiple Feature/Resolution Philosophy*, Proceedings of IWFHR'94, pp. 57-66, 1994

38. K. Chung, Y. Shin, and J. Yoon, *An Application of Feature Selection Based on Neural Network Pruning to Handwritten Character Recognition*, Proceedings of IWFHR'96, pp. 253-266, 1996

39. C. Kaynak and E. Alpaydin, *Multistage Classification by Cascaded Classifiers*, International Symposium on Intelligent Control, pp. 95-100, 1997

40. C. Y. Suen, C. Nadal, R. Legault, T. A. Mai, and L. Lam, *Computer Recognition of Unconstrained Handwritten Numerals*, Proceedings of The IEEE, Vol. 80, No. 7, pp. 1162-1180, 1992

41. L. Xu, A. Krzyzak, and C. Y. Suen, *Methods of Combining Multiple Classifiers and Their Applications to Handwritten Recognition*, IEEE Transactions on Systems, Man, and Cybernetics, Vol. 22. No. 3, pp. 418-435, 1992

42. C. L. Kuan and S. N. Srihari, *A Stroke-based Approach to Handwritten Numeral Recognition*, in Proceeding of U. S. Postal Service Advanced Technology Conference, pp. 539-548, 1988

43. P. D. Gader, D. Hepp, B. Forester, T. Peurach, and B. T. Mitchell, *Pipelined System for Recognition of Handwritten Digits in USPS ZIP Codes*, Proceeding of the Fourth U. S. Postal Service Advanced Technology Conference, pp. 539-548, 1990

44. S. N. Srihari, *Recognition of handwritten and Machine Printed Text of Postal Address Interpretation*, Pattern Recognition Letters, Vol. 14, pp. 291-302, 1993

45. P. Ahmed and C. Y. Suen, *Computer Recognition of Totally Unstrained Handwritten ZIP Codes*, Int. J. Pattern Recognition and Artificial Intelligence, Vol. 1, No. 1, pp. 1-15, 1987

46. T. K. Ho, J. J. Hull, and S. N. Srihari, *Decision Combination in Multiple Classifier System*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-16, No. 1, pp. 66-75, 1994

47. E. Cohen, J. J. Hull, and S. N. Srihari, *Understanding Handwritten Text in a Structural Environment: Determining ZIP Codes from Address*, International Journal of Pattern Recognition and Artificial Intelligence, Vol. 5, Nos. 1&2, pp. 221-264, 1991

48. C. Nadal, R. Legault, and C. Y. Suen, *Complementary Algorithms for the Recognition of Totally Unconstrained Handwritten Numerals*, in Proc. 10th Int. Conf. Pattern Recognition. pp. 443-449, 1990

49. F. Kimura and M. Shridhar, *Handwritten Numeral Recognition Based on Multiple Algorithms*, Pattern Recognition, Vol. 24, No. 10, pp. 976-983, 1991

50. J. J. Hull, A. Commike, and T. -K. Ho, *Multiple Algorithms for Handwritten Character Recognition*, in Proc. Int. Workshop Frontiers in Handwriting Recognition, pp. 117-124, 1990

51. L. Eikvil, K. Aas, and M. Holden, *Tools for Automatic Recognition of Character Strings in Maps*, Proceedings of the 6th International Conference, pp. 741, 1995

52. N. Otsu, *A Threshold Selection Method from Gray-level Histograms*, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-9, No. 1, pp. 62-66, 1979

53. W. K. Pratt, *Digital Image Processing*, John Wiley & Sons, Inc., 1991

54. Y. Amit and D. Geman, *Shape Quantization and Recognition with Randomized Trees*, Neural Computation, No. 9, pp. 1545-1588, 1997

55. S. N. Srihari, E. Cohen, J. J. Hull, and K. Kuan, *A System to Locate and Recognize Zip Codes in Handwritten Addresses*, IJRE, Vol. 1, pp. 37-45, 1989

56. C. S. Lin and C. L. Huang, *New Forms of Shape Invariants from Elliptic Fourier Descriptors,* Pattern Recognition. Vol. 20, No. 5, pp. 535-545, 1987

57. F. P. Kuhl and C. R. Giardina, *Elliptic Fourier Features of a Closed Contour*, Computer Vision, Graphics and Image Processing. Vol. 18, pp. 236-258, 1982

58. P. Wunsch and A. F. Laine, *Wavelet Descriptors for Multiresolution Recognition of Handwritten Characters*, Pattern Recognition, Vol. 28, No. 8, pp. 1237-1249, 1995

59. I. -S. Oh, C. Y. Suen, *A Feature for Character Recognition Based on Directional Distance Distribution*, Proceedings of 4th International Conference on Document Analysis and Recognition, Vol. 1, pp. 288-292, 1997

60. R. J. Schalkoff, *Pattern Recognition: Statistical, Structural and Neural Approaches*, John Wiley & Sons, Inc. 1992

61. J. Cai and Z. Liu, *Integration of Structural and Statistical Information for Unconstrained Handwritten Numeral Recognition*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 21, No. 3, pp. 263-270, 1999

62. S. -B. Cho, *Neural-Network Classifiers for Recognizing Totally Unconstrained Handwritten Numerals*, IEEE Transactions on Neural Networks, Vol. 8, No. 1, 1997

63. H. Yamagata, H. Nishida, T. Suzuki, M. Tachikama, Y. Nakajima, and G. Sato, *A Handwritten Character Recognition System by Efficient Combination of Multiple Classifiers*, IEICE Transactions on Information and Systems, Vol. E79-D, No. 5, pp. 498-503, 1996

64. C. M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press Inc., 1995

65. K. Fukushima, *Neocognition: A Hirerarchical Neural Network Capable of Visual Pattern Recognition*, Neural Networks, Vol. 1, pp. 119-130, 1988

66. H. Schwenk and M. Milgram, *Structured Diabolo-networks for Handwritten Character Recognition*, International Conference on Artificial Neural Networks, pp. 985-988, Springer-Verlag, 1994

67. S. Lee, *Off-line Recognition of Totally Unconstrained Handwritten Numeral Using Multilayer Cluster Neural Network*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 18, No. 6, pp. 648-652, 1996

68. M. P. Craven, K. M. Curtis, B. R. Hayes-Gill, and C. D. Thursfield, *A Hybrid Neural Network/Rule-based Technique for On-line Gesture and Hand-written Character Recognition*, Proceedings of the Forth IEEE International Conference on Electronics, Circuits and Systems, Vol. 2, pp. 850-853, 1997

69. M. T. Hagan, H. B. Demuth, and M. Beale, *Neural Network Design*, PWS Publishing Co., 1996

70. S. Haykin, *Neural Networks: A Comprehensive Foundation*, Prentice Hall, 1999

71. D. R. Lovell, *The Neocognitron As A System for Handwritten Character Recognition: Limitations and Improvements*, PhD Thesis, University of Queensland, 1994

72. P. Simard, Y. LeCun, and J. Denker, *Efficient Pattern Recognition Using a New Transformation Distance*, in Neural Information Processing Systems, pp. 50-58, Morgan Kaufmann, 1993

73. K. T. Blackwell, T. P. Vogl, S. D. Hyman, G. S. Barbour, and D. L. Alkon, *A New Approach to Handwritten Character Recognition*, Pattern Recognition, Vol. 25, No. 6, pp. 655-666, 1992

80. J. M. Keller and J. Osborn, *Training the Fuzzy Integral*, International Journal of Approximate Reasoning, 15:1-24, 1996

81. H. Tahani and J. M. Keller, *Information Fusion in Computer Vision Using the Fuzzy Integral*, IEEE Transactions on Systems, Man, and Cybernetics, Vol. 20, No. 3, pp. 733-741, 1990

82. S. Beiraghi, M. Ahmadi, M. Shridhar, and M. Sid Ahmed, *Application of Fuzzy Integrals in Fusion of Classifiers for Low Error Rate Handwritten Numerals Recognition*, International Conference on Pattern Recognition, Vol. 2, pp. 487-490, 2000

VITA AUCTORIS

NAME:	Xinyu Ye
PLACE OF BIRTH	Zhejiang, China
YEAR OF BIRTH	1971
EDUCATION	Zhejiang University of Technology, Hangzhou, China
	1989 - 1993 B.Sc.
	University of Windsor, Windsor, Ontario
	1999 - 2001 M.A.Sc.