FUSIONS OF CNN AND SVM CLASSIFIERS FOR RECOGNIZING HANDWRITTEN CHARACTERS

Xiaoxiao Niu

A Thesis
in
The Department
of
Computer Science and Software Engineering

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

April 2011

CONCORDIA UNIVERSITY
School of Graduate Studies

This is to certify that the thesis prepared

By: Xiaoxiao Niu

Entitled: Fusions of CNN and SVM Classifiers for Recognizing Handwritten Characters

and submitted in partial fulfillment of the requirements for the degree of

Master of Computer Science

complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

______________________________________ Chair
Dr. V. Haarslev

______________________________________ Examiner
Dr. S. Bergler

______________________________________ Examiner
Dr. A. Krzyzak

______________________________________ Co-supervisor
Dr. T. D. Bui

______________________________________ Co-supervisor
Dr. C. Y. Suen

Approved by ______________________________________
Chair of Department or Graduate Program Director

____
Dr. Drew, Robin A. L., Dean
Faculty of Engineering and Computer Science

Date ______________________________________

ABSTRACT

Fusions of CNN and SVM Classifiers for Recognizing Handwritten Characters

Xiaoxiao Niu

Off-line handwritten character recognition plays an important role on a very large scale in handwriting recognition systems. The ultimate goal of this research field is to let the machine read generic materials written by human beings. In order to achieve this, it is necessary to further improve the recognition accuracy and the reliability of current off-line handwritten character recognition systems.

The main contribution of this thesis is to present several ways of integrating the synergy of two superior classifiers: Convolutional Neural Network (CNN) and Support Vector Machine (SVM) which have proven results in recognizing different types of patterns. Two models of new fusions have been investigated. In the hybrid model, CNN works as a trainable feature extractor and SVM performs as a recognizer. It automatically extracts the features from the raw images and generates predictions. In the regular combination model, the CNN classifier is trained with raw images but with normalized sizes, and the SVM classifier is trained with handcrafted features. The reliability for both models has been realized through the introduction of a rejection mechanism.

The comparisons between the two proposed models were tested on handwritten digits and handwritten letters in the English language, respectively. For the handwritten digit recognition, experiments were conducted on the well-known MNIST database. Experimental results and comparisons with other works on the same database showed that the best results were achieved by the proposed hybrid model: An error rate of 0.19%

without rejection (compared to the most recent error rate of 0.35%), and a recognition rate of 94.40% under 100% reliability with rejection (compared to a recognition rate of 91.51% under 100% reliability from other studies to our best knowledge).

For the experiments on handwritten letters, the NIST database was applied. Three strategies were adopted in classifying handwritten letters into different types of classes. They are a 26-class problem in uppercases and a 26-class problem in lowercases, a 26-metaclass problem, and a 52-class problem. The recognition results without rejection by the hybrid model for the 26-class problem in uppercases and the 26-class problem in lowercases were 96.2289% and 90.2410%, respectively. The recognition rates without rejection were 92.0744% for the 26-metaclass problem and 70.2408% for the 52-class problem. Results showed that the hybrid model outperformed other single classifiers (SVM, CNN) in our experiments. However, the combination system had slight recognition improvements compared to the hybrid model on the 26-metaclass and the 52-class problems. Besides, the combination system was proven to be more reliable.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to Dr. Ching Y. Suen and Dr. Tien D. Bui. Both supervisors provided their generous support to my research and my daily life. Dr. Suen offered me a wealth of opportunities for increasing my research capabilities. His valuable suggestions and enthusiastic guidance finally led to this thesis. Dr. Bui gave me many inspiring comments, heartful encouragement and sufficient assistance during my studies. Their expertise, thought and personalities have strongly influenced me.

I would like to deeply thank to Dr. Louisa Lam, who gave me lots of constructive comments for my research. She also offered her invaluable experiences, goodwill and encouragement to me.

My grateful thanks go to my colleagues and friends at the Centre for Pattern Recognition and Machine Intelligence (CENPARMI). They are: Dr. Huiqun Deng, Dr. Wumo Pan, Dr. Chunlei He, Dr. Didier Guillevic, Dr. Sara Izadi Nia, Ms. Guiling Guo, Wu Ding, Ying Li, Xiaohua Xie, Bing Zhang, Khoa Luu, Malik Waqas Sagheer, and George Stathopoulos. Special thanks to our Research Manager, Mr. Nicola Nobile, for his excellent technical support and to Ms. Shira Katz for her considerable efforts in editing and proofreading my thesis. I also wish to thank our Secretary, Ms. Marleah Blom, who cheers up every student at CENPARMI.

I would like to acknowledge research support received from the Natural Sciences and Engineering Research Council of Canada and Concordia University.

Finally, it is beyond words to express my appreciation for the love and encouragement from my parents, my brother, my nephew and all of my loved friends.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter, the research topic, handwritten character recognition, is introduced. We begin with a description of the motivation behind this research in Section 1.1. The challenges faced are presented in Section 1.2. Some of the previous works are reviewed in Section 1.3. Based on these previous studies, we propose our methods in Section 1.4. Finally, the outline of this thesis is given in Section 1.5.

1.1 Motivation

Handwriting recognition is an important and challenging problem, and has been intensively studied for the last four decades. The ultimate goal of this field is to let the machine read a human's handwriting. To realize this, many achievements have been made in recognizing on-line and off-line handwritten characters. Lots of research results have indicated that for machines, the on-line handwriting recognition tasks are much easier than off-line tasks; and machines can achieve higher online recognition rates compared to offline. Thus, a lot of research is still needed in the off-line handwriting recognition applications.

Text recognition of the generic content is the long-term goal of off-line handwriting recognition. In the past decades, most research on the text recognition task has focused on specific applications with a restricted vocabulary, such as the zip-code reading in the postal mailing system, bank cheque processing in the financial system, etc. Some commercial systems have been developed and have been applied in the market [1, 2, 3].

Nowadays, the generic applications require the machine to recognize large vocabularies consisting of thousands or tens of thousands of words. For example, some common application tasks that work with large vocabularies include [4]:

- Postal applications: recognizing the postal addresses on the envelopes, such as the name, street, city, country, etc.;
- Genetic text transcriptions: reading unconstrained handwritten notes;
- Information retrieval: extracting the specified fields from handwritten forms;
- Reading handwritten fields in forms: the forms include tax forms, census forms, insurance forms and other business forms.

In these cases, handwritten character recognition plays an important role in the handwriting recognition system. Since current systems designed for a large vocabulary of off-line handwriting recognition problems are based on the character-model-based string recognition: the recognizer takes the character or pseudo-character as the basic recognition unit. Using the word as the recognition unit works very well on small vocabulary recognition applications, but it is no longer suitable for the large vocabulary tasks. Besides, a good recognition system for large vocabularies demands a large database for training and testing. However, the existing handwritten word datasets are quite limited both in the amount of words and the diversity of the words. On the other hand, the isolated handwritten characters (10 digits and 52 letters for English language) have sufficient available databases to use, such as the MNIST digit database [5], the CENPARMI digit database [6], the CEDAR character database [7], the NIST character database [8], the C-cube English letter database [9], etc. Furthermore, to create a new isolated handwritten character database, samples can be much more easily collected in terms of labor and time costs, compared with building a new large vocabulary database.

Driven by the goal of text recognition of generic content and to ultimately realize the machine simulation of human reading, our research is on off-line handwritten character recognition, and more specifically, on handwritten digit recognition and handwritten English letter recognition. The methods to improve the recognition accuracy and to enhance the reliability on both recognition problems are addressed in this thesis.

1.2 Challenge

Although numerous achievements have been made in the handwritten character recognition field, there are still big gaps between research studies and the demands of practical applications. Two main challenges have been encountered in the real world: the improvement of the recognition accuracy and the enhancement of a system's reliability. We will describe these two challenges, respectively, below.

Firstly, we give the definitions of the recognition rate and the reliability rate below:

$$\mathrm{Re}\,cognition = \frac{\#correctly \quad recognized \quad samples}{\#total \quad testing \quad samples} \tag{1}$$

$$\mathrm{Re}\,liability = \frac{\#total \quad testing \quad samples \quad - \quad \#errors}{\#total \quad testing \quad samples} \tag{2}$$

- **Improving the recognition accuracy**

The previously reported results on handwritten digit recognition have varied from 97.60% to 99.77% [10], depending on different databases, classification strategies and experimental conditions. One specific example of handwritten digit recognition research is the work of Ciresan et al. on the MNIST database, where the most recent and lowest error rate was 0.35%, obtained by using the 6-layer Neural Network [11]. However, a higher recognition rate on handwritten digits is always desired in real-life applications. When we consider recognizing a numeral string, the recognition probability of this string is the multiplication of the recognition probability of each isolated digit (assuming that each digit is correctly separated from the numeral string by the segmentation process). For example, take a five digit ZIP code, if we assume that the recognition rate of each isolated digit is 99.65%, then the recognition probability of this string is 98.26% ($\approx 0.9965^5$). Thus, to increase the recognition accuracy for handwritten numeral strings, it is necessary to boost the recognition performance of each individual digit in the string.

As for the unconstrained handwritten letter recognition, the recognition rates have been much lower. The recognition rates have often varied from 69% to 93% [12, 13, 14, 15, 16], due to the complex essence of this problem. There are 52 letters in the English language: 26 uppercases (A-Z) and 26 lowercases (a-z). The difficulties are caused by the fact that the writing style of the unconstrained letter is not known a priori to a recognition system. The recognizer should discover and handle different writing styles. Furthermore, the unconstrained handwritten letters can have ambiguous shapes within one class and similar shapes among different classes, when they are isolated from the cursively written word strings. Consequently, there are many unsolved research problems on recognizing unconstrained handwritten letters and these problems require further investigations.

- **Enhancing the reliability**

The demands of the industry applications require the most reliable recognition systems. As for the automatic bank cheque processing, a small error in reading the courtesy amount or the legal amount could lead to a huge financial loss. Moreover, it is very unrealistic to design a handwritten character recognition system with 100% recognition accuracy; even human beings cannot achieve 100% correct accuracy in distinguishing others' cursive handwritings. Hence, the reliability is much more important than the recognition accuracy in real-life systems. However, most papers published on handwritten character recognition have focused on the recognition performance, and only a few of them have discussed the reliability of the recognition system. In this thesis, our research directions are not only to improve the current recognition performance, but also to seek a way to enhance the reliability for the handwritten character recognition systems.

1.3 Previous Work

In this section, recent studies in off-line handwritten character recognition are described. Although many achievements have been made in the handwritten character recognition field, many unsolved issues still exist. For an in-depeth review of offline handwriting recognition approaches and methods, readers may refer to [4, 10, 17].

Recently, many new learning methods have appeared in the pattern recognition field. Most of them have been applied to handwritten digits, and handwritten characters. For example, one of the most popular learning algorithms, Support Vector Machines, has achieved the highest recognition rate on handwritten isolated digits [10] and offline cursive characters [13] when compared with other classical algorithms. Most recognition studies are still based on the following algorithms: Multi-Layer Perceptrons (MLPs), Hidden Markov Models (HMMs), K-Nearest Neighbours (KNNs), Radial Basis Function networks (RBFs), etc. Therefore, the study on improving the performance of individual classifiers is one of the most popular research directions in this field.

Many studies focus on improving recognition rates by adopting different strategies on designing single classifiers. Liu [16] proposed a partial discriminative training scheme for the classification of overlapping classes in handwritten characters. This training scheme was applied to the neural networks, SVM classifiers, and the classifiers trained by the MCE method. Experiments on the offline handwritten letter recognition and online handwritten symbol recognition have demonstrated that the proposed training scheme mostly outperforms ordinary discriminative training and merged metaclass classification when evaluated at the metaclass level. Ranzato et al. [18] introduced an energy-based model for unsupervised learning of sparse overcomplete representations. The model was composed of a linear encoder, and a linear decoder preceded by a sparsifying non-linearity which turned a code vector into a quasi-binary sparse code vector. The proposed unsupervised method was applied to initialize the first layer of the LeNet5 Convolutional Neural Network. The error rate on the MNIST database was reported to be as low as 0.39% with distortions on training samples. Fabien Lauer et al. [19] proposed a system such that a trainable feature extraction was trained by the LeNet5 Convolutional Neural Network and the classification task was performed by Support Vector Machines. Results on the MNIST handwritten digit database showed that the system outperformed both SVMs and LetNet5. Lecun et al. [20] applied several learning methods, including KNN, SVM and CNN, to recognize generic objects, and found that CNN was more efficient and robust than the other classifiers. Milgram et al. [21] compared two strategies for multiclass SVMs: the "one against one" and the "one against all" strategies, by applying several post-processing methods for estimating posterior probability. Experiments were conducted on the handwritten character recognition problems. The “one-against-all” strategy appeared to be more accurate than the “one-against-one” strategy. However, the “one-against-one” strategy was substantially faster in the training process and more suitable for problems with a large scale in the number of classes.

Ensemble learning methods (classifier combinations) have been proposed as a new study direction in order to improve the performance of individual classifiers. In the long run, the combined classifiers are supposedly superior to the best individual classifiers. There are many mathematical reasons for considering an ensemble system, but the intrinsic reason can be elicited from our daily lives [22]. When people make an important decision related to financial, medical or social consequences, they will usually ask different individuals’ opinions, weigh them and combine them after some thought process in order to make a final decision. Ensemble systems follow the exact same process by asking the opinions of different classifiers and combining their outputs with more confidence. The design, implementation and application of ensemble systems are another main focus of research in this field.

Several researchers have investigated on the combination strategy of multiple classifiers. Fu et al. [23] proposed a classifier fusion strategy to train Modified Quadratic Discriminate Function (MQDF) classifiers by using a cascade structure and the maximum rule based fusion was applied in the testing procedure. The experimental results on the handwritten Chinese characters showed that an error reduction of at least 10% was achieved by this method. Suen et al. [24] discussed several combination methodologies for different levels of classifier outputs: the abstract level, a ranked list of classes, and measurements. They concluded that better recognition rates may be accompanied by higher costs in computation requirements, in the quantity of training data, and in the difficulty of the theoretical analysis. A cascade ensemble classifier system, proposed by Zhang in [25], was applied to recognize the MNIST handwritten digits. The results showed that the proposed system can enhance the recognition rate and increase the reliability performance at the same time. Lam et al. [26] implemented a Bayesian formulation and a weighted majority vote was used as a rule to combine seven classifiers. The performances of these combined classifiers were evaluated on handwritten numerals. Nishimura et al. [27] improved the performance of an HMM-based handwritten character recognition system by using the bagging method. Kuncheva [28] has worked on classification combinations since 1990, and has focused on the important and widely studied issue of how to combine several classifiers in order to achieve an improved recognition performance.

Based on these previous studies, we explore the handwritten character recognition problem from two aspects: the individual classification algorithm and the combination technique. Therefore, a hybrid model and a combination system which are based on both the CNN classifier and the SVM classifier are proposed in this thesis. They will be discussed in the next section.

1.4 The Proposal

Improving the accuracy and the reliability for the handwritten character recognition system can be achieved via different processing modules: preprocessing, feature extraction, classifier design, post-processing, etc. In our research, we mainly focus on these issues related to the classifier design. A hybrid CNN & SVM model and a CNN and SVM combination system are proposed.

To boost the recognition rate, feature extraction is one key factor in the success of a recognition system. It requires that features should have the most distinguishable characteristics among different classes and should retain invariant characteristics within one class. There are two categories in feature extraction methods: hand-designed approaches and automatic trainable feature extraction approaches. Since many classifiers cannot process the raw image, most hand-designed features for the handwritten characters consist of statistical features and structural features. As for the automatic extraction methods, features can be learned directly from raw images. One specific example of automatic feature extractors is the trainable feature extraction based on the Convolutional Neural Network described in [19], which showed a high performance for handwritten digit recognition. By using the trainable feature extractor plus elastic distortions or affine distortions, the system obtained the recognition error rates of 0.56% and 0.54%, respectively. Inspired by this particular work, we propose the hybrid CNN & SVM model.

The combination of classifiers is another successful method that has been used to improve the performance of handwriting recognition. However, as far as we know, there is no published paper to date on the comparison between the combination system and the hybrid system. Therefore, this is the first time that the performances of both architectures are compared on the recognition of handwritten characters.

In this thesis, a hybrid CNN & SVM model and a CNN and SVM combination system are proposed for handwritten character recognition. The hybrid CNN & SVM model automatically retrieves features based on the CNN architecture, and recognizes the unknown pattern by using the SVM classifier. In order to assess the performance of such a hybrid model, a combination system of CNN and SVM is investigated. An SVM is trained with hand-designed features, and a CNN model is trained with raw images but with normalized sizes. Then, the two classifiers are combined by using the weighted multiplication combination scheme. The reliabilities of the proposed systems have been achieved through rejecting errors by a certain rejection rule. Finally, to verify the feasibility of our methodology, the MNIST digit database and the NIST character database are tested.

1.5 Thesis Outline

The rest of this thesis is organized as follows:

Chapter 2 provides the theoretical background of the classification algorithms adopted in our research. Specifically, we introduce the preliminary theory of Support Vector Machines (SVMs), including two-class SVMs and multiclass SVMs. Then, the concept and the structure of Convolutional Neural Networks (CNNs) are presented.

Chapter 3 proposes two models for solving handwritten character recognition problems. The first one is the hybrid CNN & SVM model. We describe the architecture of this model and make the analysis of its merits. Next, the second model, the CNN and SVM combination system, is presented. Its structure and the combination method are discussed at the end of this chapter.

Chapter 4 applies proposed models on the handwritten digit recognition problem. Furthermore, the reliability of both systems is considered by using the error rejection method. We test on the MNIST database. The experimental results show that the hybrid model makes the best achievements on both recognition and reliability performance, when compared with others in the literature.

Chapter 5 discusses the unconstrained English letter recognition problem by applying both the hybrid CNN & SVM model and the CNN and SVM combination model. To tackle this complex problem, we classify and recognize unconstrained handwritten letters into three types of classes: the 26-class uppercase problem and the 26-class lowercase problem, the 26-metaclass problem and the 52-class problem. Experiments and analysis are conducted on a subset of the NIST database.

Chapter 6 draws conclusions. The contributions of this thesis are given. The analysis of the proposed methods is discussed. Finally, further research directions are provided.

Chapter 2

Theoretical Background of Classification Algorithms

Our proposed systems were designed based on the SVM and the CNN classifiers. Researchers have demonstrated that both classifiers are superior to other classifier algorithms in handwritten digit recognition, with higher recognition rates when experiments are conducted on the same database. We will firstly introduce the SVM theory and structure used in our experiments in Section 2.1, and the CNN theory and structure in Section 2.2. Then, the hybrid CNN & SVM trainable feature extractor model and the combination system will be described in the next chapter.

2.1 Support Vector Machines

The Support Vector Machine [29] is a statistical method that has shown great success in many practical applications in the pattern recognition field, such as handwritten digit recognition [30], text classification [31], face recognition [32], speech recognition [33], etc. Specifically, Table 1 summarizes some of the most important contributions in the field of handwritten character recognition.

Table 1. SVM performances on handwriting recognition applications found in the literature.

Field	Reference	# Class	Recognition rate (%)	Error rate (%)	Rejection rate (%)	Database
Isolated digits	Teow et al. [34]	10	99.41	0.59	0	MNIST (10,000)
	Liu et al. [10]	10	99.58	0.42	0	MNIST (10,000)
	DeCoste et al. [35]	10	99.58	0.42	0	MNIST (10,000)
	Dong [36]	10	99.62	0.38	0	MNIST (10,000)
	Dong [36]	10	98.70	1.30	0	CENPARMI (2,000)
	Oliverira et al. [37]	10	99.20	0.80	0	NIST- SD 19 (60,089)
Isolated handwritten letters	Milgram et al. [21]	26	96.82	10.07	0.5	NIST- SD 19 Uppercase characters (12,092)
	Dong et al. [38]	26	92.34	7.66	0	NIST- SD 19 Lowercase characters (10,688)
	Liu [16]	26	93.35	6.65	0	C-Cube (19,133)
	Camastra [14]	52	89.20	10.80	0	C-Cube (19,133)
	Camastra [14]	26	89.61	10.39	0	C-Cube (19,133)

Support Vector Machines with different kernel functions can transform a non-linear separable problem into a linear separable problem by projecting data into the feature space and then finding the optimal separate hyperplane. This method was initially proposed to solve two-class problems. Later, a few strategies were suggested to extend this technique to multiclass classification problems. For more details about SVMs, the authors in [39] provide a very good and comprehensive theory. In this section, we will briefly introduce two-class SVMs first, and multiclass SVMs later.

2.1.1 Two-Class Support Vector Machines

Here, we talk about Support Vector machines for two-class problems. First, we introduce the linear Support Vector Machines, which is the training data that are linearly separable in the input space. Second, we discuss the nonlinear Support Vector Machines, which is the training data that can not be linearly separated in the input space and they need to be mapped into the high dimensional feature space via kernel functions.

- ***Linear Support Vector Machines***

Suppose a training set of a binary classification task is given by:

$$S = \{(\vec{x}_i, y_i) \mid (\vec{x}_i, y_i) \in R^m \times R, i = 1, 2, \ldots, l\}$$

where $y_i \in \{-1, 1\}, (i = 1, 2, \ldots l)$. The set S can be linearly separated by a maximum margin hyperplane (3):

$$\vec{w}^{\mathrm{T}} \cdot \vec{x} + b = 0 \tag{3}$$

where $\vec{w}$ is an m-dimensional vector, and b is a scalar. For the linear separable case, $\vec{w}$ and b can be solved by the following optimization problem consisting of (4) and (5) below, which is a "dual" problem of a Lagrangian formulation:

Minimize:

$$W(\vec{\alpha}) = -\sum_{i=1}^{l} \alpha_i + \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j (\vec{x}_i \cdot \vec{x}_j) \tag{4}$$

$$\text{subject to:} \quad \begin{cases} \sum_{i-1}^{l} y_i \alpha_i = 0 \\ 0 \le \alpha_i \le C, i = 1, 2, \ldots l \end{cases} \tag{5}$$

where $\alpha_i (i = 1,...,l)$ is a positive Lagrange multiplier, introduced by the Lagrangian formulation. Vector $\vec{x}_i$ is one of the Support Vectors (SVs) when $0 < \alpha_i \le C$. The $\vec{w}$ and b of hyperplane (3) are calculated below as (6) and (7) by Karush-Kuhn-Tucker (KKT) conditions:

$$\vec{w} = \sum_{\vec{x}_i \in SV_s} \alpha_i y_i \vec{x}_i \tag{6}$$

$$b = \frac{1}{N_{SV_s}} \sum_{\vec{x}_i \in SV_s} b_i \tag{7}$$

where N_{SV_s} is the number of Support Vectors (SVs). The decision function in the test phase is defined as:

$$g(\vec{x}) = sign\{\vec{w}^T \cdot \vec{x} + b\} \tag{8}$$

- ***Nonlinear Support Vector Machines***

Usually, the training data are not linearly separable in the input space, and the solution discussed above is no longer feasible. Therefore, we extend the Support Vector Machines to deal with nonlinear separable data in the input feature space, which is called the nonlinear Support Vector Machines.

For a nonlinear separable case, kernel functions ($K(\vec{x}_i, \vec{x}_j)$) are used without explicitly mapping the training data from a “low dimensional space” to a “high dimensional space.” The optimization problem denoted by (4) and (5) changes to (9) and (10):

Minimize:

$$W(\vec{\alpha}) = -\sum_{i=1}^{l} \alpha_i + \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j K(\vec{x}_i, \vec{x}_j) \tag{9}$$

$$\text{subject to: } \begin{cases} \sum_{i-1}^{l} y_i \alpha_i = 0 \\ 0 \le \alpha_i \le C, i = 1,2,...l \end{cases} \tag{10}$$

The decision function is defined as formula (11) below:

$$g(\vec{x}) = sign\{\sum_{\vec{x}_i \in SV_s} \alpha_i y_i K(\vec{x}_i, \vec{x}) + b\} \tag{11}$$

The common kernels include:

$$\text{Polynomial: } K(\vec{x}_i, \vec{x}_j) = (\vec{x}_i \cdot \vec{x}_j + 1)^d \tag{12}$$

$$\text{Radial Basis Function: } K(\vec{x}_i, \vec{x}_j) = e^{-\sigma \|\vec{x}_i - \vec{x}_j\|^2} \tag{13}$$

$$\text{Hyperbolic tangent: } K(\vec{x}_i, \vec{x}_j) = \tanh(\kappa \vec{x}_i \cdot \vec{x}_j + \delta) \tag{14}$$

Where d, σ, κ and δ are kernel parameters.

2.1.2 Multiclass Support Vector Machines

Now, we have the basic concept of Support Vector Machines on two-class problems. Next, we extend the Support Vector Machines to handle multiclass problems. It is called the Multiclass Support Vector Machines [40].

The problem of the multiclass data is defined in this way. Consider a k-class problem, such that we have l training samples: $\{\vec{x}_1, y_1\},...,\{\vec{x}_l, y_l\}$, where $\vec{x}_i \in R^m, i = 1,...,l$ are m-dimensional feature vectors and $y_i \in \{1,...,k\}$ are the corresponding class labels.

Due to the fact that two-class Support Vector Machines employ direct decision functions, it is not straightforward to apply such functions to the multiclass problem. Thus, some strategies are used. In the following sections, we introduce three typical Multiclass Support Vector Machines which are based on the two-class Support Vector

Machines. They are: one-against-all Support Vector Machines, one-against-one Support Vector Machines, and Directed Acyclic Graph Support Vector Machines.

- ***One-against-all***

The one-against-all method constructs k two-class SVMs where k is the number of classes. The i th SVM is trained with the i th class training data with positive labels, and all the other training data are trained with negative labels. The i th SVM solves the following optimization problem:

Minimize:

$$L(\vec{w}_i, \xi_j^i) = \frac{1}{2}\left\|\vec{w}_i^T\right\|^2 + C\sum_{j=1}^{l}\xi_j^i \tag{15}$$

$$\text{subject to:} \quad \begin{cases} (\vec{w}_i^T)\phi(\vec{x}_j) + b_i \geq 1 - \xi_j^i, y_i = i \\ (\vec{w}_i^T)\phi(\vec{x}_j) + b_i \leq -1 + \xi_j^i, y_i \neq i \\ \xi_j^i \geq 0, j = 1, \ldots, l \end{cases} \tag{16}$$

where training data $\vec{x}_i$ are mapped to a higher-dimensional space by the function $\phi(\bullet)$, and $\xi_j^i, i = 1, \ldots, l$ are the slack variables, while C is the penalty parameter.

In the classification phase, a new data $\vec{x}$ is classified as belonging to the class which has the largest value of the decision function:

$$\arg\max_{i=1,\ldots,k}(\vec{w}_i^T\phi(\vec{x}) + b_i) \tag{17}$$

- ***One-against-one***

The one-against-one method constructs $k(k-1)/2$ classifiers, where each classifier uses the training data from two classes chosen out of k classes. For training data from the i th and the j th classes, we need to solve the following optimization problem:

Minimize:

$$L(\vec{w}^{ij}, \xi_n^{ij}) = \frac{1}{2}\left\|(\vec{w}^{ij})^T\right\|^2 + C\sum_n \xi_n^{ij} \tag{18}$$

$$\text{subject to:} \quad \begin{cases} (\vec{w}^{ij})^T \phi(\vec{x}_n) + b^{ij} \geq 1 - \xi_n^{ij}, y_n = i \\ (\vec{w}^{ij})^T \phi(\vec{x}_n) + b^{ij} \leq -1 + \xi_n^{ij}, y_n \neq i \\ \xi_n^{ij} \geq 0, n = 1, \ldots, k(k-1)/2 \end{cases} \tag{19}$$

The most popular method for the class identification in the one-against-one method is the "max wins" algorithm. Each classifier gives one vote to its determined class, and the final result is determined by the class which wins the most votes, e.g.:

$$\arg\max_i \sum_{j \neq i, j=1}^{k} sign((\vec{w}^{ij})^T g(\vec{x}) + b^{ij}) \tag{20}$$

- ***Directed Acyclic Graph Support Vector Machines***

The training phase for this method is the same as that for the one-against-one method, but its testing takes less time than that of the one-against-one method. In the testing phase, it uses a rooted binary directed acyclic graph which has $k(k-1)/2$ internal nodes and k leaves. Each node is a binary SVM of i th and j th classes. To define the class of one new data $\vec{x}$, we start at the root node. The binary decision function ($g_{ij}(\vec{x})$) at this node is evaluated. If $g_{ij}(\vec{x}) > 0$, we say that $\vec{x}$ does not belong to class j. If $g_{ij}(\vec{x}) < 0$, we consider that $\vec{x}$ does not belong to class i. Then, we move to either the left or the right of the root node depending on the output value of the decision function. This procedure is repeated at each level of the graph. When reaching a leaf, a class is determined for $\vec{x}$.

Figure 1 illustrates an example of a four-class problem. At the top level, class 1 and class 4 are chosen. If $g_{14}(\vec{x}) > 0$, $\vec{x}$ does not belong to class 4. Then we move to its right child where the classification pair is class 1 and class 3. If $g_{13}(\vec{x}) < 0$, $\vec{x}$ does not belong to class 1. Then we go to its left child where the classification pair is class 2 and class 3. If $g_{23}(\vec{x}) > 0$, $\vec{x}$ does not belong to class 3. Finally, we classify $\vec{x}$ into class 2.

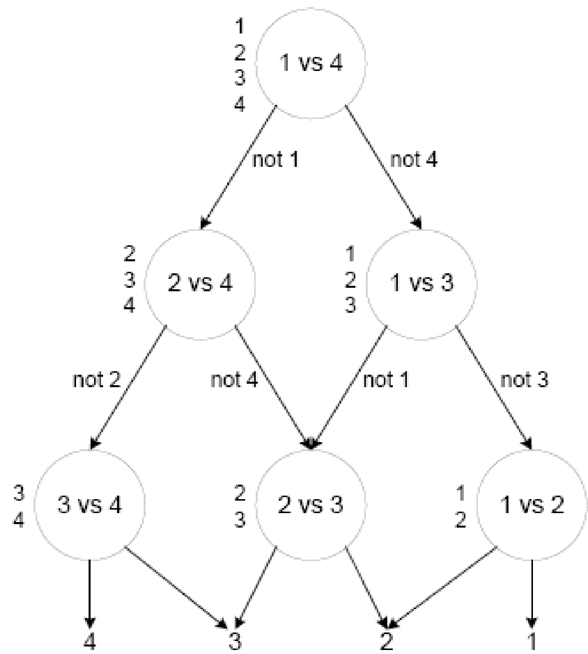


Figure 1. A DAG-SVM for four classes [39].

2.1.3 SVM Library Adopted in the Experiments

LIBSVM [41] has been used to build the SVM models in our experiments. LIBSVM is a simple, easy-to-use, and efficient software for SVM classification and regression. In our experiments, the SVM classifiers were trained to make the predictions with probabilities. Judging on the probability values, post-processing could easily be applied in deciding whether to accept or to reject the candidate. The Radial Basis Function (RBF) was used as the kernel function. The one-against-one approach was applied on the multiclass SVMs.

2.2 Convolutional Neural Networks

In this section, we discuss the Convolutional Neural Networks. In Section 2.2.1, the principle of the CNN and the structure of the classical model – LeNet5 are described.

Then, the structure of the CNN model adopted in our experiments is presented in Section 2.2.2.

2.2.1 Theory of CNN

A Convolutional Neural Network [42] is a multi-layer neural network with a deep supervised learning architecture that can be viewed as the composition of two parts: an automatic feature extractor and a trainable classifier. The feature extractor is composed of alternate convolutional layers and subsampling layers. It retrieves discriminating features from the raw images via two operations: convolutional filtering and down sampling. The convolutional layer has many feature planes. Each unit on one feature plane receives 25 inputs from a 5 by 5 square area in its previous layer. Each square area is called the receptive field. Then, the value of this unit is calculated by multiplying 25 coefficients plus a trainable bias. All the units in one feature plane share the same set of coefficients and the bias. The subsampling layer that follows the convolutional layer comprises the same number of feature planes as the previous convolutional layer, but with half the number of rows and columns. Each unit is connected with a 2 by 2 receptive field of the previous layers. The trainable classifier is composed of one fully connected layer and one output layer. The system is trained by a back-propagation algorithm.

Figure 2 illustrates the typical CNN known as LeNet5 [42]. The input handwritten image, with 32 by 32 pixels, is sent to the first convolutional layer, which has 6 feature planes of 28 by 28 pixels. These planes are reduced into half their sizes with 14 by 14 pixels for one feature plane in the subsampling layer. The next convolutional layer, $C3$, extends the number of feature planes to sixteen. Each unit in each feature plane is connected to several receptive fields at an identical location in a subset of $S2$'s feature planes. Later, in the subsampling layer $S4$, 16 feature planes are reduced to half their sizes, with 5 by 5 pixels. The last convolutional layer has 120 feature planes. Each unit is connected to a 5 by 5 neighborhood on all feature planes of $S4$. The fully connected layer $F6$ contains 84 units connected to the 120 units of $C5$. Like the classical neural networks, units in $F6$ are computed as a dot product between their input vector and trainable coefficients, are added with a bias, and are passed on to a sigmoidal activation function. Finally, the output layer is a Euclidean RBF layer with 10 units. This output layer is used to predict the class label of the input pattern, which is the one with the minimum output value.

Convolutional Neural Networks were designed particularly for recognizing characters. One interesting aspect of Convolutional Neural Networks is that this technique considers the feature extractor as a black box model. The Convolutional Neural Network is fed with almost raw inputs (e.g. size normalized images) and it automatically extracts features. Therefore, this learning algorithm is unlike other classical learning algorithms which need independent hand-designed feature extractors. Furthermore, images have a strong 2D local structure and pixels that are spatially or temporally near each other have high correlations. Convolutional Neural Networks can extract local features through the receptive fields of hidden units. As a result, these features are sensitive to the topology of the input image.

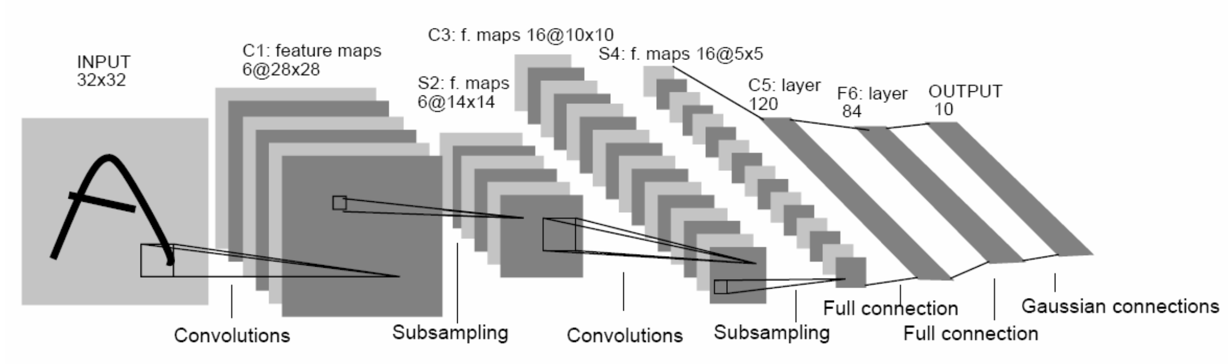


Figure 2. Architecture of LeNet5 [42].

2.2.2 Structure of CNN Model in the Experiments

Instead of using LeNet5, we adopted the same simplified CNN structure that was presented in [43]. In this section, we introduce the simplified structure of the CNN model used in our research.

The architecture of the CNN model is shown in Figure 3. There are five layers. The input layer is a matrix of the normalized and centralized pattern. Two feature map layers (N1 and N2) are used to compute the features, and each layer completes both convolutional filtering and down sampling operations. Each neuron on one feature map connects 25 inputs with its previous layers, and they are defined by the 5 by 5 convolutional filtering kernel (known as the “receptive field”). All the neurons in one feature map share the same kernel and connecting weights (known as the “sharing weights”). The trainable classifier is the fully connected Multi-Layer Perceptron (MLP), with one hidden layer (N3) and one output layer (N4).

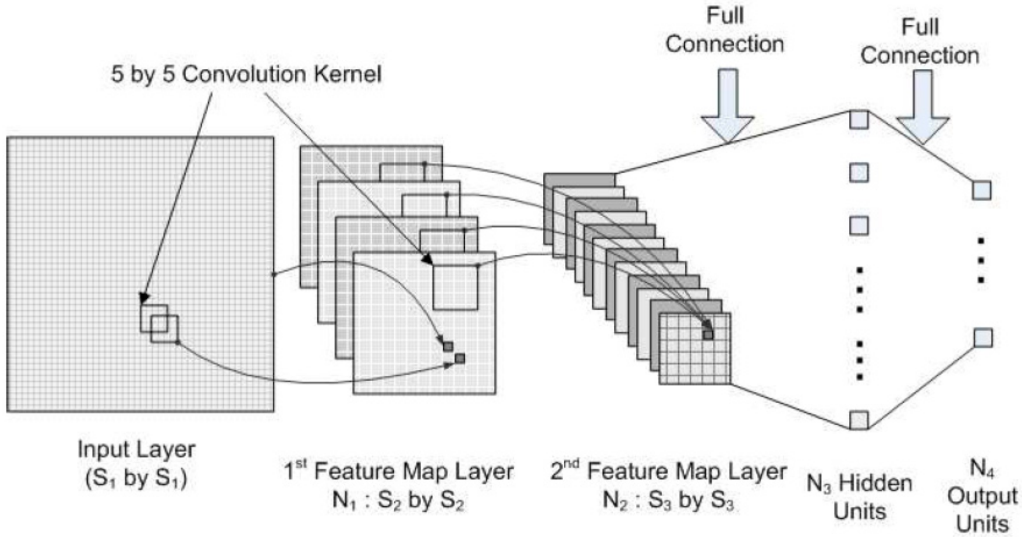


Figure 3. The structure of the adopted CNN model.

Based on the SVM and CNN classifiers, we propose the hybrid CNN & SVM model for recognizing handwritten characters. The proposed model replaces the trainable classifier of a CNN model with an SVM classifier. The details will be described in Chapter 3.

Chapter 3

Hybrid CNN & SVM Model and Combination System

In this chapter, a hybrid CNN & SVM model and a CNN and SVM combination system are proposed for handwritten character recognition. In the hybrid CNN & SVM model, CNN works as a trainable feature extractor and SVM performs as a classifier. This model automatically extracts the features from the raw images and generates predictions. In the CNN and SVM combination system, the CNN classifier is trained with raw images but with normalized sizes, while the SVM classifier is trained with hand-designed features. In the following paragraphs, we describe the hybrid CNN & SVM model in Section 3.1, and discuss the combination CNN and SVM system in Section 3.2.

3.1 Hybrid CNN & SVM Model

There is one specific example of using CNN as a trainable feature extractor which resulted in a high performance on recognizing handwritten digits [19]. Inspired by this method, we propose the hybrid CNN & SVM model. In Section 3.1.1, the structure of the hybrid CNN & SVM model is presented, followed by the analysis of its merits described in Section 3.1.2.

3.1.1 Architecture of Hybrid CNN & SVM Model

The architecture of the hybrid CNN & SVM model was designed by replacing the last layer (N4) of the CNN model (as shown in Chapter 2, Figure 3) with an SVM classifier in the testing phase. For output units of the N4 layer in the CNN network, they are the estimated probabilities for the input sample. Each output probability is calculated by an activation function. The input of the activation function is the linear combination of the outputs from the previous N3 layer with trainable weights, plus a bias term. Looking at the output values of N3 is meaningless, but only makes sense to the CNN network itself; however, these values can be treated as features for any other classifiers.

Figure 4 shows the structure of the hybrid CNN & SVM model. The prediction of the unknown sample is made by an SVM classifier instead of the N4 layer. After the original CNN has been trained by the back-propagation algorithm, the outputs produced from Layer N3 are extracted as the new features. They are sent to the SVM classifier for training. Once the SVM classifier has been well trained, it conducts the recognition task with corresponding features from the testing data. Firstly, the normalized and centred input image is sent to the input layer, and the original CNN with the output unit (N4) is trained with several epochs until the training process converges. Then, the SVM with an RBF kernel replaces the output layer N4. The SVM takes the outputs from the N3 layer as a new feature vector. Finally, the trained SVM makes new decisions on testing images with such automatically extracted features. In our experiments, we set the number for each layer as: N1 = 25, N2 = 50 and N3 = 100.

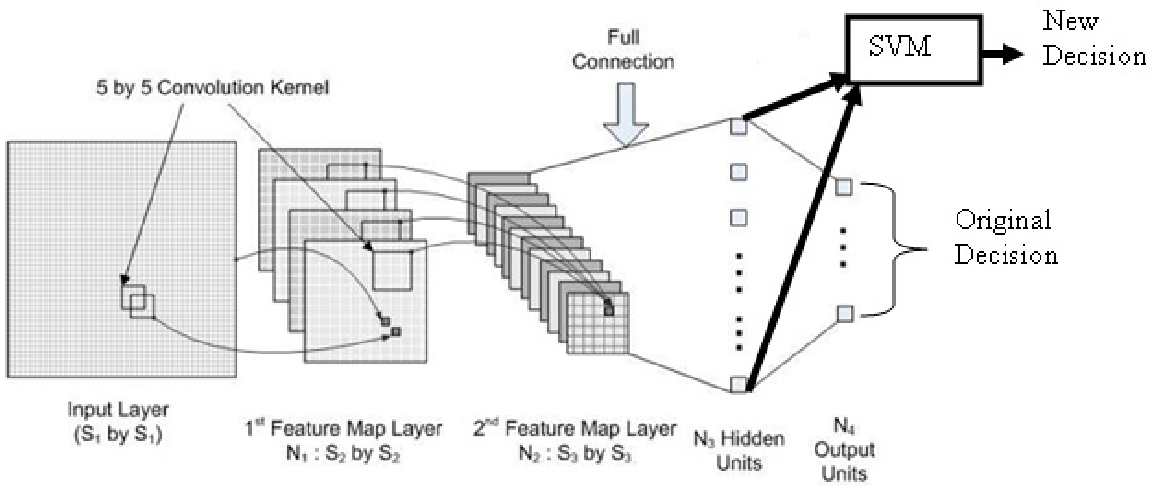


Figure 4. Structure of the hybrid CNN & SVM model.

3.1.2 Merits of Hybrid CNN & SVM Model

In this section, we analyze the advantages of the proposed hybrid CNN & SVM model, through a discussion on the relative merits of the CNN and SVM classifiers.

Our expectation that the hybrid CNN & SVM model will outperform each individual classifier is based on the fact that the hybrid system compensates the limits of the CNN and SVM classifiers by incorporating the merits of both classifiers. Since the theoretical learning method of CNN is the same as that for the Multi-Layer Perceptron (MLP), it is an extension model of the MLP. The learning algorithm of MLP is based on the Empirical Risk Minimization, which attempts to minimize the errors in the training set. When the first separating hyperplane is found by the back-propagation algorithm, no matter whether it is the local or the global minima, the training process stops and the algorithm does not improve the separating hyperplane solution. Therefore, the generalization ability of MLP is lower than that of SVM. On the other hand, the SVM classifier aims to minimize the generalization errors on the unseen data with a fixed distribution of the training set, by using the Structural Risk Minimization principle. The separating hyperplane is a global optimum solution. It is calculated by solving a quadratic programming problem, and the margin area between two classes of training samples reaches its maximum. As a result, the generalization ability of SVM is maximized. Due to the good generalization ability of the SVM, it should enhance the classification accuracy after its replacement of the N4 output units from the CNN.

Another limit of MLP is that it tends to assign a high confidence value to the misclassified samples which are located near the separating boundary. This causes difficulties in rejecting such errors in practical applications. But the SVM classifier calculates a more reliable estimated probability for the classification decision, which helps in the design of a simple and efficient rejection mechanism. Its design will be described in Chapter 4.

The advantage of the CNN classifier is that it automatically extracts the salient features of the input image. The features are invariant in a certain degree to the shift and shape distortions of the input characters. This invariance occurs because CNN adopts the weight sharing technique on one feature map. On the contrary, the hand-designed feature extractor needs elaborately designed features or even applies different types of features to achieve the distortion invariance. Furthermore, the topology of handwritten characters is very important because pixels located near each other in the space have strong connections. The elementary features like the corners, endpoints, etc. are composed of these nearby pixels. CNN uses the receptive field concept successfully to obtain such local visual features. However, the hand-designed feature extraction methods ignore and lose such topology of the inputs in most cases. Therefore, the trainable features of CNN can be used instead of the hand-designed features to collect more representative and relevant information, especially for the handwritten digits.

3.2 Combination of CNN and SVM Classifiers

The combination of classifiers is another successful method that has been used to improve the performance of handwriting recognition. However, as far as we know, there is no published paper to date on the comparison between the combination system and the hybrid system based on the SVM and CNN classifiers. Therefore, this is the first time that the performances of both architectures are compared on the applications of handwritten characters. In Section 3.2.1, we present the architecture of the CNN and SVM combination system. Then, the combination rule is described in Section 3.2.2.

3.2.1 Architecture of Combination System

We implemented a combination system of CNN and SVM to see how well it performs compared with the hybrid CNN & SVM model. The architecture of the combination system is illustrated in Figure 5. In this system, the CNN is trained with normalized images, while the SVM classifier is trained with hand-designed features which will be described in Chapter 4. Then, the two classification results are processed by a combination scheme described below, and this scheme generates a ranked list of predictions for the input image.

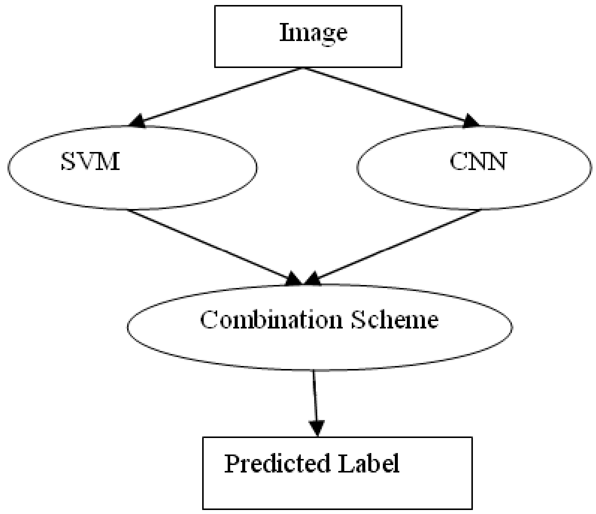


Figure 5. Architecture of CNN and SVM combination system.

3.2.2 Combination Rule

The combination scheme adopted in our experiments is the weighted multiplication method [44], because it generates better results compared with other combination methods, such as majority vote, sum, product, etc. The weighted multiplication method combines the SVM and CNN classifiers by multiplying the probabilities of each model with their corresponding weighting factors. The formula is defined as follows:

$$P(c_i \mid S, C) = P(c_i \mid S)^{w_S} \times P(c_i \mid C)^{w_C}, \qquad i = 1..n. \tag{21}$$

where $P(c_i \mid C)$ is a conditional probability for one class (i), computed from the CNN model; $P(c_i \mid S)$ represents a posterior probability for the same class (i) given by the SVM model; and $P(c_i \mid S, C)$ is the combination probability for the class (i). The weighting factors w_c and w_S are derived from the performances of CNN and SVM, respectively. Finally, a ranked list of candidates is obtained with a decreasing order of probabilities after the combination process. The top candidate is then chosen as the predicted class for the input pattern.

To see the feasibility of the hybrid CNN & SVM model and the combination system, they are applied on the handwritten character recognition applications. The handwritten characters include handwritten digits and handwritten alphabetical letters in this thesis. Thus, the experiments on handwritten digit recognition are described in Chapter 4, and the experiments for recognizing unconstrained handwritten letters in the English language are presented in Chapter 5.

Chapter 4

Hybrid and Combination of CNN and SVM Classifiers for Recognizing Handwritten Digits

In this chapter, the proposed hybrid CNN & SVM model and the combination CNN and SVM system are applied on the recognition of handwritten digits. To verify the feasibility of our methodology, we tested it on the MNIST digit database. Here, our goals are not only to improve the current recognition performance but also to seek the highest reliability on the applications of handwritten digits.

The rest of this chapter is organized as follows: the hand-designed features for training the SVM classifier are described in Section 4.1. Experimental results and the analysis on the hybrid CNN & SVM model and on the combination system are illustrated in Section 4.2. Section 4.3 compares the differences between machine recognition and human classification on the MNIST database. Conclusions are drawn in Section 4.4.

4.1 Feature Extraction for the SVM Model

To achieve a good performance from an SVM classifier, we extracted different types of hand-designed features, and concatenated them together. The extracted features included the gradient feature, the distance feature, and the chain feature [45], which have been proven by researchers to be efficient in recognizing handwritten characters [10]. These features are described below.

- ***Gradient feature***

Gradient features are among the most effective features in character recognition [46, 47]. First, we applied the Sobel operator to calculate the gradient vector for each pixel in the input image. Then, the input pattern was partitioned into 6 × 4 zones. We calculated the local orientation histograms by decomposing those gradient vectors into four equally spaced standard directions, starting from 0 degrees. If a gradient vector was lying between two standard directions, it was decomposed into two components in the two standard directions, as shown in Figure 6. The gradient feature vector was built by concatenating these local orientation histograms and normalizing all the values to [0, 1]. In total, we had 96 attributes in a gradient feature vector.

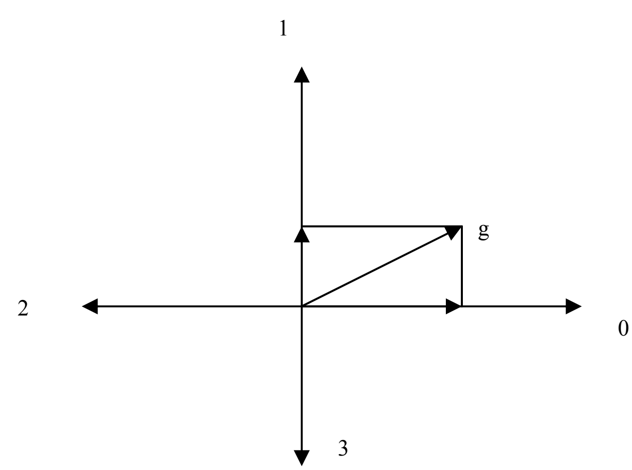


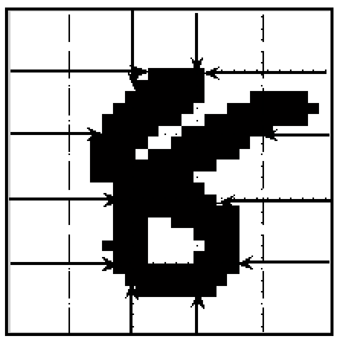
Figure 6. Decomposition of a gradient vector into two standard directions.

- ***Distance feature***

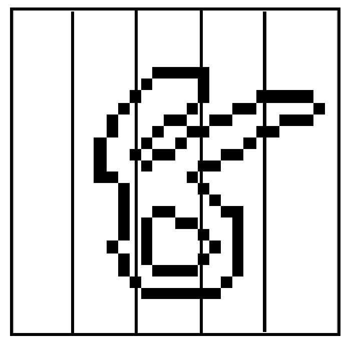
The distance feature records the distance from each of the four boundaries of the binary image to the first black pixel of the character, as shown by the arrows in Figure 7(a). The number of samplings on each boundary in our experiments was 17, so the dimension of the distance feature vector was 68.

- ***Chain feature***

The chain feature counts the number of same directional pixels in each of the subdivided local regions on the contour image. The divided contour image is shown in Figure 7(b). Eight directions and 16 divisions were considered in our experiments; as a result, the dimension of the chain feature vector was 128.



(a) Distance feature for a binary image



(b) Subdivision of contour image for chain feature

Figure 7. Different feature extraction methods.

In total, there were 292 variables (96 + 68 + 128) in one feature vector, computed from each of the three above-mentioned features, respectively.

4.2 Experiments

To evaluate which method would be more effective, the hybrid versus the combination system, we conducted experiments on a public handwritten numeral dataset known as MNIST. MNIST is a handwritten digit dataset that researchers use as a benchmark. It contains 60,000 training samples and 10,000 testing samples. It is a subset originally belonging to the NIST dataset. The images in the MNIST dataset are grayscale numeral bitmaps that have been centred and size normalized to 28 ×28 pixels. These images were downloaded from [5]. Some samples in this database are illustrated in Figure 8.

The experiments are discussed in this way: Section 4.2.1 presents the results by using the hybrid CNN & SVM model without rejection; and the experiments applied by the combination system without rejection are discussed in Section 4.2.2. To find the reliability performance of our proposed models, we analyze each system's reliability by introducing a rejection mechanism. The details are presented in Section 4.2.3. In Section 4.2.4, the complexity of the proposed hybrid model is compared with the SVM classifier and the CNN classifier on the MNIST testing dataset.

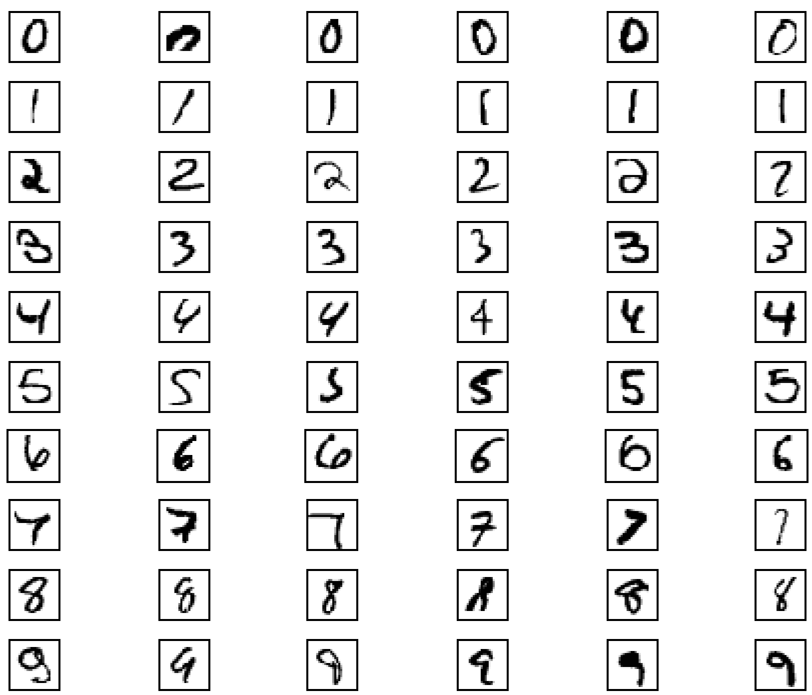


Figure 8. Sample images in MNIST database.

4.2.1 Experiments on Hybrid CNN & SVM Model

When training the CNN network, we used the MNIST dataset directly without any preprocessing. However, previous researchers [18, 19, 48] have proven that better generalization can be achieved with an expanded training dataset by using distortion techniques. In our experiments, Simard's elastic distortions [48] with scaling and rotation transforms were applied in the CNN training phase. The training procedure was stopped after 500 epochs as it converged to a fixed value (around 0.28), as shown in Figure 9. The recognition error on the testing dataset was 0.59% by this CNN learning classifier.

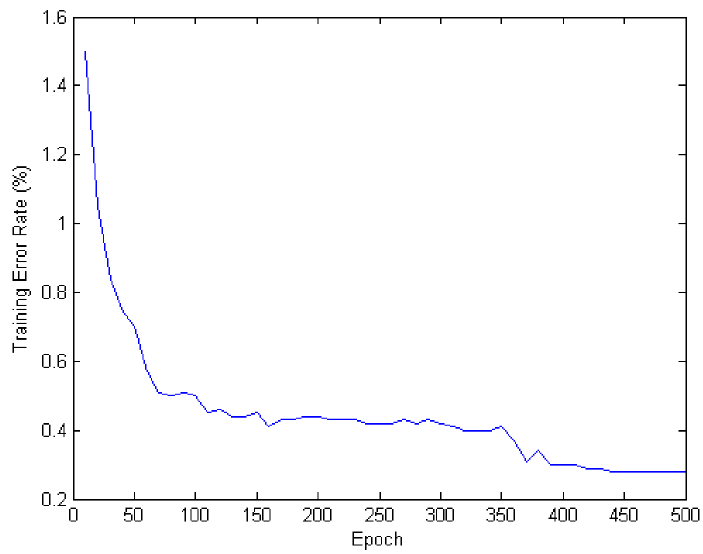


Figure 9. The trend of training error rates of CNN on the MNIST dataset.

Next, the hybrid CNN & SVM model was built and trained. It adopted an SVM classifier to make decisions instead of using the last fully connected layer of CNN to predict labels. One hundred values from the layer N3 of the trained CNN network were used as a new feature vector to represent the input pattern, and were fed to the SVM for learning and testing. When training the SVM, we used the RBF kernel and chose the optimal parameters ($C = 128$ and $\sigma = 2^{-11}$) by using the 5-fold cross validation method on the training dataset. In the testing phase, we achieved an error rate of 0.19% on the 10,000 testing data, which corresponds to the highest recognition rate to date as far as we know. Comparisons with other results published on the MNIST dataset are listed in Table 2.

Table 2. Comparison of testing results on MNIST dataset.

Reference	Method	Distortion	Error rate (%)
F. Lauer et al. [19]	TFE-SVM	Affine	0.54
P.Y. Simard et al. [48]	Convolutional NN	Elastic	0.40
M. Ranzato et al. [18]	Convolutional NN	Elastic	0.39
D.C. Ciresan et al. [11]	6-layer NN	Elastic	0.35
Y. Mizukami et al. [49]	KNN	Displacement computation	0.57
J. X. Dong et al. [30]	VSVM[a]	-	0.44
X. Chen et al. [50]	Gaussian Mixture Model	-	0.53
This thesis	Hybrid CNN & SVM	Elastic Scaling Rotation	**0.19**

Figure 10 shows all of the 19 misclassified samples, and Table 3 indicates the confusion matrix. From these error cases, we found that they can be categorized into two types: (1) the most frequent confusing pairs, in this case "4-9" and "5-3", which have similar shapes and structures due to people's cursive writing habits. For example, when we closely examine the image "2131.tif" in Figure 10, even human eyes cannot distinguish whether it is a "4" or a "9" without its true label; (2) the degraded quality of digit images, such as missing strokes ("2655.tiff", "3423.tif", "3559.tif"), broken numerals ("6572.tif"), intruder noises ("3226.tif", "5655.tif") and stroke connections ("948.tif", "9730.tif"). These cases could be caused by people's poor handwritings, or introduced by the scanning procedure, the size normalization, and improper segmentations. For the second error category, it is extremely difficult for a machine to make a correct prediction with such ambiguous and degraded inputs.

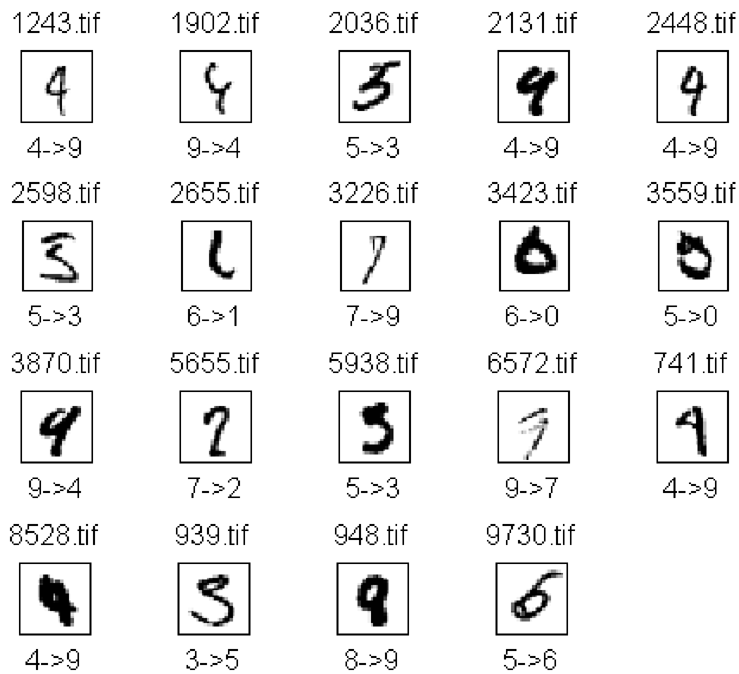


Figure 10. Nineteen digit images were misclassified by the hybrid CNN & SVM model. The upper title is the name of the image in the MNIST testing dataset, and the lower subtitle indicates the corresponding labels (truth -> prediction).

Table 3. Confusion matrix of the hybrid model on the MNIST testing dataset.

Prediction Truth	0	1	2	3	4	5	6	7	8	9
0										
1										
2										
3						1				
4										5
5	1			3			1			
6	1	1								
7			1							1
8										1
9					2			1		

4.2.2 Experiments on the Combination of CNN and SVM

We built the CNN network as mentioned earlier, in order to combine it with another expert; we trained an SVM classifier. First, the grey-level digit images were transformed into binary images for the convenience of the feature extraction step. Then, three types of features, including gradient, distance and chain features, were retrieved and concatenated as described in Section 4.1. Next, these hand-designed features extracted from the training dataset were sent to train the SVM classifier. The optimal parameters of SVM were also generated by 5-fold cross validation ($C = 32$ and $\sigma = 2^{-5}$). Finally, we got the error rate of 1.69% on the testing dataset, which was not the best result obtained by using SVM when compared with other researchers' results [5]. However, our aim in this section is to investigate the performance of the combination of classifiers, and the most important emphasis is on the diversity of each individual classifier while not requiring each classifier to be the best one.

Then, the SVM was combined with CNN by using the weighted multiplication rule. During the experiments, we noticed that the optimal combination result could not be achieved by directly using the recognition performance of each classifier. Therefore, weighting factors were determined by an exhaustive search in the range of (0, 1), in steps of 0.1 increments. As a result, when $w_c = 1$ and $w_s = 0.2$, the testing error was 0.54%, which was the optimal combination recognition rate. In fact, we also tried other combination methods, such as the majority vote, the sum, the max, etc., but their performances were inferior to the weighted multiplication method. Hence, we chose the weighted multiplication combination rule and present its result here.

Furthermore, we combined the hybrid and the combination systems. To distinguish such a new combination, we denoted it as the ensemble system. The ensemble architecture also adopted the weighted multiplication combination scheme. After exhaustive searching of optimal weights, the best classification result achieved a 0.20%

error rate, when $w_{Hybrid} = 1$ and $w_{Comb} = 0.1$. Here, w_{Hybrid} represents the weight of the hybrid CNN & SVM system and w_{Comb} represents the weight of the combination CNN and SVM system, respectively. Table 4 shows the confusion matrix of the ensemble system. When comparing Table 3 and Table 4, we noticed that the only difference appears in the pair "4->9", where one more testing digit "4" is misrecognized as digit "9" in the ensemble system.

Table 4. Confusion matrix of the ensemble system on the MNIST testing dataset.

Prediction / Truth	0	1	2	3	4	5	6	7	8	9
0										
1										
2										
3						1				
4										6
5	1			3			1			
6	1	1								
7			1							1
8										1
9					2			1		

For comparative purposes, we summarized the recognition error rates on the MNIST testing dataset in Table 5 by using different classification strategies. The table obviously shows that the combination can enhance the classification ability due to the diversity of each individual recognizer. However, such an improvement has its limitations, and a significant achievement was made by the hybrid method and the ensemble system. The highest classification performance was achieved by the hybrid CNN & SVM with an error rate of 0.19%. In this case, we conclude that the hybrid architecture has a superior recognition performance to the combination technique on the MNIST digit database.

Table 5. Test error rates (%) of classifiers on MNIST dataset without rejection.

SVM	CNN (distortion)	Combination CNN and SVM	Hybrid CNN & SVM	Ensemble system
1.69	0.59	0.54	0.19	0.20

4.2.3 Reliability Performances

Industrial applications require the most reliable system. Small errors can cause large mistakes and introduce tedious human labour to correct them. Hence, it is necessary to investigate reliabilities of the proposed systems. In this subsection, we worked on the rejection mechanism, and showed that the proposed systems can achieve a 100% reliability rate while maintaining a reasonably high recognition rate at the same time.

In our experiments, the test sample is rejected by measuring the difference between the top two confidence values in the ranked predictions. If the difference is less than a predefined threshold, then the test pattern is rejected; otherwise it is accepted. To determine the threshold, we applied two methods: one uses the exhaustive search in the range of (0, 1) with certain steps; the other calculates the mean distance between the top two confidence values of all the training data [51]. We applied both methods on the hybrid system and the ensemble system.

In the following subsections, the details for the hybrid model with rejection are described in Section 4.2.3.1, followed by the discussion of the ensemble system with rejection in Section 4.2.3.2.

4.2.3.1 The Hybrid Model with Rejection

Table 6 lists the recognition and reliability rates of the hybrid CNN & SVM system, with different threshold values. In the first part, the threshold increases its value within (0, 0.9) with an incremental step 0.1, and within (0.9, 1) with an incremental step 0.01.

We noticed that when the value of the threshold increases, the recognition rate decreases while the reliability rate increases. The reliability rate reaches 100% with a high recognition rate of 94.40% when the threshold is equal to 0.99. In the last row of Table 6, the result is generated by the mean distance method. The threshold is 0.996105. The reliability rate is 100%, but the recognition rate is 85.52% which is lower than the one obtained by the threshold equal to 0.99.

Table 6. Recognition rates of the hybrid system with rejection.

Threshold	Recognition(%)	Rejection(#)	Error(#)	Reliability(%)
0.0	99.81	0	19	99.81
0.1	99.77	5	18	99.82
0.2	99.76	9	15	99.85
0.3	99.76	12	12	99.88
0.4	99.76	13	11	99.89
0.5	99.72	17	11	99.89
0.6	99.70	21	9	99.91
0.7	99.66	26	8	99.92
0.8	99.61	32	7	99.93
0.9	99.22	75	3	99.97
0.91	99.10	87	3	99.97
0.92	98.95	103	2	99.98
0.93	98.79	120	1	99.99
0.94	98.61	138	1	99.99
0.95	98.37	162	1	99.99
0.96	98.08	191	1	99.99
0.97	97.58	241	1	99.99
0.98	96.73	326	1	99.99
0.99	**94.40**	**560**	**0**	**100.00**
Threshold (mean distance)				
0.996105	85.52	1448	0	100.00

4.2.3.2 The Ensemble System with Rejection

The results of the ensemble system with rejection are shown in Table 7. The structure of Table 7 is the same as that of Table 6. From the table, we found that when the threshold equals 0.97, the reliability rate is 100% and the recognition rate reaches as high as 92.84%. When the threshold is calculated by the mean distance, the value is 0.973471. The reliability also achieves a rate of 100%, but the recognition rate drops to 89.61%.

After carefully examining Table 6 and Table 7, we observed that the hybrid model outperforms the ensemble system under the same reliability rates in general. When both systems firstly reached zero-level errors, the rejection rate was 5.60% for the hybrid system and 7.16% for the ensemble system. Figure 11 shows the error and rejection rates of the hybrid system and the ensemble system. It is obvious that the hybrid system provides lower rejection rates than the ensemble system under the same amount of errors. Therefore, the hybrid system is really a robust learning model for recognizing handwritten digits.

As for the two decision methods of the threshold, the second one (mean distance) can automatically calculate the value and make both systems achieve 100% reliability, but the recognition rate is not quite as satisfying. While the first method can generate a better recognition rate, it is time consuming in searching the best threshold value. Hence, how to compute an optimal threshold to achieve the most satisfactory performance really depends on the requirements of practical applications.

Table 7. Recognition rates of the ensemble system with rejection.

Threshold	Recognition(%)	Rejection(#)	Error(#)	Reliability(%)
0.0	99.80	0	20	99.80
0.1	99.76	6	18	99.82
0.2	99.76	8	16	99.84
0.3	99.76	11	13	99.87
0.4	99.73	16	11	99.89
0.5	99.71	19	10	99.90
0.6	99.66	26	8	99.92
0.7	99.59	33	8	99.92
0.8	99.33	61	6	99.94
0.9	98.49	150	1	99.99
0.91	98.28	171	1	99.99
0.92	98.03	196	1	99.99
0.93	97.77	222	1	99.99
0.94	97.29	270	1	99.99
0.95	96.52	347	1	99.99
0.96	95.47	452	1	99.99
0.97	**92.84**	**716**	**0**	**100.00**
0.98	47.97	5203	0	100.00
0.99	0.04	9996	0	100.00
Threshold (mean distance)				
0.973471	89.61	1039	0	100.00

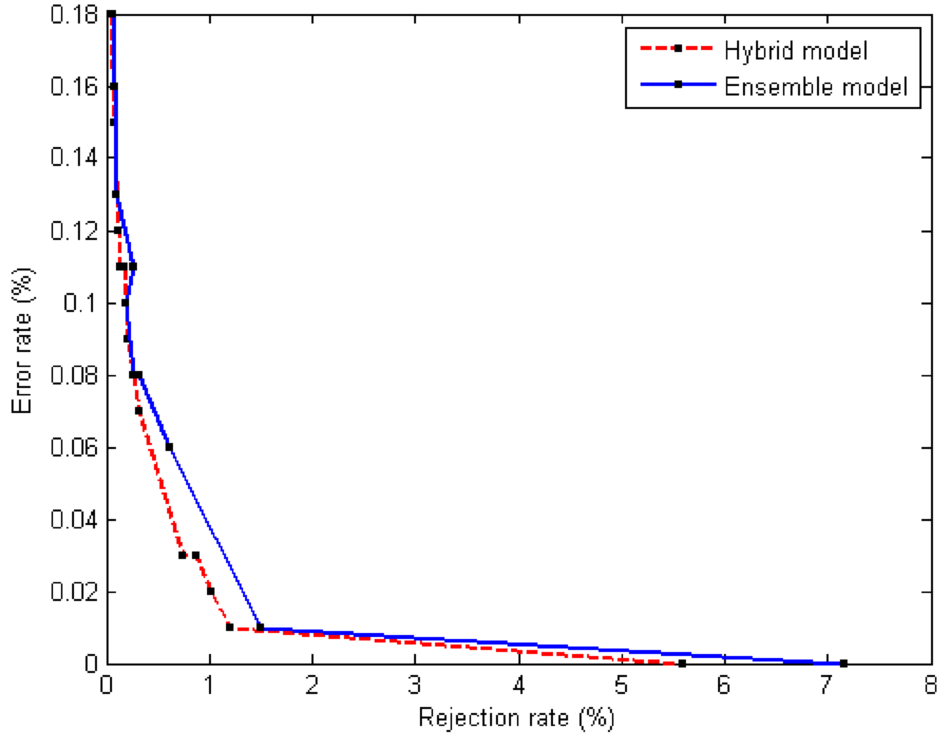


Figure 11. Error-reject analysis of hybrid model and ensemble system for the MNIST testing dataset.

4.2.4 Comparing the Complexity of Individual Classifiers

The complexity analysis was conducted on the Red hat operating system, which is an open source software based on Linux. It was installed on a PC with Intel Pentium D CPU 3.40GHZ, and 4.00GB of RAM.

To discuss the complexity of the proposed hybrid model, we compared it with SVM and CNN classifiers on the testing process. Three factors were considered: testing speed, memory usage, and the number of Support Vectors (SVs). We ignored the analysis of classifiers on the training speed, because it is obvious that the training speed of the hybrid model roughly equals the summation of that on training a CNN classifier and an SVM classifier separately.

Table 8 shows the complexity comparison among the hybrid model, the SVM classifier and the CNN classifier on the 10,000 MNIST testing samples. The number of SVs plays an important role in analyzing the complexity of an SVM classifier. It reflects the size of the weight model of an SVM, and directly influences the speed of the decision procedure. From Table 8, we observed that the hybrid model had nearly one sixth the number of SVs when compared with the SVM classifier; while the testing speed of the hybrid model on 10,000 samples was more than 10 times faster than SVM, and the memory usage was less than 10 times that of SVM in space. The complexity of the CNN was in the middle of the three classifiers. Therefore, in conclusion, even the hybrid model needs more time in the training process, it runs faster, requires little memory space, and has higher generalization ability than other single classifiers in our experiments.

Table 8. Comparison of three classifiers in terms of complexity on the 10,000 digits in the MNIST testing dataset.

	SVM	CNN	Hybrid CNN & SVM model
Total testing speed (Seconds)	90	50	8
Memory usage (MB)	13.2	1.6	1.3
# SVs	5,991	_	1,034

4.3 Human Classification versus Machine Recognition

In this section, we will compare and discuss the differences between humans and machines in the recognition of handwritten digits. Figure 12 summarizes the percentage of human participants with the correct classifications on the 45 digit images when using the labels provided by MNIST. This survey was conducted at CENPARMI with 30 people participating in 2004 [52]. After investigating the errors made by the five classifiers: GPR, VSVM[b], VSV2, LeNet5, and POE [53], 45 digits were selected as the most difficult numerals for machines to recognize.

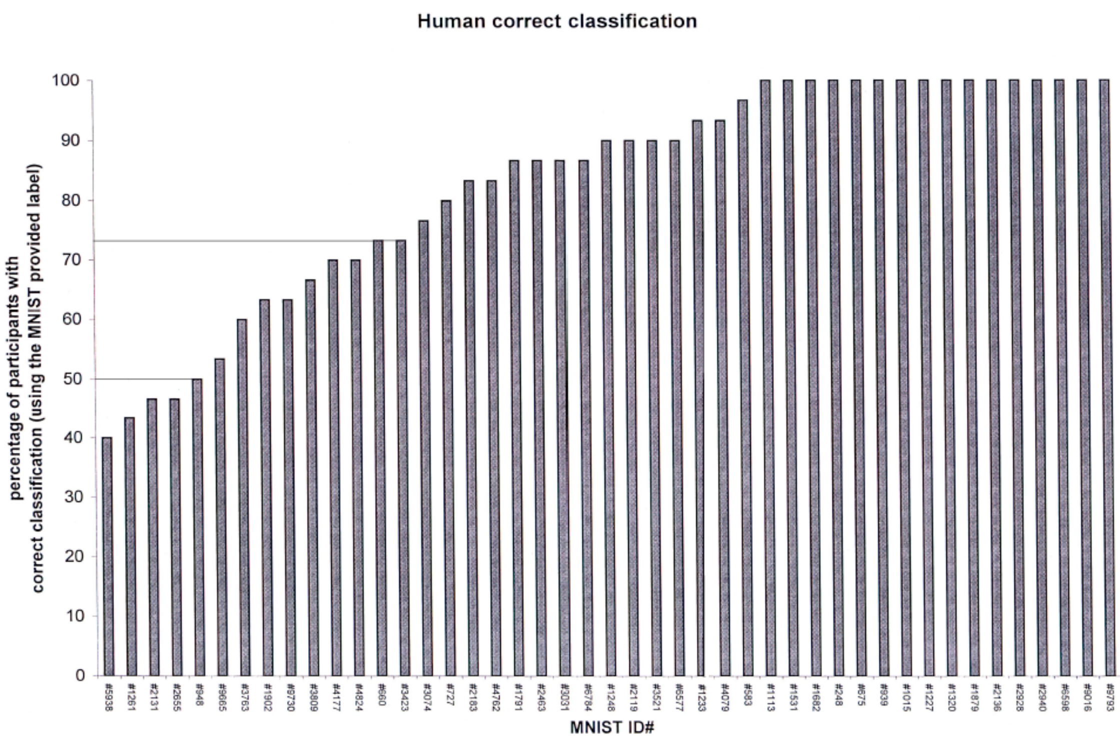


Figure 12. The percentage of correct classification by humans on 45 testing images, using the MNIST provided labels [52].

From Figure 12, we noticed that 16/45 digit images were correctly classified by all the participants; and the rest (29 / 45 digit images) could be recognized by only a portion of people. There were seven common errors in total, made by both humans and our proposed hybrid model without rejections. The ID numbers of those common errors are: #5938, #2131, #2655, #948, #1902, #9730, and #3423. All of them could be correctly recognized by less than 72% of participants. For the first five digits in Figure 12 (from left to right) that could be correctly classified by less than half of participants, four of them were misrecognized by the hybrid model. In this case, the handwritten digits that cause difficulty in being recognized by the majority of people can also cause difficulty in being correctly classified by the machine.

There is one special case such that image #939 is correctly recognized by 100% of participants while the hybrid model misclassifies this digit. When examining this image (in Section 4.2.1, Figure 10), we see that a stroke is connected in the upper part of the image and makes a loop. Even for humans with no difficulties in identifying it, this stroke causes a big problem for the machine. The reason for the misrecognition by the hybrid model might be due to the lack of training samples with similar stroke structures. To solve this problem, one way is to import more unseen training samples into the database, and the other way is to use the rejection mechanism to reject it.

4.4 Conclusion

We presented a hybrid system and a combination system by using CNN and SVM for the recognition of handwritten digits. The hybrid CNN & SVM system took the CNN as an automatic feature extractor and it allowed SVM to be the output predictor. The combination system combined CNN and SVM classifiers by using a weighted multiplication rule. The experiments were conducted on the MNIST digit database. Experimental results showed the benefit of the proposed hybrid model. It had a 0.19% testing error rate without rejection, and reached a high recognition rate of 94.40% pertaining to a 100% reliability rate with rejection. This was the first time that the hybrid system and the combination system of CNN and SVM classifiers were compared on the handwritten digit database.

Chapter 5

Applying Hybrid and Combination of CNN and SVM Classifiers to Recognize Cursive English Letters

The goal of this chapter is to show that the proposed hybrid models can be easily applied to the unconstrained handwritten letter recognition. First, we briefly introduce the background of this problem. Then, we present possible solutions. Next, we describe the experimental results. Finally, the discussion and conclusions are drawn.

5.1 Introduction

Recognition of unconstrained cursive characters is an important and challenging problem and has been intensively studied in the field of handwriting recognition. Nowadays, many researchers work on document analysis by manipulating very large vocabularies, which are mostly based on the recognition of handwritten cursive characters. However, the unconstrained handwritten character recognition problem is much more complicated compared to the constrained recognition of letters for specific writing styles, such as uppercases or lowercases. For the recognition of unconstrained handwritten characters, 52 English characters (uppercases (A-Z) and lowercases (a-z)) have numerous ambiguous shapes both within and among the classes. For example, in Figure 13, some letters, like lowercases “o”, “x”, and “w”, are written cursively in identical shapes of their corresponding uppercases. Some other cursive characters, such as “e” and “l”, “i” and “l”, “u” and “n”, have similar shapes when they are isolated from the handwritten words.

To find a better solution on recognizing unconstrained handwritten letters, this chapter investigates three strategies of categorizing the handwritten letters into different classes. Experiments were conducted on the NIST-SD19 database. The results show that satisfying performance can be achieved by our proposed models.

The rest of this chapter is organized as follows: the details of our system design are presented in Section 5.2. In Section 5.3, the preprocessing of size normalization on the cursive letters is described. Experimental results and comparisons are illustrated in Section 5.4. The discussion and analysis are presented in Section 5.5. Finally, we draw the conclusion in Section 5.6.

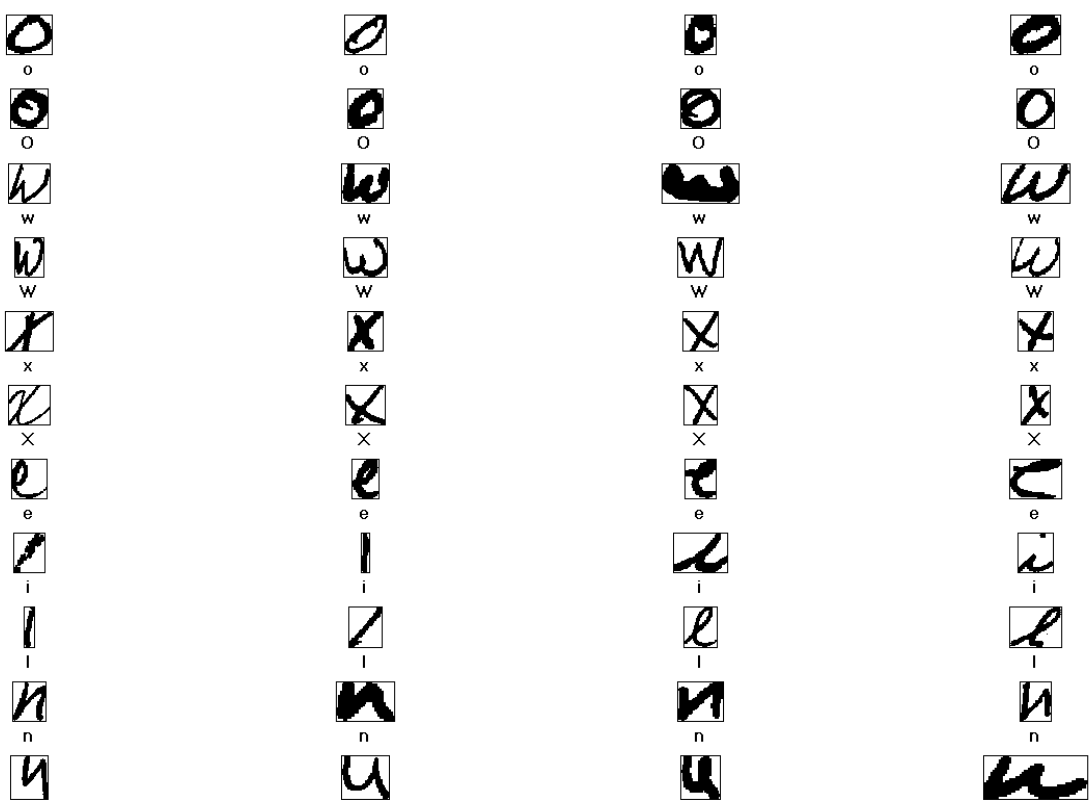


Figure 13. Cursive letter examples with similar shapes (true labels are under the images).

5.2 The Design of Classifiers

The aim of this chapter is to explain our method of recognizing unconstrained handwritten letters, so we designed three classification strategies to handle this problem. The handwritten letters are classified into three categories of classes. They are: a 26-class problem in uppercases and a 26-class problem in lowercases, a 26-metaclass problem, and a 52-class problem. These categories are defined as follows:

- a 26-class problem in uppercases and a 26-class problem in lowercases: two classifiers are trained respectively, one with 26 uppercase outputs and the other with 26 lowercase outputs.
- a 26-metaclass problem: the uppercase and the lowercase are merged into one class which is called a "metaclass". For example, {A, a} belongs to one class. The 26-metaclass recognizer generates 26 outputs.
- a 52-class problem: the uppercase and the lowercase represent distinct classes. For example, {a} denotes class one and {A} denotes class two. There are 52 outputs for the 52-class recognizer.

5.3 Preprocessing

In this section, the size normalization method is described in details.

The isolated cursive English letters in the NIST-SD19 database are available in different sizes, so they need to be size normalized before the next processing steps. One reason lies in the training requirement of the CNN classifier: each input image should be in a standard size. The other reason is for the fair and facilitative comparisons between two feature extraction methods: the hand-designed features and the automatically trainable features.

In the size normalization technique adopted in this research, geometric moments play an important role. The geometric moments are defined as follows: Let $f(x, y)$ be a real function of two variables for the image. Its $(p,q)th$ order geometric moment $m_{p,q}$ is defined as:

$$m_{p,q} = \iint x^p y^q f(x, y) dxdy \tag{22}$$

With geometric moments: $m_{0,0}, m_{0,1}$, and $m_{1,0}$, we can calculate the centroid (centre of gravity) as:

$$c_x = m_{1,0} / m_{0,0} \tag{23}$$

$$c_y = m_{0,1} / m_{0,0} \tag{24}$$

Furthermore, we can calculate the centred geometric moments $u_{p,q}$ as:

$$u_{p,q} = \iint (x - c_x)^p (y - c_y)^q f(x, y) dxdy \tag{25}$$

The basic idea of the moment-based size normalization technique [46] is to first shift the given pattern so that its centroid (c_x, c_y) coincides with its geometric centre. Then, the pattern is reframed with a rectangular R region having a dimension of $[c_x - 2\sqrt{\mu_{2,0}}, c_x + 2\sqrt{\mu_{2,0}}] \times [c_y - 2\sqrt{\mu_{0,2}}, c_y + 2\sqrt{\mu_{0,2}}]$. Finally, the rectangular R is mapped linearly to the target pattern plane with a fixed size. During this mapping, we explicitly maintain the aspect ratio of the input pattern. Figure 14 illustrates this size normalization procedure.

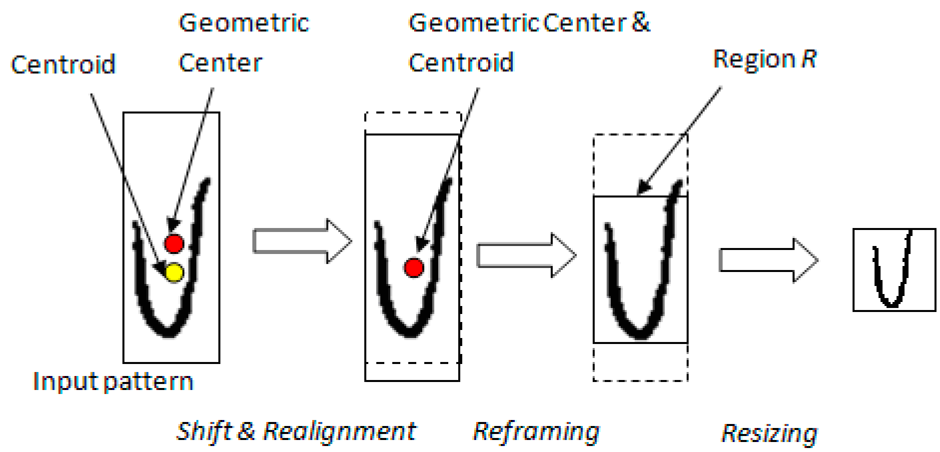
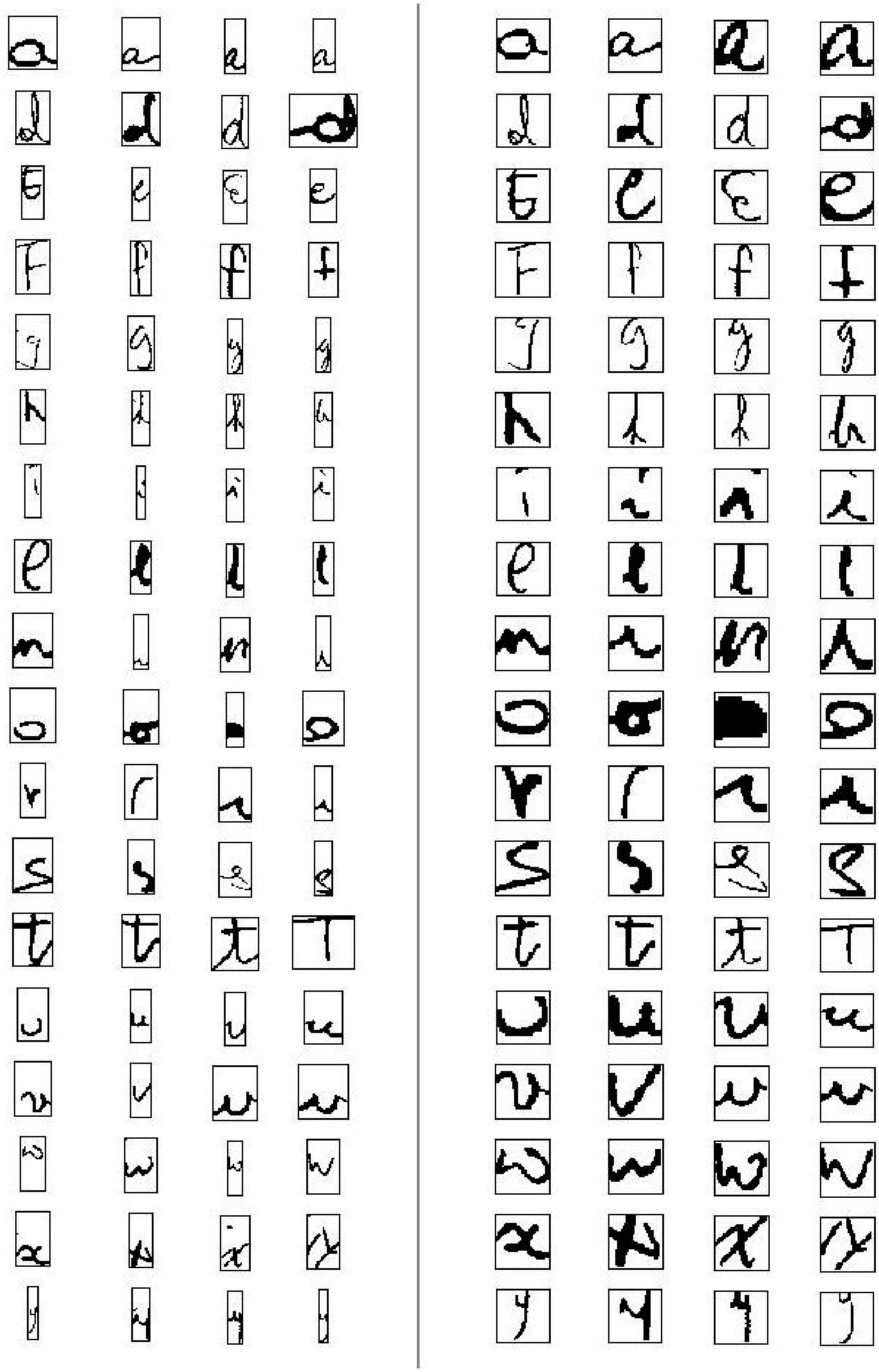


Figure 14. Demonstration of moment-based normalization procedure.

The moment-based size normalization method provides a good performance in handwritten character recognition. By shifting the pattern's centroid, the variations of stroke positions among samples are largely decreased. At the same time, this technique cuts the tails of some elongated strokes and retains most of the classification-related information in the pattern.

The size of each handwritten letter image was normalized to 48 × 48 pixels by using the moment-based normalization method. Figure 15 shows some examples of the isolated characters, e.g. before preprocessing in Figure 15(a) and after size normalization in Figure 15(b).



(a) (b)

Figure 15. Examples of isolated characters: (a) before and (b) after preprocessing.

After image preprocessing, features were retrieved for training the SVM classifier. The concatenation of features, extracted from normalized cursive letter images, was the same process as that applied on the handwritten digits, including the same features: the gradient feature, the distance feature, and the chain feature that were described in Chapter 4, Section 4.1. As for training the CNN classifier, the normalized images were used directly.

5.4 Experiments

We conducted the experiments on the NIST-SD19 database [8], which is a public database of isolated cursive letters and handwritten digits. English cursive characters were segmented from binary images of Handwriting Sample Forms, including uppercases and lowercases.

There are three sets in the NIST-SD19 database: ‘hsf_0123”, “hsf_4” and “hsf_7”. For making the comparison easier with other research results, we followed Cavalin’s [54] way of taking “hsf_0123” as the training set and “hsf_7” as the testing set. The total number of letters in the “hsf_0123” set is 339,248, composed of 184,033 uppercases and 155,215 lowercases. The testing set “hsf_7” has 23,670 alphabetic letters, with 12,092 uppercases and 11,578 lowercases. All the letters in the “hsf_7” set were used for the testing samples. However, due to the huge number of samples in the set “hsf_0123”, we just selected parts of data from the “hsf_0123” set in order to speed up the learning process, and to provide a reasonable proportion of the training set samples to the testing set. The number of training and testing samples for building recognizers according to the three classification strategies mentioned in Section 5.2 are as follows:

- The 26-class uppercase recognizer and the 26-class lowercase recognizer:

- The 26-class uppercase recognizer: 1,660 uppercase letters are selected from each class in “hsf_0123”, which creates a total of 43,160 training data. The number of testing data in uppercases is 12,092 in “hsf_7”.
- The 26-class lowercase recognizer: 1,440 lowercase letters are taken from each class in “hsf_0123”, which creates a total of 37,440 training samples. The number of lowercase letters for the testing data is 11,578 in “hsf_7”.

- The 26-metaclass recognizer and the 52-class recognizer: both recognizers are trained and tested on the same data set. One fourth of the data is randomly extracted from the “hsf_0123” set, that creates 84,834 samples in total, including uppercases and lowercases. All of the 23,670 samples from the set “hsf_7” are used as testing data.

Our experiments are discussed below. In Section 5.4.1, the experimental results for the 26-class uppercase recognizer and the 26-class lowercase recognizer are presented; and the experiments on the 26-metaclass recognizer and the 52-class recognizer are followed in Section 5.4.2.

5.4.1 Experiments on the 26-Class Uppercases and the 26-Class Lowercases

The 26-class uppercase problem and the 26-class lowercase problem were considered by applying three individual classifiers, consisting of the SVM, CNN and hybrid CNN & SVM model. The parameters for training each classifier were configured and experimentally set. These parameters are shown in Table 9, where c and σ represent the training parameters of the SVM classifier with the RBF kernel; #layers means the number of feature maps in layer 2, layer 3, and layer 4 of the CNN model; and the #epoch indicates the number of running epochs when the CNN model converges and the learning process stops.

Table 9. Classifier parameters on the 26-class uppercase problem and the 26-class lowercase problem.

	SVM		CNN		Hybrid CNN & SVM	
	c	σ	#layers	#epoch	c	σ
26 uppercases	32	0.03125	25-50-100	370	8	0.001953125
26 lowercases	8	0.0078125	25-50-100	420	8	0.001953125

Table 10 lists the testing results on uppercase characters and lowercase characters by using different individual classifiers. The proposed hybrid CNN & SVM classifier produced the highest recognition rates on both 26-class uppercases and 26-class lowercases, with rates of 96.2289% and 90.2410%, respectively. The second highest recognition rates were generated by the SVM classifier, and the testing results given by the CNN classifier were followed. The aim of these experiments was to verify the generalization ability of the hybrid CNN & SVM model on both uppercase and lowercase problems. Although the recognition performances among these three classifiers exhibited slight differences, the performance of the proposed hybrid CNN & SVM model was satisfactory and better than other classifiers.

Table 10. Recognition rates for the 26-class uppercases and 26-class lowercases by three different classifiers.

	SVM (%)	CNN (%)	Hybrid CNN & SVM (%)
26 uppercases	95.6583	95.5200	**96.2289**
26 lowercases	89.8946	89.5500	**90.2410**

Table 11 lists the comparison of the complexity among three classifiers on the 26-class problem in uppercases and the 26-class problem in lowercases, respectively. The symbol #SVs indicates the number of Support Vectors. When comparing these three classifiers, we noticed that the hybrid CNN & SVM model runs the fastest in the decision process on both upper and lower cases. Besides, this model occupies almost the least amount of memory space. The complexity of the CNN classifier was in the middle, and the SVM classifier had the highest complexity. Thus, when considering the complexity, the hybrid model is more effective than other single classifiers in our experiments.

Table 11. Comparison of three classifiers in terms of complexity on the 26-class uppercase letters and the 26-class lowercase letters in the NIST database.

	26 uppercases			26 lowercases		
	SVM	CNN	Hybrid CNN & SVM	SVM	CNN	Hybrid CNN & SVM
Total testing speed (Second)	240	208	26	252	201	39
Memory usage (MB)	31.5	5.4	3.5	34.2	5.4	5.7
#SVs	12,537	_	2,776	14,187	_	4,637

In order to find the reliability of the hybrid model, we applied the same rejection approach as that adopted on the handwritten digit recognition experiments in Chapter 4, Section 4.2.3. For space saving purposes and easy comparisons with other researchers, the results with nearly 1% error rates are presented in Table 12, each on the 26-class uppercase recognizer and the 26-class lowercase recognizer. The reliability rate was 98.99% and the rejection rate was 8.72% with a 1.01% error rate on the 26 uppercase characters. As for the 26-class lowercase recognizer, the rejection rate increased to 29.07% when we ensured that the error rate remained around 1%.

Table 12. Testing results on the 26 uppercase characters and 26 lowercase characters with around 1% error rate, using the hybrid CNN & SVM model.

	Threshold	Recognition (%)	Rejection (%)	Error (%)	Reliability (%)
26 uppercases	0.910	90.27	8.72	1.01	98.99
26 lowercases	0.968	70.01	29.07	0.92	99.08

Table 13 compares our results with other research results. Even though there have been many achievements on these problems, it is hard to compare the results in a fair way. This difficulty occurs because most of authors have reported their results based on different databases, or on the same database with different training sample sets and testing sample sets. The only research results published on the 26 uppercase characters by using the same dataset with the same amount of training and testing samples can be found in [21, 54, 55, 56]. These results are included in Table 13. However, as found in [12, 57], all the experiments conducted on the 26 class lowercases used our lowercase testing set as their validation set. But, we also included their validation results in Table 13 in order to provide the most comprehensive information for the reader and for future researchers.

From Table 13, we noticed that our proposed method has a good generalization ability and a better reliability than most other methods, and it is favourably comparable to the best result achieved by the SVM on the 26 uppercase characters [21]. Moreover, the hybrid model has two good characteristics. One is that this model can automatically extract the discriminating features from input images, while a successful SVM recognizer depends heavily on the representative features to be extracted by hand. The other is that the recognition performance of the hybrid CNN & SVM model can be further improved through the fine tuning of the model's structure and its parameters, such as the number of feature planes in each layer, the number of running epochs, the size of input samples, etc. Therefore, our proposed method is quite promising in the handwriting recognition research field.

Table 13. Experimental results for comparisons on the NIST handwritten letter database.

	Method	Recognition (%) (zero rejection)		Rejection (%)	Error (%)
26 uppercases	Ensemble HMMs [54]	93.24±0.07		39.36	1.00
	SVM [21]	96.82		10.07	0.50
	Ensemble MLPs [55]	95.98		0	4.02
	Ensemble KNNs [56]	94.16		0	5.84
	Proposed hybrid model	**96.23**		**8.72**	**1.01**
26 lowercases	MLP [12]	90.06		0	9.94
	MLP [57]	88.88		0	11.12
	Proposed hybrid model	**90.24**		**29.07**	**0.92**

Our goal was to recognize the unconstrained characters, without knowing which type of case that a character belongs to in a priori. So we continue our work on investigating the 26-metaclass problem and the 52-class problem in the following section.

5.4.2 Experiments on the 26-Metaclass and 52-Class Problems

To solve the 26-metaclass problem and the 52-class problem, we applied the five classification models that were applied on the digit recognition problem. That is: SVM, CNN, hybrid CNN & SVM, combination CNN and SVM, and the ensemble system. In this way, we could provide a more comprehensive analysis on the efficiency of our proposed systems.

In this section, we describe the experiments on the 26-metaclass recognition problem and the 52-class recognition problem without rejection in Section 5.4.2.1. Then, the reliability performances are analyzed on both recognition systems in Section 5.4.2.2.

5.4.2.1 Experiments on the 26-Metaclass and 52-Class Problems without Rejection

When training each single classifier, there are some crucial parameters that need be set and optimized firstly. To build the SVM classifier, the 5-fold cross validation technique is conducted on the training dataset and it generates the optimal parameters $C = 8$ and $\sigma = 0.03125$. As for the hybrid CNN & SVM, the values of the optimal parameters are produced with $C = 8$ and $\sigma = 0.0078125$. For the CNN model, the number of each feature map layer of CNN is set to be 25-50-100, which is the same as the structure applied on the handwritten digits. However, the training process converges and stops at 1,000 epochs, because the cursive letter recognition problem is more difficult and complicated compared with the handwritten digit recognition problem.

Table 14 lists the testing results on the 26-metaclass problem and the 52-class problem by using SVM, CNN and hybrid CNN & SVM classifiers. Comparing the recognition rates, the best testing results were achieved by the hybrid CNN & SVM model on both problems: a rate of 92.0744% for the 26-metaclass problem and 70.2408% for the 52-

class problem. These results prove that our proposed hybrid CNN & SVM model is superior to the SVM and the CNN classifier in the performance when dealing with the handwritten character (both digits and letters) recognition.

It is clear to see that the recognition rate on the 26-metaclass problem is significantly higher than that on the 52-class problem. That is to say, it is much easier to recognize letters in the insensitive cases (A recognizer does not need to know the uppercase or the lowercase of the sample) compared to the sensitive cases (A recognizer need to know the uppercase or the lowercase of the sample) for the English cursive letters. We need to look at why there is such a large recognition gap between these two problems. One instinctive reason is that by adding more number of classes, the complexity of the training process is increased for the classifier under the same amount of training samples. It is quite probable that the recognition ability would be enhanced by expanding the training set for the 52-class problem. Another reason is that the differences among the classes are decreased after the corresponding uppercases and lowercases are merged. For example, some uppercase and lowercase letters, such as “C” and “c”, “O” and “o”, “V” and “v”, “X” and “x”, etc., have very similar shapes in human handwritings. Treating them as separate classes may cause many confusions to the classifier’s training. But after merging them, the variances among different classes can be largely eliminated. As a result, a better recognizer with a higher generalization ability is achieved.

Table 14. Recognition rates on NIST database by using different classifiers without rejection.

#Class	SVM (%)	CNN (%)	Hybrid CNN & SVM (%)
26	91.5040	91.6800	**92.0744**
52	69.7169	69.7700	**70.2408**

For comparative purposes, we built up the combination/ensemble systems for recognizing the unconstrained handwritten letters as similar to those systems applied to the handwritten digits. The experiments are described in the following paragraphs.

When the individual classifier models are well trained, they are combined through the weighted multiplication rule. It has been demonstrated as better than other rules (e.g. majority vote, sum, and product) in the performance of recognizing handwritten digits. The weighting factors for the combination CNN and SVM system and the ensemble system were optimized for each recognition problem. Table 15 shows their values, where w_c denotes the weighting factor of the CNN classifier, w_s denotes the weighting factor of the SVM classifier, w_H represents the weighting factor of the hybrid CNN & SVM model, and w_C represents the weighting factor of the combination CNN and SVM system.

The experimental results on the testing set by using the combination system and the ensemble system are shown in Table 16. The slightly higher recognition rates were produced by the ensemble system on both 26-metaclass and 52-class problems, with rates of 92.6785% and 70.9844%, respectively. Comparing the results with those in Table 14, the performances of the hybrid CNN & SVM model are favourably comparable with the combination/ensemble system. That means, the strategy of combining different classifiers does not improve the recognition ability too much for this unconstrained English letter problem.

Table 15. The values of weighting factors for the combination system and the ensemble system.

Parameter \ # Class		26	52
Combination system	w_c	1.0	1.0
	w_s	0.3	0.3
Ensemble system	w_H	0.9	0.9
	w_C	1.0	1.0

Table 16. Recognition results by applying different combination methods without rejection.

#Class	Method	Recognition rate (%)
26	Combination SVM and CNN	92.5306
	Ensemble system (Combination + Hybrid)	**92.6785**
52	Combination SVM and CNN	70.5619
	Ensemble System (Combination + Hybrid)	**70.9844**

5.4.2.2 Reliability Experiments on the 26-Metaclass and 52-Class Problems

Furthermore, we extended our experiments in order to get reliable recognizers on the 26-metaclass and the 52-class problems. The reliability is much more important than the recognition accuracy in practical applications. Hence, we will discuss the reliability performances of the hybrid CNN & SVM model and the ensemble system on the 26-metaclass and 52-class problems respectively, by using the same rejection mechanism that was described in Chapter 4, Section 4.2.3.

Table 17 lists the testing results with two types of rejection thresholds on the 26-metaclass problem by using the hybrid model, and Table 18 shows the experimental results by applying the ensemble system. The classification results for the 52-class problem are presented in the same way: the results generated by the hybrid model are given in Table 19, and those produced by the ensemble system are displayed in Table 20. For an easier comparison, we also illustrate the error-reject evaluation for the 26-metaclass in Figure 16, and Figure 17 shows the analysis for the 52-class problem. We only illustrate the results that have error rates of less than 8%. From Figure 16 and Figure 17, it is evident that the ensemble system makes fewer rejections than the hybrid model under the same error rates. This means that the ensemble system performs more reliably on the 26-metaclass and the 52-class problems. From these observations, we conclude that combining distinct classifiers can enhance the reliability of the performance, compared with the usage of a single recognizer on the unconstrained handwritten letter recognition problem. However, for the 1% error rate, the rejections on the 26-metaclass problem were much smaller than those on the 52-class problem, which proves again that recognizing letters in the insensitive case is an easier task compared to that on the sensitive case.

Table 17. Testing results of 26-metaclass problem with rejection by using the hybrid model.

Threshold	*Recognition(%)*	*#Rejection*	*#Error*	*Reliability(%)*
0.0	92.07	0	1876	92.07
0.1	91.58	294	1699	92.82
0.2	91.06	557	1558	93.41
0.3	90.61	791	1431	93.95
0.4	90.09	1021	1324	94.40
0.5	89.40	1290	1217	94.85
0.6	88.67	1599	1082	95.42
0.7	87.07	2172	887	96.25
0.8	84.89	2956	620	97.38
0.9	81.90	3823	460	98.05
0.91	81.29	3986	441	98.13
0.92	80.64	4159	422	98.21
0.93	79.89	4358	402	98.30
0.94	79.06	4581	375	98.41
0.95	78.14	4826	346	98.53
0.96	76.95	5147	308	98.69
0.97	75.05	5633	271	98.85
0.98	71.58	6502	224	99.05
0.99	60.36	9229	152	99.35
Threshold (mean distance)				
0.958246	77.19	5085	315	98.67

Table 18. Testing results of 26-metaclass problem with rejection by using the ensemble system.

Threshold	*Recognition(%)*	*#Rejection*	*#Error*	*Reliability(%)*
0.0	92.67	0	1733	92.67
0.1	91.30	702	1356	94.27
0.2	89.90	1258	1132	95.21
0.3	88.79	1687	966	95.91
0.4	87.37	2163	825	96.51
0.5	85.89	2632	707	97.01
0.6	84.12	3185	573	97.57
0.7	81.39	4014	389	98.35
0.8	77.27	5199	179	99.24
0.9	69.53	7121	89	99.62
0.91	68.16	7458	78	99.67
0.92	66.52	7855	68	99.71
0.93	64.59	8322	58	99.75
0.94	61.88	8974	49	99.79
0.95	58.42	9802	38	99.83
0.96	52.80	11135	37	99.84
0.97	39.87	14216	15	99.93
0.98	0.97	23439	0	100.00
0.99	0.10	23646	0	100.00
Threshold (mean distance)				
0.904632	68.96	7260	87	99.63

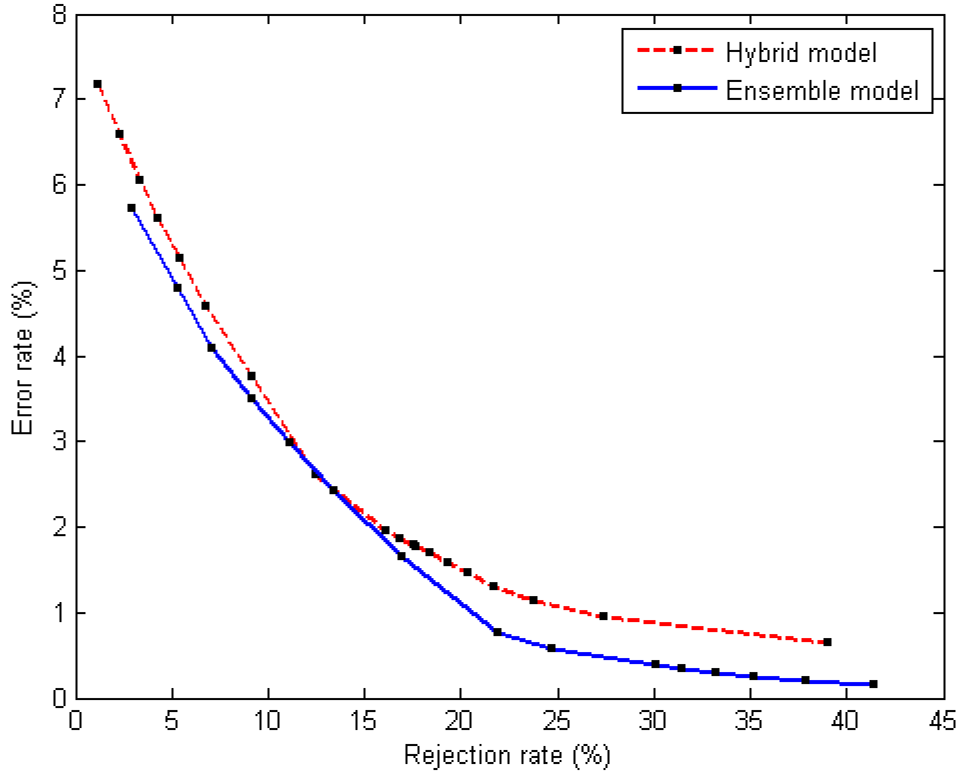


Figure 16. Error-reject analysis of hybrid model and ensemble system for the 26-metaclass problem.

Table 19. Testing results of 52-class problem with rejection by using the hybrid model.

Threshold	*Recognition(%)*	*#Rejection*	*#Error*	*Reliability(%)*
0.0	70.24	0	7044	70.24
0.1	69.22	722	6562	72.27
0.2	68.33	1262	6232	73.67
0.3	67.47	1766	5932	74.93
0.4	66.65	2288	5605	76.32
0.5	65.56	2865	5285	77.67
0.6	64.18	3532	4945	79.10
0.7	60.95	4942	4301	81.82
0.8	57.19	6484	3647	84.59
0.9	43.94	11424	1844	92.20
0.91	41.47	12250	1604	93.22
0.92	38.97	13075	1369	94.21
0.93	35.78	14039	1161	95.09
0.94	32.20	15090	957	95.95
0.95	28.78	16076	781	96.70
0.96	24.25	17364	566	97.60
0.97	19.91	18551	405	98.28
0.98	15.43	19752	265	98.88
0.99	8.55	21509	137	99.42
Threshold (mean distance)				
0.892577	45.72	10845	2002	91.54

Table 20. Testing results of 52-class problem with rejection by using the ensemble system.

Threshold	*Recognition(%)*	*#Rejection*	*#Error*	*Reliability(%)*
0.0	70.98	0	6868	70.98
0.1	67.59	1873	5797	75.50
0.2	65.31	2943	5267	77.74
0.3	63.23	3906	4797	79.73
0.4	60.86	4949	4315	81.77
0.5	57.88	6170	3798	83.95
0.6	53.28	7973	3085	86.96
0.7	47.29	10215	2260	90.45
0.8	37.27	13774	1072	95.47
0.9	20.00	18913	21	99.91
0.91	17.26	19568	15	99.93
0.92	14.15	20308	11	99.95
0.93	11.22	21004	9	99.96
0.94	7.76	21827	4	99.98
0.95	1.89	23222	0	100.00
0.96	0.25	23610	0	100.00
0.97	0.13	23637	0	100.00
0.98	0.08	23650	0	100.00
0.99	0.03	23662	0	100.00
Threshold (mean distance)				
0.766184	41.46	12318	1539	93.50

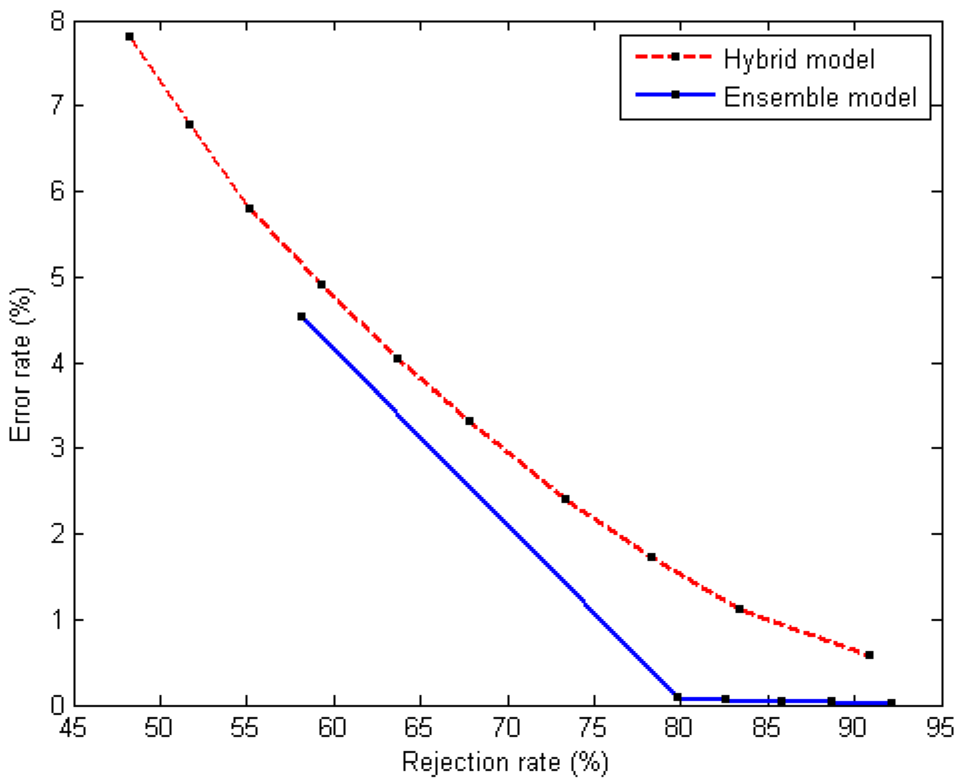


Figure 17. Error-reject analysis of hybrid model and ensemble system for the 52-class problem.

5.5 Discussion

In this section, we discuss two questions pertaining to the motivation and the main concerns of our research in this chapter.

1. Which classification algorithm is better for recognizing unconstrained English letters?

When only considering an individual classifier, including the SVM, CNN, and the hybrid CNN & SVM model, there is no doubt that the proposed hybrid model outperforms the other classifiers on all recognition problems pertaining to 26-class uppercases, 26-class lowercases, 26-metaclass, and 52-class on the NIST database. Although the recognition performances on these problems by using the hybrid model are not the best ones when compared with the result reported in literature [21], the proposed hybrid model has a superiority over the other existing classifier models in two ways: one is that the salient features can be automatically extracted by the hybrid model, while the success of most other traditional classifiers relies largely on the retrieval of good hand-designed features. The other lies in that the hybrid model combines the advantages of SVM and CNN, as both are the most popular and successful classifiers in the handwritten character recognition field.

If we compare the recognition accuracy between the hybrid model and the combination/ensemble systems, the combination/ensemble approaches make improvements on the unconstrained handwritten letter recognition problem more than the former, but to a limited extent. However, the reliability of the combination/ensemble systems is much better than the hybrid model. For example, it is interesting to note that the reliability is boosted as the recognition performance is increased under the zero rejection condition. In conclusion, to obtain higher recognition and reliability achievements on such complicated unconstrained letter recognition problems, the design of a more sophisticated classification system is required, such as the combination/ensemble approaches that adopted different classifiers in our experiments.

2. Which classification strategy is more efficient in solving the unconstrained letter recognition problem?

In practical applications, sometimes, it is not important for the recognizer to know about the writing styles (uppercases or lowercases) but only to know which letter pertains to the sample. The writing style can be known according to the context information or can be further processed by the post-processing technique. This is called a case insensitive situation. To cope with this situation, the 26-metaclass classification problem is obviously the easiest one to be solved with the highest recognition accuracy in our experiments.

However, if considering the case sensitive situation, which requires the recognizer to know the uppercases and lowercases in the classification procedure, then the recognition performance is not satisfactory when the 52-class classification strategy is directly applied. Thus, some other approaches are worthy of further exploration. As we have already built up the 26-class uppercase recognizer and 26-class lowercase recognizer, both recognition rates exceed 90% without rejection. It was very encouraging for us to discover a good method to combine them, as suggested by the literature [12]. Additionally, the metaclass strategy can be improved by merging not all uppercases and corresponding lowercases, but only the ones which have similar shapes between the uppercase letters and the lowercase letters.

5.6 Conclusion

This chapter has investigated the unconstrained English handwritten letter recognition problem by applying different classification algorithms, including SVM, CNN, the proposed hybrid CNN & SVM model, and their combinations. To deal with such a complex problem, three classification strategies to determine the number of classes were taken into account: the 26-class problem in uppercases and 26-class problem in lowercases, the 26-metaclass problem, and the 52-class problem.

The experiments were conducted on the NIST handwritten alphabet database. Results showed that the hybrid system outperforms other individual classifiers on all three classification strategies, under the zero rejection condition. However, the combination/ensemble systems have proved more reliable than individual recognizers on this problem. Furthermore, from experimental observations, we found that the task of recognizing letters in the insensitive case is much easier compared to the sensitive case. Therefore, future studies should be related to the question of how to improve the recognition performance on the sensitive case.

Chapter 6

Conclusion

In this thesis, I worked on recognizing off-line handwritten characters. This research topic was driven by the goal of generic context recognition for off-line handwriting recognition. Two types of recognition models were proposed. One was the hybrid CNN & SVM model. The other was the CNN and SVM combination system. The efficiency and feasibility of the proposed models were evaluated from two aspects: the recognition accuracy and the reliability performance. Experiments were conducted on handwritten digits and handwritten letters in the English language, respectively. Two public benchmark databases were used for our testing: the MNIST digit database and the NIST letter dataset. The proposed hybrid model achieved the best results on recognizing handwritten digits. It produced high and satisfactory performances on the handwritten English letter recognition. Moreover, this model has the potential to be improved further. In this section, the thesis contributions are summarized and the future work is proposed.

6.1 Contributions

In this thesis, my research focuses on the classifier design, with the aim of increasing the recognition accuracy and the reliability performance of the current handwritten character recognition system. The main contributions of this thesis are summarized in the following paragraphs.

Firstly, the proposed hybrid model has demonstrated its robustness and success on the recognition of handwritten characters. For the handwritten digits, the lowest error rate with no rejections was achieved at a rate of 0.19% on the MNIST database. It is the best result up to date, compared with the latest one of 0.35% reported in [11]. Moreover, 100% reliability with a 0.56% rejection rate has been obtained by applying the hybrid model. Its reliability performance exceeds Dong's results in [36], which is 100% reliability with an 8.49% rejection rate by using the HeroSVM library under the same rejection mechanism. For the handwritten letter recognition, satisfactory achievements have also been achieved by the hybrid model. The recognition rates without rejections on the NIST database for the 26-class uppercase letters and the 26-class lowercase letters were 96.2289% and 90.2410%, respectively. These rates dropped to 92.0744% for the 26-metaclass problem and 70.2408% for the 52-class problem. For all the cases, the hybrid model outperformed other single classifiers that were trained in our experiments, like the SVM model and the CNN model. Even though it is difficult to make a fair comparison with other researchers, due to the different databases, the number of training and testing sets, and the experimental conditions, etc., we believe that our recognition performances are among the top rankings in general in the handwritten character recognition field. However, there is one special case that we can compare with other researchers, due to the same training set and testing set on the adopted NIST database. It is the 26-class uppercase problem (see Chapter 5, Table 13). Even though the recognition rate of the hybrid model is not the highest one; it is comparable to the best result. Moreover, the proposed model is very promising and the recognition accuracy has the potential of being further improved.

Secondly, this is the first time that a comparison has been made between the hybrid CNN & SVM model and the combination of CNN and SVM system on handwritten character recognition problems. According to common sense, the combination of different classifiers may outperform the single recognizer. However, our experimental results showed that it depends on the applications. For handwritten digit recognition, the hybrid CNN & SVM model is superior to the combination/ensemble systems. As for recognizing unconstrained handwritten letters in the English language, the conclusion is reversed and the combination/ensemble systems perform slightly better than the hybrid model. Therefore, in conclusion, we can not find a universal approach to solve all the problems. Specific issues should be analyzed in a case by case manner.

Thirdly, the comprehensive analysis of the generalization ability and the reliability of the handwritten character recognition systems were provided in this thesis. Most researchers have focused on improving the recognition accuracy, but only a few of them have mentioned the reliability performance. In order to meet the requirements of real-life applications and apply the proposed methods into practical fields in the future, I considered both of them for the handwritten digits and the handwritten letters. The reliability performance has been realized through the rejection rule: the pattern was rejected when the difference between the top two confidence values in the output rank list was smaller than a predefined threshold. But, as for the question about how to choose a suitable threshold for the trade-off between the recognition and the reliability performances, it is really a problem which depends on different application demands.

Last but not least, this thesis has proven that automatically extracted features in the hybrid model are superior to hand-designed features. The experimental results on the handwritten characters showed that the recognition accuracy of the hybrid model using automatic features was higher than that of the SVM model using hand-designed features.

However, it is undeniable that there are many other hand-designed features that may be more suitable to the handwritten character recognition problems. But the features (the gradient feature, the distance feature and the chain feature) extracted for building the SVM classifiers in this thesis are quite representative and have been shown to be very efficient in the literature [10]. Besides, the design of good features by human beings is really an elaborate and time-consuming task. Thus, the automatic features extracted by the hybrid model are easier to implement and they outperform the hand-designed features on handwritten character recognition applications.

6.2 Future Work

The research process on this topic never ends. Future work can be conducted on this thesis topic with the following aspects:

Research on the hybrid CNN & SVM learning model is still at an early stage. The performance of the hybrid model can be further improved through the fine tuning of its structure and its parameters. For example, improvements might be made based on the size of the input layer, the number of feature layer maps in layer 2, layer 3, and layer 4, the stopping criterion of the learning process, the kernel functions used in the model, etc.

It is desired to improve the recognition accuracy on unconstrained handwritten letters, especially for the case sensitive recognition problem. There are some strategies that can be considered as solutions. One is to improve the recognition rate on the 52-class problem directly. I believe that this improvement can be achieved by enlarging the amount of training samples. The second strategy is to find a good way to combine the 26-class uppercase recognizer and the 26-class lowercase recognizer that were discussed in Chapter 5, Section 5.4.1. The third approach is to further improve the metaclass strategy by only merging the uppercases and the corresponding lowercases that have similar shapes into one class.

Extending the proposed hybrid model to other applications is a task worth investigating. It is very easy to apply our work on the isolated special symbols, such as “,”, “.”, “?”, “!” etc. Without being limited to the English handwritten character recognition, characters in other languages, such as Arabic, French, etc., can also be studied.

References

[1] N. Gorski, V. Anisimov, E. Augustin, O. Baret, and S. Maximov, “Industrial bank check processing: the A2iA CheckReaderTM,” International Journal on Document Analysis and Recognition, 3: 196-206, 2001.

[2] G. Dzuba, A. Filatov, D. Gershuny, I. Kil, and V. Nikitin, “Check amount recognition based on the cross validation of courtesy and legal amount fields,” International Journal of Pattern Recognition Artificial Intelligence, 11(4): 639-655, 1997.

[3] Lockheed Martin Delivers Last Automated Package Processing System to the U.S. Postal Service, http://www.lockheedmartin.com/news/press_releases/2006/LockheedMartinDeliversLastAutomated.html.

[4] A. L. Koerich, R. Sabourin, and C. Y. Suen, “Large vocabulary off-line handwriting recognition: a survey,” Pattern Analysis and Applications, 6(2): 97–121, 2002.

[5] The MNIST Database of Handwritten Digits, http://yann.lecun.com/exdb/mnist/.

[6] C. Y. Suen, C. Nadal, R. Legault, T. A. Mai, and L. Lam, “Computer recognition of unconstrained handwritten numerals,” Proceedings of the IEEE, 80(7): 1162–1180, July 1992.

[7] USPS Office of Advanced Technology Database of Handwritten Cities, States, ZIP Codes, Digits, and Alphabetic Characters, http://www.cedar.buffalo.edu/Databases/CDROM1/.

[8] P. J. Grother, “NIST special database 19 - handprinted forms and characters database,” National Institute of Standards and Technology, March 1995.

[9] F. Camastra, M. Spinetti, and A. Vinciarelli, “Offline cursive character challenge: a new benchmark for machine learning and pattern recognition algorithms,” In Proceedings of 18th International Conference on Pattern Recognition, Hong Kong, 2: 913-916, 2006.

[10] C. L. Liu, K. Nakashima, H. Sako, and H. Fujisawa, “Handwritten digit recognition: benchmarking of state-of-the-art techniques,” Pattern Recognition, 36(10): 2271-2285, October 2003.

[11] D. C. Ciresan, U. Meier, L. M. Gambardella, and J. Schmidhuber, “Deep big simple neural nets excel on handwritten digit recognition,” CoRR, abs/1003.0358, 2010. (informal publication)

[12] A. L. Koerich and P.R. Kalva, “Unconstrained handwritten character recognition using metaclasses of characters,” In Proceedings of IEEE International Conference on Image Processing, Genoa, Italy, 2:II-542-5, 2005.

[13] J. R. Thornton, J. B. Faichney, M. M. Blumenstein, V. M. Nguyen, and T. J. Hine, “Offline cursive character recognition: a state of the art comparison,” Advances in Graphonomics: Proceedings of IGS 2009, pp. 148-151, 2009.

[14] F. Camastra, “A SVM-based cursive character recognizer,” Pattern Recognition, 40(12): 3721-3727, 2007.

[15] M. Blumenstein, X. Y. Liu, and B. Verma, “An investigation of the modified direction feature for cursive character recognition,” Pattern Recognition, 40(2): 376-388, February 2007.

[16] C. L. Liu, “Partial discriminative training for classification of overlapping classes in document analysis,” International Journal on Document Analysis and Recognition, 11(2): 53-65, 2008.

[17] N. Arica and F. T. Yarman-Vural, “An overview of character recognition focused on off-line handwriting,” IEEE Transactions on Systems, Man and Cybernetics, Part C: Applications and Reviews, 31(2): 216-233, 2001.

[18] M. Ranzato, C. Poultney, S. Chopra, and Y. LeCun, “Efficient learning of sparse representations with an energy-based model,” In J. Platt et al. (Eds.), Advances in Neural Information Processing Systems, MIT Press, pp. 1137-1144, 2006.

[19] F. Lauer, C. Y. Suen, and G. Bloch, “A trainable feature extractor for handwritten digit recognition,” Pattern Recognition, 40(6): 1816-1824, June 2007.

[20] Y. LeCun, F. J. Huang and L. Bottou, “Learning Methods for Generic Object Recognition with Invariance to Pose and Lighting,” In Proceedings of Computer Vision and Pattern Recognition, Washington, DC, USA, pp.97-104, 2004.

[21] J. Milgram, M. Cheriet, and R. Sabourin, "“One Against One” or “One Against All”: which one is better for handwriting recognition with SVMs?" In Proceedings of 10th International Workshop on Frontiers in Handwriting Recognition, La Baule, France, 2006. [inria-00103955 − version 1]

[22] R. Polikar, "Ensemble based systems in decision making," IEEE Circuits and Systems Magazine, 6(3): 21-45, 2006.

[23] Q. Fu, X. Q. Ding, T. Z. Li, and C. S. Liu, "An effective and practical classifier fusion strategy for improving handwritten character recognition," In Proceedings of International Conference on Document Analysis and Recognition, Curitiba, Brazil, pp. 1038-1042, 2007.

[24] C. Y. Suen and L. Lam, "Multiple classifier combination methodologies for different output levels," Invited Paper, In Proceedings of the International Workshop on Multiple Classifier Systems, Cagliari, Italy, pp. 52-66, June 2000.

[25] P. Zhang, "Reliable recognition of handwritten digits using a cascade ensemble classifier system and hybrid features," Ph.D. thesis, Computer Science Department, Concordia University, Montreal, April 2006.

[26] L. Lam and C. Y. Suen, "Optimal combinations of pattern classifiers," Pattern Recognition Letters, 16: 945-954, September 1995.

[27] H. Nishimura, M. Kobayashi, M. Maruyama, and Y. Nakano, "Offline character recognition using HMM by multiple directional feature extraction and voting with bagging algorithm," In Proceedings of 5th International Conference on Document Analysis and Recognition, Bangalore, India, pp. 49–52, 1999.

[28] L. I. Kuncheva, "Combining pattern classifiers: methods and algorithms," Wiley, 2004.

[29] C. J. Burges, "A tutorial on support vector machines for pattern recognition," Data Mining and Knowledge Discovery, 2(2): 121 – 167, 1998.

[30] J. X. Dong, A. Krzyzak, and C. Y. Suen, "Fast parallel SVM training algorithm and its applications to very large dataset," IEEE Transactions on Pattern Analysis and Machine Intelligence, 27(4): 603-618, 2005.

[31] T. Joachims, “Text categorization with support vector machines: learning with many relevant features,” In Proceedings of the European Conference on Machine Learning, Chemnitz, Germany, pp. 137-142, 1998.

[32] G. Guo, S. Z. Li, and K. Chan, “Face recognition by support vector machines,” In Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition, Grenoble, France, pp. 196-201, March 2000.

[33] Y. P. Estevan, V. Wan, and O. Scharenborg, “Finding maximum margin segments in speech,” In Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, Hawaii, USA, pp. 15-20, April 2007.

[34] L. N. Teow and K. F Loe, “Robust vision-based features and classification schemes for off-line handwritten digit recognition,” Pattern Recognition, 35 (11): 2355-2364, 2002.

[35] D. DeCoste and B. Schölkopf, “Training invariant support vector machines,” Machine Learning Journal, 46(1-3): 61-190, 2002.

[36] J. X. Dong, “Speed and accuracy: large-scale machine learning algorithms and their applications,” Ph.D. thesis, Department of Computer Science, Concordia University, Montreal, 2003.

[37] L. S. Oliveira and R. Sabourin, “Support vector machines for handwritten numerical string recognition,” In Proceedings of 9th International Workshop on Frontiers in Handwriting Recognition, Kokubunji, Tokyo, Japan, pp. 39-44, October 2004.

[38] J. X. Dong, A. Krzyzak, and C. Y. Suen, “Local learning framework for handwritten character recognition,” Engineering Applications of Artificial Intelligence, 15(2): 151--159, April 2002.

[39] S. Abe, “Support vector machines for pattern classification,” Springer-Verlag, London, 2005.

[40] C. W. Hsu and C. J. Lin, “A comparison of methods for multiclass support vector machines,” IEEE Transactions on Neural Networks, 13(2): 415, 2002.

[41] C. C. Chang and C. J. Lin, “LIBSVM: a library for support vector machines,” 2001, Software Available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[42] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, 86(11): 2278-2324, November 1998.

[43] W. Pan, T. D. Bui, and C. Y. Suen, "Isolated handwritten Farsi numerals recognition using sparse and over-complete representations," In Proceedings of International Conference on Document Analysis and Recognition, Barcelona, Spain, pp. 586-590, July 2009.

[44] J. H. Kim, K. K. Kim, and C. Y. Suen, "An HMM-MLP hybrid model for cursive script recognition," Pattern Analysis and Application, 3(4): 314–324, 2000.

[45] J. H. Kim, K. K. Kim, and C. Y. Suen, "Hybrid schemes of homogeneous and heterogeneous classifiers for cursive word recognition," In Proceedings of 7th International Workshop on Frontiers in Handwriting Recognition, Amsterdam, Netherlands, pp. 433–442, 2000.

[46] C. L. Liu, K. Nakashima, H. Sako, and H. Fujisawa, "Handwritten digit recognition: investigation of normalization and feature extraction techniques," Pattern Recognition, 37(2): 265-279, 2004.

[47] M. Shi, Y. Fujisawa, T. Wakabayashi, and F. Kimura, "Handwritten numeral recognition using gradient and curvature of gray scale image," Pattern Recognition, 35(10): 2051-205, 2002.

[48] P. Y. Simard, D. Steinkraus, and J. C. Platt, "Best practice for convolutional neural networks applied to visual document analysis," In Proceedings of International Conference on Document Analysis and Recognition, Edinburgh, Scotland, 2: 958—962, 2003.

[49] Y. Mizukami, K. Yamaguchi, J. Warrell, P. Li, and S. Prince, "CUDA Implementation of deformable pattern recognition and its application to MNIST handwritten digit database," In Proceedings of 20th International Conference on Pattern Recognition, Istanbul, Turkey, pp. 2000-2004, 2010.

[50] X. Chen, X. Liu, and Y. Jia, "Discriminative structure selection method of Gaussian Mixture Models with its application to handwritten digit recognition," Neurocomputing, 74 (6): 954 – 961, 2011.

[51] H. Cecotti and A. Belaid, "Rejection strategy for convolutional neural network for adaptive topology applied to handwritten digits recognition," In Proceedings of International Conference on Document Analysis and Recognition, Seoul, Korea, pp. 765-769, 2005.

[52] C. Y. Suen, K. Zanibbi, and A. Ring, "Human recognition of MNIST digits," Technical Report, CENPARMI, Concordia University, April 2006.

[53] C. Y. Suen and J. N. Tan, "Analysis of errors on handwritten digits made by a multitude of classifiers," Invited Paper, Pattern Recognition Letters, 26: 369-379, 2005.

[54] P. R. Cavalin, R. Sabourin, C. Y. Suen, and A. S. Britto, "Evaluation of incremental learning algorithms for HMM in the recognition of alphanumeric characters," Pattern Recognition, 42(12): 3241-3253, 2009.

[55] P. V. W. Radtke, R. Sabourin, and T. Wong, "Classification system optimization with multi-objective genetic algorithm," In Proceedings of 10th International Workshop on Frontiers in Handwriting Recognition, La Baule, France, pp. 6359—6366, 2006.

[56] E. M. Dos Santos, R. Sabourin, and P. Maupin, "A dynamic overproduce-and-choose strategy for the selection of classifier ensembles," Pattern Recognition, 41: 2993–3009, 2008.

[57] A. L. Koerich, "Large vocabulary off-line handwritten word recognition," Ph.D. thesis, École de Technologie Supérieure, Montreal-Canada, August, 2002.