

Automatic Verification of the Outputs of Multiple Classifiers for  
Unconstrained Handwritten Numerals

Jinna Michelle Tan

A Thesis  
in  
The Department  
of  
Computer Science

Presented in Partial Fulfillment of the Requirements  
for the Degree of Master of Computer Science at  
Concordia University  
Montreal, Quebec, Canada

March, 2004

© Jinna Michelle Tan, 2004



National Library  
of Canada

Bibliothèque nationale  
du Canada

Acquisitions and  
Bibliographic Services

Acquisitons et  
services bibliographiques

395 Wellington Street  
Ottawa ON K1A 0N4  
Canada

395, rue Wellington  
Ottawa ON K1A 0N4  
Canada

*Your file* *Votre référence*  
*ISBN: 0-612-91121-7*  
*Our file* *Notre référence*  
*ISBN: 0-612-91121-7*

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

---

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this dissertation.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de ce manuscrit.

While these forms may be included in the document page count, their removal does not represent any loss of content from the dissertation.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

**Canada**



# Abstract

Automatic Verification of the Outputs of Multiple Classifiers for Unconstrained

Handwritten Numerals

Jinna Michelle Tan

Recognition of unconstrained handwritten characters has gained considerable attention in different areas due to its many possible applications. Since the late 60's, research in this area has made impressive progress and many systems have been developed. Some of them achieved high recognition rates of 98% to 99%. However, these systems still misrecognize some patterns that are easily recognized by humans, thereby, decreasing the reliability of the system.

For the purpose of satisfying the high demand of reliability in some practical applications such as bank cheque processing, we proposed a verifier based on structural features.

The verifier is a prototype-based system. Different numbers of prototypes have been constructed for each digit according to the complexity of its structure. Prototypes are built using the structural primitives of a numeral and the relations among them. The local differences between some confusing pairs of numerals such as 4-9, 0-6 are also addressed in the prototypes.

Three classifiers: SVM, LeNet5, and MQDF are used in the recognition stage of the proposed system. SVM is the primary classifier. Patterns rejected by the SVM are passed to the parallel combination of the three classifiers. The proposed verifier is then applied to the classification results.

The proposed system yielded a reliability rate of 99.92% and a recognition rate of 96.50% on MNIST Database. The reliability increased from 99.06% to 99.92% after applying the verifier. Hence, we can conclude that the proposed system has successfully achieved high reliability while maintaining a reasonable recognition rate.

# Acknowledgements

First and foremost, I would like to take this opportunity to express my sincerest thanks to my thesis supervisor, Dr. C. Y. Suen, for his guidance, enthusiastic support, valuable suggestion and encouragement throughout the course of my studies.

I would further like to thank all my friends, especially those in CENPARMI. Thank you for your thoughtful discussion and helpful suggestions; thank you for sharing the good time and for being there during the bad time; your friendship has made this a memorable experience.

Last but never the least; I would like to thank my family for being with me, loving me, believing in me and supporting me. Without their continuous support and encouragement, I would not be able to complete my work.

# TABLE OF CONTENTS

<b>LIST OF FIGURES .....</b>	<b>VIII</b>
<b>LIST OF TABLES .....</b>	<b>X</b>
<b>1. INTRODUCTION.....</b>	<b>1</b>
1.1 STATE OF THE ARTS.....	1
1.2 SYSTEM OVERVIEW .....	7
1.3 OUTLINE OF THE THESIS .....	10
<b>2. FEATURE EXTRACTION .....</b>	<b>11</b>
2.1 PREPROCESSING .....	11
2.2 THINNING.....	13
2.3 FEATURE EXTRACTION .....	13
<b>3. THE VERIFIER.....</b>	<b>23</b>
3.1 PROTOTYPES .....	24
3.2 PROTOTYPE RULES .....	33
3.3 SPECIAL PROTOTYPES.....	36
3.3.1 <i>Special feature extraction technique</i> .....	36
3.3.2 <i>Prototypes based on features extracted from binary image/contour</i> .....	40
3.3.2.1 Numeral one .....	40
3.3.2.2 Numeral six .....	42
3.3.2.3 Numeral eight.....	48
3.3.3 <i>Confusing pairs of numerals</i> .....	52
3.3.3.1 Numerals four and nine.....	52
3.3.3.2 Numerals zero and six.....	55
3.3.3.3 Numerals four and six .....	57
<b>4. THE CLASSIFIERS .....</b>	<b>59</b>
4.1 SVM [37].....	59
4.2 LENET-5[38].....	61
4.3 MQDF [39].....	64
4.4 COMBINATION OF THE CLASSIFIERS .....	66
<b>5. EXPERIMENT.....</b>	<b>72</b>
5.1 VERIFICATION ON THE OUTPUT OF COMBINATION OF CLASSIFIERS .....	72
5.1.1 <i>Database</i> .....	72
5.1.2 <i>Design of experiment</i> .....	72
5.1.3 <i>Experimental result</i> .....	74
5.2 VERIFICATION OF THE COLLECTION OF MISCLASSIFIED DATA .....	77
5.3 ERROR ANALYSIS.....	82
5.3.1 <i>Misrecognized data</i> .....	82

5.3.2 <i>Mis-verified data</i> .....	86
5.3.3 <i>Experiment of human recognition of ambiguous data</i> .....	90
<b>6. CONCLUSIONS</b> .....	<b>95</b>
<b>BIBLIOGRAPHY</b> .....	<b>97</b>
<b>APPENDIX A</b> .....	<b>104</b>



# List of Figures

<i>Figure 1: Block diagram of the proposed system.</i>	9
<i>Figure 2: The patterns used in Smoothing Algorithm.</i>	11
<i>Figure 3: Merging two nearby holes into one hole.</i>	12
<i>Figure 4: Filled hole.</i>	12
<i>Figure 5: Point zone classification.</i>	14
<i>Figure 6: Directions of chain code.</i>	15
<i>Figure 7: A, V, C, D type curves.</i>	19
<i>Figure 8: Z, S type curves.</i>	19
<i>Figure 9: Direction changes for detecting 3- curve</i>	21
<i>Figure 10: Two types of circle.</i>	22
<i>Figure 11: Confusing pair of numerals</i>	27
<i>Figure 12: Variations of the numeral three.</i>	28
<i>Figure 13: Instances of numeral four.</i>	28
<i>Figure 14: Confusing pair of numerals</i>	29
<i>Figure 15: Prototypes of the numeral eight.</i>	35
<i>Figure 16: Numeral eight</i>	36
<i>Figure 17: Procedure for extracting the features of the numeral six.</i>	39
<i>Figure 18: Possible break points of numeral six.</i>	40
<i>Figure 19: Filled numeral nine.</i>	41
<i>Figure 20: Filled six.</i>	42
<i>Figure 21: Comparison of two and six</i>	44
<i>Figure 22: Numeral six.</i>	47
<i>Figure 23: Incorrect skeletons of eight.</i>	48
<i>Figure 24: Left and right contour extractions of eight</i>	49
<i>Figure 25: Contour models of numeral eight.</i>	50
<i>Figure 26 Mis-verified numeral two</i>	51
<i>Figure 27: Two types of numerals four and nine.</i>	52
<i>Figure 28: Features to distinguish numerals four and nine with a loop.</i>	53
<i>Figure 29: Features to distinguish numerals four and nine with the feature ‘almost-loop’</i>	54
<i>Figure 30: Two instances of numeral six.</i>	56
<i>Figure 31: Confusing pairs of numerals six and four</i>	57
<i>Figure 32: Architecture of LeNet-5</i>	61
<i>Figure 33: Data distribution of classifier SVM.</i>	68
<i>Figure 34: Data distributions of MQDF and LeNet5</i>	69
<i>Figure 35: Performance of the GPR with the Verifier</i>	76
<i>Figure 36: Images correctly rejected by the verifier.</i>	76
<i>Figure 37: Errors of the GPR with verifier.</i>	77
<i>Figure 38: Mis-verified images by VSVM<sup>b</sup>.</i>	78
<i>Figure 39: Mis-verified images by VSV2.</i>	78
<i>Figure 40: Mis-verified images by LeNet5.</i>	79
<i>Figure 41: Mis-verified images by POE.</i>	79
<i>Figure 42: Mis-verified images by VSVM.</i>	80

<i>Figure 43: Mis-verified images by VSVM.</i> .....	81
<i>Figure 44: Mis-verified images by MLP</i> .....	82
<i>Figure 45: An instance of numeral nine</i> .....	90
<i>Figure 46: Two instances of numeral four</i> .....	90
<i>Figure 47: Data of the experiment</i> .....	92

## List of Tables

<i>Table 1: Some results reported in the literature.....</i>	<i>5</i>
<i>Table 2: Top 10 confusing digits with frequency.....</i>	<i>23</i>
<i>Table 3: Connection matrix of <math>C_3</math> feature maps and <math>S_2</math> feature maps.....</i>	<i>62</i>
<i>Table 4: Test results with and without verifier.....</i>	<i>75</i>
<i>Table 5: Three categories of misrecognized data.....</i>	<i>83</i>
<i>Table 6: Common mistakes of classifiers.....</i>	<i>84</i>
<i>Table 7: Distribution of mis-verified data.....</i>	<i>87</i>
<i>Table 8: Experimental result of human recognition of ambiguous numerals.....</i>	<i>94</i>

# Chapter 1

## Introduction

### 1.1 State of the arts

Optical Character Recognition (OCR) is one of the most successful applications of automatic pattern recognition. Since the mid 1950s, there have been steady research efforts devoted to automatic processing and recognition of handwritten characters. Nowadays, amazing progress in handwritten characters recognition has been made by using powerful computer along with efficient tools; for instance, Neural Network, Support Vector Machine and Hidden Markov Model.

Recognition of unconstrained isolated handwritten numerals is an important aspect of the OCR. Some real-life applications of numeral recognition are useful in fields such as automatic postal sorting, automatic bank cheque, and financial slip processing. Many techniques have been devoted to improve the recognition performance; some of the most popular techniques used are various feature selection schemes, classification methods, and different system architectures such as multi-experts, and verification modules.

Selection of an effective feature extraction scheme is one of the most important factors in achieving high recognition performance. Devijver and Kittler [1] defined feature extraction as the problem of “extracting from the raw data the information which is most

relevant for classification purposes, in the sense of minimizing the within-class pattern variability while enhancing the between-class pattern variability". A good survey of feature selection is presented in Trier et al [2].

Basically, there are three types of features;

- In the first type of feature, the gray-level or the binary-level themselves are used as the feature vectors. They are usually represented by an N-dimensional vector, where N is the number of pixels in the images. The convenient way of using this pixel vector is template matching; the similarity (or dissimilarity) between the input pixel vector and the templates are computed. One fast evolving approach is using pixel features as direct input to neural networks. Desirable result has been reported [38].
- The second type is the structural feature. The structural features are a set of perceptual entities of a character such as end points, intersection points, bend points, curves, distance information, and directional features [28, 33, 3, and 4].
- The third type of features is the statistical feature. It is the results of global mathematical transformations on an image; moments [5], Fourier descriptions [6], and wavelet transforms [7] are some of these transformations.

Some researchers have tried to combine structural features and statistical features to improve the recognition performance. Cai et al [11] proposed an approach that integrates the statistical and the structural information for the recognition of unconstrained handwritten numerals. This merge has achieved high performance in terms of recognition speed and accuracy.

Besides feature extraction schemes, many classification methods have been used to achieve better recognition performance;

- Prototype methods. The classification of an unseen pattern is to assign its class to the label to the closest prototype by a distance measure. Several approaches such as 1-nearest neighbor and generalized learning vector quantization (GLVQ) [8] belong to this domain.
- Statistical techniques. In this method, the probability of that an observed pattern belongs to a certain class is determined by statistical decision functions and a set of optimal criteria. K-Nearest-Neighbor [9], Bayesian classifier [10], Hidden Markov Model (HMM) [11], and Support Vector Machine [40] are popular handwriting recognition approaches within this domain. Good results have been reported using these approaches.
- Structural techniques. In structural handwritten recognition, characters are represented as unions of structural primitives. The recognition system needs to identify a character's primitives and the particular structural relations among them. [28, 33].
- Neural network. Neural network classifiers are the most commonly used traditional classifiers. It is defined as “a computing structure consisting of a massively parallel interconnection of adaptative neural processors” [55]. Architectures of neural network include Multi-Layer Perceptron (MLP) [12], Convolutional Networks [38], Self-Organized Maps [13], Radial Basis Function [12], Space Displacement Neural Networks [14], Time Delay Neural Networks [15], and Quantum Neural Networks [48].

During the past decade, there has been a tendency to combine the decisions of several classifiers in order to improve recognition results. In terms of implementation, categorization of combined method was made by considering the combination topologies or structures employed. These topologies can be broadly classified into four categories [47].

- **Conditional Topology.** Under this structure, a primary classifier is used. The secondary classifier is deployed when the primary classifier rejects a pattern due to its inability to give a classification or when there is a low confidence in the classification decision. The first classifier handles most of the recognizable patterns while the second classifier deals with the more difficult ones. An example of this topology is the Dual Radial Basis function network designed by Chim et al [16]. One network processes the samples rejected by the other, and vice versa.
- **Hierarchical (Serial) Topology.** Classifiers are applied in succession, with each classifier producing a reduced set of possible classes for each pattern. The classifiers of a stage focus specifically on the reduced set. Examples under these structural can be found in papers [17, 18].
- **Hybrid Topology.** Certain classifiers have better performance on particular types of pattern. Thus, when certain features or parameters of the pattern are extracted, this preference information is used to select the classifier to be used to run in the multiple classifiers system. An example of such an approach is presented in the paper of Powalka et al [19], in which the selection between the applications of

wholistic and segmentation-based word recognizers is based on the estimated length of the word to be processed.

Multiple (Parallel) topology. In this topology, multiple classifiers are first operated in parallel to produce classification decisions of the pattern. Then, these decisions are combined to produce the final result. Commonly used combination strategies are: majority vote [20], Max Rule, Min Rule, Median rule [21], and so on.

Author	Database	Recognition	Error	Rejection	Reliability
Mayraz et al [52]	MNIST	98.32%	1.68%	0.00%	98.32%
Li et al et al [24]	MNIST	99.06%	0.94%	0.00%	99.06%
Lecun et al [38]	MNIST	99.18%	0.82%	0.00%	99.18%
Liu et al [58]	MNIST	99.28%	0.72%	0.00%	99.28%
Decoste et al [51]	MNIST	99.44%	0.56%	0.00%	99.44%
Kussul et al [57]	MNIST	99.50%	0.50%	0.00%	99.50%
Suen et al [44]	CENPARMI	93.05%	0.00%	6.95%	100.00%
Cho [22]	CENPARMI	96.05%	3.95%	0.00%	96.05%
CENPARMI [23]	CENPARMI	98.85%	1.05%	0.00%	98.85%
Liu et al [58]	CENPARMI	98.90%	1.10%	0.00%	98.90%
Liu et al [59]	CENPARMI	99.15%	0.85%	0.00%	99.15%
Oliveira et al [60]	NIST SD19	99.16%	0.84%	0.00%	99.16%
Liu et al [59]	NIST SD19	99.47%	0.53%	0.00%	99.47%

Table 1: Some results reported in the literature.

Table 1 shows some results reported in the literature of recent years. Although some current methods perform efficiently in certain situations, few have achieved a satisfactory level for practical applications demanding high reliability such as bank cheque processing. The system proposed by Suen et al [44] is an exception; it is the only one that has achieved 100% reliability. This system is a combination of four experts using majority vote. The main reason of the zero substitution is the complementary nature of the four independent experts. The majority of them do not make the same errors on the test data of CENPARMI. However, we have observed that some errors are common to some



classifiers with different structures [37, 38, 39, and 52] on MNIST database. So there is no guarantee that the system proposed by Suen et al can achieve zero substitution on database other than CENPARMI's. Moreover, the recognition rate of the system is 93.05%; a higher recognition performance is expected by practical applications.

In financial applications, errors are more intolerable than rejections since extra effort is necessary to detect errors; therefore, very high reliability is expected from the system. Consequently, a module that can achieve a high reliability in relatively easy way is necessary. Moreover, this module should require a small implementation cost while maintaining a reasonable recognition rate. This module is defined as the verifier. It is an expert which confirms or negates a classification result from a general-purpose recognizer. Therefore, the verifier improves the system reliability. Recently, some researches about the verifier in the domain of handwritten recognition have been done.

Takahashi [35] is among the earliest that mentioned the concept of verification in the document analysis domain. In his system, linear tournament verification is executed using a small one-to-one network verifier to improve the ordering of the top candidates. Another example is Lee et al [25] who incorporated a verifier into the system for handwritten numerals. Zhou et al [48] designed and implemented a quantum neural network (QNN) based verification enhanced system for handwritten numerals. The importance and functionality of a verifier was analyzed and consciously emphasized in Zhou et al [48]'s paper. It also introduced a testing hypothesis to evaluate the precision rate of the system. The reliability of the system increased from 97.84% to 99.04% after the verifier was embedded into the system.

Further investigation of verifier is presented in Luiz et al [55]. It introduced the concept of levels of verification, where two levels are considered: high-level and low-level. High-level verifiers are defined as those that deal with a sub-set of the classes considered by the general-purpose recognizer. The goal of the verifier at this level is to confirm or deny the hypotheses produced by the general-purpose recognizer by recognizing them. The low-level verifiers are defined as those that deal with meta-classes of the system such as parts of characters. The purpose of a low-level verifier is to determine whether a hypothesis generated by the general-purpose recognizer is valid or not. The authors implemented the absolute verifiers and pair-wise verifiers and combined them with the general-purpose recognizer respectively. Both of these two verification strategies improved the reliability of the system. The absolute verifiers show a better performance than the pair-wise verifiers due to the lack of a training set of confusing pairs of numerals. Moreover, verification strategy was applied to the recognition of numeral strings [26], words [27], and so on. Since these research fields are not the focus of our research, detailed discussions will not be provided here.

## **1.2 System Overview**

The focus of our work is the verification of unconstrained isolated handwritten numerals. Unconstrained handwritten numerals are written in free styles, using various handwriting instruments, and probably sloppily. These numerals are written by people when they enter the zip codes on envelopes and courtesy amounts on cheques.

Some practical systems, such as those for automatic bank cheque processing, require a very high reliability. Errors are more intolerable than rejections since the costs to find out

the errors are much higher. The goal of the proposed system is to decrease the error rate while maintaining a reasonable recognition rate, thereby achieving a high reliability of the system.

Basically, the system is composed of four parts: preprocessing, feature extraction, recognition and verification. The block diagram of the proposed system is presented in Figure 1.

In the preprocessing stage, the system applies a variety of smoothing techniques to fill small holes and remove noises. The next stage is the feature extraction. A numeral is a union of structural primitives. Human beings extract the feature of a numeral by identifying the structural primitives and the geometric relations among them. In order to simulate human recognition, features used in the proposed system are structural ones, such as end points, loops, and concavities.

The third part of the system is the recognition. Three classifiers: SVM, LeNet5, and MQDF proposed by [37], [38], and [39], respectively, are combined in the proposed system to achieve a more reliable recognition performance. From the output of the classifier, only the ranked candidates are sent as the input of the verifier. Since recognition is not the focus of this thesis, detailed description of the three classifiers is not presented in the thesis.

The last part of the system is the verifier. The verification is the focus of the thesis.. Generally, the verifier will confirm a candidate, which is given by the classifiers, if the confidence value is equal or larger than a threshold; otherwise, it will reject the candidate. Some verifiers also correct recognition results. However, the major functionality of a verifier is to determine whether an input pattern has the characteristics of a particular

class. However, it does not decide to which class the pattern belongs; using a verifier to perform a classifier's work is risky. To some extent it is preferable for a system to make no decision rather than a wrong decision since the cost to detect and correct an error is much higher than a rejection. Therefore, the proposed verifier will not correct recognition results.

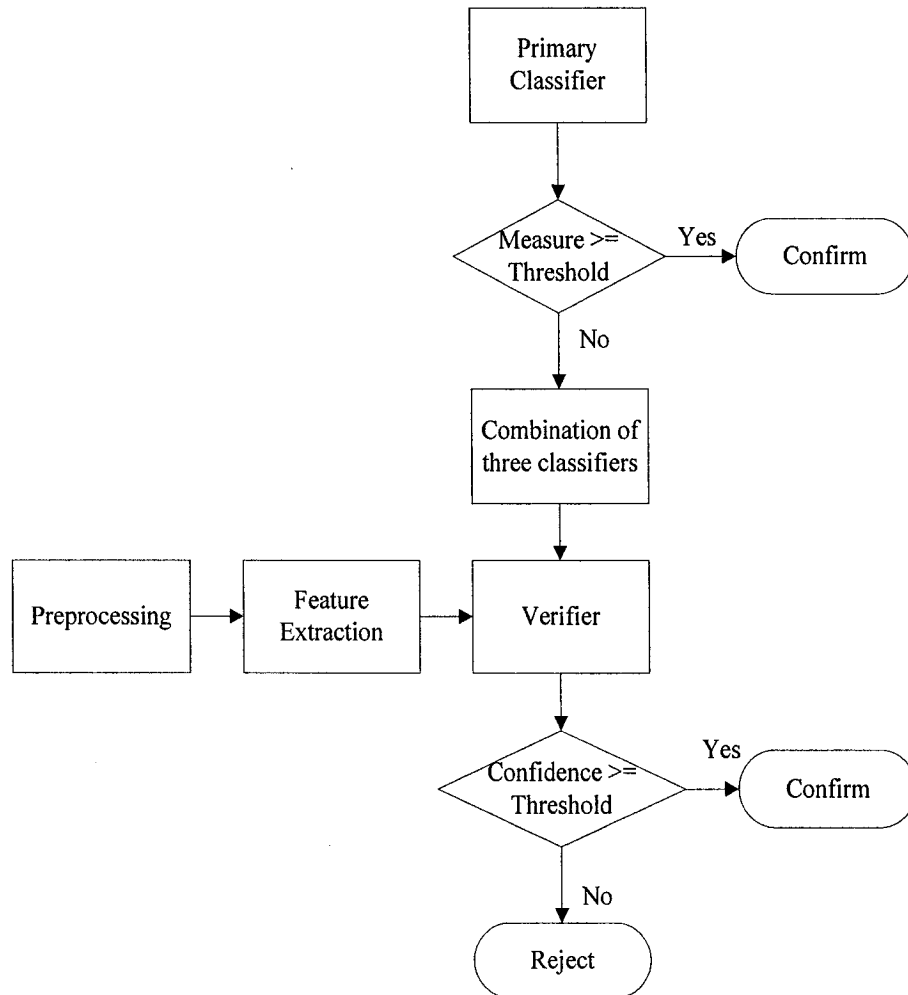


Figure 1: Block diagram of the proposed system.

## 1.3 Outline of the thesis

The thesis comprises six chapters organized as follows:

The first chapter gives the introduction: it reviews the state-of-art of OCR and introduces the focus of the thesis.

The second chapter describes the preprocessing techniques and thinning algorithm used in the system. Then, the definition and detailed description of structural features used in the verifier are given.

The third chapter introduces the most important component of the system: the prototype based verifier. Description of prototypes of each numeral is provided. Distinguishing confusing pairs of data is emphasized in this chapter.

The fourth chapter briefly describes the three classifiers used in the system. Then, detailed descriptions of the combination of the three classifiers are presented. A conditional combination topology is first applied. The SVM has been selected as the primary classifier. Then, test data that are rejected by the primary classifier are passed to the combination of classifiers, and combined by a parallel topology.

The fifth chapter presents the experiments and results of the proposed system: first, the experimental design and result of the proposed system on MNIST database is given; second, the experimental result of the verification on some misrecognized data produced by some classifiers found in references is illustrated; finally, an analysis of errors made by the classifiers and verifiers is described.

The conclusion is drawn in the sixth chapter. It summarizes the performance and limitation of the proposed system. Future work is also outlined in this chapter.

The appendix contains a collection of misrecognized images of the selected classifiers.

## Chapter 2

### Feature Extraction

#### 2.1 Preprocessing

An image may have small holes and small spurs. This problem can be handled by applying template matching, where the center pixel is processed when the template finds a match in the image. Templates are shown in Figure 2 [28]. In these templates, '0' represents a background pixel, "1" represents a black pixel, and "x" represents a "don't care" pixel. The center pixels of patterns (a)-(c) and their rotated patterns by  $90^\circ$ ,  $180^\circ$ ,  $270^\circ$  are filled. The center pixels of patterns (d)-(e) and their rotated patterns by  $90^\circ$ ,  $180^\circ$ ,  $270^\circ$  are deleted.

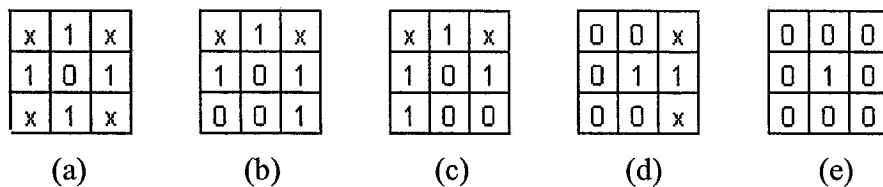


Figure 2: The patterns used in Smoothing Algorithm.

However, besides the problem of small holes and small spurs, there are many images where the positions of holes are weird and the number of holes is larger than normal in a specific position. In order to solve this problem, we applied the three techniques proposed by Legault & Suen [29]. First, if there is a hole with an area of less than three pixels, the hole will be filled; second, if there are more than three holes, only the largest three are

kept; finally, if there are more than one hole at the top or bottom part of the image, these holes will be merged if they are nearby. Figure 3 shows an example of merging holes.

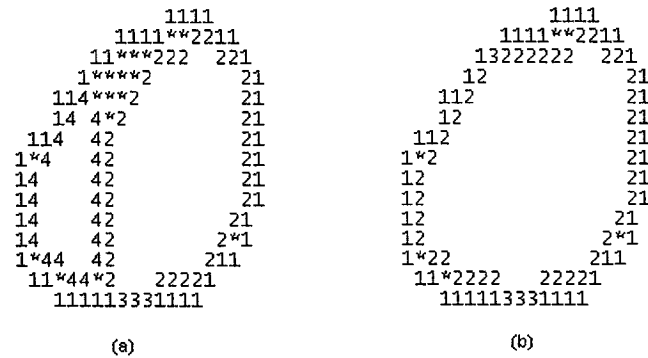


Figure 3: Merging two nearby holes into one hole.

The three techniques of Legault & Suen are applied preceding the template matching to avoid the situation as shown in Figure 4. In the figure, “\*” represents a black pixel and “1”, “2”, “3”, and “4” represent background pixels. If the template matching is applied first, then pixels 2 and 4 will be filled in the process of template matching as illustrated in Figure 4b. Then, the three techniques of Legault & Suen will fill pixels 1 and 3 since the hole is less than three pixels as shown in Figure 4c. Thereby, the hole is completely filled. To prevent this error, we applied the three techniques of Legault & Suen, and applied the template matching afterwards. Holes whose areas are larger than three pixels will be marked in stage one and skip the template matching to preserve more information of holes.

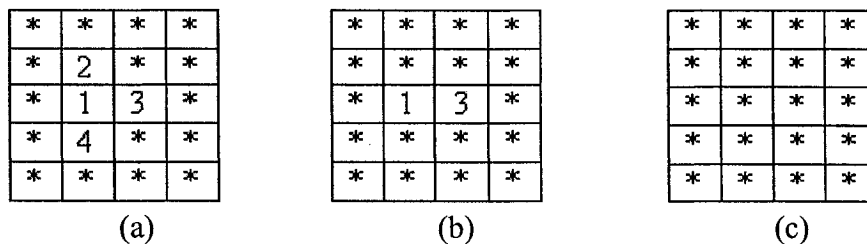


Figure 4: Filled hole.

## **2.2 Thinning**

Thinning is defined as a process to remove the outer layer of an object shape in an image until a one-pixel width skeleton is obtained. Many thinning algorithms have been proposed in the past; most of them remove the outer part layers one after the other. The Zhang & Suen [30] thinning algorithm is one of them. It is used to get the skeleton of the smoothed image in the proposed system. The Zhang & Suen thinning algorithm consists of two successive iteration thinning algorithms. In the first iteration, it removes all boundary pixels from the southeast side of an object, and then the intermediate result is saved. In the second iteration, it removes all boundary pixels from the northwest side of the object obtained in the first iteration. The Zhang & Suen thinning algorithm is fast and stable although it may produce some extra branches as other pixel-based thinning algorithms do. We observed that the length of the majority of extra branches produced by the thinning algorithms is small. Thus, this problem can be solved by removing branches whose length is smaller than a threshold.

We also tried to apply some template-based thinning algorithms, such as those described in Wu et al[31] and Han et al[32], but Zhang & Suen has a better performance than the others. Therefore, the Zhang & Suen algorithm was the one used for further development of the system.

## **2.3 Feature Extraction**

Humans recognize a character by identifying the unions of structural primitives of the character and the relations among them. In order to simulate humans' recognition process, structural features are used in the system.



Feature Extraction is based on the skeleton of an image. There are different approaches to decompose the skeleton into primitives; some of these approaches are based on the polygonal approximation of the skeleton and on the detection of feature points. The proposed system used the latter approach.

The geometric distribution of features is one essential characteristic of features. In order to describe the position of a feature clearly and precisely, the image is divided into nine zones as illustrated in Figure 5. The minimum rectangle boundary of an image is first detected; then, it is divided into nine even zones. A feature may occur in one or more zones.

zone 7	zone 8	zone 9
zone 4	zone 5	zone 6
zone 1	zone 2	zone 3

Figure 5: Point zone classification.

The features used in the proposed system are similar to those proposed by Siy and Chen [33], but with more new features added.

First, two standard feature points are detected: terminal point and intersection point.

- 1) A terminal point is the point that has only one black pixel in its eight neighbors.
- 2) An intersection point is the point that has more than two black pixels in its eight neighbors and these black pixels cannot be a direct neighbor of one another.

Based on the two standard feature points, a skeleton can be decomposed into branches.

- 3) A branch of the skeleton is the skeleton segment that starts from a standard feature point and ends with a standard feature point.

Branches are stored in the Freeman chain code to facilitate the extraction of other features.

Figure 6 shows the 8 directions of the chain code.

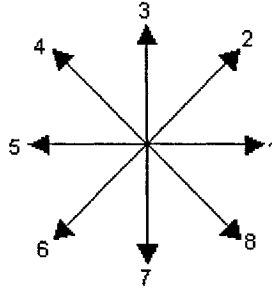


Figure 6: Directions of chain code.

Second, a non-standard feature point: bend point is extracted. A bend point is the point at which there is an abrupt change in the slope of the segment. Bend point is an important feature which can serve to differentiate numerals. Without this kind of feature points, the structure of some numerals cannot be described clearly. The bend points are detected according to the algorithm proposed by Baptista and Kulkarni [34]:

If the length of current branch is greater than six pixels, a record of the last seven points tracked along the current branch is kept. Let the coordinates of the seven pixels be  $(r_1, c_1), (r_2, c_2), \dots, (r_7, c_7)$ .

If  $\{ABS[\sum_{i=1}^3 (r_{i+1} - r_i) - \sum_{i=4}^6 (r_{i+1} - r_i)] + ABS[\sum_{i=1}^3 (c_{i+1} - c_i) - \sum_{i=4}^6 (c_{i+1} - c_i)]\} > 3$ , then a bend

point would be identified at the 4<sup>th</sup> last point tracked.

Third, other features such as line and concavity features are extracted.

Straight Line features:

- 4) horizontal straight line
- 5) vertical straight line
- 6) straight line with positive slope
- 7) straight line with negative slope

The slopes of the positive and negative straight lines are also extracted to facilitate the prototype construction.

Portion of circle features:

- 8) C curve
- 9) D curve
- 10) A curve
- 11) V curve
- 12) S curve
- 13) Z curve
- 14) 3 curve

The circle itself is defined also:

- 15) circle

#### **Definition of lines and concavities**

The opening and depth of a concavity is an important factor which can define the above-mentioned features. The definitions of the opening and depth are presented below.

Let a given branch  $F$  be a node list  $\{n_1, n_2, \dots, n_m\}$ , the coordinates of node  $n_i$  is  $(r_i, c_i)$ , where  $r$  stands for rows and  $c$  stands for columns.

The opening of a concavity is defined as the distance between the two terminal nodes of the branch. Let the coordinates of the two terminal nodes of the branch be  $(r_1, c_1)$ , and  $(r_m, c_m)$ , the opening of the curve is

$$Opening(F) = \sqrt{(r_m - r_1)^2 + (c_m - c_1)^2}$$

The depth of a concavity is defined as

$$Depth(F) = \text{Max}_{1 \leq i \leq m} \{d_i\}$$

where  $d_i$ , which is the perpendicular distance of node  $n_i$  from the line joining the two terminal nodes  $n_1$  and  $n_m$  of the branch, is given by:

$$d_i = \frac{|(r_1 - r_i)(c_m - c_1) - (c_1 - c_i)(r_m - r_1)|}{\sqrt{(r_m - r_1)^2 + (c_m - c_1)^2}}$$

### 1) Definition of straight line

The straightness is measured by two factors:

$$\begin{aligned} Measure1 &= Depth(F) / Opening(F) \\ Measure2 &= MSE \end{aligned}$$

where the Mean Square Error (MSE) of the regression equation  $R$  on  $C$  ( $R=A+BC$ ) is calculated in the following procedure:

$$MSE = (\sum_{i=1}^m r_i^2 + mA^2 + B^2 \sum_{i=1}^m c_i^2 - 2A \sum_{i=1}^m r_i - 2B \sum_{i=1}^m c_i r_i + 2AB \sum_{i=1}^m c_i) / m$$

$$A = (\sum_{i=1}^m r_i + \sum_{i=1}^m c_i^2 - \sum_{i=1}^m c_i \sum_{i=1}^m c_i r_i) / (m \sum_{i=1}^m c_i^2 - (\sum_{i=1}^m c_i)^2)$$

$$B = (m \sum_{i=1}^m c_i r_i - \sum_{i=1}^m c_i \sum_{i=1}^m r_i) / (m \sum_{i=1}^m c_i^2 - (\sum_{i=1}^m c_i)^2)$$

$m =$  the total number of nodes in the given branch.

A given branch is determined as a straight line when Measure1  $< 0.15$  and Measure2  $< 0.4$ . Otherwise, the branch will be classified as a portion of circle. These two thresholds are determined empirically.

The classification of a straight line is based on the slope of the best fit line of the given branch.

A branch  $\in$  horizontal straight line if  $-\tan(1/16 * \pi) \leq slope \leq \tan(1/16 * \pi)$

A branch  $\in$  vertical straight line if  $slope \geq \tan(7/16 * \pi) \parallel slope \leq -\tan(7/16 * \pi)$

A branch  $\in$  positive straight line if  $\tan(1/16 * \pi) < slope < \tan(7/16 * \pi)$

A branch  $\in$  negative straight line if  $-\tan(7/16 * \pi) < slope < -\tan(1/16 * \pi)$

## 2) Definition of portion of Circle

First, a portion of circle is classified into two subsets: horizontal curve (HC) and vertical curve (VC). This classification is based on the slope of the line LC, which is connecting the two end points of the portion of circle.

A branch  $\in$  HC if  $-\tan(1/4 * \pi) < slope < \tan(1/4 * \pi)$

A branch  $\in$  VC if  $slope \geq \tan(1/4 * \pi) \parallel slope \leq -\tan(1/4 * \pi)$

If a branch  $\in$  HC, four lines parallel to LC are drawn as illustrated in Figure 7a. Two lines LA1, LA2 are drawn above LC, then, other two LB1, LB2 are drawn below LC.

Similarly, if a branch  $\in$  VC, four lines parallel to LC are drawn (Figure 7b); two lines LL1, LL2 are drawn above LC, and the other two LR1, LR2 are drawn below LC.

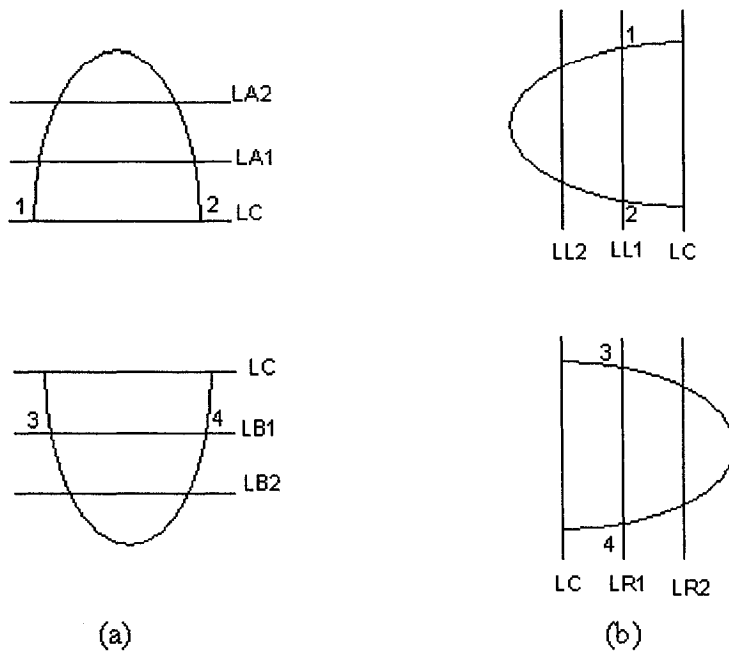


Figure 7: A, V, C, D type curves:  
 (a) Horizontal curve, (b) Vertical curve

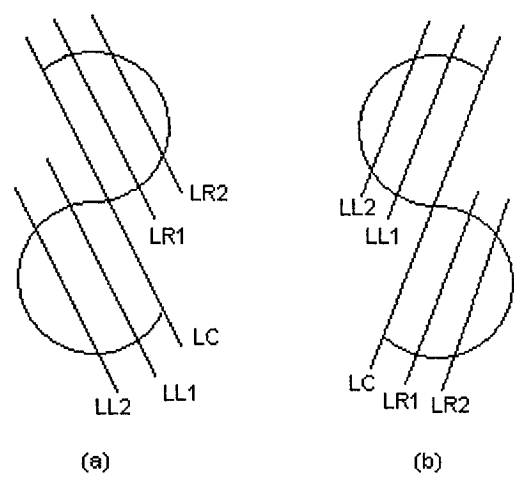


Figure 8: Z, S type curves:  
 (a) Z curve, (b) S curve

Let the number of intersection points of lines LC, LA1, LA2, LB1, LB2, LL1, LL2, LR1, LR2 and the given branch be denoted as IC, IA1, IA2, IB1, IB2, IL1, IL2, IR1, IR2, respectively.

Let the two intersection points of line LL1 (or LA1) and the given branch be labeled as  $(x_1, y_1)$  and  $(x_2, y_2)$ , and the two intersection points of line LR1 (or LB1) and the given branch be identified as  $(x_3, y_3)$  and  $(x_4, y_4)$ .

The condition of a given portion of a circle contains feature set C, D, A, V, S, Z, which are listed below:

$$A = \{b \mid b \in HC \wedge IC = 2 \wedge IA1 = 2 \wedge IA2 = 2 \wedge IB1 = 0 \wedge IB2 = 0\}$$

$$V = \{b \mid b \in HC \wedge IC = 2 \wedge IA1 = 0 \wedge IA2 = 0 \wedge IB1 = 2 \wedge IB2 = 2\}$$

$$C = \{b \mid b \in VC \wedge IC = 2 \wedge IL1 = 2 \wedge IL2 = 2 \wedge IR1 = 0 \wedge IR2 = 0\}$$

$$D = \{b \mid b \in VC \wedge IC = 2 \wedge IL1 = 0 \wedge IL2 = 0 \wedge IR1 = 2 \wedge IR2 = 2\}$$

$$S = \{b \mid [b \in VC \wedge IC = 3 \wedge IL1 = 2 \wedge IL2 = 2 \wedge IR1 = 2 \wedge IR2 = 2 \wedge \max(y_1, y_2) > \min(y_3, y_4)] \vee [b \in HC \wedge IC = 3 \wedge IA1 = 2 \wedge IA2 = 2 \wedge IB1 = 2 \wedge IB2 = 2 \wedge \max(x_1, x_2) > \min(x_3, x_4)]\}$$

$$Z = \{b \mid [b \in VC \wedge IC = 3 \wedge IL1 = 2 \wedge IL2 = 2 \wedge IR1 = 2 \wedge IR2 = 2 \wedge \max(y_3, y_4) > \min(y_1, y_2)] \vee [b \in HC \wedge IC = 3 \wedge IA1 = 2 \wedge IA2 = 2 \wedge IB1 = 2 \wedge IB2 = 2 \wedge \max(x_3, x_4) > \min(x_1, x_2)]\}$$

For the curves C, D, A, and V, the opening and the depth of the concavity are important characteristics; therefore, they are saved together with the features.

To detect the feature called 3-curve in a given branch, we first trace the direction change of the chain code of the given branch. If there exists a direction change from direction 6 to 8 (direction from 6 to 7, and then to 8 is also valid), or from direction 4 to 2 (direction from 4 to 3, and then to 2 is also valid), as illustrated in Figure 9, and the direction change point M is in zone 4, zone 5 or zone 6; then, point M is treated as the midpoint of curve 3. The branch is broken into two segments: one above the midpoint, the other one below the midpoint.

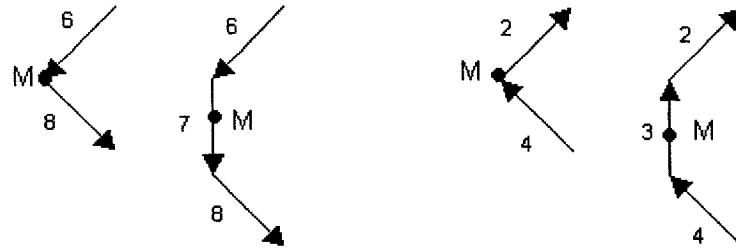


Figure 9: Direction changes for detecting 3- curve

Let an end point other than point M of upper segment be  $(x_1, y_1)$ , and an end point other than point M of the lower segment be  $(x_2, y_2)$ ; a branch is a 3-curve if the following conditions are met:

- a. upper segment is D curve or A curve;
- b. lower segment is D curve or V curve;
- c.  $y_1 > y_2$

### 3) Definition of a circle

If a segment is circular, it contains the feature called circle.

A circle can occur in one branch, which happens in most cases of numeral zero; or it can be formed by many branches. Two samples of circle are shown in Figure 10. In the illustration, figure a is a circle constructed by one branch and figure b is a circle constructed by two branches.



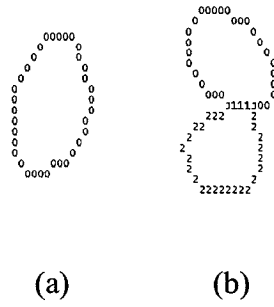


Figure 10: Two types of circle.

In this chapter, we first presented the preprocessing techniques and the Zhang & Suen thinning algorithm. Then the definitions and the extraction methods of the structural features used in the proposed system: lines, portions of circle, and the circle itself were described in detail. The prototypes of the verifier are constructed based on these features.

# Chapter 3

## The Verifier

The goal of a verifier is to evaluate precisely the result produced by a classifier in order to compensate for its weakness due to particular training. This evaluation, consequently, makes the whole system more reliable.

The proposed verifier is designed to “plug and play” and is applied after a general-purpose recognizer. It is used without prior knowledge of the implementation details of the recognition system.

There are three types of verifiers as defined by Takahashi & Griffin [35]: absolute verification for each class (i.e., is it a numeral “1”?), pair wise verification between two classes (i.e., is it a numeral “4” or “9”?), and verification in clustered, visually similar, classes (i.e., is it a “0”, “6”, or “9”).

The proposed verifier is an absolute verifier. However, when building prototypes and match rules, we put efforts in measuring the local differences between confusing pairs of numerals, such as pair-wise digits in Table 2. Digits in Table 2 are obtained from misrecognized data of some classifiers [37, 38, and 39].

confusion	frequency	confusion	frequency
8 - 9	27	7 - 9	17
4 - 9	26	3 - 8	13
3 - 5	22	3 - 7	12
5 - 8	20	3 - 9	12
2 - 7	18	5 - 6	12

Table 2: Top 10 confusing digits with frequency.

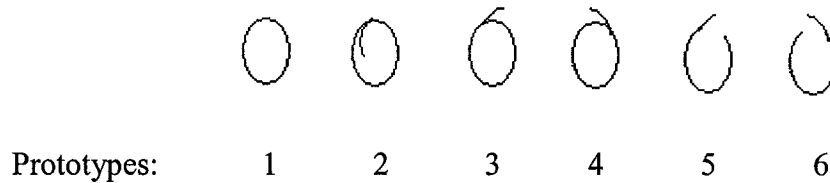
### 3.1 Prototypes

There are ten modules in the proposed verifier, one module for each numeral. Several prototypes and matching rules are built for each numeral according to the detected primitives, geometrical properties of features, and the relations among features. Because the shapes of some numerals are stable, but some numerals will vary due to the writing habit, the number of prototypes for each numeral is determined by the shape variations of a numeral.

For a given input, it is first determined to which prototype the patten belongs based on its structure and geometrical properties of features. Then, we evaluate the input image using the matching rules of the determined prototype. An image that has a confidence value higher than the threshold will be confirmed; otherwise, it will be rejected.

Prototypes of each numeral are briefly described below:

- Module 0



The structure of the numeral zero is simple. We observed that most of them consist of a circular loop, which is an easy-to-detect and stable feature. It is an important feature to distinguish zero from other numerals without a circle, such as numeral one and seven. Then we continue to look at other features that can further confirm an image as a numeral zero.

The structure of each prototype in module zero is shown above. We need to point out that it shows only some instances of the prototypes. It does not mean that the

input image must have a shape similar to one of the instances; shape variation is allowed. For example, as long as an image has a circle and a small stroke inside the circle, no matter in what position the short stroke is located, the image belongs to prototype 2. Then, further investigation will be processed according to the evaluation rules such as the length of the stroke, the slope of the stroke, etc.

Six prototypes were built according to the variation of shapes in the numeral zero. Since the most confusing numeral with the numeral zero is the numeral six, we paid special attention to the local differences between zero and six when designed their prototypes. For instance, prototypes 3 and 4 are similar; the difference between them is the direction of the upper stroke, which is on top of the circle. If the stroke has the direction as prototype 3, it has a higher probability to be a numeral six. Therefore, rules of prototype 3 were built more restrictive than prototype 4. The same situation applied to prototypes 5 and 6.

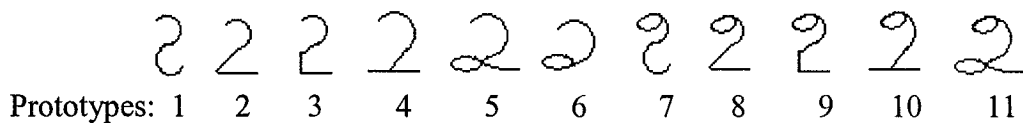
- Module 1

Prototypes:		(	∟	⊥
	1	2	3	4

Numeral one has the simplest structure. Its common structure is a vertical (or almost vertical) stroke. However, since the image we examined is handwritten, the stroke may not be as straight as that of a printed numeral one. Therefore, besides the prototype 1, which is a straight line, we constructed prototype 2 with a small curve. Though the graphs of prototypes 1 and 2 show only the vertical stroke/curve, a stroke with a positive slope such as “/” or a stroke with a negative slope as “\” also belong to

these prototypes. The confidence value of the image depends on the straightness and the slope of the line/curve. Although images in the structures of prototypes 3 and 4 are not as common as prototypes 1 and 2, they are still being written from time to time; thus, prototype 3 and 4 need to be considered when designing prototypes. The structures of images in prototypes 1 and 2 are unique; they will not be confused with other numerals. However, images in prototype 3 or prototype 4 are probably confused with numeral seven or numeral two respectively. Thus, local differences between these pairs of confusing numerals are measured closely in the matching rules.

- Module 2



The number of handwriting styles of the numeral two is larger than that of numerals zero and one. Therefore, the prototype design is more complex. Twelve prototypes were constructed for the numeral two. As we observed, prototypes 1 to 5 are the most usual writing styles. Though prototypes 6 to 11 do not occur as often as prototypes 1 to 5, they are still common writing styles. Ignoring them will lead to a high rejection rate.

Although the upper D curve at the top of prototypes 1 to 5 is a smooth curve, images having the same structures but with a sharper D curve also belong to these prototypes. However, in some prototypes such as prototype 4, if a sharp D curve exists, we need to apply some rules to check whether it is a numeral one. As illustrated in Figure 11,

the numeral one and two have the same structure. We need to measure the angle 'a' and compare the length of segment 1 and segment 2 to distinguish them.

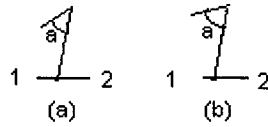
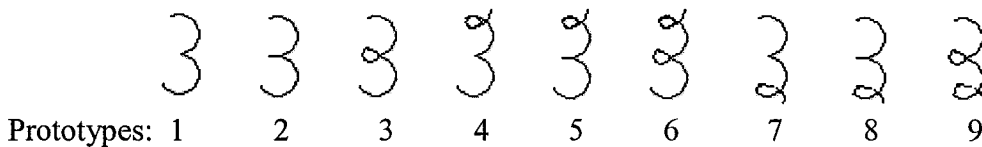


Figure 11: Confusing pair of numerals:  
(a) numeral one, (b) numeral two

We also observed that the numeral two is easily confused with numerals seven and three. Thus we paid attention of distinct characteristics among these numerals when building their prototypes and matching rules.

- Module 3



The structures of the numeral three are not complex. In general, they have two D curves, one on top of the other one. These two D curves have different connecting ways; this enlarges the number of prototypes. They may occur in one branch as illustrated in prototype 1; they may be connected with a short stroke by a junction point as shown in prototype 2; or they may be connected to a small circle by a junction point as shown in prototype 3. We also considered cases where a top circle is connected to the top D curve or a bottom circle connected to the bottom D curve as in prototypes 4 to 9.

However, as illustrated in Figure 12, there are many variations within some prototypes. For instance, there is a bend point between the two D curves in figure a,

but none in figure b. Moreover, in figures c and d, the depth of concavity of bottom curve is so small and can hardly be claimed as a D curve. Further more, the orientations of the bottom curve of figures c and d are totally different. These variations make the matching rules more complicated.

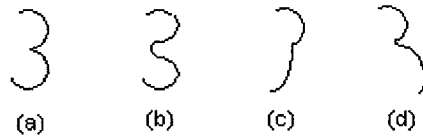
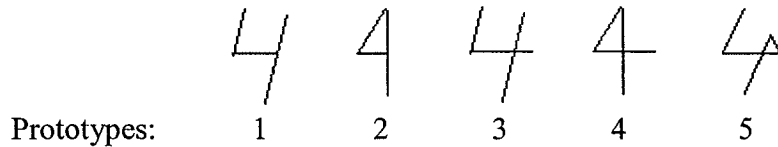


Figure 12: Variations of the numeral three.

- Module 4



There are five prototypes in Module 4. The structures of the numeral four are not very complex. However, building matching rules is difficult because the numeral four is easily confused with numerals nine and six in certain writing styles. For example, for the image in prototype 3, if it is written with a sharp C curve as displayed in Figure 13a, it is clear that it is a numeral four; on the other hand, if it is written with a smooth C curve as presented in Figure 13b, it is hard to determine whether it is a four or nine. Rules to distinguish the numeral four from nine and six will be discussed in detail in a later section.

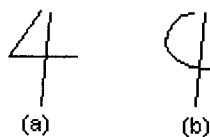
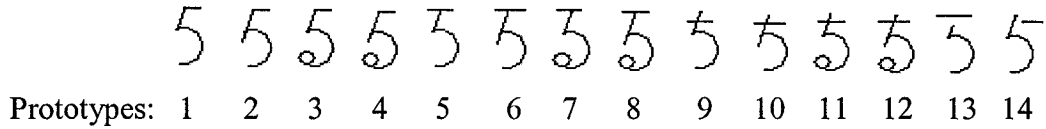


Figure 13: Instances of numeral four.

- Module 5



Fourteen prototypes were constructed for Module 5 as shown above. Prototypes allow variations. For instance, the D type curve in the bottom part of the 5 can be a smooth curve like ‘5’ or a sharp curve like ‘5’. They belong to the same prototype.

When dealing with images in prototypes 5 to 8, we need to measure the local differences between numerals five and three. Some numerals three may have the same structure as the numeral five, such as the image displayed in Figure 14b. The only difference between them is the length of segment 2. Thus, the ratio of the length of segment 2 to the length of segment 1 is an important feature in prototypes 5 to 8.

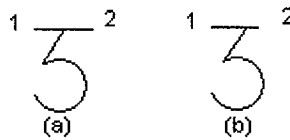
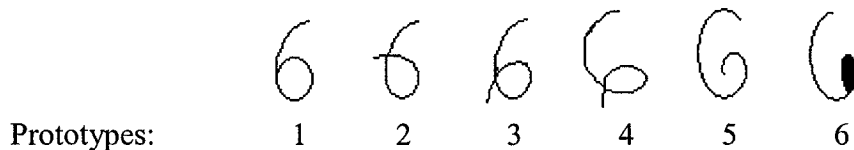


Figure 14: Confusing pair of numerals:  
(a) numeral five, (b) numeral three

- Module 6



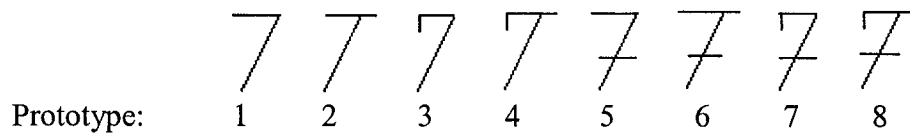
Six models were built for the numeral six as displayed above. Prototype 1 consists of an upper stroke and a lower circle. Prototypes 2 to 4 are similar to prototype 1 but with one more lower stroke. The structure of prototype 5 is different from prototypes



1 to 4. It is in one branch and does not contain a circle. Prototype 6 handles images that have a filled circle at the bottom.

Obviously, the structures of prototypes 2, 3 and 4 look alike. The only difference among these prototypes is the position of the junction point, which connects the upper stroke, the lower stroke and the lower circle of the six. In prototype 2, the junction point is located at the top of the circle; in prototype 3, the junction point occurs in the middle-left of the circle; in prototype 4, the junction point is positioned at the lower-left of the circle. The position of the junction point is an important feature. It determines with which numeral the pattern is easily confused. Numerals in prototype 2 are confused with numeral eight if the slope of the lower stroke is negative and the lower stroke is long. On the other hand, numerals in prototypes 3 and 4 are confused with numeral four if the lower stroke is long and barely vertical. The measuring rules in prototype 4 are more restrictive than prototype 3 because the position of junction point of prototype 3 increases the probability of the image to be a numeral four. Further discussion on the confusing pair of numerals four and six will be presented in a later section.

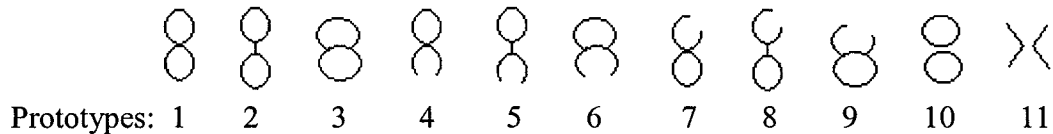
- Module 7




Eight models were built for Module 7. They can be separated into two groups. Prototypes 1 to 4 belong to one group, prototypes 5 to 8 belong to the second group;

this last group has one more horizontal bar in the lower part of the image. Both prototypes 1 and 3 consist of a long horizontal stroke at the top and a long barely-vertical stroke connected to the left end point of the horizontal stroke. If the angle of these two strokes is small and the length of horizontal stroke is short, prototypes 1 and 3 may be confused with numeral one. On the other hand, the second group of prototypes may be confused with numeral two if the bottom vertical stroke is short. These characteristics are closely measured in the corresponding prototypes.

- Module 8



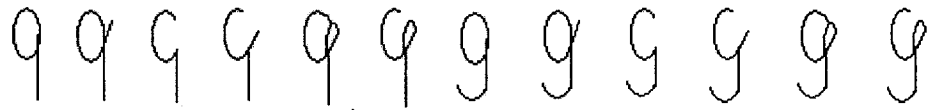
Eleven models were constructed for the numeral 8 as shown above. Variations are allowed in the prototypes. For example, the case  is still in prototype 1 although an extra stroke is present at the top of the numeral. The confidence value of the image changes according to the length and position of the extra stroke.

Basically, prototypes 1, 2, and 3 belong to one group; prototypes 4, 5, and 6 belong to another group; finally, prototypes 7, 8 and 9 belong to the third group. The structures of members in the first group are similar, except the way of connecting the top circle and the bottom circle. In prototype 1, the two circles of eight are connected by one junction point; in prototype 2, one circle connects to one end point of a small vertical stroke, the other circle connects to the other end point of the small stroke; in prototype 3, the two circles are connected by a common stroke. The same situations

apply to the second and third group. Prototype 10 represents the numeral eight with two disconnected circles.

Prototypes 1 to 10 rely on information extracted from skeleton of an image. However, skeleton has its own weakness that it may not contain all information of a pattern. In numeral eight, it is not rare that the lower circle is filled, which can not be described accurately by the skeleton. Thus, the last prototype is designed to compensate for the weakness of the skeleton. This prototype is based on the information extracted from the contour. Detailed information of this prototype will be described in a later section.

- Module 9



Prototypes: 1 2 3 4 5 6 7 8 9 10 11 12

There are 12 prototypes in Module 9. The first six prototypes are similar to the last six prototypes. The only difference is the shape of the bottom branch connected to the top circle or the feature ‘almost-loop’. In the first six prototypes, this branch is a straight line; however, the bottom branch in the last six prototypes is a curve. The reason to differentiate these two types of image is that a numeral nine with a straight line is easier to be confused with other numerals, such as four and six. Therefore, the shape of the upper part of the numeral nine in the first six prototypes is examined closely to look for the local differences to distinguish nine from its confusing numerals. The detailed implementation of rules to distinguish confusing pair of numerals four and nine will be introduced in a later section.

The prototypes should describe all possible differences among members of the same class. In fact, in real life, the description of a single member of a class may be quite different from one another. It is not possible to have prototypes that can represent all members of the same class. Hence, we construct prototypes that have common shapes and that can represent the majority written styles of numerals.



### **3.2 Prototype rules**

In the previous section, prototypes of each numeral were briefly described. In this section, an example is given to demonstrate how the prototype and its matching rules work to verify a numeral.

The input pattern of the verifier is a numeral eight with two connected circles and a short stroke connected to the right-top part of the top circle. The original image is shown in Figure 16a; the skeleton of the numeral is shown in Figure 16b.

Given that the classification result of this numeral is eight, the system first needs to check to which prototype of the numeral eight the input pattern belongs. A block diagram of the structure of prototypes of the numeral eight is displayed in Figure 15. The input pattern has two circles connected by one small vertical stroke, whose two termination points are in the two circles respectively; thus, the pattern belongs to prototype 2. Because of space limitation, the block diagram shows only the structure of prototype 2, the structures of other prototypes are not presented here.

Knowing the input pattern belongs to prototype 2, the next step is to check the relative positions of the two circles. As a numeral eight, one circle must be on top of the other. Otherwise, the prototype 2 rejects the pattern.

Then two evaluation functions: CheckUpperCircle and CheckLowerCircle are called to evaluate whether the two circles satisfy the requirements. Some samples of numeral eight have a small stroke connected to the top left or top right corner of the top circle in the forms of  and . Since it is a common writing style, the confidence value of the pattern will not decrease. However, confidence value will be deducted in the two evaluation functions for all other extra strokes according to their length and curvature.

If the confidence value is equal to or larger than the threshold, which is determined empirically, the input pattern is confirmed as a numeral eight. Otherwise, it is rejected by prototype 2. Though a pattern is rejected by prototype 2, it does not mean that it is rejected by the system. It will be passed to prototype 11. Prototype 11 is a special prototype of the numeral eight. When the system can not recognize the input pattern from its skeleton, the system will try to recognize it from the contour. Differing from other prototypes, which are constructed on the basis of the skeleton, prototype 11 is constructed by the features extracted from the contour. Detailed description of this prototype will be given in a later section. Input patterns that cannot satisfy the requirements of prototypes 1 to 10 are passed to prototype 11.

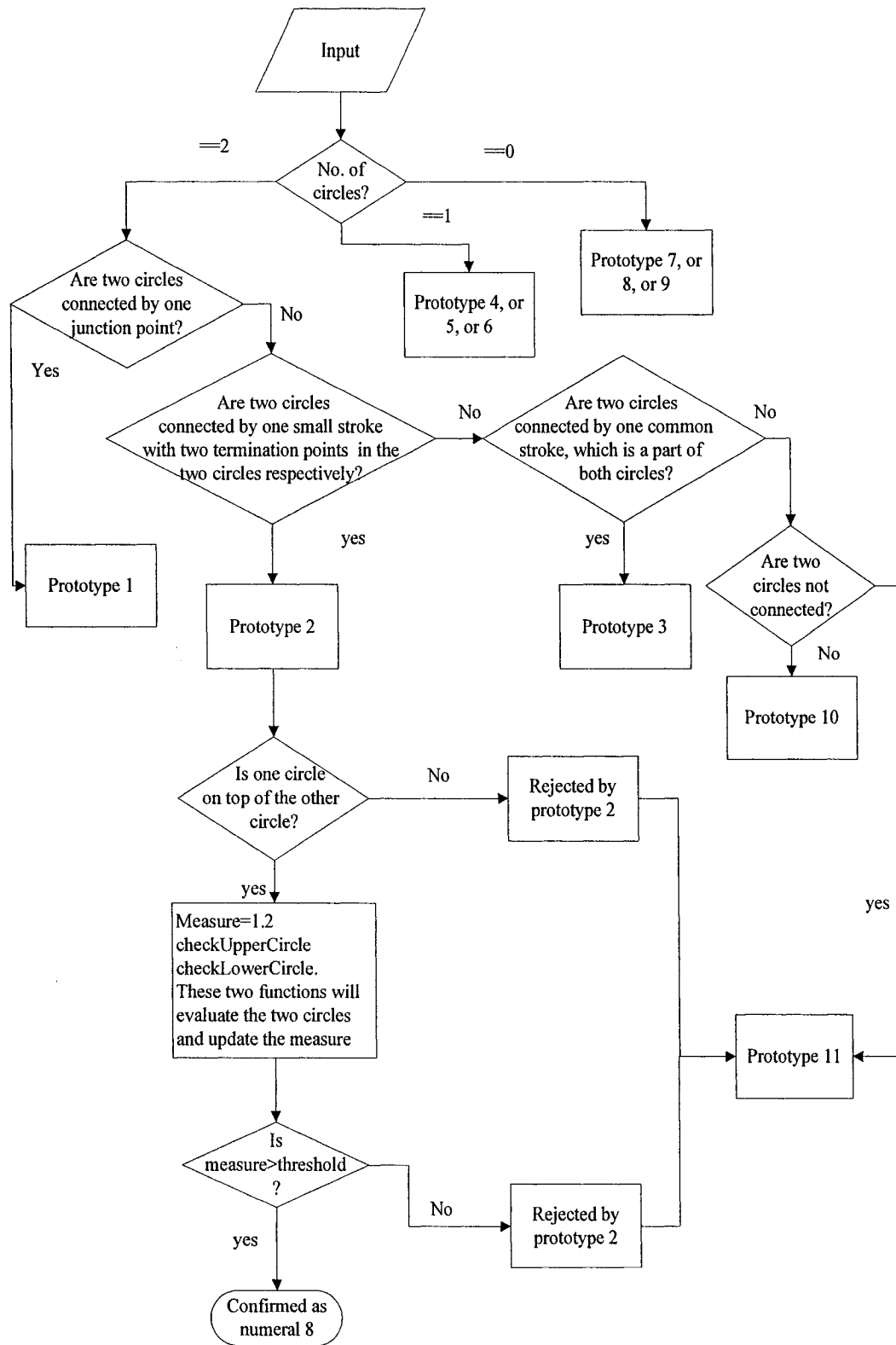


Figure 15: Prototypes of the numeral eight

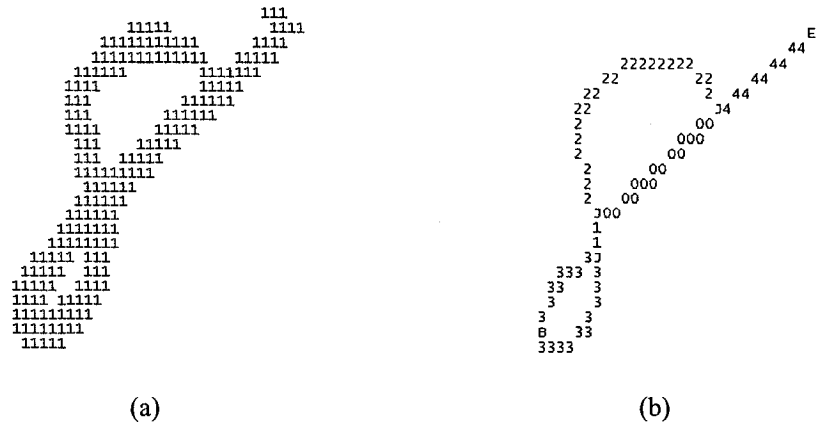


Figure 16: Numeral eight:  
(a) original image, (b) skeleton

### 3.3 Special prototypes

Most of prototypes are constructed by features extracted from the skeleton and relations among them as presented in section 2.3. However, there are special cases in which the system can not solely rely on this information. Some patterns have important features that can not be extracted by general feature extraction techniques; their essential features need to be extracted from the binary image or from the contour instead of the skeleton. Some patterns are easily confused with other numerals; therefore, more features need to be extracted to exhibit the local differences between confusing numerals.

#### 3.3.1 Special feature extraction technique

Prototypes are constructed on the basis of the properties of extracted features and the relations among extracted features. However, some features of numerals with special shapes may not be extracted using the general feature extraction techniques mentioned in section 2.3. In this case, special feature extraction techniques have to be used for some particular prototypes.

In this section, we will use the fifth prototype of the numeral six as an example to demonstrate special feature extraction techniques. Numerals satisfying the following conditions will be classified into the fifth prototype of the numeral six.

- There is one Long Branch. The Long Branch is defined when the height of the minimum rectangle boundary of the branch is larger than three quarters of the height of the minimum rectangle boundary of the image. The other criterion of the Long Branch is that the width of the minimum rectangle boundary of the branch is larger than three quarters of the width of the minimum rectangle boundary of the image. These conditions ensure that the Long Branch is the major part of the image and represents the basic structure of the image.
- There is a D curve in the right bottom part and a V curve at the bottom of the Long Branch. These two connected curves form a feature ‘almost-loop’ at the bottom of the numeral six.
- There is a line/curve connected to the feature ‘almost-loop’ in the upper part of the branch.

In this prototype, features such as the D curve and V curve cannot be extracted according to the definitions of the features. According to the definitions of curves of type D and V, the number of intersection points of the branch and the line connected the two end points (p1, p2 in Figure17) of the branch should be two. However, in this case, the number of intersection points of numerals in this prototype is three. Thus, the curve types D and V cannot be extracted. Therefore, it is impossible to extract sufficient information by simply applying the feature extraction technique in the Long Branch. We need to break down the



branch into several segments and use some particular segments to extract particular features.

We extract features of this prototype using the following algorithm:

Let the branch from p2 to p1 be a node list  $\{n_1, n_2, \dots, n_m\}$ , the coordinates of node  $n_i$  be  $(r_i, c_i)$ , p2 be  $n_1$ , p1 be  $n_m$ ,

1. Scan the branch start from p2 to p1 to find p3. P3 is the first node  $n_j$  that has  $r_j = \max_{1 \leq i \leq m}(r_i)$ . This step is shown in Figure 17b.
2. Scan the segment from p2 to p3 to detect p4 and p5. Let p3 be  $n_k$ , the segment from p2 to p3 be a node list of  $\{n_1, n_2, \dots, n_k\}$ . p4 is the first node  $n_j$  that has  $c_j = \min_{1 \leq i \leq k}(c_i)$ ; p5 is the first node  $n_j$  that has  $r_j = \min_{1 \leq i \leq k}(r_i)$ . This step is described in Figure 17c.
3. Scan the segment from p3 to p1 to find the point p6. Let p4 be  $n_a$ , p5 be  $n_b$ , p6 be  $n_c$ . p6 is the node which lies within segment p3 to p1 and satisfies  $r_c = r_{a+\text{int}[(a+b)/2]}$  and  $c_c < c_{a+\text{int}[(a+b)/2]}$  as shown in Figure 17d.
4. Use the segment from p4 to p3 as a branch to extract feature D curve and its depth and opening.
5. Use the segment from p5 to p6 as a branch to extract feature V curve and its depth and opening.
6. The segment p1 to p6 will be treated as the line/curve that is in the upper part of the numeral six and connected to the feature 'almost-loop', which is the segment from p2 to p6.

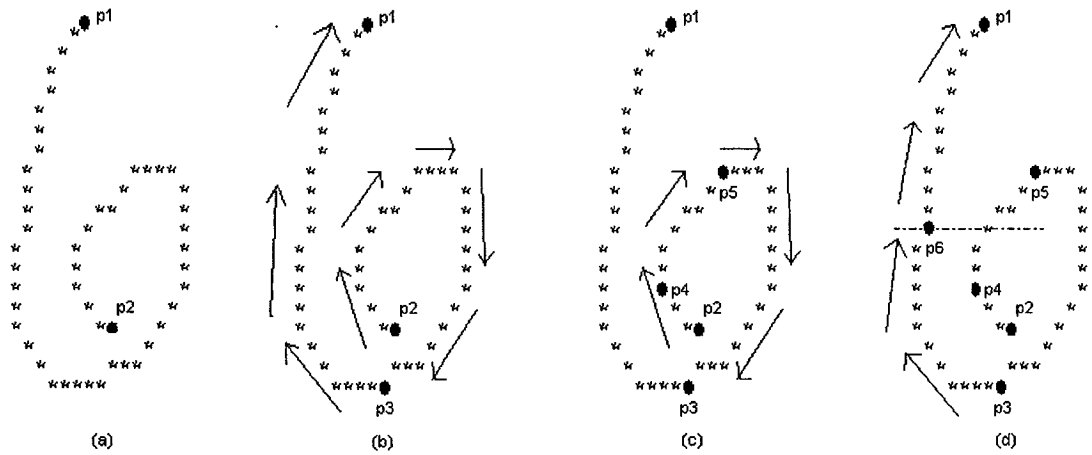


Figure 17: Procedure for extracting the features of the numeral six.

Since the length of the upper line/curve will be compared to the height of the ‘almost-loop’ to calculate the confidence of a given image. To determine the break point is an important task to make the confidence value more accurate. This point separates the branch into the upper line/curve and the lower ‘almost-loop’ . We use p6 as the break point in the proposed system. Instead of p6, we ever considered p7 and p8 as the break point. P7 is on the same row as p5, and p8 is on the same row as p4, as described in Figure 18. On one hand, using p7 will underestimate the length of the line/curve in some cases such as the image shown in Figure 17. This caused more rejection of the numeral six. Another case occurs when using p8 as the break point; it will overestimate the length of the line/curve. This will lead to misconfirmation of a zero as a numeral six. Thereby, it will lead to another problem: increment of the error rate. Intuitively, a point between p7 and p8 will be the optimal break point. After the experiment, we observed that using p6 as the break point yields the best result.

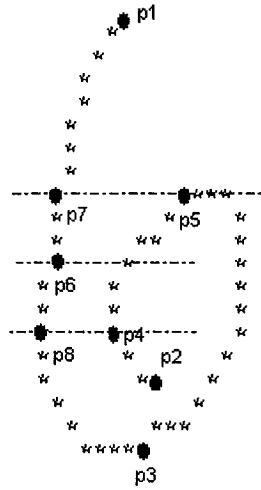


Figure 18: Possible break points of numeral six.

### 3.3.2 Prototypes based on features extracted from binary image/contour

The construction of the prototypes and matching rules are based on the features extracted from the skeleton of an input image. This is advantageous because a skeleton describes the structure of an image explicitly and clearly. On the other hand, due to writing instruments and limitations of skeleton extraction techniques, a skeleton has the disadvantage of losing information: often part of it and sometimes important details such as filled concavities or holes. In such cases, the original shape information kept on the binary image or the contour helps to identify the proper features. Therefore, extracting information from the binary image or the contour can compensate for the weakness of the skeleton.

#### 3.3.2.1 Numeral one

The prototype of numeral one is simple. It is not a tough job to determine whether an instance of the numeral one belongs to a particular prototype. However, for some

prototypes, it is dangerous to solely rely on the skeletons of images. For instance, the image shown in Figure 19 is a numeral nine, but its upper circle is filled due to the writing instrument or the preprocessing technique. In this case, the skeleton of the image is a positive straight line. If we rely only on rules of the first prototype of the numeral one, the numeral will be determined as a numeral one since it fits in the structure of the first prototype and obtains a fairly high confidence value. This limitation also applies to the second prototype. In order to avoid this type of mistake, more measurements based on the binary image should be added to the first and the second prototype of the numeral one to make the verification more reliable.

As illustrated in Figure 19, the image is divided evenly into four parts in the vertical direction. Then, we compare the stroke width of these four parts. The stroke width of each part should be similar if the input image is a numeral one. On contrast, the stroke width of part 1 or both parts 1 and 2 should be larger than that of parts 3 and 4 in the numeral nine; the stroke width of part 3 or both parts 3 and 4 should be larger than that of parts 1 and 2 in the numeral six. Based on the above measurements, we can distinguish a numeral one from a six and a nine with a filled circle.

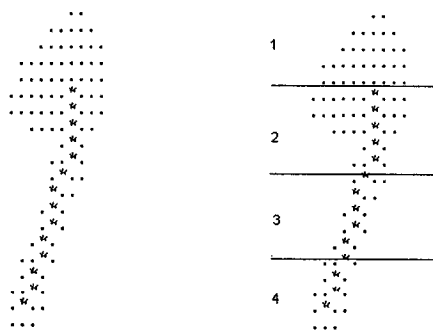


Figure 19: Filled numeral nine.

### 3.3.2.2 Numeral six

A circle at the bottom is an important characteristic of the standard shape of the numeral six. However, due to writing instruments and the filling techniques, sometimes the circle will be filled. In this case, the skeleton of a six will not have a circle at the bottom. Instead, the skeleton will have the shape of a hook, as shown in Figure 20. The notation ‘\*’ represents the skeleton of six. Numerals with this shape will be passed to the sixth prototype of numeral six for further investigation.

It is hard to determine the numeral based on the skeleton alone. The skeleton of Figure 20b can be either six or zero. Obviously, more features are needed to recognize this type of numeral six.

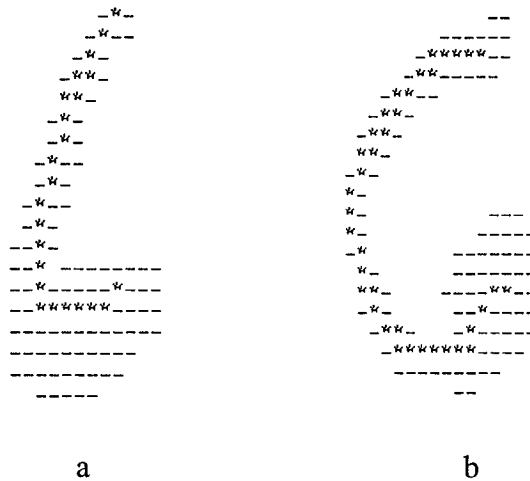


Figure 20: Filled six.

It is intuitive that a numeral six with filled circle is not symmetrical about the y-axis. The y-coordinate of the center of gravity should sit in the bottom part of the image. A statistic feature: the symmetry or skewness of the distribution of black pixels in rows [36] is helpful in discovering this characteristic.

The symmetry is measured by means of the index

$$\alpha_3 = \frac{\mu_3}{\sigma^3}$$

where  $\mu_r = E[(X - \mu)^r] = \sum_x (x - \mu)^r \cdot f(x)$

$$\mu = E(X) = \sum_x x \cdot f(x)$$

$$\sigma = \sqrt{\mu_2}$$

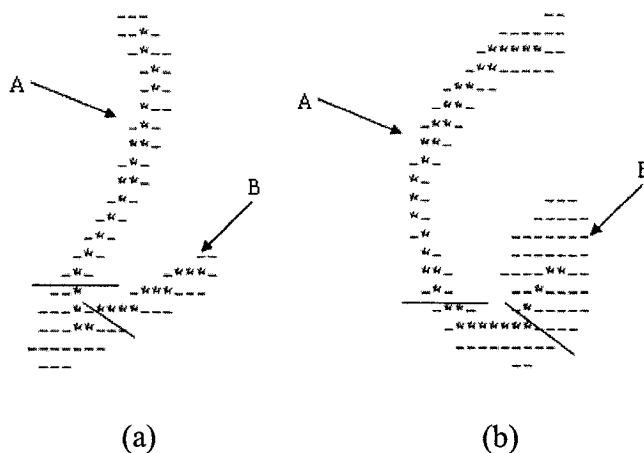
and  $x$  is the  $x$ th row in the image,  $f(x)$  is the number of black pixels in a row

The closer  $\alpha_3$  is to zero, the more symmetric the numeral is. If  $\alpha_3$  is positive, the y-coordinate of the center of gravity will be at the top of the numeral, i.e. positively skewed; on the contrary, if  $\alpha_3$  is negative, the y-coordinate of the center of gravity is in the bottom part of the numeral; i.e., it is negatively skewed.

We observed that it is also possible for numerals zero and one to have the skeleton of a shape of a hook. The average symmetry indices of zero and one in a hook shape are  $-2.61$  and  $-0.63$  respectively. The average symmetry index of six in a hook shape, which is  $-16.08$ , is much lower than those of zero and one. Therefore, when the skeleton of a numeral has the shape of hook, the symmetry index of the numeral will be checked. If its value is smaller than the threshold, which is determined empirically, the image satisfies one condition of being a numeral six.

However, some special cases of numeral two also have the 'hook' shape skeleton and a low symmetry index as numeral six, such as the numeral two presented in Figure 21a. Since the six in a hook shape has a filled circle, it is intuitive that the stroke width of the filled circle must be much larger than that of the upper line/curve. Let the upper line/curve be denoted as stroke A, the filled circle as stroke B. we observed that in

numeral six the stroke width of stroke B is larger than that of stroke A; on the other hand, in numeral two, the stroke width of stroke B is almost the same as stroke A. This observation shows that we can use the ratio of stroke width of stroke B over stroke A to distinguish the numeral two with a hook shape from the numeral six with a filled circle.



(a) (b)  
Figure 21: Comparison of two and six:  
Numeral two, (b) Numeral six

In order to obtain the stroke widths of strokes A and B, the first step we need to do is to detect which portion of an image belongs to stroke A and which belongs to stroke B. The algorithm to extract stroke A and stroke B and their stroke widths is introduced below (see Figure 22):

Let the end point p1 at the top of the skeleton be  $n_1$ , the other end point p2 is set to be  $n_m$ , the branch from p1 to p2 is a node list  $\{n_1, n_2, \dots, n_m\}$ , the coordinate of node  $n_i$  is  $(r_i, c_i)$ .

1. Extract initial stroke A. At the point  $n_{\text{int}(m/3)}$ , search its left and right contour points, which are in the same row as the point  $n_{\text{int}(m/3)}$ . The image is broken into two parts by the line connecting the left and right contour points as shown in Figure 22a. The part of image with skeleton from  $n_1$  to  $n_{\text{int}(m/3)}$  is the initial stroke A.

2. Calculate the stroke width of the initial stroke A.

$$\text{stroke width of strokeA} = \frac{\text{number of pixels in initial strokeA}}{\text{int}(m/3)},$$

where the denominator is the length of the skeleton of the initial stroke A.

3. Find stroke A. Stroke A may longer than the initial stroke A.

Initialize the *number of pixels in strokeA*:

$$\text{number of pixels in strokeA} = \text{number of pixels in initial stroke A}$$

Scan from  $n_{\text{int}(m/3)+1}$  to p2. For each node  $n_i$

- If  $r_i$  equal to  $r_{i-1}$ , skip  $n_i$ , proceed to  $n_{i+1}$ .
- Find the left and right contours of  $n_i$  in the horizontal direction. Let the left contour be  $c_{left}$  and right contour be  $c_{right}$ .
- If  $c_{right} - c_{left} + 1 \geq 2.5 \times \text{stroke width of strokeA}$ , stop scanning and go to step 4. The part of image with skeleton from  $n_1$  to  $n_{i-1}$  is stroke A.  $n_{i-1}$  is denoted as point p3.
- Stroke width of stroke A changes as the node changes.

$$\text{number of pixels in strokeA} += c_{right} - c_{left} + 1$$

$$\text{stroke width of strokeA} = \frac{\text{number of pixels in strokeA}}{i}$$

4. Find the break point to extract the portion of the filled circle. Scan from p2 to p3 to find p4, which is the first node in the lowest row. It is the point to break the filled circle from the image. However, there are cases that p4 happens to be the point p2. These cases are rare, but they do occur. In such case, p4 will be chosen between two candidates: the last node in the lowest row, and the point in the node



list located five positions preceding p2. The candidate that is closer to p2 will be the winner and set as p4. Some may wonder whether p4 will also be in the same location as p2. The answer is definitely no. Since if it is, the shape of the skeleton will not be a hoop, thus will not be considered in this criterion.

5. After we have determined the break point, we need to determine the direction to break the filled circle from the image. The breaking direction should be perpendicular to the direction of the filled circle (see Figure 22b). Let p4 be  $n_k$ . The slope of the node list  $\{n_k, n_{k+1}, \dots, n_{k+4}\}$  can roughly estimate the direction of the filled circle. The slope of the node list is calculated using Mean Square Error, and is denoted as *slope1*. The break point and the direction form the breaking line. The breaking line passes through the break point and has the slope of  $\frac{1}{slope1}$ . The filled circle portion of the image is stroke B.

6. Calculate the stroke width of stroke as follows:

$$stroke\ width\ of\ strokeB = \frac{number\ of\ pixels\ in\ strokeB}{m - k + 1},$$

where the denominator is the length of the skeleton of stroke B. The skeleton of stroke B detected by the thinning algorithm may be slightly shorter than the real skeleton of stroke B, which occurs in most thinning algorithms. When the stroke is long, such as stroke A, a small difference in skeleton length will not affect the accuracy of the value of stroke width much. However, stroke B is usually shorter. Using the length of original skeleton as denominator in the above formula will produce a stroke width of stroke B that is much larger than the accurate value. To solve this problem, we extend the skeleton to obtain a more accurate skeleton

length. The skeleton is extended from p2 to a contour point along the line that goes through p4 and with slope of  $slope1$ . This process is shown in Figure 22c.

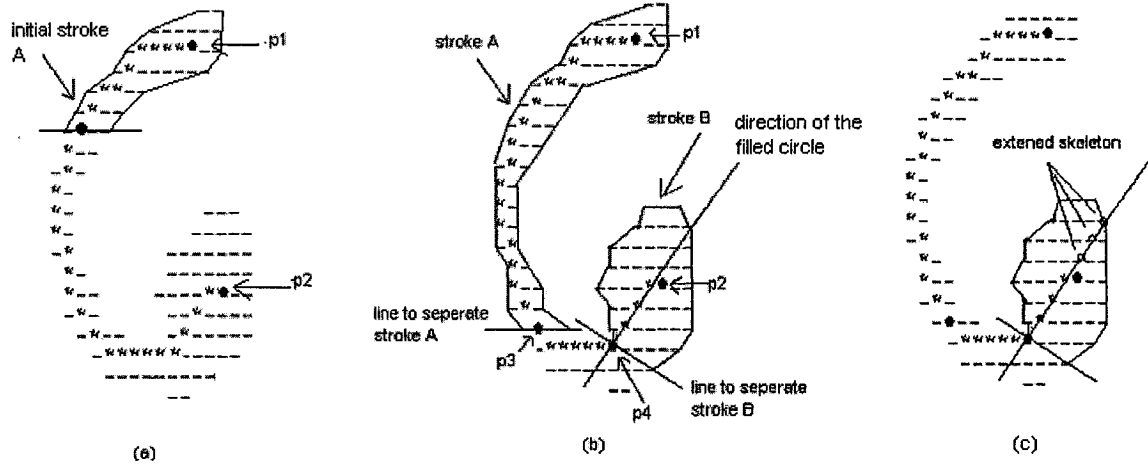


Figure 22: Numeral six.

By following the above-mentioned steps, we obtain the stroke widths of strokes A and B. If the ratio of stroke widths of B to A is equal to or larger than the threshold, the image has satisfied this criterion of numeral six. The threshold is set as 1.5, which is determined empirically.

In conclusion, if an image satisfies all of the conditions below, it belongs to prototype 6 of numeral six.

- The skeleton of the image has 'hook' shape
- Symmetry  $\leq -7.45$
- Stroke width of B / Stroke width of A  $\geq 1.5$

### 3.3.2.3 Numeral eight

Like the numeral six, in some cases of the numeral eight, the skeleton does not contain essential characteristics. For instance, if the bottom circle of eight is filled, the skeleton will look exactly like a numeral nine (Figure 23a). Moreover, the skeleton may miss some strokes of the image such as the case in Figure 23b where the skeleton does not contain the upper right stroke of eight.

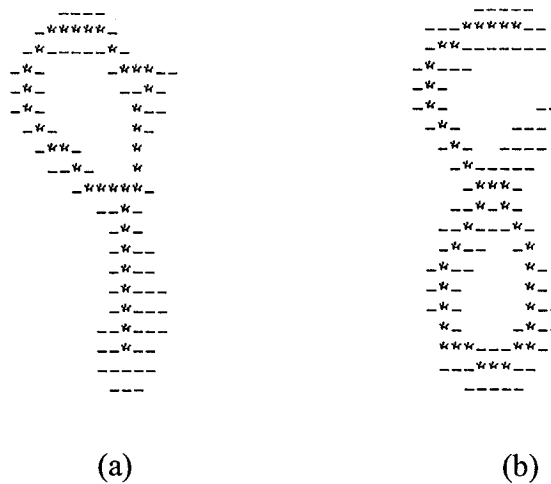


Figure 23: Incorrect skeletons of eight.

Without this useful information, it is difficult for the system to make a correct decision. On the other hand, for some images, although there are distortions, the outer contour of the image still contains all the necessary information to recognize a numeral. For the image in Figure 23a, even though the lower circle is filled, we can still discover that it is an eight. The features that help us to recognize that is the shape ‘)’ around the middle of the image. Thus, examining the direction change of the middle part of the outer contour of numeral eight is a great way to distinguish it from other numerals.

The first step in this method is to detect the outer contour of the image. Numeral eight should have the shape ‘)’ in the left middle part and ‘(’ in the right middle part. So what is

needed to be inspected is the left middle and right middle parts of the outer contour. Originally, we attempted to extract the outer contour using the Moor-Neighbor contour tracing and then break the contour into left and right parts. There is no problem doing this in images with two circles as the one shown in Figure 23a. This image is easy to break into left and right parts and easy to trace the direction changes the in the middle part of it. However, when dealing with images that have more complex outer contours such as the one in Figure 23b, it is more difficult to break it correctly into left and right contours. Moreover, the right contour does not have the shape of '3' and contains some other information we do not need. This increases the difficulty of extracting the expected shape from the contour.

To prevent making mistake in extracting the feature '3' ('), we proposed another method to detect the left and right contours instead of the contour tracing. This method is described in Figure24.

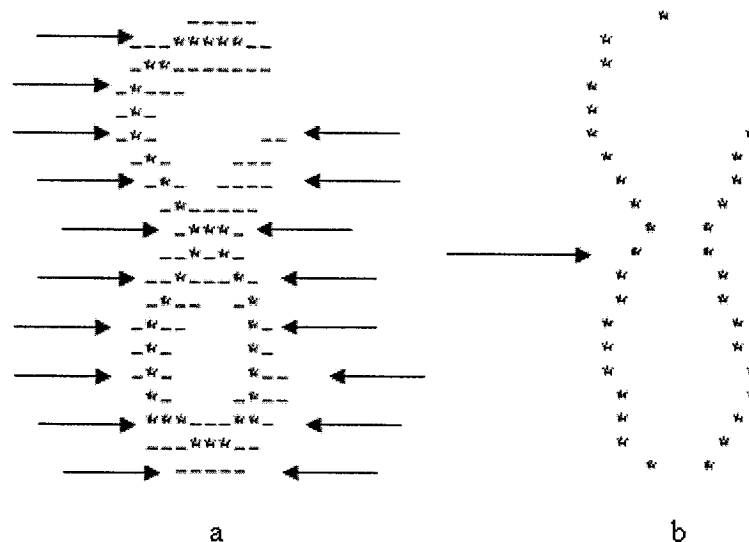


Figure 24: Left and right contour extractions of eight:  
 (a) original of eight, (b) left and right contour of eight

The image is scanned row by row from bottom to top. If the upper part of eight is a feature 'almost-loop', the scanning will stop at the opening of the feature 'almost-loop' as shown in right contour of Figure 24b. To extract the left contour, for each row, the image is scanned from left to right until it reaches a black pixel, which is marked as the left contour. On the other hand, to extract the right contour, the image is scanned from right to left until it reaches a black pixel, which is the right contour.

After the left and right contours have been extracted, the system will study the shape of these two contours. The system will detect whether these two contours have the shape similar to ')(' around the middle of the numeral. If this is true, the system will determine to which one of the following models (Figure 25) the numeral belongs. Afterwards, the corresponding confidence value will be assigned to the numeral.

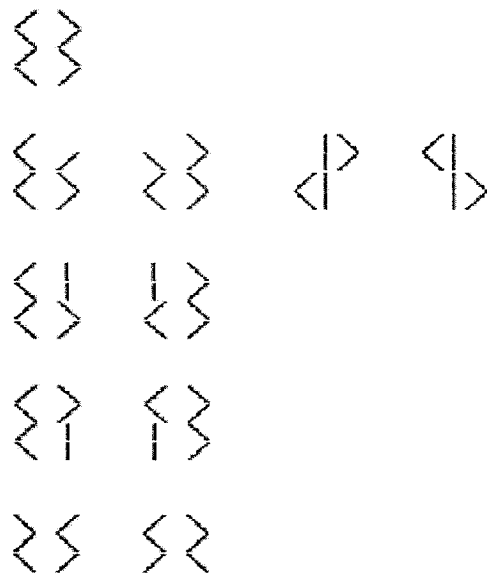


Figure 25: Contour models of numeral eight.

It was noticed that some samples of the numeral eight that are distorted; this is when the upper or lower circle of eight is filled. Thus, these distorted numerals will have skeletons of the shape of numeral six, nine, or even one while both circles are filled. These images

may be misclassified and mis-verified. However, the contour of these images should still have those characteristics mentioned above. Therefore, when the system verifies a numeral given that its value is six, nine or one by the classifier, besides from verifying the structure of its skeleton, it will also check whether the contour of the numeral belongs to one of the models of eight listed in Figure 25. If it belongs to one of these models, the numeral is probably an eight. Therefore, the system will not confirm the numeral. By going through this step, it can decrease the substitution rate of the system.

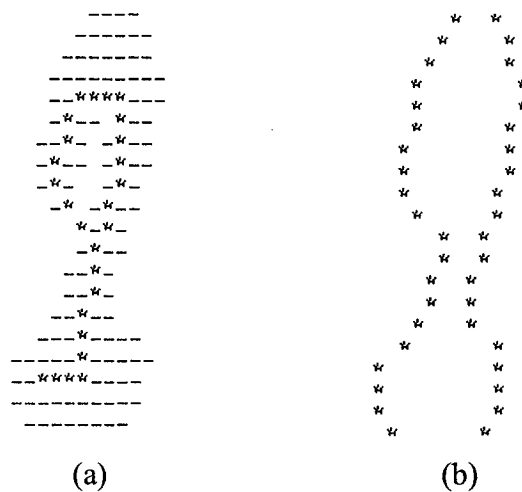


Figure 26 Mis-verified numeral two:  
 (a) original of two, (b) left and right contours of two

However, there exist some numerals two whose contour have the shape of  $\langle \rangle$ , as the one shown in Figure 26. This numeral two is written using an instrument with a thick tip, which makes the lower part of two look like a filled circle. Moreover, the end point of the upper two is touching the middle of the image, which forms a loop. Consequently, this image has a contour that perfectly fits the contour models of the numeral eight. It is difficult to differentiate it from the numeral eight with a filled circle. This image is confusing even to human beings. Since it is tough to distinguish this image from numeral eight and it is an extremely rare case, the system will confirm it as a numeral eight.

### 3.3.3 Confusing pairs of numerals

Some pairs of numeral have similar structures. They are the main reason of misclassification. Hence, the system needs to look closely at certain parts of an image to distinguish it from its confusing images. In the following sections, differentiation of three of the most confusing pairs of numerals will be introduced.

#### 3.3.3.1 Numerals four and nine

Numerals four and nine form the most confusing pair of numerals. They have similar shape and structure. For instance, for the first pair shown in Figure 27a, both of them have a circle at the top and a stroke at the bottom. Moreover, the positions of these features and the slopes of the strokes are similar. For the second pair shown in Figure 27b, both of them have a V type curve in the top part and a barely-vertical stroke in the bottom part. The positions of these features are similar too.

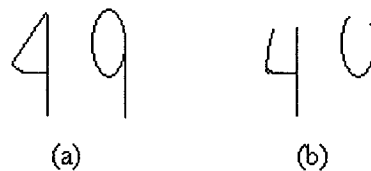


Figure 27: Two types of numerals four and nine.

There is no difference in the structure of these two numerals; the only difference between them is the curvature changes in some parts of the image. Thus, it is impossible to differentiate four and nine based on global features. We need to use the local features to distinguish them.

Usually, four and nine are written in two types of structures: one with a circle at the top as shown in Figure 27a; the other one with a V curve at the top as shown in Figure 27b.

Different features are extracted from an image corresponding to its structure.

The following features are extracted and examined to distinguish numerals four and nine with a circle in the top part and a stroke in the bottom part.

- The bend point at the right-top part of the circle (p1 in Figure 28a). If the bend point exists, the pattern is more likely to be numeral four.
- The bend point at the left-bottom part of the circle (p2 in Figure 28a). If the bend point exists, the pattern is more likely to be numeral four.
- The angle between the circle and the stroke (angle a in Figure 28). If the angle is smaller than the threshold, the pattern is more likely to be numeral nine.
- The vertical short tail connected to the circle. If the short tail exists, the longer the tail, the higher the confidence value of the numeral four.

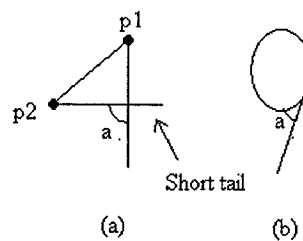


Figure 28: Features to distinguish numerals four and nine with a loop.

The following features are extracted and examined to distinguish numerals four and nine with a feature 'almost-loop' in the top part and a stroke in the bottom part. These features are shown in Figure 29.

- The bend point at the left-bottom part of the curve starts from p1 and ends at p2. If the bend point exists, the pattern is more likely to be numeral four.



- The straightness of the two segments of curve p1p2. Segments 1 and 2 are displayed in Figure 29a and 29b. If the bend point at the left-bottom part of the curve p1p2 exists, the bend point is the break point to separate segment 1 and segment 2. Otherwise, the breaking point is the point within the curve p1p2 and has the maximum distance with the line connecting p1 and p2. The straighter the segments are, the more probable the pattern is numeral four.
- The A curve at the top of the ‘almost-loop’ (Figure 29 c). If the A curve exists, the pattern is more likely to be numeral nine.
- The horizontal distance of the opening of the ‘almost-loop’ and the maximum horizontal distance of the ‘almost-loop’ (Figure 29d and 29e). Compare these two distances and get the ratio of horizontal distance of the opening of the ‘almost-loop’ to the maximum horizontal distance of the ‘almost-loop’. When the ratio is larger than a threshold, the larger the ratio is, the more likely the image is numeral four.
- The angle between the ‘almost-loop’ and the stroke. If the angle is smaller than the threshold, the pattern is more likely to be numeral nine.
- The vertical tail connected to the ‘almost-loop’. If the vertical tail exists, the longer the tail, the higher confidence value of numeral four.

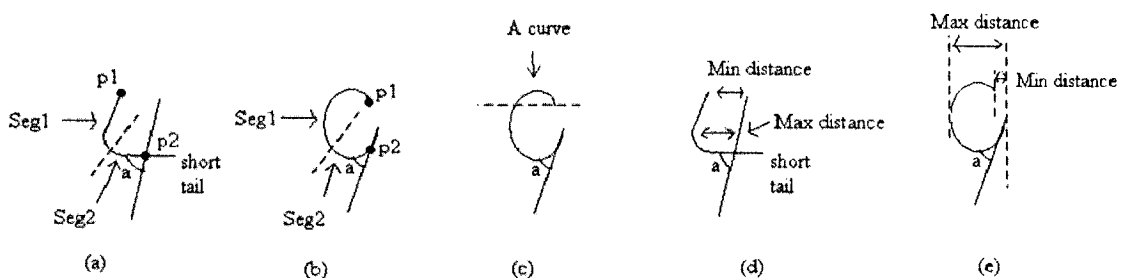


Figure 29: Features to distinguish numerals four and nine with the feature ‘almost-loop’

Based on the above criteria, numerals four and six can be differentiated.

### **3.3.3.2 Numerals zero and six**

The most common shape of the numeral zero is a circle while the numeral six consists of a circle and a stroke on top of the circle. In most cases numerals zero and six are not confusing. Unfortunately, not all people write numeral zero as a pure circle. There are some small extra strokes located outside the circle and that are connected to the circle. When the stroke is connected to the circle at the top, it can be easily confused with the numeral six. Similar, when the upper stroke of the numeral six is short, it is confused with the numeral zero. Thus, the position and the length of the upper stroke are critical features to distinguish zero from six.

If there is a circle at the bottom and a stroke connected to the circle at the top, such as images in prototypes 1 and 2 of numeral six, the following features will be examined.

- The position of the junction point, which is connecting the circle and the upper stroke. The circle will be divided into 9 even zones as shown in Figure 5. If the junction point is in zone 7, 4, 8, 9, favor is given to numeral six; otherwise, favor is given to numeral zero.
- The slope of the upper stroke. If the slope of the stroke is equal or larger than  $\tan(30^\circ)$  or equal and smaller than  $-\tan(30^\circ)$ , favor is given to numeral six; otherwise, favor is given to numeral zero.
- The length of upper stroke. The longer the length is, the more probable the image is numeral six.
- The vertical distance between two end points of the upper stroke. The larger the distance, the more likely image is a numeral six.

Originally, the system uses only the first three criteria to distinguish numerals two and six. Then we observed that two images with the same length of upper stroke represent different numerals. As shown in Figure 30, the lengths of the upper strokes are the same of those in the two images. Since the upper stroke of the left image is almost vertical while that of right image is flatter, the left image looks more likely to be a numeral six. In this case, the vertical distance of the upper stroke is more important. Thus, the last criterion has been added to include this characteristic.

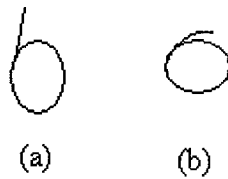


Figure 30: Two instances of numeral six

If numerals zero and six are written in another style: an ‘almost-loop’ at the bottom and an upper stroke connected to the ‘almost-loop’ as the image shown in Figure 18, the system first needs to break the image into an ‘almost-loop’ and an upper stroke. Unlike the written style in Figure 30, in which the circle and the upper stroke are easy to separate by the junction point, the breaking point of this style is not so intuitive to be detected. The system will use the  $p_6$  of Figure 18 as the breaking point. The reason for using this point as break point and the process of detecting this point have been described in the previous section.

The following features will be closely examined.

- The length of upper stroke. The longer the length is, the more probable the image is a numeral six.

- The vertical distance between two end points of the upper stroke. The larger the distance is, the more likely the image is to be a numeral six.
- The minimum distance between p6 and any point of the ‘almost-loop’. It is common that the distance is small for both numerals zero and six. It is also normal that the distance is large for numeral six, and it is rare that the distance is large for number zero. Thus, when the distance is larger than half of the maximum horizontal distance of any two points of the ‘almost-loop’, the larger the distance is, the bigger the chance for the image to be a numeral six.

### 3.3.3.3 Numerals four and six

In most written styles of numerals four and six, their structures are totally different. However, images in prototype 5 of numeral four have a structure similar to images in prototypes 3 and 4 of numeral six (see prototype description in section 3.2). All of these prototypes have one right circle, a C curve and lower stroke as shown in Figure 31. Thus we need to look at these features closely to detect distinct characteristics.

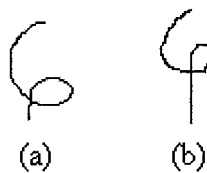


Figure 31: Confusing pairs of numerals six and four:  
(a) six, (b) four

First, the right circle is examined. If there are two bend points located at the top and right bottom of the circle respectively, there is a bigger chance that the image belongs to numeral four and, consequently, a smaller chance that the image is a numeral six.

Second, the C curve is inspected. If the curvature changes of this curve are smooth, favor is given to numeral six; on the other hand, if there is a bend point at the right bottom of

the curve, favor is given for it to be a numeral six. Moreover, for some samples of the numeral six, the C curve also contains an A curve. However, it is rare that the numeral four contains an A curve. Hence, when an A curve is detected, the probability that the image is a numeral six increases.

Finally, the lower stroke is checked. Two features of the stroke are measured: the slope of the stroke, and the length of the stroke. If the slope is within the range  $[-\tan(45^\circ), \tan(45^\circ)]$ , it is more likely to be a numeral six; otherwise, it has an equal probability to be a numeral four or six. The length of the stroke is another important feature to distinguish the numerals four and six. It is compared with the height of the minimum rectangle boundary of the C curve. The smaller the ratio of the length of the stroke to the height of the minimum boundary of the C curve is, the higher the score is assigned to numeral six; on the contrary, the larger the ratio is, the higher the score is assigned to numeral four.

Combining these three measurements, the system gets the scores of the numeral being a numeral four or six and makes a decision accordingly.

In this chapter, the most important component of the system: the verifier was described in detail. First, a brief description of the prototypes of each numeral was presented. Second, an example was given to demonstrate the process of verification. Third, some prototypes which need special techniques or features were provided to illustrate the complexity of the verifier. Finally, measurements of some confusing pair of numerals were defined.

## Chapter 4

### The Classifiers

The purpose of a verifier is to confirm or reject the outputs of the classifiers. The images have to pass through the classifiers first. Here, three classifiers, SVM proposed by Dong et al [37], LeNet5 proposed by Lecun et al [38], and MQDF proposed by Kimura et al [39] will be used to obtain the possible outputs of the images. The following sections will briefly introduce the three classifiers. Detailed descriptions of these classifiers can be found in [37, 38, and 39].

#### 4.1 SVM [37]

In the past several years, support vector machine (SVM) has played an increasingly important role in the pattern recognition system due to its excellent generalization performance in a wide variety of learning applications such as handwritten digit recognition and object recognition.

Given that training sample  $\{X_i, y_i\}$ ,  $y_i \in \{-1, 1\}$ ,  $X_i \in R^n$  where  $y_i$  is the class label and  $i=1, \dots, N$ .

The support vector machine first map the data to a Hilbert space  $H$ , which can be considered as a generalization of Euclidean space, using a mapping  $\Phi$ ,

$$\Phi: R^n \rightarrow H$$

The mapping  $\Phi$  depends on a kernel function  $K$  that satisfies the Mercer's conditions [40, 41] such that  $K(X_i, X_j) = \Phi(X_i) \bullet \Phi(X_j)$ . Then, in the space  $H$ , we need to find an optimal hyperplane by maximizing the margins and bounding the number of training errors. More specifically, we need to compute the sign of  $f(X)$

$$\begin{aligned} \text{where } f(X) &= W\Phi(X) + b \\ &= \sum_{i=1}^N \alpha_i y_i \Phi(X_i) \Phi(X) + b \\ &= \sum_{i=1}^N \alpha_i y_i K(X_i, X) + b \end{aligned}$$

Here,  $b$  is a threshold. The data  $X_i$  for which  $\alpha_i > 0$  are called support vector. We can avoid computing  $\Phi(X)$  explicitly and use the kernel function  $K$  instead.

Training an SVM is to find  $\alpha_i$ ,  $i = 1, \dots, N$ , which can be obtained by minimizing the following quadratic cost function:

$$L_D(\alpha) = \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j K(X_i, X_j)$$

subject to  $0 \leq \alpha_i \leq C \quad i=1 \dots N$

$$\sum_{i=1}^N \alpha_i y_i = 0$$

where  $C$  is a parameter decided by the user, the larger the value of  $C$ , the higher penalty allocated to the training errors.

However, the training of SVM is slow. Thus the author proposed a fast SVM training algorithm. The algorithm applies Keerthi et al 's [42] SMO to solve the optimization of a sub-problem in working set, which is a subset of training set, in combination with some

effective techniques such as kernel caching, “digest” strategy, shrinking strategies, and queue technique of selecting a new working set.

## 4.2 LeNet-5[38]

LetNet-5 is a convolutional neural network. It comprises seven layers, excluding the output layer, all of which contain trainable parameters/weights. These layers contain convolutional layers (labeled as  $C_i$ ), sub-sampling layers (labeled as  $S_i$ ), and fully-connected layers (labeled as  $F_i$ ). The architecture of the leNet-5 is shown in Figure 32.

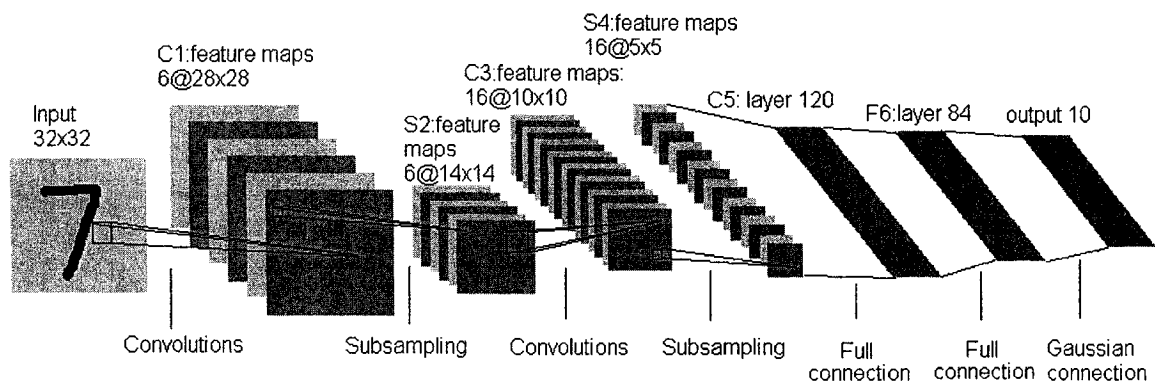


Figure 32: Architecture of LeNet-5

The layer  $C_1$  is a convolutional layer, which is composed of six feature maps of the size of  $28 \times 28$  with different weight vectors. A unit in a feature map has 25 inputs connected to a 5 by 5 area of the input, which is the receptive field of the unit. The receptive fields of neighboring units are overlapped. The layer  $C_1$  contains 156 trainable parameters and 122,304 connections.

The layer  $S_2$  is a sub-sampling layer with 6 feature maps of size  $14 \times 14$ . Each unit is connected to the  $2 \times 2$  area in the corresponding feature maps of  $C_1$ . The reason of sub-



sampling is to reduce the precision of some feature information. Precise information of certain features is harmful because the information is likely to vary in different samples of a numeral. For instance, we only need to know that there is an end point in the top left corner of a numeral seven; we do not need to know the exact coordination of this end point since the coordination varies in instances of numeral seven. The value of a unit is obtained by adding the four inputs from  $C_1$ , then multiplying a trainable coefficient, and adding to a trainable bias, finally passing the result through a sigmoid function. The layer  $S_2$  has 12 trainable parameter sand 5,880 connections.

The layer  $C_3$  is a convolutional layer with 16 feature maps of size  $10 \times 10$ . Each unit is connected to  $5 \times 5$  neighbors at the identical locations of  $S_2$ 's feature maps. The connection matrix of  $C_3$  features maps and  $S_2$  feature maps are shown in Table 3. The layer  $C_3$  has 1,516 trainable parameters and 156,000 connections.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	X				X	X	X			X	X	X	X		X	X
1	X	X				X	X	X			X	X	X	X		X
2	X	X	X				X	X	X			X		X	X	X
3		X	X	X			X	X	X	X			X		X	X
4			X	X	X			X	X	X	X		X	X		X
5				X	X	X			X	X	X	X		X	X	X

Table 3: Connection matrix of  $C_3$  feature maps and  $S_2$  feature maps.

The column indicates the feature map in  $C_3$ ; the row indicates the feature maps in  $S_2$ . The table indicates which feature maps in  $S_2$  are combined by a particular feature map of  $C_3$ .

The layer  $S_4$  is a sub-sampling layer with 16 feature maps of size  $5 \times 5$ . Each unit in each feature map is connected to the feature maps of the previous layer in a similar way as  $C_1$  and  $S_2$ . Layer  $S_4$  has 32 trainable parameters and 2,000 connections.

The layer  $C_5$  is a convolutional layer with 120 feature maps of size  $1 \times 1$ . Each unit is connected to  $5 \times 5$  neighbors at the identical locations of all  $S_4$ 's feature maps. It contains 48,120 trainable connections.

The layer  $F_6$  is a fully-connected layer with 84 units. It has 10,164 trainable parameters.

Up to layer F6, The state of unit  $i$ , denoted by  $x_i$ , is computed by a sigmoid squashing function:

$$x_i = f(a_i)$$

where  $a_i$  is the weighted sum of unit  $i$ . For each unit, the weighted sum is produced by adding a trainable bias to a dot product between the input vector and its weight vector.

The squashing function is defined as

$$f(a) = A \tanh(Sa)$$

where  $A$  is the amplitude of the function and  $S$  determines its slope at the origin. The author set  $A=1.7159$  and  $S=\frac{2}{3}$ .

Finally, the output layer is composed of Euclidean Radial Basis Function units (RBF), one for each class, with 84 inputs each. The RBF unit is computed using the following formula:

$$y_i = \sum_j (x_j - w_{ij})^2$$

The larger the RBF is, the less fitness is between the input pattern and the model of the class associated with the RBF.

The loss function of this neural network is:

$$E(W) = \frac{1}{P} \sum_{p=1}^P \{y_{D^p}(Z^p, W) + \log(e^{-j} + \sum_i e^{-y_i(Z^p, W)})\}$$

where  $y_{D_p}$  is the output of the  $D_p$ -th RBF unit,  $Z^p$  is the  $p$ -input pattern, and  $W$  represents the collection of adjustable parameters in the system.

The first term is an MSE criterion, which pushes down the penalty of the correct class; the second term plays a “competitive” role, it pulls up the penalties of the incorrect classes.

### 4.3 MQDF [39]

A quadratic discriminant function (QDF) can be written in the orthogonal expansion form:

$$g_0(x) = \sum_{i=1}^N \frac{1}{\lambda_i} \{\varphi_i'(x - \mu_M)\}^2 + \log \prod_{i=1}^n \lambda_i$$

with

$$\Sigma_M = \sum_{i=1}^N \lambda_i \varphi_i \varphi_i'$$

where the  $\mu_M$  and  $\Sigma_M$  denote the maximum likelihood estimates of the mean and the covariance, respectively, and  $\lambda_i$  ( $\lambda_i \geq \lambda_{i+1}$ ) and  $\varphi_i$  denote the  $i$ th eigenvalue and the eigenvector of the matrix  $\Sigma_M$ .

However, the QDF uses the maximum likelihood estimate of the covariance matrix, which is sensitive to the estimation error of the covariance matrix. Thus, the author proposed a Modified Quadratic Discriminant Function (MQDF) that employs a kind of a pseudo Bayesian estimate of the covariance matrix. MQDF is less sensitive to the error and requires less computation time and storage while achieving a better performance.

Performance of the discriminant function is improved by using the following pseudo-Bayesian estimate [43] of the covariance:

$$\Sigma_p = \Sigma_M + h^2 I$$

where  $I$  is the identity matrix and  $h^2$  is a constant.

From the above formula,

$$\varphi_i \Sigma_p \varphi_i^t = \varphi_i \Sigma_M \varphi_i^t + h^2 = \lambda_i + h^2$$

Thus the  $i$ th eigenvalue and eigenvector of  $\Sigma_p$  are equal to  $\lambda_i + h^2$  and  $\varphi_i$ , respectively. A

modified quadratic MQDF1 is given as

$$g_1(x) = \sum_{i=1}^N \frac{1}{\lambda_i + h^2} \{\varphi_i^t(x - \mu_M)\}^2 + \log \prod_{i=1}^n (\lambda_i + h^2)$$

MQDF1 is less sensitive to the error of the estimate of the covariance matrix, but it requires  $O(n^2)$  computation time and storage as QDF does. In order to decrease the computation time and storage, the author made another modification MQDF2 of the discriminant function. By substituting  $h^2$  for all of the eigenvalues  $\lambda_i$ ,  $i \geq k+1$  of  $\Sigma_M$  in QDF, we obtain MQDF2:

$$g_2(x) = \sum_{i=1}^k \left(\frac{1}{\lambda_i}\right) \{\varphi_i^t(x - \mu_M)\}^2 + \sum_{i=k+1}^n \left(\frac{1}{h^2}\right) \{\varphi_i^t(x - \mu_M)\}^2 + \log(h^{2(n-k)} \prod_{i=1}^n \lambda_i)$$

By using the equation

$$\sum_{i=1}^n \{\varphi_i^t(x - \mu_M)\}^2 = \|x - \mu_M\|^2$$

The MQDF2 is rewritten as

$$g_2(x) = \frac{1}{h^2} [\|x - \mu_M\|^2 - \sum_{i=1}^k \left(1 - \frac{h^2}{\lambda_i}\right) \{\varphi_i^t(x - \mu_M)\}^2] + \log(h^{2(n-k)} \prod_{i=1}^n \lambda_i)$$

It is obvious that the required computation time and storage of the MQDF2 are about  $k/n$  times those of QDF and MQDF1. Also, the MQDF2 as well as the MQDF1 are less sensitive to the estimation error of the covariance matrix if  $h^2$  and  $k$  are suitably chosen.

#### **4.4 Combination of the Classifiers**

Some commercial applications demand high reliability, which is extremely difficult to achieve by a single classifier. It is reasonable to consider combining several classifiers to achieve a higher accuracy. In recent years, a lot of combinations have been implemented using different strategies, such as majority vote approaches [44], statistical approaches [45], formulations based on Bayesian and Dempster-Shafer theories of evidence [46]. Using a combination of classifiers usually results in an improvement on reliability. This observation motivates us to combine the three classifiers described in the previous sections.

To improve efficiency, we first apply the Conditional Topology [47]. Under this topology, a primary classifier is first used. When it rejects a pattern due to its inability to give a classification, or when a low confidence value is given, other classifiers are deployed.

First, we need to choose one classifier as the primary classifier, which should have a clear boundary between the distribution of the correctly recognized data and the erroneous data. Once this is decided, we can set the boundary as the threshold of rejection.

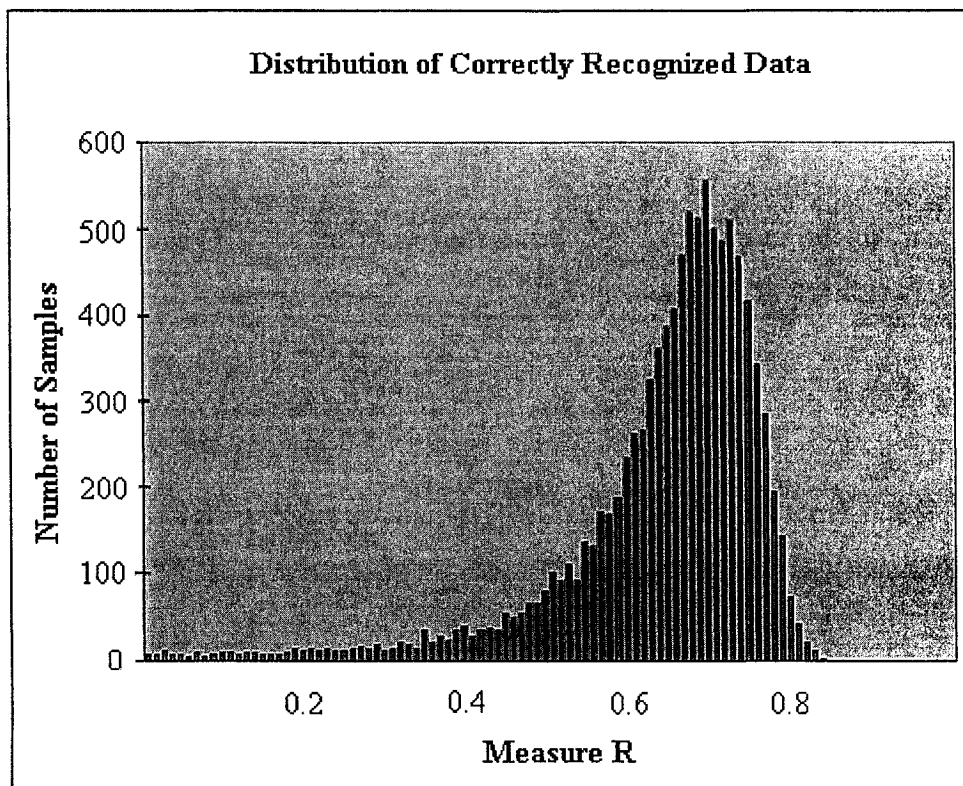
Each classifier will output the rank and the confidence value of each class (ten classes in the case of numerals recognition). We tried several measures to observe the distribution of data based on different measures. Only using the measure R, which is the ratio of the

distances of confidence values of rank 1 and rank 2 to the average distance of confidence value of rank  $i$  and rank  $i+1$  for  $1 \leq i \leq 9$ , shows a clearer boundary between the distribution of the correctly recognized data and the erroneous data.

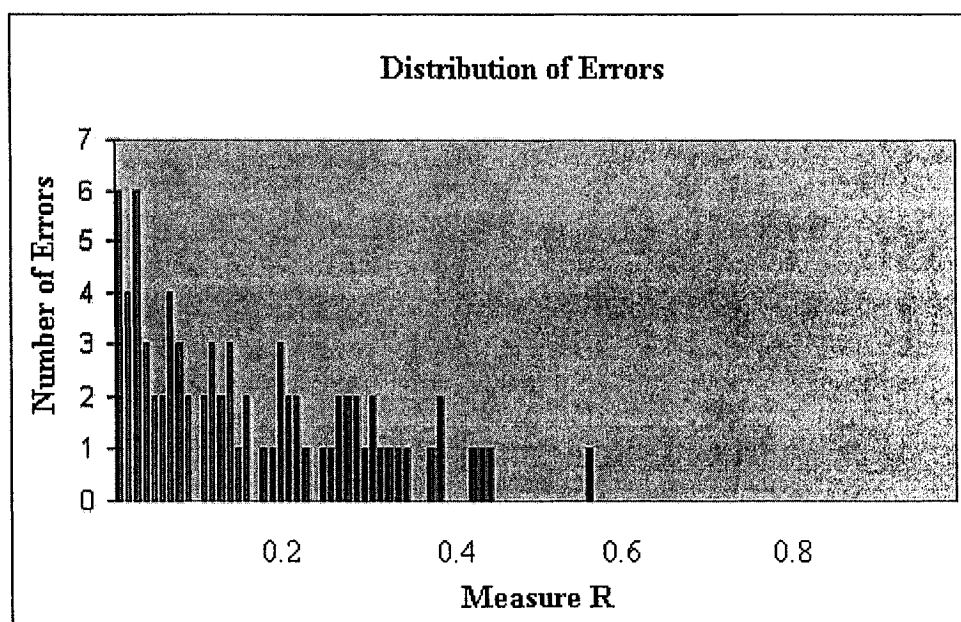
Given that the confidence value of rank  $i$  as  $C_i$ , measure R is defined as:

$$R = \frac{|C_1 - C_2|}{\frac{1}{9} \sum_{i=1}^9 |C_i - C_{i+1}|}$$

In our experiment, 10,000 numerals were randomly selected from the training set and passed SVM, LeNet5, and MQDF individually. The distribution of the correctly recognized data and erroneous data based on the measure R of the classifier SVM are shown in Figure 33a and 33b, respectively. As we can observe, a majority of correctly recognized data are distributed in the right part of the figure while the erroneous data are in the left part of the figure. This phenomenon does not occur in the classifiers MQDF and LeNet5 whose data distributions are displayed in Figure 34. In MQDF, most of correctly recognized data and errors are located in the left part of the figure. On the other hand, in LeNet5, both of correctly recognized data and errors are distributed all over the x-axis. We can not see a clear boundary between the correctly recognized data and the erroneous data in these last two classifiers. As a result of the observation of the distribution of data, SVM is more suitable to be the primary classifier than the others. We set the threshold of rejection as 5.64603, which is the maximum measure the erroneous data have. If the measure R of a pattern in the classifier SVM is smaller or equal to the threshold, the secondary classifier is deployed.



(a)



(b)

Figure 33: Data distribution of classifier SVM

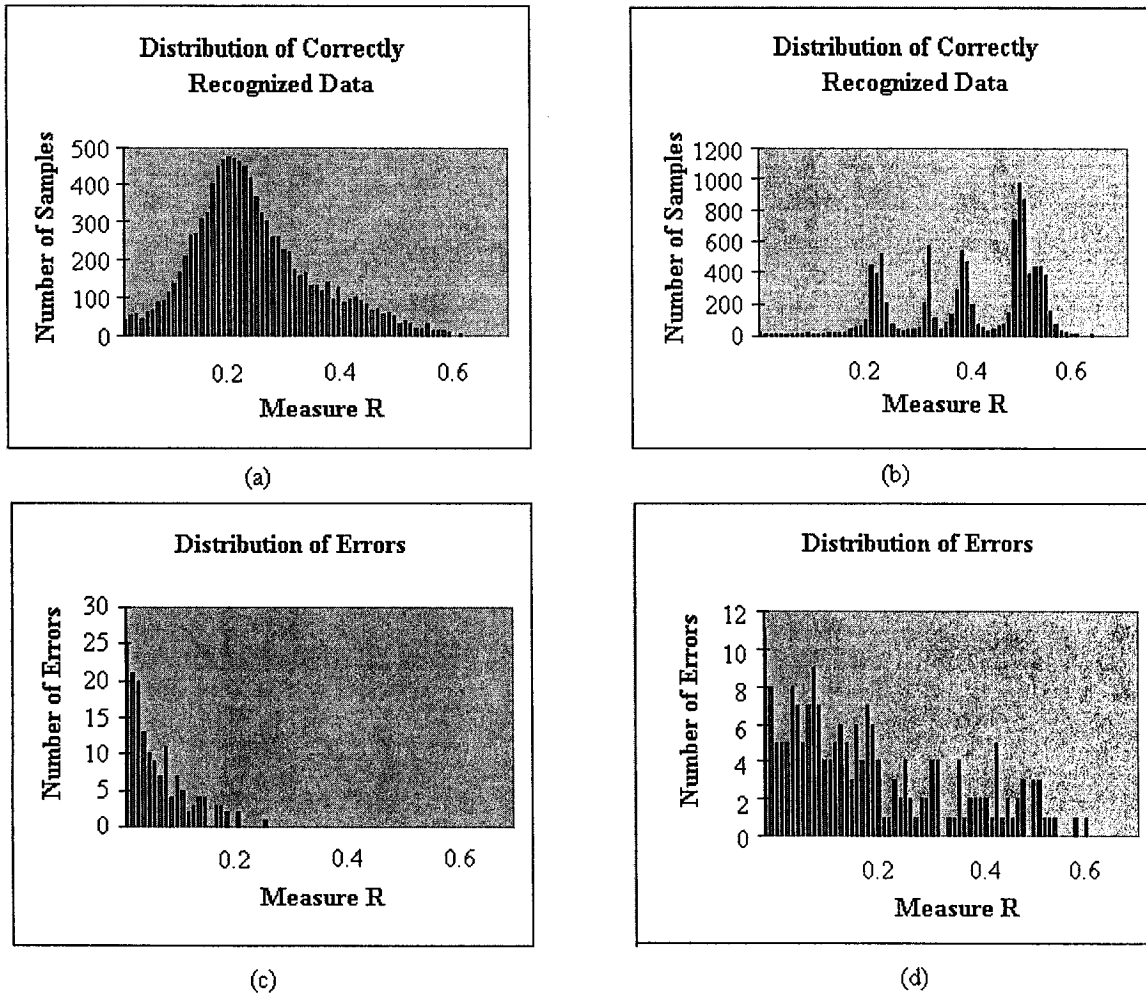


Figure 34: Data distributions of MQDF and LeNet5:  
 (a) and (c) distribution of correctly recognized and errors of classifier MQDF, respectively; (b) and (d) distribution of correctly recognized and errors of classifier LeNet5, respectively.

In the proposed system, the secondary classifier is a parallel combination of the three classifiers. The three classifiers operate in parallel to produce classifications of a pattern, after which the decisions are combined to yield a final decision.

The system uses the ranked list of the ten numerals of each classifier as the input information of the parallel combination. The system assigned weights according to the overall performance of each classifier on the training set.



Let  $C$  be a  $10 \times 10$  matrix in which the entry  $C_{ij}^{(k)}$  denotes the number of patterns with actual class  $i$  that is assigned to class  $j$  by the classifier  $k$ .  $N_i$  denotes the number of patterns whose actual class is  $i$ . The conditional probability that the classifier  $k$  assigns a pattern to class  $j$ , given that the pattern  $x$  actually belongs to class  $i$ , is estimated as

$$P(\text{classifier}_k(x) = j | x \in C_i) = \frac{C_{ij}^{(k)}}{N_i}$$

Let  $Conf_{ik}$  be the confidence value of a pattern being class  $i$  in classifier  $k$ , it is defined as

$$Conf_{ik} = \begin{cases} 2(10 - R_{ik}) & \text{if } R_{ik} = 1 \\ 10 - R_{ik} & \text{if } R_{ik} = 2, 3, \dots, 10 \end{cases}$$

where  $R_{ik}$  is the rank of class  $i$  in classifier  $k$ . Since the lower value of the rank, the higher probability the pattern belongs to class  $i$ , we transform the rank to confidence value by using  $(10 - R_{ik})$  when the rank is not the top one. Because the top one has a much higher probability to be the true label than others, the confidence value of the top one is doubled by the formula  $2(10 - R_{ik})$ .

Let  $M_i$  be the confidence value of class  $i$  of the parallel combination.  $M_i$  is calculated using the following formula

$$M_i = \sum_{k=1}^3 Conf_{ik} \times P(\text{classifier}_k(x) = i | x \in C_i)$$

where  $Conf_{ik}$  represents the confidence value of a pattern being class  $i$  in classifier  $k$ , while  $P(\text{classifier}_k(x) = i | x \in C_i)$  represents the weight of the classifier  $k$ , consequently,  $M_i$  represents the confidence value of class  $i$  in the parallel combination.

The ten classes are re-ranked according to the value of  $M_i$ . The larger the  $M_i$ , the higher rank of class  $i$  is. The class with maximum  $M_i$  is ranked as number one; the class with second maximum  $M_i$  is ranked as number two; and so on.

In conclusion, this chapter first briefly introduced the three classifiers: SVM, LeNet5, and MQDF. For the purpose to achieve a better recognition performance, the three classifiers were combined. First, the Conditional Topology was applied: the SVM was chosen as the primary classifier. When the primary classifier failed to give a classification, the second topology: parallel combination of the three classifiers was deployed to yield a final decision.

# Chapter 5

## Experiment

### 5.1 Verification on the output of combination of classifiers

#### 5.1.1 Database

The first set of experiment was conducted on MNIST database. MNIST database is a widely known handwritten digit recognition benchmark. It contains binary images of handwritten digits constructed from the NIST's Special Database 3 and Special Database 1. SD-3 is much clearer and easier to recognize than SD-1 because SD-3 was collected from Census Bureau employees, while SD-1 was collected from high school students. The training set contains 60,000 images, half of them from SD-1 and the other half from SD-3. The test set has 10,000 test images: 5,000 from SD-1 and 5,000 from SD-3. Images in the training set and the test set are generated by different writers. The images were first size normalized to fit into a box of  $20 \times 20$  pixels, and then centered in a  $28 \times 28$  image [38].

#### 5.1.2 Design of experiment

We used the general-purpose recognizer (GPR), which combined three individual classifiers, to compare the original results of the classifiers and the final result after adding the verification module.

As mentioned by Zhou et al [48], the perfect status of a verifier is to confirm all the true decisions and negate all the wrong decisions of a recognizer. However, in real-life applications, this status cannot be achieved. It is a dilemma to keep the balance between the false positive and false negative. False positive is the confirmation of a wrong decision while false negative is the negation of a true decision. It is usually impossible to make both of them small at the same time [49].

The power of verification can be defined as  $(1 - \text{false positive})$ . The higher the verification power, the higher the possibility of false negative is. How to control the verification power is a problem of how to balance the reliability and the recognition rate. It is a problem related issue. In our problem, as stated in the introduction, the main goal of the system is to achieve a high reliability. We want a high verification power to satisfy the requirements needed to confirm an image.

The proposed verifier is prototype-based. The task of building a set of prototypes should describe all possible differences among members of the same class. In fact, in real life, the description of single members of a class may be quite different from one another. It is not possible to have prototypes that can represent all members of the same class. Images that do not match any prototypes that were previously built will be rejected by the verifier. Moreover, since the confirmation conditions are restrictive for the reason mentioned above, a lot of images will be rejected. These two facts lead to a low recognition rate.

While achieving a high reliability, we want to maintain a reasonable high recognition rate. A way to solve this problem is to control the inflow of the verifier. The verifier is plugged into a combination of classifiers, which is more reliable than individual classifiers. Also, as we observed, a majority of outputs of classifiers are reliable. Only

small portions of the outputs with lower confidence values are erroneous. Therefore, it is not necessary to pass the reliable data into the verifier. Hence, the inflow to the verifier will be controlled; only patterns that cannot satisfy any of the following conditions will be passed to the verifier.

1. Measure R (defined in section 4.4) in classifier SVM  $\geq$  primary classifier rejection threshold (5.64603), which is determined empirically.
2. Top1 candidate of SVM = top1 or top2 candidate of parallel combination of classifiers.
3. Top1 candidate of SVM = top1 or top2 candidate of classifier MQDF.

These conditions ensure only patterns with unreliable classification results are passed to the verifier.

### 5.1.3 Experimental result

In Table 4, we list the performance of the general-purpose recognizer (GPR), which combines three individual classifiers, and its performance of using the verifier. The Recognition rate, substitution rate, rejection rate and reliability are defined by

$$\text{Recognition rate} = \frac{\text{Number of correctly recognized samples}}{\text{Total number of test samples}} \times 100\%$$

$$\text{Substitution rate} = \frac{\text{Number of incorrectly recognized samples}}{\text{Total number of test samples}} \times 100\%$$

$$\text{Rejection rate} = \frac{\text{Number of rejected samples}}{\text{Total number of test samples}} \times 100\%$$

$$\text{Reliability} = \frac{\text{Recognition rate}}{100\% - \text{Rejection rate}}$$

As seen from the Table 4, the reliability of the system has been improved after applying the verifier. By lowering the rejection threshold, the substitution rate can be reduced at

the expense of an increased rejection. Since a minimized error risk is a priority, the optimal rejection is set to be 3.42%, with which the system achieves a high reliability with a reasonable recognition rate. When the rejection is higher than 3.42%, the small decrement of the substitution rate is at the expense of large decrement of recognition rate. With the rejection rate of 3.42%, the general-purpose recognizer along with the verifier yields a reliability rate of 99.92% and a recognition rate of 96.50% on MNIST Database. The substitution rate is dropped dramatically from 0.94% to 0.08%. However, the recognition rate also drops about 2.5%. On the other hand, the reliability increased from 99.06% to 99.92%. Among the 8 errors, 3 are due to the classifiers, 5 are due to the verifier. The proposed system achieves high reliability coupled with a reasonable recognition rate. It satisfied the requirement of some applications with a high demand of reliability such as bank cheque processing and other financial transactions.

From the literature [48, 55], we notice that only a few verifiers have been built for unconstrained handwritten numerals. Since the verifiers in the literature are based on classifiers different from the classifiers used in the proposed system, and the performances of the verifiers are affected largely by the classifiers. This made it hard to compare our performance with others.

Method	Recognition	Rejection	Substitution	Reliability
GPR	99.06%	0.0%	0.94%	99.06%
Verifier	99.00%	0.19%	0.81%	99.19%
	98.50%	1.05%	0.45%	99.56%
	98.00%	1.75%	0.25%	99.75%
	97.50%	2.34%	0.16%	99.84%
	97.00%	2.86%	0.14%	99.86%
	<b>96.50%</b>	<b>3.42%</b>	<b>0.08%</b>	<b>99.92%</b>
	90.00%	9.93%	0.07%	99.92%
	80.00%	19.95%	0.05%	99.94%

Table 4: Test results with and without verifier.

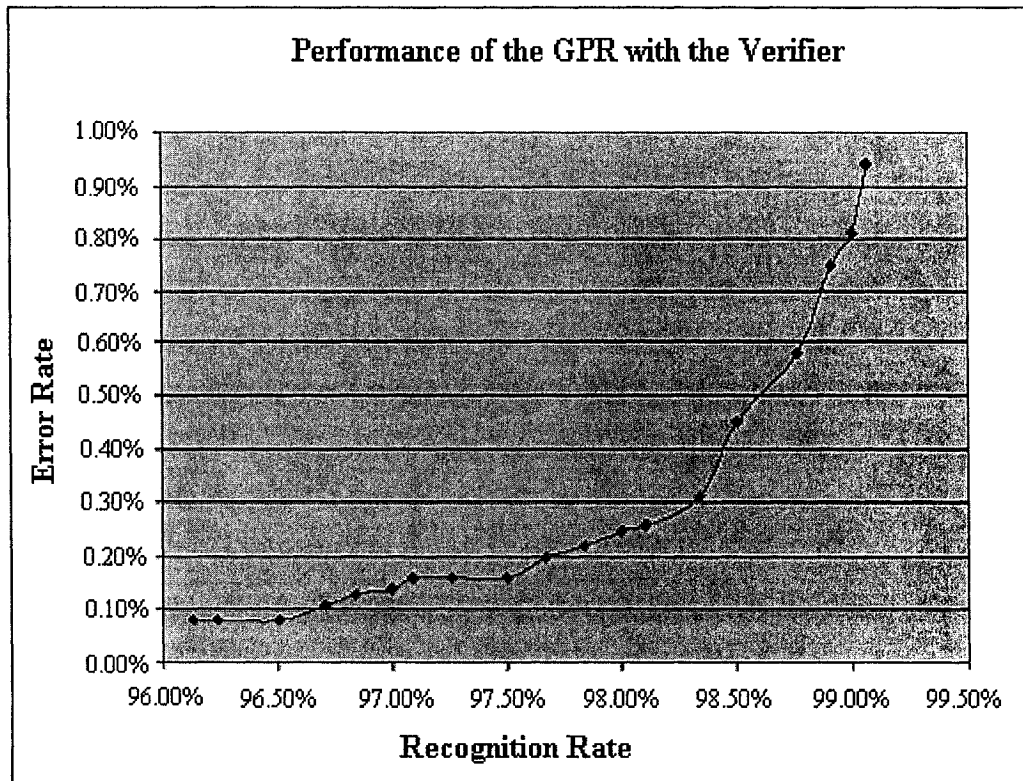


Figure 35: Performance of the GPR with the Verifier

Figure 36 gives some samples misclassified by GPR but rejected by the verifiers. Some ambiguous images, which are confusing even to humans, are rejected by the verifier. The result shows the ability of the verifier in detecting erroneous decisions.

4->9	6->0	3->2	9->7	7->3	4->9	8->9	6->4
448	2119	2928	6572	4239	4383	6556	3521

Figure 36: Images correctly rejected by the verifier.

The first digit on the top of each image indicates the true label. The second digit on the top specifies the estimated digit label. The number on the bottom of each image is the sequence number of the image in the database.

Figure 37 displays the eight errors of the GPR with the verifier. The first three images are misclassified by GPR but do not pass to the verifier since they have a high confidence value. The last six images are misclassified by GPR and misinterpreted by the verifier. As

we can see, some of them are confusing even to human beings because they look more like the estimated digit than the true label of the database. An example of this kind of patten is the image with sequence number 9506. Thus, it is reasonable that the verifier misinterpreted these data.



Figure 37: Errors of the GPR with verifier.

The first digit on the top of each image indicates the true label. The second digit on the top specifies the estimated digit label. The number on the bottom of each image is the sequence number of the image in the database.

## 5.2 Verification of the collection of Misclassified data

In order to know the pure strength of the verifier and analyze errors made by the verifier, we test the verifier with a collection of misclassified images produced by different classifiers.

The following are the collection of misclassified images produced by a variety of classifiers.

### VSVM<sup>b</sup>

VSVM<sup>b</sup> is a virtual support vector machine with multi-scale virtual patterns proposed by Dong[50]. The classifier is tested with the MNIST database. The 38 errors produced by VSVM<sup>b</sup> are displayed in Appendix A.

We used these 38 images and their recognized label as input to test the verifier. The verifier rejected 36 of these misrecognized images and incorrectly confirmed 2 of them. The mis-verified images are shown in Figure 38.





7→2	4→9
	
9506	9793

Figure 38: Mis-verified images by VSVM<sup>b</sup>.

The first digit on the top of each image indicates the true label. The second digit on the top specifies the estimated digit label. The number on the bottom of each image is the sequence number of the image in the database.

## VSV2

VSV2 is a virtual support vector machine with 2 pixels translation proposed by Decoste et al [51]. The VSV2 used 3×3 box jitter combined with four additional translations by 2 pixels. Detailed description of the classifier can be found in the paper of Decoste [51].

The 56 errors of the MNIST test data are shown in Appendix A.

We tested the verifier with these 56 errors. It correctly rejected 50 of them but mistakenly confirmed 6 of them. These mis-confirmed images are displayed in Figure 39.







3→7	9→5	5→0	9→4	9→4	4→9
					
1682	1710	3559	3870	3986	9793

Figure 39: Mis-verified images by VSV2.

The first digit on the top of each image indicates the true label. The second digit on the top specifies the estimated digit label. The number on the bottom of each image is the sequence number of the image in the database.

## LeNet5

LeNet5 is a convolutional neural network described in the paper of Lecun et al [38]. This version of LeNet5 is trained with an enlarged training set of MNIST which was composed of the 60,000 original images plus 540,000 instances of distorted patterns. Distorted patterns are made using different types of transformation such as translation,

scaling, and horizontal shearing. LeNet5 made 82 errors, which are presented in Appendix A.

These 82 errors were passed as input to the proposed system. Among these, 72 were successfully rejected by the verifier, 10 were inaccurately confirmed by the verifier.

Figure 40 shows the mis-confirmed images.











3→5	3→7	9→4	2→0	9→4	9→4	9→4	2→8	7→2	4→9
									
450	1682	2294	2463	3870	4406	4426	8095	9016	9793

Figure 40: Mis-verified images by LeNet5.

The first digit on the top of each image indicates the true label. The second digit on the top specifies the estimated digit label. The number on the bottom of each image is the sequence number of the image in the database.

### Product of Experts Network

The Hierarchical Product of Experts is proposed by Hinton et al [52]. The classifier is also trained and tested using the MNIST database. It produced 168 errors, which are displayed in Appendix A.

The verifier correctly rejected 153 erroneous images but failed to reject 15 erroneous images. These 15 images are presented in Figure 41.
















9→4	3→7	9→4	2→0	3→5	8→9	3→7	5→0	9→5	7→0
									
360	1682	2294	2463	2953	3290	3476	3559	4285	5888
8→9	8→9	2→8	4→1	7→2					
									
6556	6756	8095	8096	9016					

Figure 41: Mis-verified images by POE.

The first digit on the top of each image indicates the true label. The second digit on the top specifies the estimated digit label. The number on the bottom of each image is the sequence number of the image in the database.

## VSVM

The virtual support vector machine is presented in [50]. Support vectors of different classes are first merged. Then, virtual patterns are generated by one pixel translation in the four directions: up, down, left, and right. RBF kernel is used in the experiment.

Two databases CENPARMI and USPS are used in the experiment.

The widely used CENPARMI database contains a training set of 4,000 binary image samples, and a test set of 2,000 binary image samples. The foreground pixel value is set to be 1, while the background pixel value is set to be 0. These data are compiled from US zip codes of dead envelopes by researchers in CENPARMI. The images are of various sizes.

VSVM produced 26 errors on the CENPARMI data, which is presented in Appendix A. Among these 26 errors, the verifier rejected 24 of them and incorrectly confirmed 2 of them. Figure 42 shows the two erroneous images.

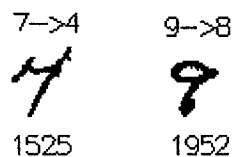


Figure 42: Mis-verified images by VSVM.

The first digit on the top of each image indicates the true label. The second digit on the top specifies the estimated digit label. The number on the bottom of each image is the sequence number of the image in the database.

The US Postal Service (USPS) database contains 9289 handwritten digits, 7,291 for the training set and 2,007 for the test set. They were collected from envelopes in Buffalo. Each digit is a 16×16 image. Images are in gray level and scaled and translated to fall within the range 1 to -1. It is known that USPS test set is rather difficult. The human

error rate is 2.5% [53]. It has also been shown that there are many confusing data and obviously mislabeled data in the test set [54].

Testing with USPS test set, VSVM misrecognized 47 images, which are listed in Appendix A. The verifier rejected 40 of them and mistakenly confirmed 7 of them as displayed in Figure 43.

8→0	3→5	4→1	5→0	6→8	5→8	1→7
8	<b>3</b>	<b>1</b>	<b>0</b>	<b>8</b>	<b>8</b>	<b>1</b>
199	792	971	994	1469	1657	1815

Figure 43: Mis-verified images by VSVM.

The first digit on the top of each image indicates the true label. The second digit on the top specifies the estimated digit label. The number on the bottom of each image is the sequence number of the image in the database. The digits enclosed in a square are mislabeled.

## MLP

The ensemble of multi-layer perceptron (MLP) is implemented by Luiz [55]. MLPs are trained with the gradient descent applied to a sum-of-squares error function. The transfer function employed is the familiar sigmoid function. This experiment is carried out on the NIST SD 19 database, which is organized in eight series, denoted by  $hsf_{\{0,1,2,3,4,6,7,8\}}$ . The training set of the recognizer is extracted from  $hsf_{\{0,1,2,3\}}$ ; the test set is extracted from  $hsf_7$ .

We obtained 119 images that were misrecognized by the recognizer, the verifier correctly rejected 93 of them while mis-verified 26 of them.

1→2	1→2	1→2	1→2	1→2	1→7	1→7	1→8	2→8	2→8
3375	4564	5137	588	593	5025	5785	2813	2686	2919
2→8	3→0	3→5	3→8	3→8	3→9	5→3	5→6	6→0	6→0
5218	840	1567	2890	3599	5196	2104	4206	158	3978
7→2	7→2	8→3	8→4	9→4	9→4				
3324	4123	1759	2396	1302	4663				

Figure 44: Mis-verified images by MLP.

The first digit on the top of each image indicates the true label. The second digit on the top specifies the estimated digit label. The number on the bottom of each image is the sequence number of the image in the database. The digits enclosed in a square are mislabeled.

## 5.3 Error Analysis

For a better understand of the reasons of misrecognition and mis-verification, we analyze the collection of misrecognized data and mis-verified data.

### 5.3.1 Misrecognized data

In the collection of misrecognized data, some of them are clear and in a typical shape of a numeral, while some of them are ambiguous. After observation, we divided the collection of misrecognized data into three categories from a human's perspective. Category 1 is for the images that are easily recognized by humans without any ambiguity. Images in this category are clear, unambiguous, and in a common shape of a numeral. Category 2 is for the images that are easily confused with other numerals because of similar structures. As

we known, some pairs of numerals are easier to be confused than others; such numerals are 4-9, 0-6. Images in the category 2 usually belong to these confusing pairs. Category 3 is for the images that are difficult to be identified by humans because of noise, filled loop, and over-segmentation, etc. Degradation and distortion are probably due to the quality of the scanner, the width of the tip of the instrument and the writing habit of certain writers. The data distributions of these three categories of each classifier are listed in Table 5.

Classifier	Total number of error	Category 1	Category 2	Category 3
MNIST database				
GPR	94	59	24	11
VSVM <sup>b</sup>	38	17	15	16
VSV2	56	32	15	9
LeNet5	82	49	17	14
POE	168	118	41	9
CENPARMI database				
VSVM	26	16	6	4
USPS database				
VSVM	47	21	13	13
NIST database				
MLP	119	81	30	8

Table 5: Three categories of misrecognized data

It is normal that images belong to category 3 are misrecognized since the information obtained from the image is insufficient or incorrect. For instance, it is hard to identify a numeral six by seeing only the right half of it because it can be a six or a zero. It is extremely difficult to recognize this type of numerals, so the best way of handling them is to reject them. Fortunately, the quantity of images in category 3 is small.

It is also reasonable that images in category 2 are misrecognized since they are ambiguous. Images in this category have similar structures but have different local

characteristics. Images in Category 2 can be recognized correctly if we examine their local differences. However, since some of them are confusing even to humans, it is hard to say whether the labels in the database or the estimated label is correct. Therefore, we should only attempt to recognize parts of this category that can be identified by most of the people. The rest should be rejected.

More than half of the collection of misrecognized images belongs to category 1. Humans have no difficulty in identifying these images. Some of them are clear but do not have a typical global structure of a numeral. The reason of misrecognition might be the lack of training samples of these structures. However, when some of these images are written clearly in a typical and simple shape of a numeral; it is surprising to see that the classifiers misrecognize these images. Images in this category are the targets of the verifier.

We also observed that some images are misrecognized by the majority of classifiers. The images that are misrecognized by more than three classifiers are shown in Table 6. Since most of the classifiers carried out experiments on the MNIST database, we listed only classifiers using this database.









								
Seq No	583	6577	948	2036	8409	2136	1015	1233
Label	8	7	8	5	8	6	6	9
GPR			X		X	X	X	X
VSVM <sup>b</sup>	X	X	X	X	X	X	X	X
VSV2	X	X	X	X	X	X	X	X
LeNet5	X			X	X	X	X	X
POE	X	X	X		X	X	X	X

Table 6: Common mistakes of classifiers

The first three images are perturbed numerals which belong to the category 3. The circle at the bottom of the first image is over-segmented and the right part of the upper circle is

lost due to the writing habit of the writer. The upper right part of the numeral seven is lost due to the scanner or writing instrument. The third image is a typical perturbation caused by the wide tip of the writing instrument, which filled the lower circle of the numeral eight. The image looks like a numeral nine. Some classifiers have better performance in identifying these seriously perturbed numerals. The GPR is able to identify the first two images while the LeNet-5 is able to recognize the last two images.

The fourth to seventh images are easily recognized by humans. They belong to the category 1. The images with sequence numbers of 2036 and 8409 have a clear and typical structure of numeral eight and five, respectively. However, unsatisfactory results are produced by the selected classifiers. The numeral five can be recognized correctly only by the classifiers GPR and POE. The numeral eight are misrecognized by all five classifiers. It is surprising that all classifiers misrecognized it as numeral five since there is nothing in common between this image and numeral five. All classifiers also misrecognized the two instances of numerals six whose sequence numbers are 2136 and 1015 respectively. As we can see, these two numerals six do not have a typical shape of numeral six but still can be easily recognized by human beings with the structure of a circle at the bottom and a barely vertical stroke on top, human beings can identify the image 2136 as a six easily even though the circle is small. Similarly, although the image 1015 is slanted, it has the structure of a typical numeral six; therefore, it can be identified by human beings easily. However, all classifiers also misrecognized these two numerals. The main reason of these errors may be due to the lack of training samples. Numeral six 2136 has a very small circle in the bottom part, which is unusual in numeral six; numeral six 1015 has a big slant. Thus, there are not enough training samples that are similar to



these two numerals to train the recognizer. Though there are a lot of training samples in the structure similar to the numeral eight 8409, we observed that the ratio of height to width of the minimum rectangle of the numeral eight 8409 is different from other instances of numeral eight. Usually, for a numeral eight, the ratio of height to width of the minimum rectangle is larger than 1. However, the ratio is smaller than 1 for the numeral eight 8409 because of the long stroke, which is connected to the right part of the upper circle of the eight. In this respect, the numeral eight 8409 is a rare case indicating the lack of training samples for the recognizer.

The last image also cannot be identified by all classifiers listed in the table. All classifiers recognized it as a 4. This result is reasonable since the image is confusing with numeral four.

### **5.3.2 Mis-verified data**


As shown in section 5.2, although the verifier can correctly reject most of the errors made by the classifiers, it still made some errors. The distributions of mis-verified data in each numeral are listed in Table 7. Some of the data that are misrecognized by more than one classifier and also mis-verified by the verifier is counted as one in the table. We also observed that some mis-verified images were labeled incorrectly. These mis-labeled numerals are not counted as errors of the verifier. Mis-labeled images are displayed with a square boundary shown in the figures of section 5.2. The first three of them come from the NIST database; the rest from the USPS database.

Numeral nine is the numeral that is mis-confirmed most frequently. Most of the verifiers mis-confirmed it as numeral four. This is due to the similarity of the shapes of numeral nine and four. They have the same structural feature. The major difference is the

smoothness of a particular curve. When the handwriting is cursive, the difference in the smoothness of these two numerals becomes small. This increases the difficulty of distinguishing these two numerals.

<b>Numeral</b>	<b>0</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>6</b>	<b>7</b>	<b>8</b>	<b>9</b>
MNIST database	0	1	2	4	2	1	1	3	3	8
NIST database	0	8	3	4	0	1	2	2	1	2
USPS database	0	1	0	1	0	1	1	0	1	0
CENPARMI database	0	0	0	0	1	0	0	0	0	1
<b>Total</b>	<b>0</b>	<b>10</b>	<b>5</b>	<b>9</b>	<b>3</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>5</b>	<b>11</b>

Table 7: Distribution of mis-verified data

Numeral one also has a large number of mis-verified data. 80% of them come from the the NIST database. Most of them have the shape of  and misrecognized as numeral two. The reason of these errors is due to the weak distinguishing power of the verifier on a particular shape of the numerals one and two.

Numeral three also has a lot of instances of mis-verified data. Some of them are due to the perturbation and low quality of the image and some of them are written in a form that is confusing with numeral 5. Another reason is due to the writing style: the direction change from the upper D curve to low D curve is too small to be detected by the system.

For other numerals, the possible reasons of mis-confirmation are:

- The width of the tip of the writing instrument. This causes some circles to be filled such as the lower circle of numeral eight; or some strokes to touch one another, i.e. one end of the D curve touches the other end of a numeral then forms a circle.
- Low quality of the image due to scanner or writing instrument. This makes some parts of the image broken or some noises occur; thereby, erroneous features are detected.

- Cursive writing. Some numerals are so cursive that some strokes become connected unusually. Hence, incorrect structural features are detected.
- Confusing writing style. Some people write a numeral six with a short stroke on top of a circle, which is confusing with numeral zero. Some users write a numeral seven with a small tail under the bottom horizontal bar, which is confusing with numeral two.

Besides the reasons mentioned above, which are due to the image itself, there are other reasons due to the limitation of the proposed system.

- Limitation of prototype based system. There are great variations in unconstrained handwritten numerals. Different people have different writing styles. Handwritten numerals vary even when written by the same people. Thus it is impossible to build a set of prototypes to represent all instances of each class of numeral.
- Binarization of the image. Images in some database are in gray level. Thus, we need to convert it to binary level using a threshold. In the procedure of binarization, some information may be lost.
- The limitation of the skeleton algorithm. It is a common limitation of skeleton algorithm that the skeleton of a stroke is shorter than the stroke in the original image. It is not a problem for most strokes while their lengths are long. However, it is a serious problem for strokes that are short. They may be lost in a skeleton. This will lead to mis-confirmation by the verifier. For example, for a numeral seven with a horizontal bar at the lower part, if the short stroke under the horizontal bar is lost due to the skeleton algorithm, the numeral looks more like a numeral two than a numeral seven. We tried to extend the skeleton by extending

the end point of a stroke to its real end point of the contour. In this way, the skeleton can represent the original image more accurately. We used the end point detection technique presented in the paper by Legault et al [56]. However, the detection of end point in the contour is not always accurate. It indeed correctly extended some strokes to their real end points, but it can also create some spurious strokes, which increase the error rate. Since the goal of the proposed system is high reliability, we did not apply these techniques to the proposed system.

- The limitation of the skeleton. Most features in the proposed system are detected from the skeleton. However, the skeleton cannot provide some essential information such as a filled circle. The skeleton delivers a filled circle as a stroke, which makes a numeral eight with filled lower circle look exactly like a numeral nine. Moreover, the weak feature capturing power of the skeleton is another reason of mis-verification of confusing pairs of numerals. For instance, the main discriminative characteristic of numerals four and nine is the smoothness of the C type curve. The direction change of the C curve of numeral nine is smoother than that of the numeral four. As presented in Figure 45, the left image is the original image; the right image is the skeleton of the original image. The C curve in the original image is smooth. On the contrary, in the skeleton, there is one bend point in the left lower part of the C curve, which is an important characteristic of the numeral four. Because features are extracted from the skeleton, which misrepresented the original image sometimes, the image is mis-confirmed as a numeral four.

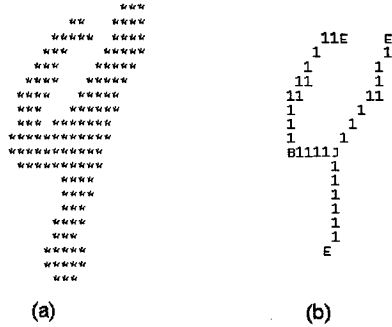


Figure 45: An instance of numeral nine:  
 (a) the original of numeral nine, (b) the skeleton of numeral nine

- Extra stroke remover. Most of the skeleton algorithms produce some extra strokes, which are harmful to the identification of a numeral. Thus, we implemented an extra stroke remover to remove some extra strokes whose length is smaller than a threshold as described in section 2.2. Unfortunately, some essential strokes are short and thus removed by the remover. Removing essential short strokes may lead to misidentification by the verifier. For instance, a numeral four with a small tail in the right has a higher probability to be recognized as a numeral four than the one without the tail (see Figure 46). Removing this tail will affect the confidence of identifying numeral four.

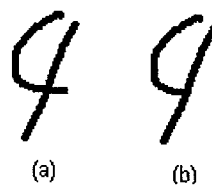


Figure 46: Two instances of numeral four:  
 (a) numeral four with small tail, (b) numeral four without tail

### 5.3.3 Experiment of human recognition of ambiguous data

As we mentioned in the previous section, a large number of errors made by the classifiers and verifier are the pattern containing ambiguous data. These data confused both the classifiers and verifiers. However, what is the recognition performance of humans? This

is one issue we want to know. The verifier extracts features and identifies numerals in a way that simulates humans' judgment. If even humans misrecognize these data, it is almost impossible for the verifier to identify them. In other words, instead of confirming these ambiguous data, it is better to reject them to avoid such mistakes.














In order to know the humans' recognition performance of ambiguous data, we designed the following experiment.

Twenty-eight ambiguous numeral images are used in this experiment: seven ambiguous data are selected from the MNIST database; thirteen are chosen from the NIST database, including two mis-labeled images; one is selected from the CENPARMI database; finally, seven data are chosen from the USPS database, including two mis-labeled images. These data are shown in Figure 47, but only images are given to the participants. The true value, the estimated value and the sequence number are not provided to the participants.








Twelve persons participated in this experiment. Considering the fact that handwriting styles for digits differ from one country to another, the participants are drawn from different countries: Canada (two participants), China (two participants), Egypt (two participants), Lebanon (one participant), Kazakh (one participant), Palestine (one participant), Nigeria (one participant), Ecuador (one participant), and Peru (one participant).

9→4	3→5	7→2	2→0	5→0	8→9	7→2
						
360	450	1227	2463	3559	6756	9506

(a)

0->4	0->6	1->2	1->7	1->7	1->7	3->5	3->8	3->9	5->3
									
4177	5334	5137	5025	516	5765	1567	2890	5196	2104
5->6	6->0	9->4							
									
4206	158	1302							

(b)  
7->4  
  
1525  
(c)

8->0	2->7	4->1	5->0	1->6	3->5	1->7
						
199	915	971	994	1335	1358	1815

(d)

Figure 47: Data of the experiment.

The first digit on the top of each image indicates the true label. The second digit on the top specifies the estimated digit. The number on the bottom of each image is the sequence number of the image in the database: (a) data from MNIST (b) data from NIST (c) data from CENPARMI (d) data from USPS

The following conditions are applied:

- Each participant must identify a unique label or label “U” for each pattern. Label “U” represents “Unknown”. It is used when the participant cannot identify a unique label.
- It is an individual work. Participants cannot consult others when they have difficulty in identifying these images.

From the experiment, we observed that four images of these numerals are labeling errors, which are displayed in bold font in Table 8. All participants identified them in the same

labels that differ from the labels in the database. Two of them come from the NIST database; the other two from the USPS database.

It can be seen that most of these images are confusing except for two images. Most of them can not be identified consistently by all participants. Therefore, it is understandable that the classifier and verifier made errors in identifying them.

A good database must reduce such ambiguous numerals as much as possible. Otherwise, it is hard to measure the real power of applications based on the database.

We also observed that humans are able to identify highly distorted numerals, such as the numeral two with sequence number of 2463 from the MNIST database. Though the image is highly distorted due to the thick tip of the instrument, 10 participants can still identify it correctly. Further study in what primitives a human relies on to recognize highly distorted numerals will be helpful in designing a better numeral recognizing system.

In conclusion, this chapter first presented the experimental design and the result of the proposed system on MNIST database: the system yielded a reliability of 99.92% along with a recognition rate of 96.50%. Then the experimental result of the proposed verifier on the collection of misrecognized data was provided. Finally, a thorough error analysis of misclassified and mis-verified data was given.



Sequence No	Label	Correctly recognized	Misrecognized	Unknown
<b>MNIST</b>				
360	9	3	5	4
450	3	11	1	0
1227	7	12	0	0
2463	2	10	0	2
3559	5	5	3	4
6758	8	9	3	0
9506	7	4	6	2
<b>NIST</b>				
4177	0	1	8	3
5334	0	4	6	2
5137	1	12	0	0
5025	1	10	2	0
516	1	11	0	1
5765	1	4	3	5
1567	3	0	9	3
2890	3	10	1	1
<b>5196</b>	<b>3</b>	<b>0</b>	<b>12</b>	<b>0</b>
<b>2104</b>	<b>5</b>	<b>0</b>	<b>12</b>	<b>0</b>
4206	5	5	3	4
158	6	8	2	2
1302	9	1	10	1
<b>CENPARMI</b>				
1525	7	8	3	1
<b>USPS</b>				
199	8	2	6	4
915	2	2	5	5
<b>971</b>	<b>4</b>	<b>0</b>	<b>12</b>	<b>0</b>
<b>994</b>	<b>5</b>	<b>0</b>	<b>12</b>	<b>0</b>
1335	1	7	2	3
1358	3	6	2	4
1815	1	11	1	0
Sum		156	129	51

Table 8: Experimental result of human recognition of ambiguous numerals.

## Chapter 6

### Conclusions

The goal of the verification system is to increase the reliability of the handwritten numerals recognition system while maintaining a reasonable recognition rate. Thereby, this satisfies the requirement of some financial document processing systems which demand high reliability.

The proposed verifier is a prototype based verifier. Different numbers of prototypes are constructed for the 10 classes of numerals according to the complexity of each numeral. Prototypes are built using the structural primitives of a numeral and the relations among them. The verifier is applied to the results of the three classifiers: SVM, LeNet5, and MQDF.

The result has shown that the reliability increased from 99.06% to 99.92% on MNIST database after applying the verifier. This performance shows the verifier's ability of raising system reliability and proves the effectiveness of this approach. The proposed verifier has successfully achieved the goal of the work.

Although the error rate drops from 0.94% (94 errors) to 0.08% (8 errors), the recognition rate also drops from 99.06% to 96.50% with the verifier. The decrement is larger than what we expected. The main reason of this result is due to the restrictive confirmation requirement of the verifier. In contrast, if the confirmation requirement loosened, the recognition rate will increase while the reliability will decrease. How to achieve a proper

trade off between the recognition rate and the reliability is still a dilemma. It depends on the goal of the system: recognition rate or reliability. In our case, the goal is reliability; thus, restrictive conditions are set in confirming the recognition result. The second main reason is the limitation of prototype based system. The prototypes of the proposed system are only for patterns in common shapes and that can represent the majority of writing styles of numerals. When the verifier meets some patterns that do not belong to any of these prototypes, it will reject the patterns.

The last main reason is the problem of skeletonization. Skeleton has the advantage of describing the structure of an image explicitly and clearly. However, it also has the disadvantage of losing important information sometimes. The majority of features of the proposed system are extracted from the skeleton. Therefore, for some patterns, insufficient features are provided to the verifier. Consequently, the verifier will reject these patterns.

In conclusion, the proposed verifier has successfully improved the reliability of the unconstrained handwritten recognition system while maintaining a reasonable recognition rate. The reliability is pretty high using the proposed verifier. However, more efforts can be devoted in the future to increase the recognition rate while maintaining high reliability, such as:

- Building more prototypes to represent more instances of numerals.
- Extracting more features from a binary image or the contour of the image.
- Extracting more statistical features and integrating them with structural features.
- Trying other dynamic methods that have a better generalization power.

## Bibliography

- [1] P. A. Devijver and J. Kittler, "Pattern recognition: a statistical approach," Prentice-Hall, London, England, 1982.
- [2] O. D. Trier, A. K. Jain, and T. Taxt, "Feature extraction methods for character recognition — a survey," *Pattern Recognition*, 29(4), 1996, pp. 641-662.
- [3] I. S. Oh and C. Y. Suen, "Distance features for neural network based recognition of handwritten characters," *International Journal on Document Analysis and Recognition*, vol. 2, 1998, pp. 73-88.
- [4] H. Nishida, "Curve description based on directional features and quasi-convexity /concavity," *Pattern Recognition*, 28(7), 1995, pp.1045-1051.
- [5] R. R. Bailey and M. Srinath, "Orthogonal moment features for use with parametric and non-parametric classifiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(4), 1996, pp. 389-399.
- [6] M. Shridhar and A. Badreldin, "High accuracy character recognition algorithm using fourier and topological descriptors," *Pattern Recognition*, 17(5), 1984, pp. 515-524.
- [7] S. Pittner and S. V. Kamarthi, "Feature extraction from wavelet coefficients for pattern recognition tasks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(1), 1999, pp. 83-88.
- [8] A. Soto and K. Yamada, "Generalized learning vector quantization," *Advances in Neural Information Processing Systems*, vol.7, MIT Press, Cambridge, MA, U.S., 1995, pp. 423-429.
- [9] D. Guillevic and C. Y. Suen, "Cursive script recognition applied to the processing of bank cheques," *Proceeding of 3<sup>rd</sup> International Conference on Document Analysis and Pattern Recognition*, vol.1, Montreal, Canada, 1995, pp.11-14.
- [10] K. W. Cheng, D. Y. Yeung, and R. T. Chin, "A Bayesian framework for deformable pattern recognition with application to handwritten character recognition," *IEEE*

*Transactions on Pattern Analysis and Machine Intelligence*, 20(12), 1998, pp. 1382-1388.

- [11] J. Cai and Z. Q. Liu, "Integration of structural and statistical information for unconstrained handwritten numeral recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(3), 1999, pp. 263-270.
  
- [12] C. M. Bishop, "Neural networks for pattern recognition," Oxford University Press, New York, U.S., 1995.
  
- [13] B. Zhang, M. Fu, H. Yan, and M. A. Fabri, "Handwritten digit recognition by adaptive-subspace self organizing map (ASSOM)," *IEEE Transactions on Neural Networks*, vol. 10, 1999, pp.939-945.
  
- [14] O. Matan, J. C. Burges, Y. LeCun, and J. S. Denker, "Multi-digit recognition using space displacement neural network," *Advances in Neural Information Processing Systems*, vol. 4, Morgan Kaufmann, 1992, pp. 488-495.
  
- [15] A. Waibel T. Hanazawa, G. Hilton, K. Shikano, and K. J. Lang, "Phoneme recognition using time-delay neural networks, Acoustics," *IEEE Transactions on Speech, Signal Processing*, 37(12), 1989, pp. 1888-1898.
  
- [16] Y. C. Chim, A. A. Kassim, and Y. Ibrahim, "Dual classifiers system for handprinted alphanumeric character recognition." *Pattern Analysis and Applications*, no. 1, 1998, pp.155-162.
  
- [17] D. Guillevic and C. Y. Suen, "HMM-KNN word recognition engine for bank cheque processing," *Proceeding of 14<sup>th</sup> International Conference of Pattern Recognition*, vol. 2, Brisbane, Australia, 1998, pp.1526-1529.
  
- [18] F. R. Rahman and M. C. Fairhurst, "Serial combination of multiple experts: a unified evaluation," *Pattern Analysis and Applications*, no. 2.1999, pp.292-311.
  
- [19] R. K. Powalka, N. Sherkat, and R. J. Whitrow, "Multiple recognizer combination topologies," *Handwriting and Drawing Research: basic and Applied Issues*, IOS Press, 1996, pp.329-342.

- [20] L. Lam and C. Y. Suen, "Optimal combinations of pattern classifiers," *Pattern Recognition Letters*, vol. 16, 1995, pp. 945-954.
- [21] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas, "On combining classifiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 3, 1998, pp. 226-329.
- [22] S. B. Cho, "Neural-network classifiers for recognizing total unconstrained handwritten numerals," *IEEE Transactions on Neural Networks*, vol. 8, no. 1, 1997, pp. 43-53.
- [23] C. Y. Suen, K. Liu, and N. W. Strathy, "Sorting and recognizing cheques and financial documents," *Proceeding of the 3<sup>rd</sup> International Association for Pattern Recognition Workshop on Document Analysis system*, Nagano, Japan, 1998, pp. 173-187.
- [24] Z. Y. Li, S. W. Tang, and H. Wang, "Pairwise coupling support vector machine and its application on handwritten digital recognition," *IEEE 2002 International Conference on Communications, Circuits and Systems and West Sino Expositions*, vol. 2, Chengdu, Sichuan, China, 2002, pp. 1194-1198.
- [25] W. Kim, J. Paik, K. Lee, H. Byun, and Y. Lee, "Handwritten digits verifier for improving recognition error," *Proceeding of 4<sup>th</sup> International Conference on Document Analysis and Pattern Recognition*, vol. 1, Ulm, Germany, 1997, pp. 368-368.
- [26] K. K. Kim, Y. K. Chung, and C. Y. Suen, "Post-processing scheme for improving recognition performance of touching handwritten numeral strings," *Proceeding of 16<sup>th</sup> International Conference on Document Analysis and Pattern Recognition*, vol. 1, Quebec, Canada, 2002, pp. 327-330.
- [27] M. Morita, R. Sabourin, F. Bortolozzi, and C. Y. Suen, "A recognition and verification strategy for handwritten word recognition," *Proceeding of 7<sup>th</sup> International Conference on Document Analysis and Pattern Recognition*, vol.1, Edinburgh, Scotland, 2003, pp. 482-486.

- [28] J. M. Hu and H. Yan, "Structural primitive extraction and coding for handwritten numeral recognition," *Pattern Recognition*, vol. 31, no. 5, 1998, pp. 493-509.
- [29] R. Legault and C. Y. Suen., "A contour-based recognition system for totally unconstrained handwritten numerals," Technical Report, Concordia, Montreal, 1989.
- [30] T. Y. Zhang and C. Y. Suen, "A fast parallel algorithm for thinning digital patterns," *Communication of the ACM*, vol. 27, 1984, pp. 236-239.
- [31] R. Y. Wu and W. H. Tsai, "A new one-pass parallel thinning algorithm for binary images," *Pattern Recognition Letters*. vol. 13, no. 5, 1992, pp. 715-723.
- [32] N. H. Han, C. W. La, and P. K. Rhee, "An efficient fully parallel thinning algorithm," *Proceeding of 4<sup>th</sup> International Conference on Document Analysis and Pattern Recognition*, vol.1. Ulm, Germany, 1997, pp. 137-141.
- [33] P. Siy and C. S. Chen, "Fuzzy logic for handwritten numeral character recognition," *IEEE Transactions on Systems, Man, and Cybernetics*, November 1974, pp. 570-575.
- [34] G. Baptista and K. M. Kulkarni, "A high accuracy algorithm for recognition of handwritten numerals," *Pattern Recognition*, vol. 21, no. 4, 1988, pp. 287-291.
- [35] H. Takahashi and T. Griffin, "Recognition enhancement by linear tournament verification," *Proceeding of 2<sup>nd</sup> International Conference on Document Analysis and Pattern Recognition*, vol.1, Tsukuba, Japan, 1993, pp. 585-588.
- [36] J. E. Freund, "Mathematical statistics," Sixth Edition, Prentice Hall, Upper Saddle River, New Jersey, 1998, pp.142, 150, 151.
- [37] J. X. Dong, A. Krzyzak, and C. Y. Suen, "A fast SVM training algorithm," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 17, no. 3, 2003, pp. 367-384.
- [38] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, 80(11), November 1998, pp. 2278-2324.

- [39] F. Kimura, K. Takashina, S. Tsuruoka, and Y. Miyake, "Modified quadratic discriminant functions and the application to Chinese character recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 9, no. 1, Jan. 1987, pp. 149-153.
- [40] C. Cortes and V. Vapnik, "Support vector networks," *Machine Learning*, vol. 20, 1995, pp. 273-297.
- [41] R. Courant and D. Hilbert, "Methods of mathematical physics," Interscience, New York, U.S., 1953.
- [42] S. S. Keerthi and E. G. Gilbert, "Convergence of a generalized SMO algorithm for SVM classifier design," *Machine Learning*, vol. 46, 2002, pp. 351-360.
- [43] R. O. Duda and P. E. Hart, "Pattern Classification and Scene Analysis," Wesley, New York, U.S., 1973. pp. 67.
- [44] C. Y. Suen, C. Nadal, T. A. Mai, R. Legault, and L. Lam, "Computer recognition of unconstrained handwritten numerals," *Proceeding of the IEEE*, vol. 80, no. 7, 1992, pp. 1162-1180.
- [45] T. K. Ho, J. J. Hull, and S. N. Srihari, "Decision combination in multiple classifier systems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 1, 1994, pp. 66-75.
- [46] J. Franke and E. Mandler, "A comparison of two approaches for combining the votes of cooperating classifiers," *Proceeding of 11<sup>th</sup> International Conference on Pattern Recognition*, vol.2, Hague, Netherlands, 1992, pp. 611-614.
- [47] L. Lam, "Classifier Combinations: Implementations and theoretical Issues," *Proceeding of 1<sup>st</sup> International Workshop on Multiple Classifier Systems*, Cagliari, Italy, 2000, pp. 77-79.
- [48] J. Zhou, A. Krzyzak, and C. Y. Suen, "Verification – a method of enhancing the recognizers of isolated and touching handwritten numerals," *Pattern Recognition*, (35), 2002, pp. 1179-1189.






































- [49] G. Casella and R. L. Berger, "Statistical inference," Wadsworth & Brooks, Pacific Grove, California, U.S., 1990.
- [50] J. X. Dong, "Speed and accuracy: large-scale machine learning algorithms and their applications," Ph.D. thesis, Concordia University, 2003.
- [51] D. Decoste and B. Scholkopf, "Training invariant support vector machines," *Machine Learning*, vol. 46, 2002, pp.161-190.
- [52] G. Mayraz and G. E. Hinton, "Recognizing handwritten digits using hierarchical products of experts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 2, 2002, pp. 189-197.
- [53] J. Bromley and E. Sackinger, "Neural-network and k-nearest-neighbor-classifiers," Technical Report 11359-910819-16TM, AT&T.
- [54] J. X. Dong, A. Krzyzak, and C. Y. Suen, "Statistical result of human performance on USPS database," Report, CENPARMI, Concordia University, 2001.
- [55] L. S. Oliveira, "Automatic recognition of handwritten numerical strings," Ph.D. thesis, Ecole De Technologie Superieure Universite du Quebec, 2003.
- [56] R. Legault, "Unconstrained handwritten numeral recognition: A contribution towards matching human performance," Ph.D. thesis, Concordia University, 1997.
- [57] E. Kussul, and T. Baidyk, "Permutative coding technique for handwritten digit recognition system," *Proceeding of the International Joint Conference on Neural Networks*, vol. 3, Portland, Oregon, U.S., 2003, pp. 2163-2168.
- [58] C. L. Liu, K. Nakashima, H. Sako, and H. Fujisawa, "Handwritten digit recognition using state-of-the-art techniques," *Proceeding of the 8<sup>th</sup> International Workshop on Frontiers in Handwriting Recognition*, Ontario, Canada, 2002, pp. 320-325.

- [59] C. L. Liu, K. Nakashima, H. Sako, and H. Fujisawa, "Handwritten digit recognition: investigation of normalization and feature extraction techniques," *Pattern Recognition*, vol. 37, 2004, pp. 265-279.
- [60] L. S. Oliveira, R. Sabourin, and C. Y. Suen, "Feature selection using multi-objective genetic algorithms for handwritten digit recognition," *Proceeding of the 16<sup>th</sup> International Conference on Pattern Recognition*, vol. 1, Quebec, Canada, 2002, pp. 568-571.


































# Appendix A

This appendix presents images misrecognized by different classifiers.

## VSVMb on MNIST database

4→6  248	8→2  583	2→7  584	3→5  939	8→9  948	6→5  1015	7→2  1227	9→4  1233	9→5  1248	7→1  1261
8→3  1879	9→4  1902	5→3  2036	7→9  2071	4→9  2131	6→1  2136	7→4  2524	3→2  2928	9→5  2940	7→9  3226
6→0  3423	6→8  3763	9→7  4164	2→7  4177	1→7  4202	3→2  4444	8→7  4498	9→8  4762	9→4  4824	7→2  5655
5→3  5938	7→1  6577	7→2  8317	8→5  8409	7→2  9506	5→6  9730	4→9  9793	2→7  9840		



## VSV2 on MNIST database

4→9  448	8→2  583	2→7  660	5→3  675	7→3  727	8→9  948	6→5  1015	4→6  1113	7→2  1227	9→4  1233
9→5  1248	5→7  1300	8→0  1320	8→7  1531	4→6  1550	3→7  1682	9→5  1710	2→8  1791	9→4  1902	5→3  2036
7→9  2071	2→0  2099	4→9  2131	6→1  2136	1→2  2183	9→6  2294	2→4  2489	6→1  2655	3→2  2928	9→7  2940
3→5  2954	6→8  3031	1→2  3074	7→9  3226	6→0  3423	6→4  3521	4→8  3535	5→0  3559	7→0  3605	6→8  3763
9→4  3870	9→4  3986	9→3  4079	9→4  4762	9→4  4824	5→3  5938	7→1  6577	0→7  6598	1→6  6784	0→6  8326
8→5  8409	2→7  9665	5→6  9730	5→6  9750	4→9  9793	0→6  9851				

LeNet5 on MNIST database




























4->6 4 248	3->5 3 450	8->2 3 583	2->1 1 660	5->3 3 675	4->8 4 830	2->8 2 927	3->5 3 939	6->5 6 1015	7->3 1 1040
9->4 4 1233	8->0 0 1320	7->8 7 1329	5->3 5 1394	8->7 8 1531	0->6 0 1622	3->7 3 1682	2->7 2 1791	8->3 8 1879	9->4 4 1902
8->2 8 1956	5->3 5 2036	4->8 4 2044	3->9 3 2110	6->0 6 2119	9->8 9 2130	4->9 4 2131	6->1 6 2136	9->4 9 2294	9->1 9 2388
9->4 9 2415	2->0 2 2463	6->1 6 2655	3->5 3 2922	3->2 3 2928	9->5 9 2940	6->0 6 3031	6->0 6 3423	6->0 6 3521	6->8 6 3763
4->6 4 3781	7->3 7 3809	9->4 9 3870	4->6 4 3942	2->7 2 4177	9->7 9 4225	4->3 4 4266	9->4 9 4370	9->4 9 4406	9->4 9 4426
8->7 8 4498	4->2 4 4576	8->4 8 4602	3->5 3 4741	8->4 8 4957	6->5 6 5737	8->5 8 5750	3->8 3 5956	3->8 3 5974	9->8 9 6072
1->5 1 6102	9->8 9 6174	6->3 6 6539	0->2 0 6558	6->5 6 6559	9->5 9 6593	0->7 0 6598	1->6 1 6784	4->9 4 7435	2->1 2 8060
2->8 2 8095	8->5 8 8409	4->9 4 8528	7->2 7 9010	7->2 7 9016	6->5 6 9680	9->7 9 9693	6->1 6 9699	5->6 5 9730	5->0 5 9771
4->9 4 9793	2->8 2 9905								

POE on MNIST database
















































9->8  152	4->6  248	9->4  265	2->7  322	9->4  360	8->5  543	8->3  583	2->8  584	2->7  660	7->3  685
4->9  741	7->3  727	8->9  948	6->5  1015	7->3  1040	4->6  1113	6->1  1182	6->5  1183	7->2  1227	9->4  1233
9->5  1248	7->1  1261	3->5  1291	8->3  1320	8->2  1365	5->3  1394	2->8  1396	7->9  1523	1->5  1528	8->7  1531
4->6  1550	9->3  1560	3->7  1682	9->3  1710	2->7  1791	6->4  1801	1->2  1869	2->8  1872	8->3  1879	9->4  1902
7->2  1904	4->8  2044	2->0  2099	3->7  2110	6->0  2119	9->2  2130	6->1  2136	1->3  2183	0->8  2186	9->4  2294
9->4  2407	6->8  2455	2->0  2463	2->4  2489	5->3  2575	9->7  2583	5->3  2598	3->5  2619	6->1  2655	9->4  2761
4->9  2772	8->0  2897	3->8  2922	9->5  2940	3->5  2953	3->5  2954	5->3  2971	9->4  3006	6->0  3031	9->7  3061
1->2  3074	8->9  3290	2->8  3331	5->9  3337	3->7  3476	9->3  3504	2->7  3512	6->4  3521	4->8  3535	5->0  3559

9->3 9 3598	7->2 7 3605	5->3 5 3703	4->9 4 3719	7->2 7 3752	3->5 3 3767	7->3 7 3768	2->8 2 3797	7->8 7 3809	9->4 9 3860
5->6 5 3894	5->3 5 3903	1->3 1 3907	4->6 4 3942	5->3 5 3969	3->5 3 3996	0->2 0 4066	9->3 9 4079	2->7 2 4177	7->9 7 4200
1->4 1 4202	9->7 9 4225	9->5 9 4285	9->5 9 4345	3->2 3 4438	0->6 0 4478	5->6 5 4549	8->9 8 4640	3->2 3 4658	8->3 8 4672
6->1 6 4700	3->5 3 4741	9->7 9 4762	5->6 5 4764	8->0 8 4808	9->4 9 4824	7->1 7 4887	3->5 3 5047	8->7 8 5279	8->7 8 5289
1->6 1 5332	1->8 1 5458	2->3 2 5635	1->8 1 5643	4->2 4 5677	6->0 6 5737	7->0 7 5888	5->3 5 5938	3->8 3 5956	5->3 5 5973
5->3 5 5982	5->3 5 5998	9->3 9 6082	9->3 9 6092	9->3 9 6167	9->3 9 6169	9->8 9 6174	0->5 0 6533	8->9 8 6556	9->7 9 6569
7->1 7 6577	0->9 0 6598	0->8 0 6652	8->9 8 6756	8->9 8 7122	3->2 3 7801	3->2 3 7822	8->0 8 7922	2->1 2 8060	2->1 2 8070
2->8 2 8095	4->1 4 8096	2->8 2 8113	3->9 3 8247	3->8 3 8278	3->8 3 8297	0->6 0 8326	3->5 3 8398	8->5 8 8409	4->9 4 8521
4->9 4 8528	7->2 7 9016	7->2 7 9025	2->7 2 9665	5->6 5 9730	4->7 4 9746	2->0 2 9840	0->6 0 9851		

V SVM on CENPARMI database

1→2  397	2→8  588	3→9  648	3→2  747	3→2  783	3→2  790	4→6  809	4→6  812	4→9  919	5→8  1140
6→0  1351	7→9  1452	7→9  1453	7→4  1520	7→4  1525	7→2  1527	8→5  1637	8→2  1640	8→9  1663	8→0  1668
8→9  1764	9→7  1882	9→8  1928	9→8  1952	9→8  1965	9→2  1994	9→2  1994			

V SVM on USPS database

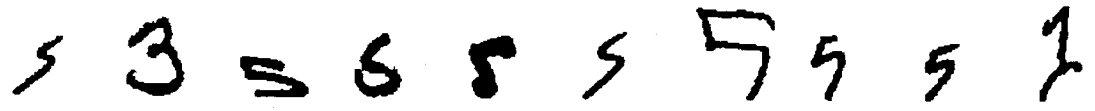
6→4  18	3→5  28	1→5  53	2→5  79	0→8  165	8→0  199	1→6  234	4→7  266	7→4  340	3→5  485
2→0  510	0→2  528	5→2  562	3→5  792	5→9  794	4→2  836	2→7  915	4→1  971	5→0  994	0→5  995
0→2  1007	7→4  1047	3→2  1105	7→2  1119	3→5  1307	1→6  1335	7→4  1354	7→4  1355	3→5  1358	4→6  1376
4→9  1380	4→9  1387	8→2  1417	3→5  1426	3→5  1432	6→8  1469	7→2  1529	9→7  1545	5→8  1657	4→9  1734
1→4  1814	1→7  1815	1→4  1816	9→8  1865	4→7  1872	5→8  1952	5→3  1978			

MLP on NIST database


0->4	0->4	0->4	0->4	0->6	0->8	0->9	1->2	1->2	1->2
4177	4179	4358	5447	5334	4945	4180	3375	359	360
1->2	1->2	1->2	1->2	1->2	1->2	1->2	1->2	1->2	1->2
361	4560	4564	4569	4959	5136	5137	5141	5145	5148
1->2	1->2	1->2	1->2	1->2	1->2	1->7	1->7	1->7	1->7
5601	586	588	590	593	594	5025	5026	514	516
1->7	1->7	1->7	1->8	2->3	2->7	2->8	2->8	2->8	2->8
5765	6103	6105	2813	1613	444	1290	2666	2919	5218
3->0	3->0	3->1	3->2	3->2	3->2	3->5	3->5	3->5	3->7
2819	840	4570	3600	4417	5692	1567	3411	5869	168
3->7	3->7	3->7	3->7	3->8	3->8	3->9	4->6	4->7	4->9
32	3965	3972	3975	2890	3599	5196	1833	3398	968



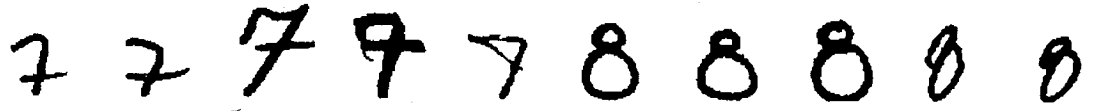
5->1 5->3 5->3 5->6 5->8 5->9 5->9 5->9 5->9 1->9

  
160 2104 494 4206 413 157 3229 813 815 516


5->9 5->9 5->9 5->9 6->0 6->0 6->0 6->1 6->1 6->5

  
517 818 819 820 158 3978 5894 1168 5486 159


7->2 7->2 7->9 7->9 7->9 8->0 8->0 8->0 8->0 8->0

  
3324 4123 1556 4012 5802 2103 2106 2110 2177 2183

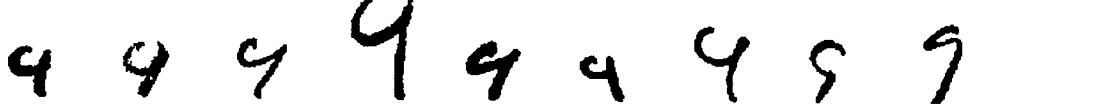
8->0 8->0 8->0 8->0 8->0 8->0 8->0 8->0 8->0 8->1

  
324 3299 3797 3798 386 418 420 421 5825 1234

8->2 8->2 8->3 8->4 8->4 8->9 9->3 9->4 9->4 9->4

  
2801 5867 1759 1850 2396 1238 3915 1296 1300 1302

9->4 9->4 9->4 9->4 9->4 9->4 9->4 9->5 9->7

  
1685 2544 2730 2805 3165 4663 4785 3052 270