4-1-2011

Neural Network Based Off-line Handwritten Text Recognition System

Changan Han
Florida International University, chan002@fiu.edu

DOI: 10.25148/etd.FI11042701

Recommended Citation

Han, Changan, "Neural Network Based Off-line Handwritten Text Recognition System" (2011). *FIU Electronic Theses and Dissertations*. 363.
https://digitalcommons.fiu.edu/etd/363

FLORIDA INTERNATIONAL UNIVERSITY

Miami, Florida

NEURAL NETWORK BASED OFF-LINE HANDWRITTEN TEXT RECOGNITION SYSTEM

A dissertation submitted in partial fulfillment of the

requirements for the degree of

DOCTOR OF PHILOSOPHY

in

ELECTRICAL ENGINEERING

by

Changan Han

2011

To: Dean Amir Mirmiran
College of Engineering and Computing

This dissertation, written by Changan Han, and entitled Neural Network Based Off-line Handwritten Text Recognition System, having been approved in respect to style and intellectual content, is referred to you for judgment.

We have read this dissertation and recommend that it be approved.

Armando Barreto

Jean Andrian

Naphtali Rishe

Malek Adjouadi, Major Professor

Date of Defense: April 1, 2011

The dissertation of Changan Han is approved.

Dean Amir Mirmiran
College of Engineering and Computing

Interim Dean Kevin O'Shea
University Graduate School

Florida International University, 2011

DEDICATION

I would like to dedicate this dissertation to my parents. There is no doubt that without their support and understanding, I could not have achieved this milestone of my life.

ACKNOWLEDGMENTS

I would like to express gratitude to my advisor, Dr. Malek Adjouadi. He not only taught me how to finish a course or a project, but also instructed me how to do research. He gave me great freedom and enough support, and advised me numerous times on my research work and publications. I truly appreciate his support, guidance, and understanding.

I extend my appreciation to my committee members: Dr. Armando Barreto, Dr. Jean Andrian, and Dr. Naphtali Rishe, for all the help they provided throughout these years.

I would like to thank the current and former members in CATE lab: Melvin Ayala, Yu Chen, Javier Delgado, Anas Salah Eddin, Mohammed Goryawala, Magno R. Guillén, Ana M. Guzman, Mouncef Lahlou, Mark Rossman, Krishna Vedala, Lu Wang, Jin Wang and Xiaozhen You. My special gratitude goes to Jin Wang and Ana M. Guzman for their help and their many useful suggestions.

Most importantly, I appreciate the support provided by the National Science Fondation under grants CNS-1042341, CNC-0959985, and HRD-0833093.

ABSTRACT OF THE DISSERTATION

NEURAL NETWORK BASED OFF-LINE HANDWRITTEN TEXT RECOGNITION SYSTEM

by

Changan Han

Florida International University, 2011

Miami, Florida

Professor Malek Adjouadi, Major Professor

This dissertation introduces a new system for handwritten text recognition based on an improved neural network design. Most of the existing neural networks treat mean square error function as the standard error function. The system as proposed in this dissertation utilizes the mean quartic error function, where the third and fourth derivatives are non-zero.

Consequently, many improvements on the training methods were achieved. The training results are carefully assessed before and after the update. To evaluate the performance of a training system, there are three essential factors to be considered, and they are from high to low importance priority: 1) error rate on testing set, 2) processing time needed to recognize a segmented character and 3) the total training time and subsequently the total testing time. It is observed that bounded training methods accelerate the training process, while semi-third order training methods, next-minimal training methods, and preprocessing operations reduce the error rate on the testing set. Empirical observations suggest that two combinations of training methods are needed for different case character recognition.

Since character segmentation is required for word and sentence recognition, this dissertation provides also an effective rule-based segmentation method, which is different from the conventional adaptive segmentation methods. Dictionary-based correction is utilized to correct mistakes resulting from the recognition and segmentation phases.

The integration of the segmentation methods with the handwritten character recognition algorithm yielded an accuracy of 92% for lower case characters and 97% for upper case characters. In the testing phase, the database consists of 20,000 handwritten characters, with 10,000 for each case. The testing phase on the recognition 10,000 handwritten characters required 8.5 seconds in processing time.

TABLE OF CONTENTS

LIST OF TABLES

TABLE PAGE

LIST OF FIGURES

FIGURE PAGE

CHAPTER I

Introduction

1.1 General Statement of the Problem Area

The main objective of this research is to find a new solution for handwritten text recognition of different fonts and styles by improving the design structure of the traditional Artificial Neural Network (ANN). ANNs have been successfully applied to pattern recognition, association and classification, forecast studies, and control applications, to name a few. The recognition results of such text or handwritten materials are then fed into Optical Character Recognition (OCR) as an electronic translation of images of handwritten, typewritten or printed text into machine-editable text. Such was the objective set out to meet the motivational needs of an automated book reader for blind persons as outlined in the first section of Chapter II. OCR is a field of research that is fully developed and has been quite useful in pattern recognition, artificial intelligence and machine vision. Consequently, typewritten text recognition that is void of any distortions is now considered largely a solved problem. However the direct use of OCR on handwritten characters remains a very difficult problem to resolve, yielding extremely low reading accuracy.

On-line handwritten character recognition involves the automatic conversion of text as it is written on a special digitizer or PDA, where a sensor picks up the pen-tip movements as well as pen-up and pen-down switching. The obtained signal is converted into letter codes, which are usable within computer and text-processing applications.

Off-line handwritten document recognition is currently a difficult problem, as different people have different handwriting styles. Scanning, segmentation and classification are the general processes that are being used to recognize handwritten documents. In previous studies, ANNs have proven to be excellent recognizers of printed characters and handwritten digits (0~9), but research into recognition of handwritten words has not always been effective [Al-Jawfi, 2009; Ebrahimpour et al., 2010].

1.2 Research Problem

This dissertation establishes a complete system that converts scanned images of handwritten characters to text documents. The system consists of a series of operations that include preprocessing, segmentation, training and recognition, dictionary-based correction, and layout recovery. Optional dynamic training of different writers improves largely the recognition accuracy.

The task of preprocessing relates to the removal of noise and variation in handwritten word patterns. Preprocessing may itself be broken down into smaller tasks such as noise removal, slant estimation and correction, skew detection, resizing, etc. By preprocessing each handwritten character, the task of subsequent recognition is simplified [Blumenstein et al., 2002]. In this dissertation, the important issue of ‘increasing diversity of training sets’ is added to the preprocessing phase.

Segmentation plays an important role in the overall process of handwriting recognition. A rule-based segmentation is applied into the system [Verma, 2003]. The algorithm follows a few steps: baseline computation, over-segmentation, and rule-based validation. Since slant and skew are removed, all baselines are horizontal and segmentation lines are vertical.

During the training and recognition, pixel images are converted into editable characters with pre-defined weights. Dynamic training gives adaptive weights for each user, which can improve the recognition accuracy for a specific user. Several computational models can be used to recognize characters, which will be discussed in Chapter 2.

In a dictionary-based correction process, a list of candidate-recognized words is identified as a comparison function of dictionary entries to various combinations of recognized characters. The process finds a most likely word and presents it to the user. Besides, on the general frequency table, words that occur in the same article are more probable to repeat again, similar to the time and space locality principles in the cache memory design.

Text layout information is recovered at the end of the system's process, using the font and format information obtained from the scanned images. The system also tells the difference between word wrap and hard wrap, so words are rearranged with respect to the page width and the number of lines in the generated text may not be identical to the number of lines in the original image.

The main research focus of this dissertation is the recognition process of segmented handwritten characters. This overall process integrates the steps of preprocessing, training and recognition, which will be discussed in Chapters 3, 4, and 5, respectively. The word and sentence recognition will be described in Chapter 6.

1.3 Significance of the Study

The system as newly developed in this dissertation contributes to digital document management and can be customized for first time users as well as for advanced users making use of different fonts and styles.

Any weights that are used for the first recognition phase are generated and integrated into the system before being distributed to the end user. Users are able to train the neural network again with their own font, and then every user could have a series of different but individualized weights.

Due to software engineering requirements, users of this system could manage the input and output layers, but they will not be able to manipulate the internal operations of the designed system.

1.4 Structure of the Research

In order to illustrate the approach and algorithm clearly, Chapter 2 briefly introduces the motivation for the handwritten character recognition, and gives a brief background of neural networks. The chapter explains why neural networks are typically used in the recognition system, and are then contrasted to other methods such support vector machines or generalized decision functions (linear and nonlinear). Three traditional neural networks (NN) that are commonly used in character recognition are briefly described in Chapter 2. These are convolutional NN, pyramidal NN, and improved pyramidal NN. Convolutional neural networks take advantage of neighbor dependency, reduce the number of weights significantly, and perform better than pyramidal and improved pyramidal neural networks.

Chapters 3 through 5 contain the main contributions to the dissertation. In these chapters, a novel approach is introduced for effective recognition of handwritten characters on the basis of an extensive database of handwritten characters that include lower and upper cases. The approach includes two preprocessing operations, architecture of the neural network, higher order training methods, and bounded stopping condition. The new approach not only improves the recognition rate, but also accelerates the training process. Mathematical formulas are included to prove the accuracy and efficiency. It should be noted that even though the training phase is computationally taxing, what matters in the end is the processing time of the testing phase once the architecture of the ANN is consolidated, and the accuracy in the recognition rate.

In Chapter 3, the pros and cons of large input size are discussed. Large input size significantly reduces error rate on training set, but improves the results for testing set only in some cases. Two preprocessing operations proposed in this chapter simulate various human writing styles and generate new characters from existing images. The horizontal shearing produces the greatest universal improvement to the training result.

From the output point of view, Chapter 4 describes the relationship between norm distance and error function. Bounded stopping condition accelerates the training process significantly and gives better or at least the same result.

Chapter 5 describes the architecture and training methods of the neural network. The higher order training method in this chapter is one of the major contributions of the dissertation, although it takes 50% more training time than traditional second order method. Researchers usually encounter some problems while training neural networks, such as local minima and over-training. Next-minimal training method is known to largely reduce the negative effects. It is observed that these two methods when appropriately put to work together could improve both the efficiency and accuracy of the neural network.

Chapter 6 provides a rule based segmentation method and applies the neural network to word on sentence recognition. Segmentation and recognition depend on each other when recognizing words and sentences. A good recognition system helps improve the segmentation mechanism. Incorrect segmentation will be rejected and a new segmentation is expected. Dictionary based correction is used to determine some unclear or uncertain characters. All of these above can improve the performance of an off-line handwritten text recognition system.

Finally, Chapter 7 concludes the disertation, describes potential improvements of the novel technologies, and envisions future research directions so that the recognition system could be extended to more application fields.

CHAPTER II

Background and Conventional Methods of Handwritten Text Recognition

2.1 Motivation for Developing Handwritten Text Recognition

The motivation of this research came about with our meetings with blind persons from Lighthouse of Broward in our research endeavor to build a book reader as a fully integrated system that is fast and yet inexpensive and effective with a high reading accuracy [Adjouadi et al. 2007]. One of their priority requests is providing the ability to read personal mail through such a device.

In this research endeavor, in terms of typewritten text, we are keenly aware of the technological advances on this front introduced by Minolta and by Xerox, and lately by Google through a US patent [Google Inc, 2009]. However what we seek and confront through the proposed new system is to include handwritten characters without compromising the important features of portability, affordability, and ease of use for blind individuals to be able to read letters and other text that is handwritten.

The system we are building, as shown in Figure 2.1 requires only two inexpensive lightweight cameras connected through USB ports, and a document holder. Such a system works using regular lighting of a home, office, library or lab environment. This system is consequently designed to address through software development (a) the recognition of text whether it is typed or handwritten; (b) and producing an audible and accurate read out through a speech synthesis engine [Amin et al., 2000; Ishitani, 2005; Nagy, 2000; Pajdla et al., 1997; Rao, 2005; Vass, 2003, Pilu, 2001, Pollard et al., 2005]. Thus, through the addition of two inexpensive cameras, any computer that is augmented with this type of software will be able to read any document with text that is either typed or handwritten. This design extends therefore the practical capabilities of the *Talking Camera* by Kurzweil [Cheung, 2006] and the flattening process of curved documents [Liang et al. 2005]. An inexpensive automatic page turner (not shown) can be included for comfort of use. The two processes shown within the red box in Figure 2.1 constitute the research thrust of this dissertation.

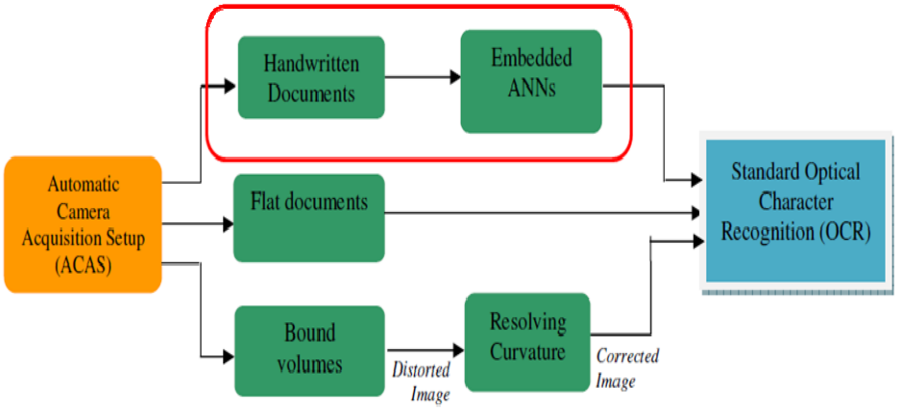
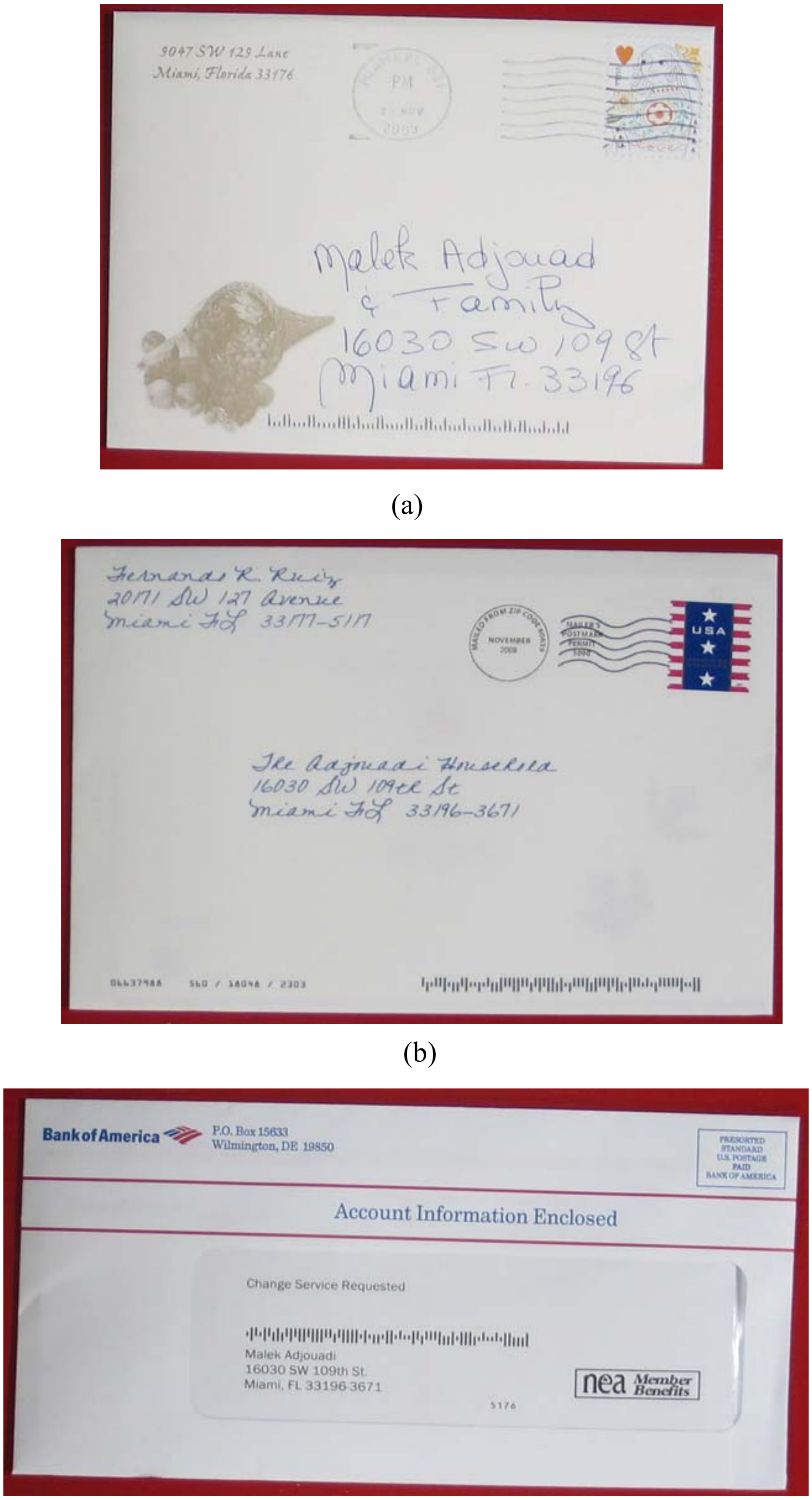


Fig. 2.1 Functional structure of the book reader design

The document reader was tested to recognize handwritten and typed characters from typical mail, as shown in Figure 2.2, these are used here as examples to express the difficulty in dealing with handwritten characters. Note that when the letters (a) and (b) are handwritten, the book reader performs poorly since the module for handwritten characters was not yet implemented in the book reader. However for typewritten mail (c), the document reader provides very good reading results as given in written form in Figure 2.3, even when the envelopes are not properly aligned.

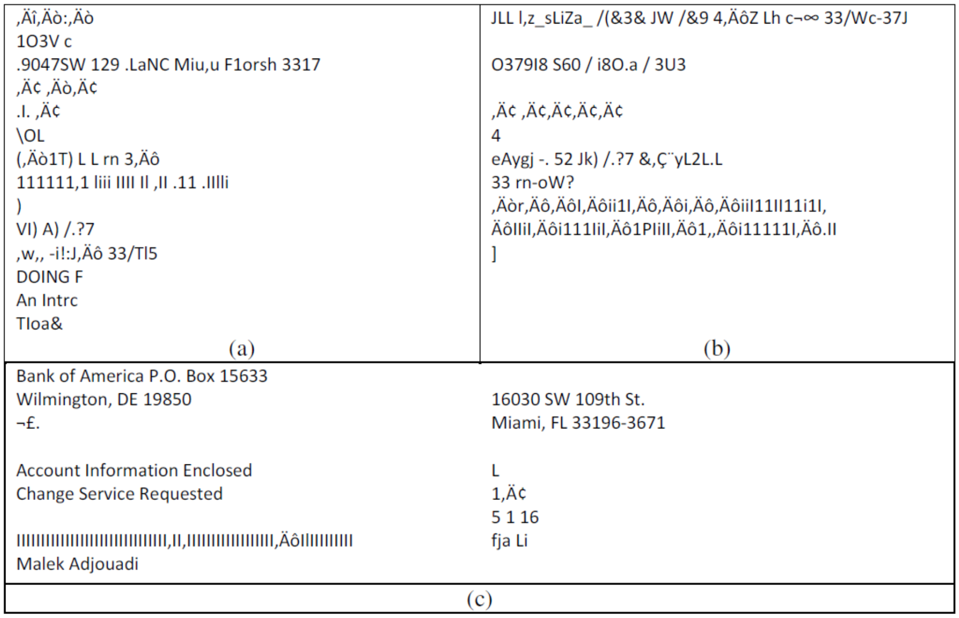


(a)

(b)

(c)

Fig. 2.2 Typical mail that can be received



```
,Äî,Äò:,Äò
1O3V c
.9047SW 129 .LaNC Miu,u F1orsh 3317
,Ä¢ ,Äò,Ä¢
.I. ,Ä¢
\OL
(,Äò1T) L L rn 3,Äô
111111,1 liii IIII II ,II .11 .IIlli
)
VI) A) /.?7
,w,, -i!:J,Äô 33/Tl5
DOING F
An Intrc
Tloa&
```

(a)

```
JLL l,z_sLiZa_ /(&3& JW /&9 4,ÄôZ Lh c¬∞ 33/Wc-37J

O379I8 S60 / i8O.a / 3U3

,Ä¢ ,Ä¢,Ä¢,Ä¢,Ä¢
4
eAygj -. 52 Jk) /.?7 &,Ç¨yL2L.L
33 rn-oW?
,Äòr,Äô,ÄôI,Äôii1I,Äô,Äôi,Äô,ÄôiiI11II11i1I,
ÄôIIiI,Äôi111IiI,Äô1PIiII,Äô1,,Äôi11111I,Äô.II
]
```

(b)

```
Bank of America P.O. Box 15633
Wilmington, DE 19850                     16030 SW 109th St.
¬£.                                      Miami, FL 33196-3671

Account Information Enclosed             L
Change Service Requested                 1,Ä¢
                                         5 1 16
IIIIIIIIIIIIIIIIIIIIIIIIII,II,IIIIIIIIIIIIIIII,ÄôIIIIIIIIII    fja Li
Malek Adjouadi
```

(c)

Fig. 2.3 Automated reading results of the documents shown in Figure 2.2

2.2 Introduction to Optical Character Recognition

Optical character recognition (OCR) is usually referred to as an off-line character recognition process to mean that the system scans and recognizes static images of the characters. On the other hand, on-line handwritten character recognition requires the identification of the movements of a pen as it is writing on a special device.

Off-line, machine printed characters and handwritten digits can be well recognized with accuracy higher than 99%. Since only human review can achieve complete recognition, the two problems are considered well solved. Currently there are several commercial software and tools available for this specific usage. However, the focus of this research is on the off-line handwritten character recognition process.

2.3 Approaches of Segmented Handwritten Character Recognition

First of all, the process of how human beings recognize characters should be considered. Most people learn the alphabet letters in kindergarten or elementary schools. After a period of study and practice, humans are able to recognize large or small, upside-down or rotated, standard or special font characters.

It is important to emphasize that it is through continuous learning that humans are able to recognize a wide array of handwritten and typed characters. Some system models such as Support Vector Machines (SVM), Decision Functions, and Artificial Neural Networks do attempt to simulate these amazing human learning activities.

2.3.1 Support Vector Machines

Support Vector Machines (SVMs) are maximum margin classifiers, and are based on the principle of structural risk minimization. Given a training set, the SVM finds a linear hyperplane that separates classes according to different labels with a maximum interclass distance or margin [Burges, 1998].

Usually, SVM makes a binary classification and gives a YES or NO answer. For the n-class problem, a SVM binary tree is needed and the depth of the tree is at least *log(n)*. The SVM can only exclude half of the candidates each cycle and is not efficient for a large number of classes.

2.3.2 Decision Functions

A Decision Function works as a line, which gives the binary classification and divides the plane into halves. For a high dimensional decision function, the function is considered as a plane (2-D) or hyper-plane (more than 2-D) and divides the hyper-space into halves. The advantage is that more than one decision function can work together and generate result at once, even for a multiclass problem. However, decision functions may not always be linear, and higher order decision functions make the computational increase almost exponential in unexpected running time, and with no guarantee for the result to converge to an optimal solution when the problem is nonlinear in nature.

2.3.3 Artificial Neural Networks

An Artificial Neural Network can be considered as the combination of support vector machines and decision functions. The network produces outputs immediately at the output layer, and the running time is independent of the number of outputs. The neural network works as a multi-layer decision function, while the extracted features are stored in the hidden layers randomly. Since neural networks execute feature extraction implicitly, users do not need to specify or be aware of the relationships between neurons and features.

2.4 Introduction to Artificial Neural Networks

An Artificial Neural Network (ANN), as exemplified in Figure 2.4, is a computational model with an interconnected group of neurons, being inspired by the structure of biological neural networks.

In pattern recognition, a neural network is an adaptive system that changes its weights and biases based on an external training set and an actual output that are predefined during the learning phase. Modern neural networks are nonlinear statistical data modeling tools used to represent complicated relationships between inputs and outputs [Zant et al., 2008; Yamada and Nakano, 1996; Gader et al., 1997; Kimura et al., 1997].

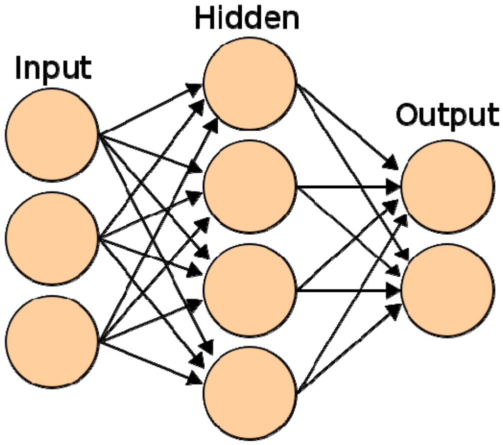


Fig. 2.4 Neural network example

In this figure, at each layer, we have

$$Y_j = f(y_j) = f(\sum_{i=0}^{n-1} w_{i,j} X_i + b_j) \tag{2.1}$$

where $f(\cdot)$ is the activation function and y_j is the weighted summation. The output of each layer depends on the input of the previous layer and the weights between both layers. The formula may be different due to different actual connections.

Usually, a sigmoid or hypertangent function is applied after each layer to make a binary decision. The functions that introduce nonlinearity to the network are called activation functions. Therefore, higher order items are not needed to add to the weighted sum. That’s why neural networks are used in this dissertation to recognize segmented handwritten characters.

2.5 Database of Handwritten Characters

The database that was used for the practical experimentation phase of this dissertation is constructed from the National Institute of Standards and Technology (NIST) special database 19 (SD-19), which contains binary images of handwritten characters of size 128 * 128 pixels. SD-19 has seven series of handwritten sample forms with isolated characters and complete forms from different groups of writers. The training set is complete with 10, 000 examples from each series, to make a full set of 70, 000 training patterns. The test set is constructed in the same way and contains 10, 000 test patterns.

2.5.1 Gray Scale

The original black and white (binary), 128 * 128, images from NIST SD-19 are resized to a smaller size using an anti-aliasing technique. The gray levels of the resulting images come from the relative portion occupied by black and white pixels in a specific area.

In Figure 2.5, the small rectangle represents a single pixel in the resulting smaller image. The number of black pixels (possibly fractional) in the rectangle divided by the number of total pixels gives the corresponding gray level value. In this process, the pixel polarity is defined to be 255 for black and 0 for white conform to the standard set out by NIST.

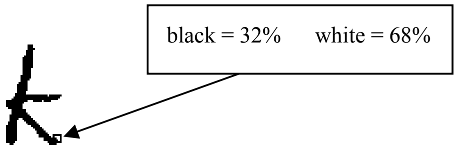


Fig. 2.5 How to assign gray level values when zooming out a bi-level image

2.5.2 Center of Mass

Determining the center of mass of handwritten characters is important for improving the recognition rate. Some people tend to draw a long tail to finish certain letters. Normalizing the characters with a long tail will compress and misplace useful information. It is therefore an improved solution to locate the center of mass and position this point at the center of the destination field.

First of all, original images should be normalized while preserving their aspect ratio to fit a smaller size region of 20 * 20. The center of mass is then given by the formula

$$R = \frac{\sum m_i r_i}{\sum m_i} \tag{2.2}$$

In this equation, r_i is the position pair and m_i is the gray level. When constructing training or test patterns, the centers of mass of normalized characters should be superimposed or kept as close as possible to the center of the patterns. The pattern size remains 28 * 28, which means that the center of mass can be up to 4 pixels away from the geometry center in any direction.

Although some information is lost due to further scaling to 20 * 20, balancing center of mass ensures that useful features of the same character be placed at almost the same position among patterns. Not surprisingly, analysis of the results shows that the overall performance has improved after this process.

2.6 Comparison of Different Neural Network Methods

In this section, three different neural networks are described and compared, convolutional NN, pyramidal NN, and improved pyramidal NN.

Convolutional neural networks have several variations. LeNet-1 network [LeCun et al., 1992], as illustrated in Figure 2.6, is one of the most effective systems. The major new point is that there are two types of layers alternatively and several maps in a same layer. The maps are designed equally but will eventually store different feature of image, due to different initial weights and biases. The greatest advantage is the weight-sharing policy, which significantly reduces the number of independent weights and makes learning and training process rather easy.

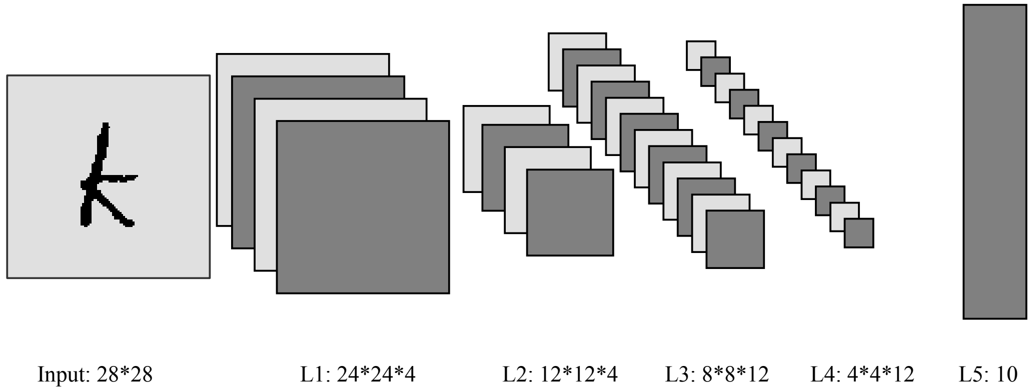


Fig. 2.6 Architecture of LeNet-1 as a five-layer neural network

Pyramidal neural network was first presented by [Phung and Bouzerdoum, 2007]. There is only one map in each layer and no subsampling layer. All 2-D layers are pyramidal layers. Since no convolution is needed, the connections of the network are straightforward, and the training and testing processes are very fast. Although the network does not give as good result as the convolutional neural network, faster recognition speed is an attraction, especially in time-sensitive cases. This type of architecture is shown in Figure 2.7.

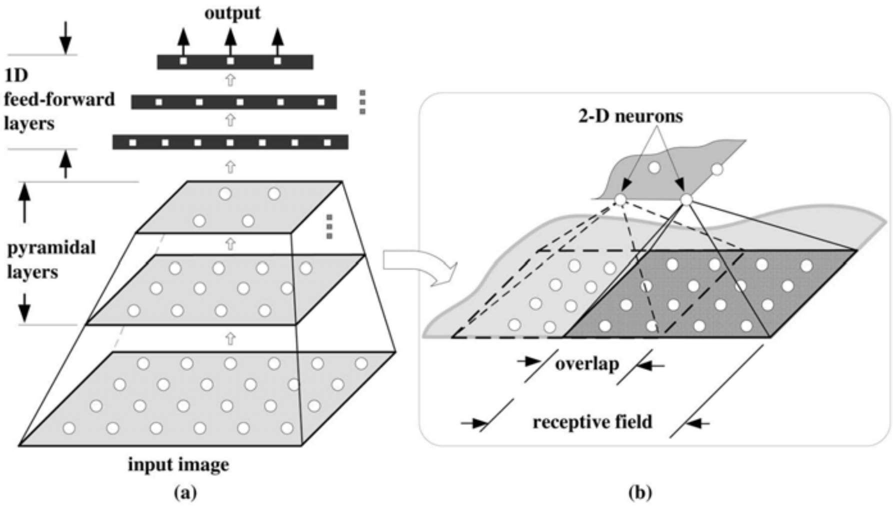


Fig. 2.7 Architecture of pyramidal NN

(a) Network layers and (b) overlapping receptive fields

Research on Improved Pyramidal Neural Network (IPNN) was first published by [Han et al., 2009]. IPNN has a hierarchical multilayered architecture with two types of processing layers: 2-D layers for feature extraction and dimensional reduction and 1-D layers for classification as illustrated in Figure 2.7. The 2-D layers consist of alternative pyramidal and subsampling layers. The first pyramidal layer is connected to the input image, followed by repeating subsampling and pyramidal layers. The last subsampling layer is fully connected to 1-D layers, which processes the features produced by the pyramidal layers. There are four 2-D layers and two 1-D layers in this proposed method. It is expected that the insertion of subsampling layers promotes efficiency and the increase of feature maps improves accuracy. The output of the last 1-D layer is considered as pattern category, which is composed of 26 units: one per letter. When a pattern belonging to class *i* is presented, the desired output is 1 for the *i*th output unit and 0 for other output units.

The architecture of IPNN is as shown in Figure 2.8. It seems similar to the convolutional neural network, but the connections between layers are pyramidal, not convolutional.

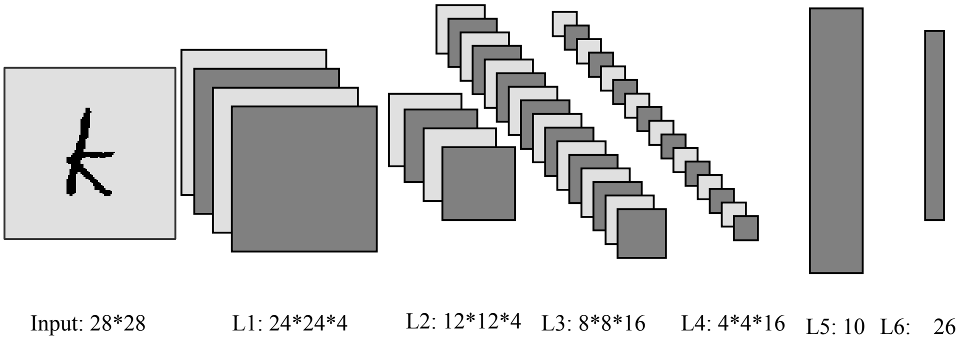


Fig. 2.8 Architecture of IPNN, a six-layer neural network

In this early phase of the research, all three ANN architectures have been implemented for comparative purposes and to gauge the merits of each. Table 2.1 provides recognition results for the three neural networks, as well as the training and testing time. Because there is great difference between recognizing upper and lower case characters, results are given separately.

Table 2.1 Comparisons of three different neural networks

Name of NN	Convolutional NN	Pyramidal NN	IPNN
Error Rate (Upper Case)	3.2%	7.0%	5.5%
Error Rate (Lower Case)	8.5%	15.9%	9.3%
Training Time (50 cycles)	60 minutes	12 minutes	25 minutes
Testing Time (10,000 patterns)	4.0 seconds	0.8 seconds	1.6 seconds

2.7 Discussion

From the results given in Table 2.1, it can be observed that the convolutional neural network has the lowest error rate while the pyramidal neural networks are the fastest to train. The IPNN on the other hand is designed to be as efficient as the pyramidal neural network, and as accurate as the convolutional neural network. However, time efficiency and recognition accuracy still require some improvements.

CHAPTER III

Preprocessing Analysis

3.1 Introduction

This Chapter introduces two useful preprocessing techniques prior to initiating the handwritten character recognition process, discusses the influence of different input size on error rate and training time, and gives a simple feature extraction strategy for predicting a possible subset.

3.2 Input Size

Input size refers to the size of the input image. It is common sense to assume that large input size will help the network recognize characters, but does not guarantee it. It is very hard to recognize a small input image, because a lot of useful information would have been lost or discarded when compressing. However, large input size may introduce over-training for the neural network. The network possibly pays more attention to the detail rather than the shape of the characters. A small change in the character will cause a character recognition failure in the network.

3.2.1 Tradeoff between Recognition Rate and Training Time

First, the comparison of two different input sizes, 28 * 28 and 36 * 36 is considered, using the same training method without any preprocessing. The results obtained are provided in Table 3.1.

Table 3.1 Comparisons of different input size

Input Size	28 * 28	36 * 36
Number of Connections	98, 442	210, 642
Error Rate (Upper Case)	3.14	3.09
Training Time (Upper Case)	135 minutes	237 minutes
Error Rate (Lower Case)	8.95	8.48
Training Time (Lower Case)	194 minutes	361 minutes
Testing Time (Both Cases)	4.4 seconds	8.5 seconds

The results in Table 3.1 show that increasing the input size improves performance minimally, but it has a considerable positive impact on lower case character recognition. The trend is more obvious after the preprocessing steps of horizontal shearing and polynomial distortion are performed. These two preprocessing steps are discussed next in Sections 3.3 and 3.4. The reasoning is straightforward; since most upper case characters are composed of straight lines with fixed shapes and are easier to distinguish. Lower case characters are more flexible and malleable, so higher resolution is needed to differentiate one from another.

The neural network with larger input size has 114% more connections, but it needs only 75% (upper case) or 86% (lower case) more training time than the smaller one. The term ‘connection’ means a path from a neuron in previous layer to a neuron in next layer. Often many such connections along with a bias are required to produce the weighted summation. Since the training time and testing time are relatively short, a larger neural network is able to replace a smaller one without significant defect.

3.2.2 Error Rate on Training Set and Testing Set

Error rates on both sets usually go up and down simultaneously. However, if the neural network is trained again and again and more than the needed information is provided, the error rate on the training set continues to decrease but it will revert on the testing set. This situation is called over-training. The relationship between error rate on training and testing sets is shown in Table 3.2.

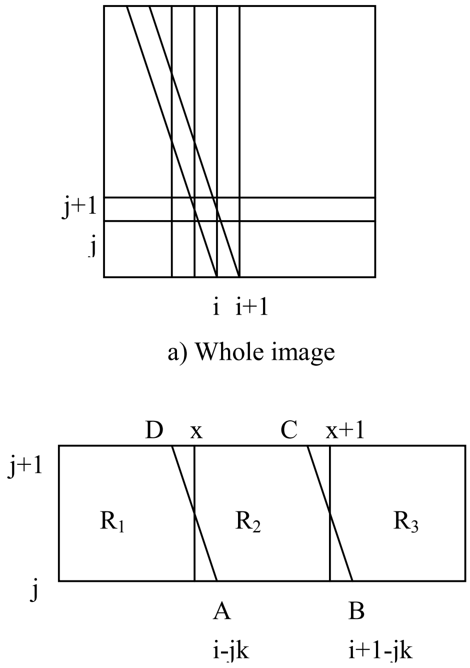
Table 3.2 Error rate (percent) of different input size

Input Size	28 * 28	36 * 36
Error Rate on Training (Upper Case)	2.02	1.13
Error Rate on Testing (Upper Case)	3.14	3.07
Error Rate on Training (Lower Case)	6.92	5.35
Error Rate on Testing (Lower Case)	8.95	8.48

The results shown in the Table 3.2 indicate that increasing the input size significantly reduces the error rate on the training set, but is much less effective on the testing set. The relationship between improved training set and stuck testing set is quite similar to over-training. That is why the input size should not be any larger, as 36 * 36 is a reasonable balanced tradeoff.

3.3 Horizontal Shearing

One effective method to improve the neural network is to increase the training set diversity and to keep the testing set consistent. The original character is horizontally sheared left and right to create two images. In the new image, the grayed scale at position (i, j) is the weighted summation of some gray scales in the original image, while the weights are computed by the intersection of the parallelogram *ABCD* and two or three small squares as shown in Figure 3.1.



a) Whole image

b) Local detail

Fig. 3.1 Horizontal Shearing

In Figure 3.1, *k* represents the slope between the slanted line (AD, BC) and the vertical line. In the example shown, the parallelogram *ABCD* intersects with three small squares R1, R2, R3. The areas of the intersections are defined as follows:

$$S_1 = \frac{1}{k}[x - i - (j+1)k]^2 \tag{3.1}$$

$$S_3 = \frac{1}{k}(i - jk - x)^2 \tag{3.2}$$

$$S_2 = 1 - S_1 - S_3 \tag{3.3}$$

where $x = \lfloor i - jk \rfloor$ and $\lfloor \cdot \rfloor$ is the floor function. Then the gray scale of new image is

$$G_{i,j} = S_1G_1 + S_2G_2 + S_3G_3 \tag{3.4}$$

Through observation, most characters in the database have their major bar within ±arctan0.15 away from the vertical line. Therefore, *k* with the value +0.15 and -0.15 are chosen respectively. Each pattern is performing two different shearing, so there are 210, 000 patterns in total to be trained.

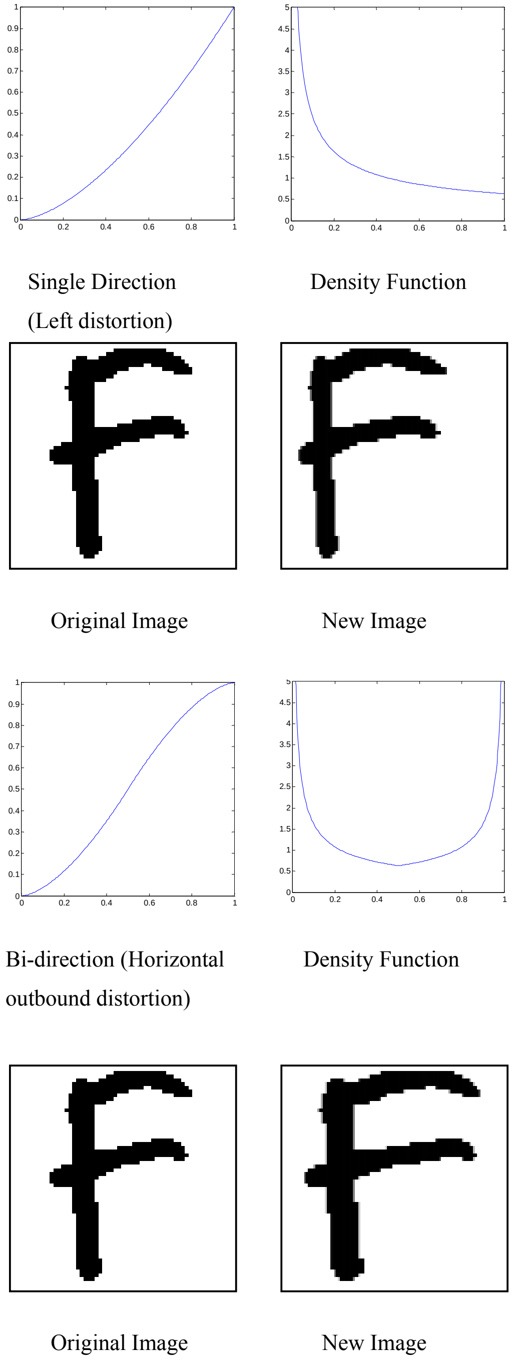
3.4 Polynomial Distortion

As indicated earlier, as some people tend to draw a long tail to begin or finish a character, this creates irregular written characters. *Distortion* is a simulation of the irregular writing, which enlarges one part of a character, and compresses the other part. This type of distortion is implemented by mapping old images to new ones.

Theoretically, distortion can be made towards any direction. In this dissertation the concentration is on four basic directions, left, right, up and down. Diagonal distortions are not included since they are considered to be combinations of existing distortions.

Besides single direction, bi-directional distortion is also effective. It can be considered as the combination of two single direction distortions, which fix the middle points of the images and push the other parts outbound or inbound. Thus each of horizontal and vertical distortion contains two subtypes. There are also four bi-directional polynomial distortions or eight in total. Figure 3.2 shows the comparison of two types of polynomial distortions.

Instead of performing 8 distortions on every pattern, an alternative method is applied. A random number 0-8 is generated for each pattern, so the pattern has an even possibility to perform a distortion of any direction or not at all. In this research, the general forms of each distortion are given in Table 3.3.



Single Direction (Left distortion)

Density Function

Original Image

New Image

Bi-direction (Horizontal outbound distortion)

Density Function

Original Image

New Image

Fig. 3.2 Polynomial Distortion Comparison

Table 3.3 General forms of polynomial distortions

Description	Note	General form
Single directional distortion, left or up	Distortion 1/2 to 1/3	$f(x) = x^{\alpha}, \quad 0 \le x \le 1$
Single directional distortion, right or down	Distortion 1/2 to 2/3	$f(x) = 1-(1-x)^{\alpha}, \quad 0 \le x \le 1$
Bi-directional distortion, outbound	Expand 1/4 and 3/4 to 1/6 and 5/6	$f(x) = \begin{cases} \frac{1}{2}(2x)^{\alpha} & 0 \le x < \frac{1}{2} \\ 1-\frac{1}{2}(2-2x)^{\alpha} & \frac{1}{2} \le x \le 1 \end{cases}$
Bi-directional distortion, inbound	Contract 1/4 and 3/4 to 1/3 and 2/3	$f(x) = \begin{cases} \frac{1}{2}-\frac{1}{2}(1-2x)^{\alpha} & 0 \le x < \frac{1}{2} \\ \frac{1}{2}+\frac{1}{2}(2x-1)^{\alpha} & \frac{1}{2} \le x \le 1 \end{cases}$

To achieve the distortion target listed in Table 3.3, the constant coefficient used is given by the following relationship:

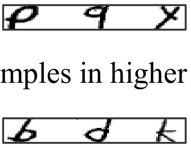
$$\alpha = \log_{1/2}(1/3) = 1.585 \tag{3.5}$$

3.5 Feature Extraction

Feature extraction is a special form of dimensional reduction in pattern recognition. It finds some common features according to target properties, and converts the original problem to an easier sub-problem. The '*black-box*' nature of artificial neural networks prevents from impeding the practical application. Therefore, techniques for feature extraction from ANNs are introduced to aid in the explanation of their classification decisions [Barakat and Bradley, 2007].

In character recognition field, feature extraction follows one or more human defined rules, and gives a possible subset.

One rule used in this research is the position of the center of mass. After carefully analyzing common handwriting, it is noticed that some letters have higher centers of mass, like p, q and y, while others have lower centers of mass like b, d and k as can be observed in Figure 3.3. Accordingly, we can build a possible subset or at least a possibility table before feeding these images into the neural network.



(a) Examples in higher center subset.

(b) Examples in lower center subset.

Fig. 3.3 The difference between higher and lower center of mass

Another rule to help predict the possible subset is the height of character. If the height is obviously low, like m, s and w, we don't need to check the position of the center of mass; instead it is predicted to be a low height subset directly. This rule only applies when the character is in a word or a sentence. It is not able to retrieve the actual height of a segmented normalized character.

In this process, one could first exclude the upper case characters and divide 26 letters into 4 subsets depending upon the height and the position of center of mass: low height, higher center, lower center, and others. If the center of mass of an input image is higher than a given threshold, a possible subset is the union of high center and others. If the center of mass is lower than a given threshold, a possible subset is the union of low center and others. Otherwise a possible set is the whole character set. The threshold is determined by observing the position of existing characters, usually 0.4 for low center characters, and 0.6 for high center characters.

Four subsets can be divided easily as shown below and the overall flow chart is given in Figure 3.4.

Low Height: a, c, e, i, m, n, o, r, s, t, v, w, x, z

High Center: g, p, q, y

Low Center: b, d, h, k, l, t,

Others: f, j

Interestingly, entering a subset recognizer does not guarantee to produce a character in the subset. Anytime when the system is unsure about which subset the input belongs to, or there is a conflict between the output and the rules, the network should always try the largest set, i.e. the whole alphabet set, in order not to miss the correct answer from the very beginning. Figure 3.5 shows the flow chart of low center recognizer, while others are similar.

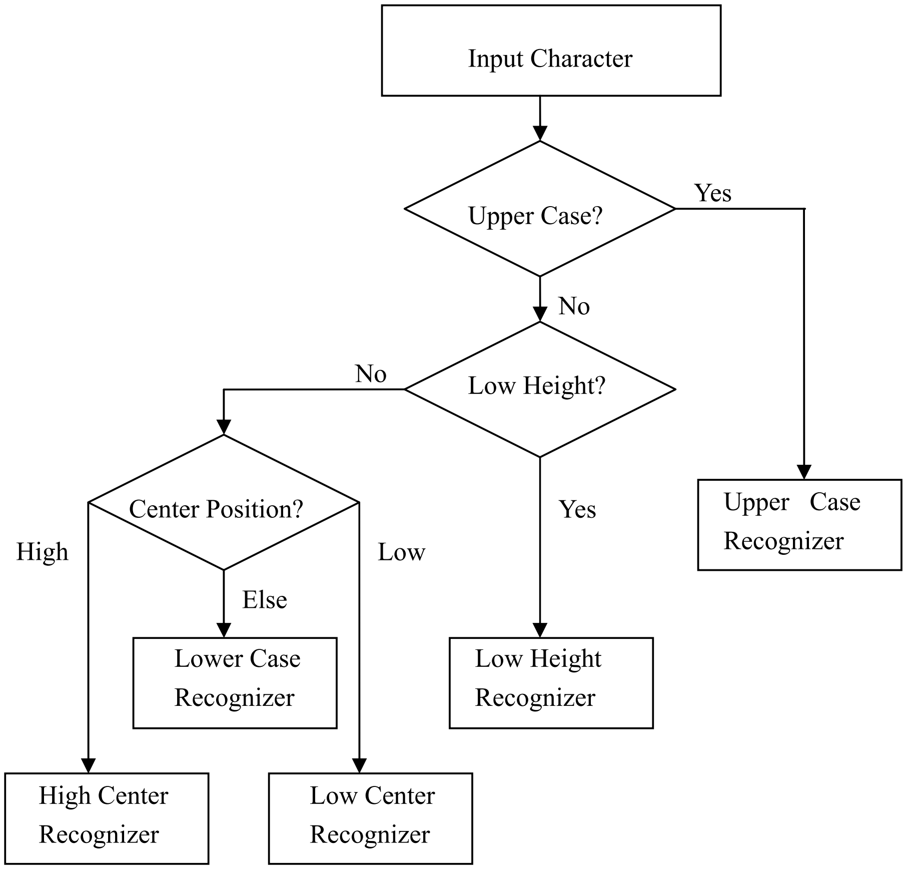


Fig. 3.4 The overall flow chart

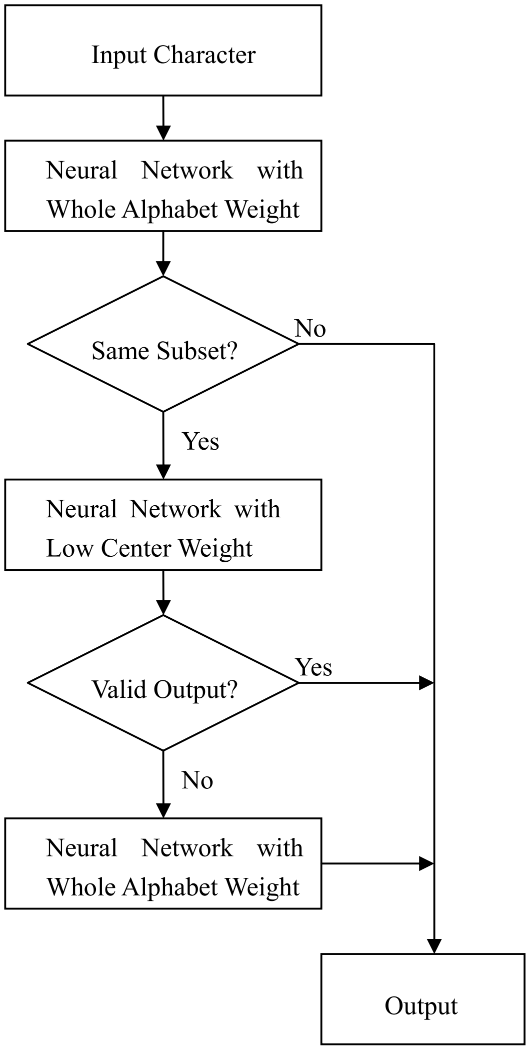

Input Character
Neural Network with Whole Alphabet Weight
Same Subset?
No
Yes
Neural Network with Low Center Weight
Valid Output?
Yes
No
Neural Network with Whole Alphabet Weight
Output

Fig. 3.5 The inner flow chart of low center recognizer

Figures 3.4 and 3.5 are structured to determine which neural network the testing data will be sent to. The neural networks are trained with different character sets to generate different weights. Detailed training methods and output conditions are discussed in Chapters 4 and 5.

CHAPTER IV

Output Improvement Analysis

4.1 Norm Distance and Error Functions

Researchers usually treat the Mean Square Error (MSE) function as the standard error function, which is the 2-norm distance between the theoretical value and actual output. In this dissertation, the neural networks trained with 4-norm (quartic) error function is found to have better performance and lower error rate than the mean square error function.

Let O_{dk} be the desired output at interface k, and O_k be the actual output. The mean square error is then given by:

$$MSE = \frac{1}{N}\sum_{k}(O_k - O_{dk})^2 \tag{4.1}$$

Similarly, mean quartic error is defined as,

$$MQE = \frac{1}{N}\sum_{k}(O_k - O_{dk})^4 \tag{4.2}$$

Since the constant coefficient and the square root do not change the relative value of the errors, MSE and MQE are equivalent to 2-norm error and 4-norm error, respectively.

According to the definition of norm, the infinity norm error turns out to be the maximum value of the errors.

$$INE = \max_{k}\left|O_k - O_{dk}\right| \tag{4.3}$$

The proposed neural network is thus designed to minimize the mean quartic error, and is henceforth called the Quartic Neural Network (QNN).

4.2 Stopping Condition: Bounded Training Method

A hyper-tangent function is used as the activation function after each layer.

$$y = f(x) = A\tanh(Sx) \tag{4.4}$$

Where $S = \frac{2}{3}$ and A is set at 1.7159 to make $f(1) = 1$ and $f(-1) = -1$. Figure 4.1 illustrates the activation function and its saturation region.

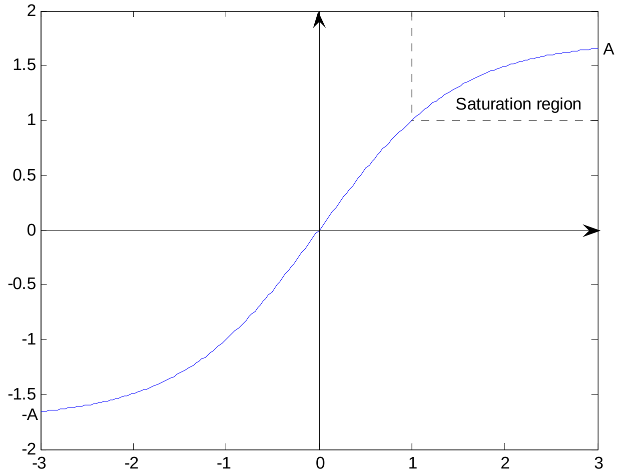


Fig. 4.1 Activation function and saturation region

In this approach, only whether the correct result is obtained is considered, but how far away it is from the desired value A is not as important. After analyzing each training pattern, the channel with the highest value is considered to be the output generated by the network. If the system gives a correct output and the output value locates in the saturation region, the errors (no matter what norm) are all set to 0 and the training process of current pattern is completely skipped. The pattern is considered correctly identified and the output does not need to be moved towards the theoretically desired value.

This modified mean quartic error is as defined in equation 4.5 below:

$$MQE = \begin{cases} 0 & \text{if } \mathrm{O_i} > 1 \text{ and } \mathrm{O_{di}} = A \\ \dfrac{1}{N}\sum_{k}(O_k - O_{dk})^4 & \text{otherwise} \end{cases} \tag{4.5}$$

where O_{di} and O_i denote any desired and actual output value. As the training process continues, more and more patterns are identified and located in the saturation region, so the training process will be significantly faster than the full cycle method.

Because upper case characters are easier to be trained and recognized than lower case ones, there are more patterns' outputs located in the saturation region and the bounded training method saves more time. According to the experimental results, the bounded training method saves 44.8%-64.1% training time for upper case character recognition, and 29%-45.6% for lower case character recognition.

As shown in Tables 4.1 and 4.2, the bounded method saves training time but keeps the same level or even achieves better recognition rate than the full cycle method. First and second order methods are generally defined based on first 1 or 2 items of Taylor's expansion. Semi-third order method, although the results of it are shown here for comparative purposes, is introduced in Chapter 6 given to highlight its merits and the impressive results that were obtained.

Table 4.1 Training time comparison (upper case)

Training Method	Full Cycle Method		Bounded Method		
	Training Time	Error Rate	Training Time	Saved Time	Error Rate
First Order Method	330 minutes	3.47	182 minutes	44.8%	3.37
Second Order Method	353 minutes	3.28	189 minutes	46.5%	3.36
Semi-Third Order Method	660 minutes	3.14	237 minutes	64.1%	3.09

Table 4.2 Training time comparison (lower case)

Training Method	Full Cycle Method		Bounded Method		
	Training Time	Error Rate	Training Time	Saved Time	Error Rate
First Order Method	328 minutes	8.51	233 minutes	29.0%	8.35
Second Order Method	354 minutes	8.04	244 minutes	31.1%	8.08
Semi-Third Order Method	664 minutes	8.37	361 minutes	45.6%	8.48

4.3 Experimental Results

This section gives the training results of different input sizes and different case characters trained with different norm error function and second order training method. The quartic error function always has the lowest error rate in most situations.

Table 4.3 Training process of upper case characters and 28*28 input size

Error Function	6 cycles	12 cycles	24 cycles	48 cycles	Final
2-norm error	6.15	5.68	5.56	4.69	4.14
4-norm error	4.33	3.84	3.74	3.39	3.35
6-norm error	4.73	3.99	3.61	3.51	3.38
Infinity norm error	4.84	4.31	4.19	3.98	3.43

Table 4.4 Training process of upper case characters and 36*36 input size

Error Function	6 cycles	12 cycles	24 cycles	48 cycles	Final
2-norm error	4.86	4.36	4.32	3.80	3.52
4-norm error	4.27	3.60	3.43	3.42	3.36
6-norm error	4.15	3.46	3.42	3.38	3.29
Infinity norm error	4.32	3.84	3.64	3.45	3.23

Table 4.5 Training process of lower case characters and 28*28 input size

Error Function	6 cycles	12 cycles	24 cycles	48 cycles	Final
2-norm error	12.36	12.06	11.38	10.75	10.42
4-norm error	10.55	9.53	9.24	9.01	8.28
6-norm error	11.35	10.88	10.12	9.87	9.69
Infinity norm error	12.30	11.44	11.04	10.47	9.58

Table 4.6 Training process of lower case characters and 36*36 input size

Error Function	6 cycles	12 cycles	24 cycles	48 cycles	Final
2-norm error	10.89	10.56	9.79	9.45	9.49
4-norm error	9.76	8.90	8.54	8.05	8.08
6-norm error	10.66	10.23	9.22	9.01	8.93
Infinity norm error	11.12	10.43	9.97	9.29	8.92

In the remaining Chapters of this dissertation, unless specified otherwise, mean error function is replaced by quartic error function, and full cycle training is replaced by bounded training method.

CHAPTER V

Architecture and Training Methods

5.1 Architecture of Quartic Neural Network

The Quartic Neural Network (QNN) as proposed below is derived from the Convolution Neural Network (LeNet-1). The difference locates in the input size, the error function and the connections between layer 2 and layer 3. In this section, we describe the architecture and mathematical implementation of QNN in details.

5.1.1 Detailed Description and Mathematical Representation

The QNN is structured into five layers, denoted by L1, L2, L3, L4 and L5 as shown in Figure 5.1. The output of one layer serves as input to the next layer. Layer L1 and L3 are shared weights convolutional layers, while L2 and L4 are sub-sampling layers. L5 is output layer with 1-D linear vector of full-connected weights.

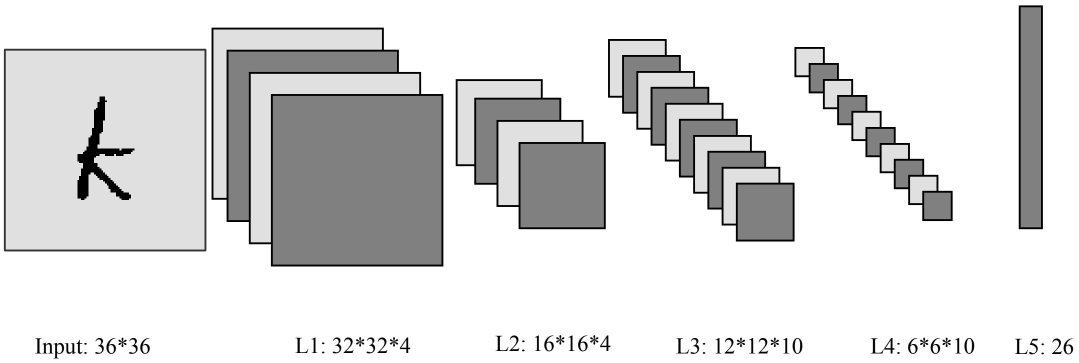


Fig. 5.1 Architecture of Quartic NN, a five-layer convolution neural network

The input is a 36 * 36 gray level plane. As a convolutional layer, L1 is composed of 4 independent 32 * 32 feature maps. Each unit in a feature map takes its receptive field from a 5 * 5 square neighborhood area on the input plane. All of the

32*32 units in map 1 of L1 use the same set of 25 weights, plus 1 bias. Of course, units in another map (still belong to L1) share another set of 25 weights and 1 bias. There are 106,496 connections and only 104 independent parameters from input layer to L1. The different paths from the previous layer to the next layer define connections. Independent parameters refer to the connections with unrelated weights or biases. Table 5.1 provides all of the nomenclature and related mathematical relations.

Table 5.1 Notations for QNN

Description	Symbol	Note
Input image	$x_{i,j}$	$1 \le i \le 36,\ 1 \le j \le 36$
Layer index	l	$1 \le l \le 5$
Map index	n	$\begin{cases} 1 \le n \le 4 & \text{For layer 1 and 2} \\ 1 \le n \le 10 & \text{For layer 3 and 4} \end{cases}$
Convolutional weight associated with layer *l*, map *n*, at position (i, j)	$w_{i,j}^{l,n}$	$0 \le i < 5, 0 \le j < 5$
Subsampling weight associated with layer *l*, map *n*	w_n^l	
Bias of map *n* in layer *l*	b_n^l	
Weighted summation of layer *l*, map *n*, at position (i, j)	$y_{u,v}^{l,n}$	*u* and *v* is bounded by actual size of each layer
Activation function	$f(\cdot)$	$f(x) = A\tanh(Sx)$, A and S are constants
Output of layer *l*, map *n*, at position (i, j)	$Y_{u,v}^{l,n}$	$Y_{u,v}^{l,n} = f(y_{u,v}^{l,n})$
Weighted from neuron *i* in layer *l*-1 to neuron *j* in layer *l*	$w_{i,j}^l$	
Final output	O_k, O_{dk}	actual, desired output, $1 \le k \le 26$

Therefore, the output of 2-D neural (u, v) in L1 can be expressed as:

$$Y_{u,v}^{1,n} = f(y_{u,v}^{1,n}) = f(\sum_{i=0}^{4}\sum_{j=0}^{4} w_{i,j}^{1,n} x_{u+i,v+j} + b_n^{(1)}) \tag{5.1}$$

where $y_{u,v}^{1,n}$ is weighted summation and $f(\cdot)$ is the activation function.

L2 is the subsampling layer, which is composed of 4 planes of size 16 * 16. Any unit in one of these planes takes inputs from 4 different units on the corresponding plane in L1. Each L1 unit appears in one connection and each L2 unit appears in four connections. The units in a plane share a same pair of weight and bias, so we have 4 weights and 4 biases. The output of L2 is defined as follows:

$$y_{u,v}^{2,n} = w_n^{(2)} \sum_{i=0}^{1}\sum_{j=0}^{1} Y_{2u+i,2v+j}^{1,n} + b_n^{(2)} \tag{5.2}$$

L3 is composed of 10 feature maps, each of which has 144 neurons in a 12 * 12 plane. The connections between L2 and L3 are shown in Table 5.2. The planes in L3 are receiving inputs from combinatorial planes in L2. From L2 to L3, there are 24 plane-to-plane connections, which are denoted by X in Table 5.2. Thus, there are 87,840 connections and 610 independent parameters in L3.

Table 5.2 Connections between L2 and L3

	1	2	3	4	5	6	7	8	9	10
1	X	X	X				X	X	X	
2	X			X	X		X	X		X
3		X		X		X	X		X	X
4			X		X	X		X	X	X

As shown earlier in Figure 5.1, L4 consists of 10 planes of size 4 * 4, and plays the same role as L2. The units in a plane share a same pair of weight and bias, so we have 10 weights and 10 biases, or 20 independent parameters altogether. The output of L3 and L4 are as defined in equations 5.3 and 5.4, respectively

$$y_{u,v}^{3,c(n)} = \sum_{n} \sum_{i=0}^{4} \sum_{j=0}^{4} w_{i,j}^{3,n} Y_{u+i,v+j}^{2,d(n)} + b_{c(n)}^{(3)} \qquad 1 \le n \le 24 \qquad (5.3)$$

$$y_{u,v}^{4,n} = w_{n}^{(4)} \sum_{i=0}^{1} \sum_{j=0}^{1} Y_{2u+i,2v+j}^{3,n} + b_{(n)}^{4} \qquad (5.4)$$

where $c(n)$ and $d(n)$ means related map index in L2 and L3 for connection n.

L4 is rewritten to be a column linear vector while being the input to succeeding output layer L5. The multi-planes 2-D and 1-D formats are interchangeable by $Y_{u,v}^{4,n} = Y_{36n+6u+v}^{(4)}$. The final network output L5 is fully connected to L4 and contains 26 units. There are 9,386 independent connections between L4 and L5 with the same number of trainable weights. When a pattern belonging to class *i* is presented, the desired output is *A* for the *i*th output unit and -*A* for other output units.

$$y_{j}^{(5)} = \sum_{i=1}^{26} w_{i,j}^{(5)} Y_{i}^{(4)} + b_{j}^{(5)} \qquad (5.5)$$

In summary, the network has 8,242 units, 210,642 connections and 10,128 independent parameters. QNN reorganizes the connections between L2 and L3, reduces the number of planes in L3 and L4, and decreases the total number of independent parameters.

5.1.2 Possible Variations

There are several possible variations of the architecture, among them is: 1) to perform larger convolution in L1 and same convolution in L3, 2) change the connections between L2 and L3, and 3) add another 1-D layer between L4 and final output. Based on the results of performed experiments with these three variations, the proposed architecture in this dissertation contributes the lowest error rate and the highest stability in multi-times trainings.

5.2 Semi-Third Order Training Method Related works

5.2.1 Mathematical Induction from Taylor's Series

There are several first and second order training methods as reported in previous works [Battiti 1992; Verma 1997; LeCun et al., 1998; Khashman 2008]. The third derivative of mean square error function is zero since it is quadratic. Thus, up to second order method can be used as in previous works. However, third and even fourth order training methods can be applied with mean quartic error function, which is a great advantage of the QNN approach.

The semi-third order training method converges sharply on the training set, but it does not guarantee a better recognition rate because of over-training. Based on the research performed in this dissertation, the semi-third order method has better performance in upper case character recognition, while it is less effective in recognizing lower case characters.

The third order method is derived from the first three terms of the Taylor series as in equation 5.6.

$$\begin{aligned}E(W+\Delta W) &= E(W)+\nabla E(W)^T\Delta W+\frac{1}{2}\Delta W^T\nabla^2 E(W)\Delta W \\ &+\frac{1}{6}[\Delta W^T\nabla^3 E(W)\Delta W]^T\Delta W+\cdots\end{aligned} \tag{5.6}$$

Let

$$\frac{\partial E(W+\Delta W)}{\partial \Delta W}=0 \tag{5.7}$$

Then

$$\nabla E(W)^T+\Delta W^T\nabla^2 E(W)+\frac{1}{2}\Delta W^T\nabla^3 E(W)\Delta W=0 \tag{5.8}$$

If all the off-diagonal elements are eliminated, Equation 5.8 becomes an *n* quadratic equation, where *n* is the number of independent parameters.

$$\frac{1}{2}\frac{\partial^3 E}{\partial w^3}\Delta w^2+[\frac{\partial^2 E}{\partial w^2}+\lambda]\Delta w+\frac{\partial E}{\partial w}=0 \tag{5.9}$$

In Equation 5.9, *w* and *Δw* are individual values of the vector *W* and *ΔW*, respectively. Assume that

$$\frac{\partial^2 E}{\partial w^2}\geq 0 \tag{5.10}$$

A positive constant λ is added to the second derivative coefficient to keep it strictly positive and prevent the absolute value from becoming too small.

The method for calculating the derivatives will be discussed in Section 5.2.2. Here *a*, *b* and *c* are introduced to represent the three coefficients. The equation takes the form of the standard form equation and the linear coefficient *b* is positive.

$$a\Delta w^2+b\Delta w+c=0 \tag{5.11}$$

For the second order method, the quadratic coefficient *a* is zero and the linear solution is given by

$$\Delta w = -\frac{c}{b} \tag{5.12}$$

However, the situation is more complicated for third order method.

If $ac > 0$, the equation may have two, one or zero real roots, depending on the sign of the discriminant.

If the discriminant Δ is negative, the quadratic equation doesn't have a real root. However, the error function $E(W + \Delta W)$ is impossible to be monotonic. This situation is due to some dropped items when computing the second and third derivates, and higher order items of Taylor series. In this case, the quadratic item is dropped and the linear solution is used instead of complex roots.

If the discriminant is non-negative, the equation has two or one real root(s). Since the linear coefficient *b* is strictly positive, the plus root is chosen to be the solution of Equation 5.11, because it is closer to and eventually limited to the linear solution as the quadratic coefficient *a* approaches 0.

$$\lim_{a \to 0} \Delta w_+ = \lim_{a \to 0} \frac{-b + \sqrt{b^2 - 4ac}}{2a} = -\frac{c}{b} \tag{5.13}$$

Proof:

$$\lim_{a \to 0} \sqrt{b^2 - 4ac} = \lim_{a \to 0} |b| \sqrt{1 - \frac{4ac}{b^2}} = b(1 - \frac{2ac}{b^2}) = b - \frac{2ac}{b} \tag{5.14}$$

$$\lim_{a \to 0} \frac{-b + \sqrt{b^2 - 4ac}}{2a} = \lim_{a \to 0} \frac{-b + (b - \frac{2ac}{b})}{2a} = -\frac{c}{b} \tag{5.15}$$

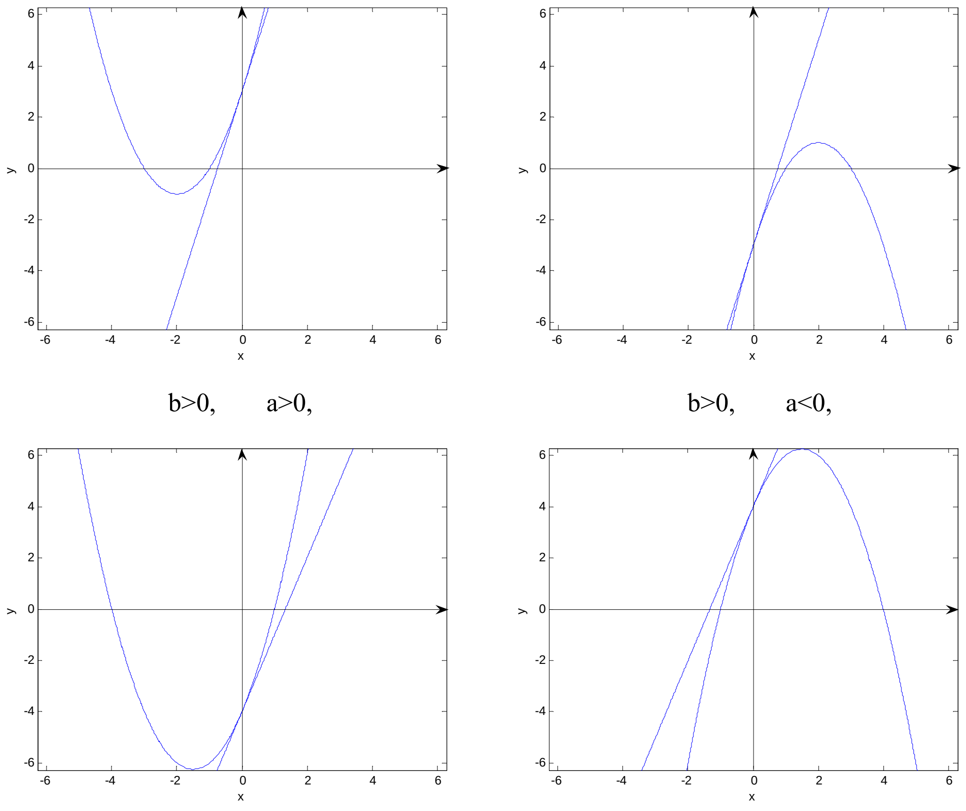
Then the range of the absolute value of the plus root is,

$$\frac{|c|}{b} \le \left| \frac{-b + \sqrt{b^2 - 4ac}}{2a} \right| \le \frac{2|c|}{b} \tag{5.16}$$

If $ac < 0$, the discriminant is guaranteed to be positive, and the range of the plus root is:

$$0 \le \left| \frac{-b + \sqrt{b^2 - 4ac}}{2a} \right| \le \frac{|c|}{b} \tag{5.17}$$

Figure 5.2 illustrates the relationship between the roots of equation $ax^2 + bx + c = 0$ and $bx + c = 0$. The plus root is closer to the linear solution. It lies between the origin point and the linear solution when $ac < 0$, or outside the linear solution when $ac > 0$.



b>0, a>0,

b>0, a<0,

b>0, a>0,

b>0, a<0,

Fig. 5.2 Graph of quadratic and linear function

From the above calculation, the third method Δw is a little larger (absolute value) than second method when $ac > 0$, but much less than (absolute value) second method when $ac < 0$ or even drop to 0. Therefore third order method is used when $ac > 0$ and the discriminant is non-negative, and second order method otherwise. This is the reason why it is called the semi-third order method. From the experimental results, semi-third order method always gives better convergence than second order method on the training set.

5.2.2 Partial derivatives with respect to weights

Similar to first and second order training, the key step of third order training is calculating the derivatives with respective to all the weights. According to equation 5.9, all three derivatives are needed while processing third order training.

According to Faà di Bruno's formula, as an identity in mathematics generalizing the following chain rule to higher derivatives:

$$(f \circ g)^{'} = (f^{'} \circ g) g^{'} \tag{5.18}$$

$$(f \circ g)^{''} = (f^{'} \circ g) g^{''} + (f^{''} \circ g)(g^{'})^2 \tag{5.19}$$

$$(f \circ g)^{'''} = (f^{'} \circ g) g^{'''} + 3(f^{''} \circ g) g^{'} g^{''} + (f^{'''} \circ g)(g^{'})^3 \tag{5.20}$$

We obtain

$$\frac{\partial E}{\partial y_l(n)} = f^{'}(y_l(n)) \frac{\partial E}{\partial Y_l(n)} \tag{5.21}$$

$$\frac{\partial^2 E}{\partial y_l^2(n)} = f^{''}(y_l(n)) \frac{\partial E}{\partial Y_l(n)} + [f^{'}(y_l(n))]^2 \frac{\partial^2 E}{\partial Y_l^2(n)} \tag{5.22}$$

$$\frac{\partial^3 E}{\partial y_l^3(n)} = f'''(y_l(n))\frac{\partial E}{\partial Y_l(n)} + 3f'(y_l(n))f''(y_l(n))\frac{\partial^2 E}{\partial Y_l^2(n)} + [f'(y_l(n))]^3\frac{\partial^3 E}{\partial Y_l^3(n)} \quad (5.23)$$

The derivatives with respect to $Y_l(n)$ are found through Equation 5.24,

$$\frac{\partial^i E}{\partial Y_l^i(n)} = \sum_m w_{l+1}^i(m,n)\frac{\partial^i E}{\partial y_{l+1}^i(m)} \qquad \mathrm{i}=1,2,3 \qquad (5.24)$$

Equation 5.24 is recursive. At each step, we are able to derive the partial derivatives with respect to the weights as shown by equation 5.25. The equations may vary due to actual network connection status.

$$\frac{\partial^i E}{\partial w_l^i(m,n)} = Y_{l-1}^i(n)\frac{\partial^i E}{\partial y_l^i(m)} \qquad \mathrm{i}=1,2,3 \qquad (5.25)$$

However, using those derivatives will possibly lead to a second derivative that is negative, which goes against the assumption made in equation 5.10. We simply drop the first term in equation 5.22, which guarantees that the second derivative estimates are non-negative.

$$\frac{\partial^2 E}{\partial y_l^2(n)} = [f'(y_l(n))]^2\frac{\partial^2 E}{\partial Y_l^2(n)} \qquad (5.26)$$

According to equations 5.25 and 5.26, the right-hand side is a sum of products of non-negative terms, so the left-hand side term is non-negative. It is demonstrated that it is quite easy to get higher derivatives in outputs. The partial derivatives with respect to all weights can be found using equations 5.25 and 5.26.

5.2.3 Summary of the Training Method

The higher order back-propagation training methods proceed as follows:

1) According to all inputs to the networks, compute the network outputs and errors. Find the maximum error.

2) Compute current and lower order derivatives of maximum error with respect to all weights.

3) Solve Δw with the equation(s) listed in Section 5.2.2

4) Go through the system and count the number of misrecognitions with new weight $w + \Delta w$. If there are fewer errors than before, then update weights and go back to step 1. Otherwise, reduce α by half and go back to step 3.

5) The system is considered to have converged when the first derivatives of weights are less than a predefined threshold.

5.3 Next-minimal Training Method

The Next-minimal training method is composed of several periods, each of which generates exactly one local minimum, where the network is considered well trained. Every period has 24 training cycles. For the first period, the training patterns are visited at a fixed randomized sequence, while the training coefficient α is decreased by half every 3 cycles. With appropriate starting coefficient λ, the training process will converge at a local minimum after a period, or 24 cycles. The local minimum is considered to be one of the training results, where the recognition rate is recorded for further comparison, and the weights are sent to the next period.

The random visiting sequence is regenerated and the training coefficient is reset to be starting coefficient λ in the second period. The next minimal point will be found in the second period. That's why the method is called next-minimal method. The error rate on the training set is usually decreasing after each period, but it is not a monotonic decrease on the testing set. The chart results in Section 5.4 show that the error rate on testing set usually bounce back after 6 periods. To avoid over-training, we train the network up to 6 periods, or 144 cycles, and select the weights with the least error rates on the training set. The final training result is given by the weights with smallest error rate on the training set, so it is probable to have better performance than the traditional method with only one detected minimum.

Next-minimal training method is thus designed to avoid the local minimum trap. The training limit of 6 periods reduces the influence of over-training as well. This method works well with the semi-third order training method and the bounded training method, each of which more or less improves the recognition rate.

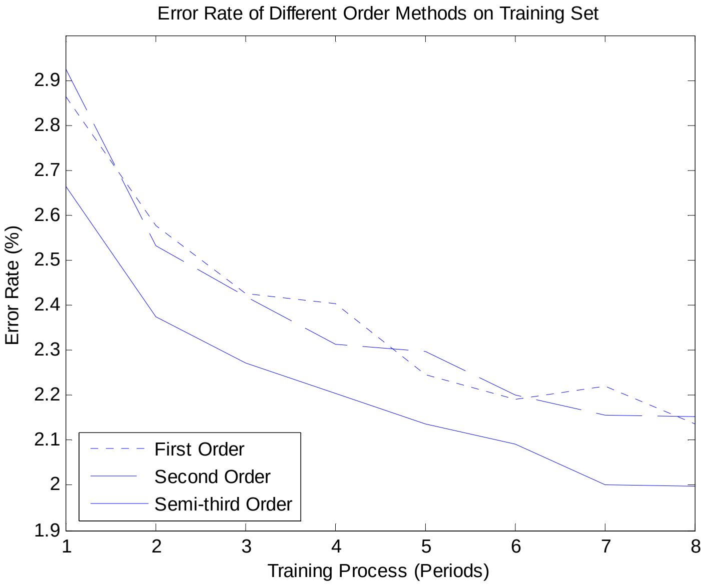
5.4 Experimental Results

5.4.1 Training results of upper case character

The training results for the upper case characters are provided in Table 5.3, and the error rate on the training as well as the testing sets is provided in Figure 5.3.

Table 5.3 Training results of upper case characters and 28*28 input size

Training method	2-norm	4-norm	6-norm	Inf. norm	Training Time
First order method	3.83	3.55	3.84	3.45	99 minutes
Second order method	4.14	3.35	3.38	3.43	110 minutes
Semi-third order method		3.14	3.24	3.51	135 minutes



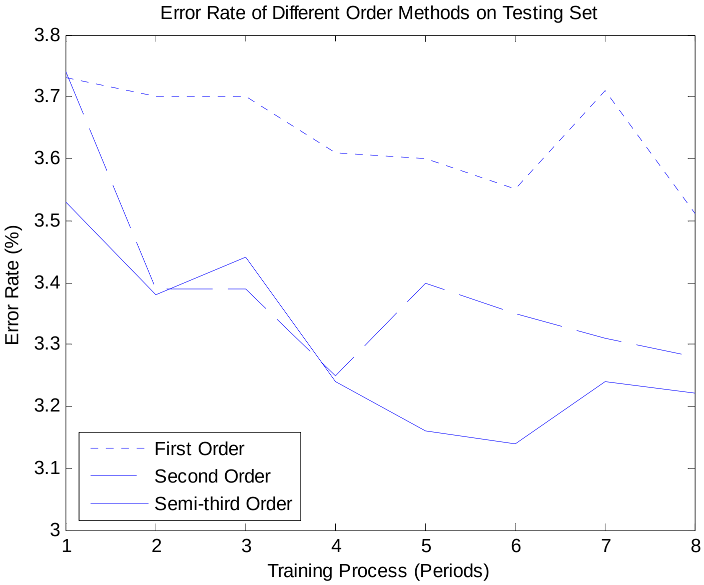


Fig. 5.3 Error rate on training and testing sets (upper case and 28*28 input size)

From the results above, quartic error function is shown to have a better performance than any other norm distance of error. Semi-third order method converges fastest on both training and testing sets, and gives the best result for upper case characters. Error rate on the training set continues to drop as the training process continues, while the error rate on the testing set bounces back to climb after 6 periods; this is caused by over-training. The weights with smallest error rate among the first 6 periods are picked as the training results.

Larger input size 36*36 takes twice the time, and has general better training result on both sets than 28*28 input size. These results are given in Table 5.4 and Figure 5.4. However, comparing with smaller size, the recognition rate on the testing set doesn't improve as much as on the training set. Similarly, semi-third order method has the best results on both sets. And the error rate on the testing set starts to climb after 6 periods.

Table 5.4 Training results of upper case characters and 36*36 input size

Training method	2-norm	4-norm	6-norm	Inf. norm	Training Time
First order method	3.39	3.37	3.50	3.27	182 minutes
Second order method	3.52	3.36	3.29	3.23	189 minutes
Semi-third order method		3.09	3.22	3.16	237 minutes

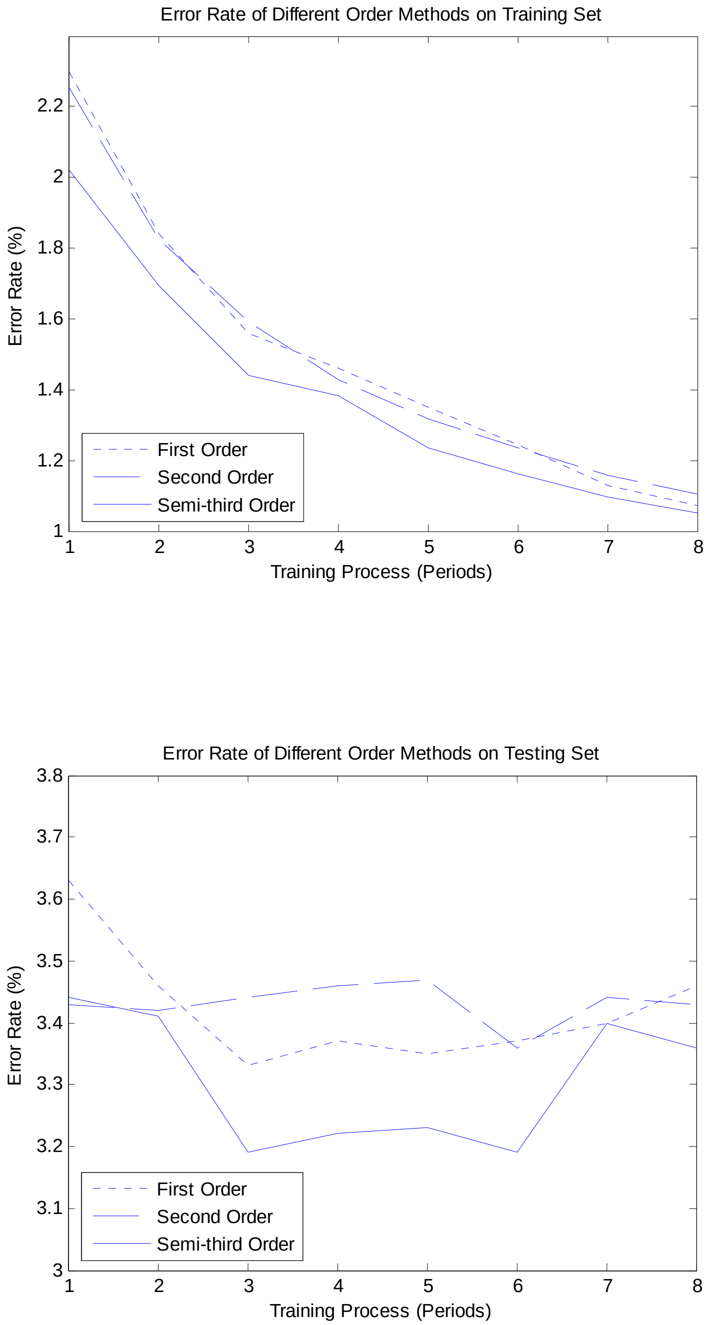


Fig. 5.4 Error rate on training and testing sets (upper case and 36*36 input size)

5.4.2 Training results of lower case character

The error rate of lower case characters is about 2.5 times higher than upper case characters, because lower case characters are more dependent on writers' style and don’t have a common or regular shape. Second order training method gives the best training result instead of semi-third order method. These results are shown in Tables 5.5 and 5.6.

Table 5.5 Training results of lower case characters and 28*28 input size

Training method	2-norm	4-norm	6-norm	Inf. norm	Training Time
First order method	13.14	8.73	10.09	9.67	122 minutes
Second order method	10.42	8.28	9.69	9.58	126 minutes
Semi-third order method		8.95	9.76	9.80	194 minutes

Table 5.6 Training results of lower case characters and 36*36 input size

Training method	2-norm	4-norm	6-norm	Inf. norm	Training Time
First order method	10.87	8.35	9.32	9.10	233 minutes
Second order method	9.49	8.08	8.93	8.92	244 minutes
Semi-third order method		8.48	8.78	9.98	361 minutes

5.4.3 Training results after preprocessing

Preprocessing operations improve performance significantly for inputs of both cases. The error rate has been reduced about 10% on average. The results show that the variance and diversity of the training set has a positive effect on the neural network. Although horizontal shearing is not among the common affine transforms, i.e., shift, scaling, and rotation, it is proven to be one of the most effective transformations. Affine transforms are usually used to process image input without caring about its content, while horizontal shearing simulates different human writing styles. That's why it's especially useful in this proposed research work. Polynomial distortion works similar to pushing the character towards one side or both sides. It simulates other writing styles that some people arrange the parts of characters differently and increases the diversity of the training set further.

Table 5.7 also shows that the training time is about 3 times more than without preprocessing. This is because the size of the training set has increased from 70,000 to 210,000. However, since the architecture of the neural network remains the same, the testing time does not change.

Table 5.7 Error rate and training time after preprocessing

Training method	Upper Case		Lower Case	
	Error Rate	Training Time	Error Rate	Training Time
First order method	3.62	537 minutes	8.66	690 minutes
Second order method	3.21	551 minutes	7.49	723 minutes
Semi-third order method	2.77	701 minutes	7.75	1128 minutes

5.4.4 Training results comparisons

Table 5.8 shows that the quartic neural network has better performance than other three neural networks, no matter whether it is with or without preprocessing.

Table 5.8 Error rate comparison with other three neural networks

Neural network type	Upper case		Lower case	
	original	preprocessing	original	preprocessing
Fully connected	6.53	5.78	11.25	10.76
Convolutional (LeNet-1)	3.42	3.18	8.95	8.45
Improved Pyramidal Neural Network	4.31	3.88	10.57	9.55
Quartic Neural Network	3.09	2.77	8.05	7.49

5.4.5 Best training results

From the results listed above, Table 5.9 shows the overall optimal final training and testing results and methods for upper and lower case characters.

Table 5.9 Optimal Training Results and Methods

	Upper Case Characters	Lower Case Characters
Lowest Error Rate	97.23%	92.51%
Best Training Method	Semi-third Order Method	Second Order Method
Training Time	701 minutes	723 minutes
Testing Time (10,000 patterns)	8.48 seconds	8.50 seconds

5.4.5.1 Training samples for upper case characters

What follows are several experiments conducted using upper case and lower case characters on randomly selected windows of the different characters that are available through the NIST SD-19 database.

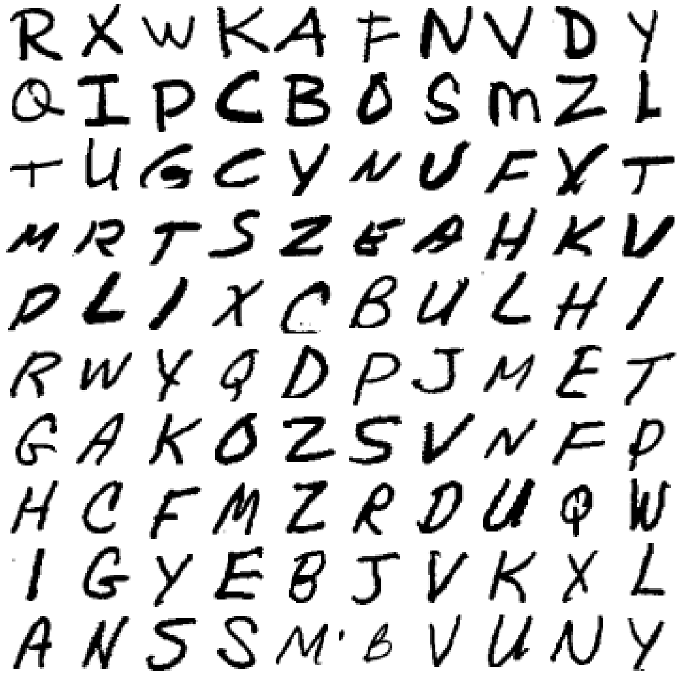


Fig 5.5 (a) Original images of upper case characters in testing set, No. 1 to 100

Recognition Result:

R	X	W	K	A	F	N	V	D	Y
Q	I	P	C	B	O	S	M	Z	L
T	U	G	C	Y	N	U	F	X	T
M	R	T	S	Z	E	A	H	K	V
P	L	I	X	C	B	U	L	H	I
R	W	Y	Q	D	P	J	M	E	T
G	A	K	O	Z	S	V	N	F	P
H	C	F	M	Z	R	D	U	Q	W
I	G	Y	E	B	J	V	K	X	L
A	N	S	S	M	B	V	U	N	Y

The recognition results for the first 100 pattern of testing set is 100% correct.

This first trial could be considered a fortunate coincidence. Other trials are thus used.

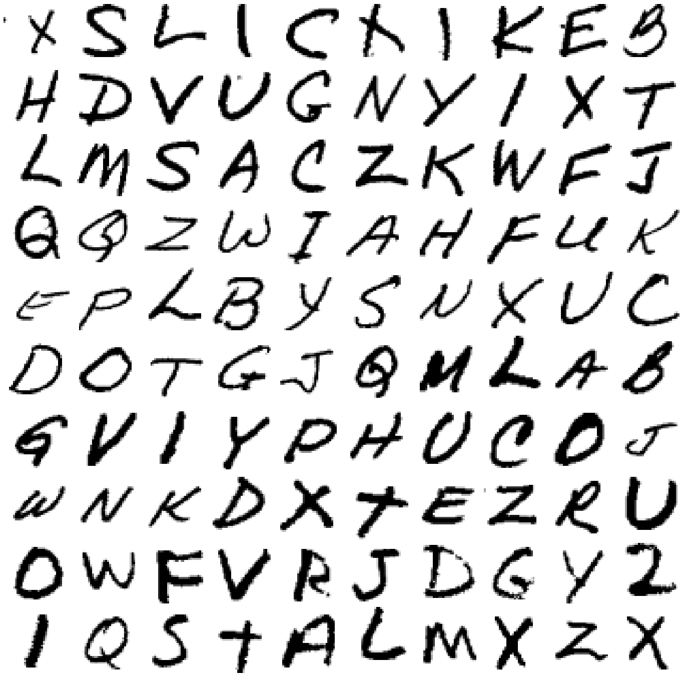


Fig 5.5 (b) Original images of upper case characters in testing set, No. 1001 to 1100

Recognition Result:

X	S	L	I	C	X	I	K	E	B
H	D	V	U	G	N	Y	I	X	T
L	M	S	A	C	Z	K	W	F	J
Q	Q	Z	W	I	A	H	F	U	K
G	P	L	B	Y	S	N	X	U	C
D	O	T	G	J	Q	M	L	A	B
G	V	I	Y	P	H	U	C	O	J
W	N	K	D	X	T	E	Z	R	U
O	W	F	V	R	J	D	G	Y	Z
I	Q	S	T	A	L	M	X	Z	X

In this case there is only 1 recognition error in the table above.

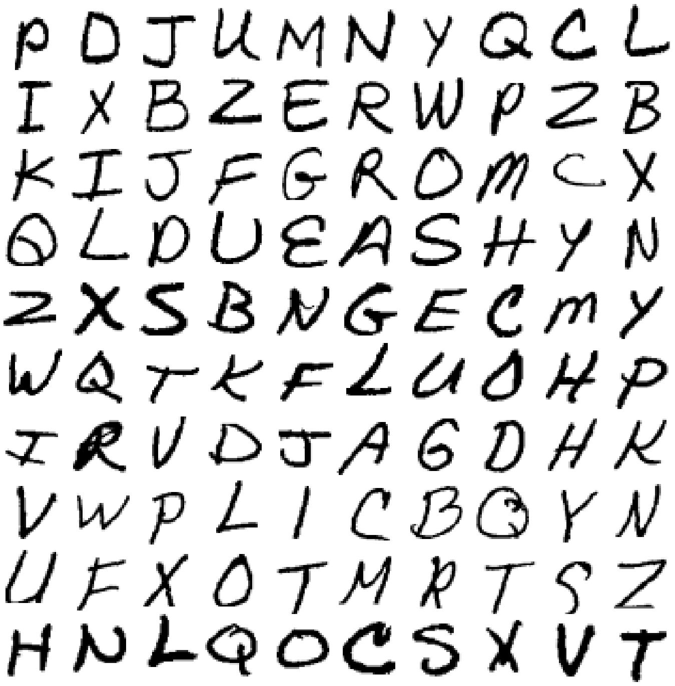


Fig 5.5 (c) Original images of upper case characters in testing set, No. 9901 to 10000

Recognition Result:

P	O	J	U	M	N	Y	Q	C	L
I	X	B	Z	E	R	W	P	Z	B
K	I	J	F	G	R	O	M	C	X
Q	L	D	U	E	A	S	H	Y	N
Z	X	S	B	N	G	E	C	M	Y
W	Q	T	K	F	L	U	O	H	P
J	R	V	D	J	A	G	D	H	K
V	W	P	L	I	C	B	Q	Y	N
U	F	X	O	T	M	P	T	S	Z
H	N	L	Q	O	C	S	X	V	T

In this case there are 3 recognition errors in the table above. From these three groups of sample images, it is shown that the recognition errors are evenly distributed over the training set.

5.4.5.2 Training samples for lower case characters

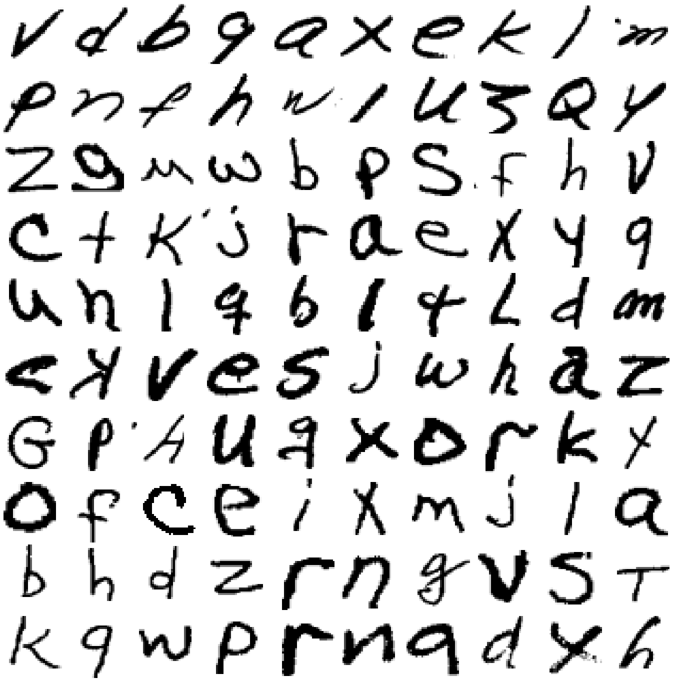


Fig 5.6 (a) Original images of lower case characters in testing set, No. 1 to 100

Recognition Result:

v	d	b	q	a	x	e	k	l	w
p	n	f	h	v	l	u	z	q	y
z	g	m	w	b	p	s	r	h	v
c	t	k	j	r	a	e	x	y	q
u	n	l	q	b	i	t	l	d	m
c	x	v	e	s	j	w	h	a	z
g	p	h	u	g	x	o	r	k	y
o	f	c	e	i	x	m	j	l	a
b	h	d	z	r	n	g	v	s	t
k	q	w	p	r	n	g	d	y	h

There are 7 recognition errors in table above. This result is reasonable since the original characters in Figure 5.6 (a) are badly written, even difficult for humans to read.

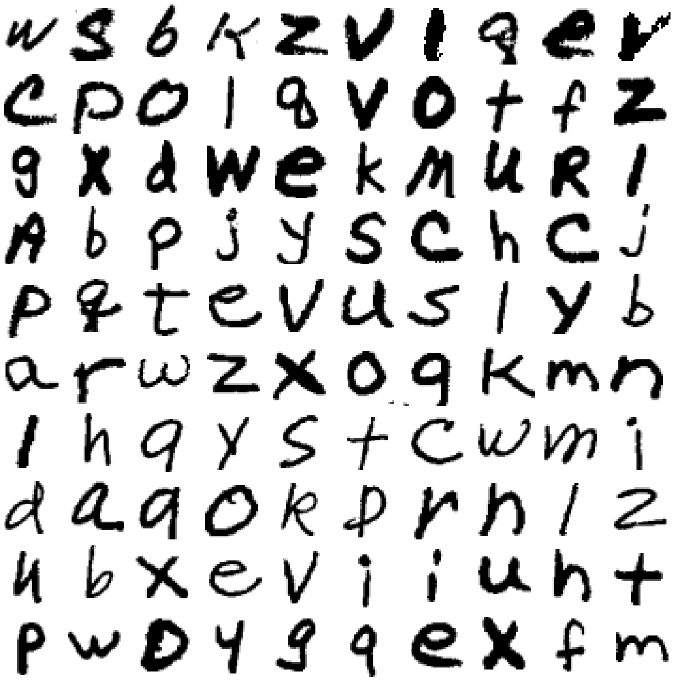


Fig 5.6 (b) Original images of lower case characters in testing set, No. 101 to 200

Recognition Result:

w	s	b	k	z	v	i	q	e	v
c	p	o	l	q	v	o	t	f	z
g	x	d	w	e	k	m	u	r	i
h	b	p	j	y	s	c	h	c	j
p	q	t	e	v	u	s	l	y	b
a	r	w	z	x	o	g	k	m	n
i	h	g	y	s	t	c	w	m	j
d	a	g	o	k	p	r	n	l	z
u	b	x	e	v	i	i	u	h	t
p	w	o	y	g	q	e	x	f	m

In this case there are only 2 recognition errors in the table. The images are well written and easier to be recognized than in the previous group given in Figure 5.6 (a).

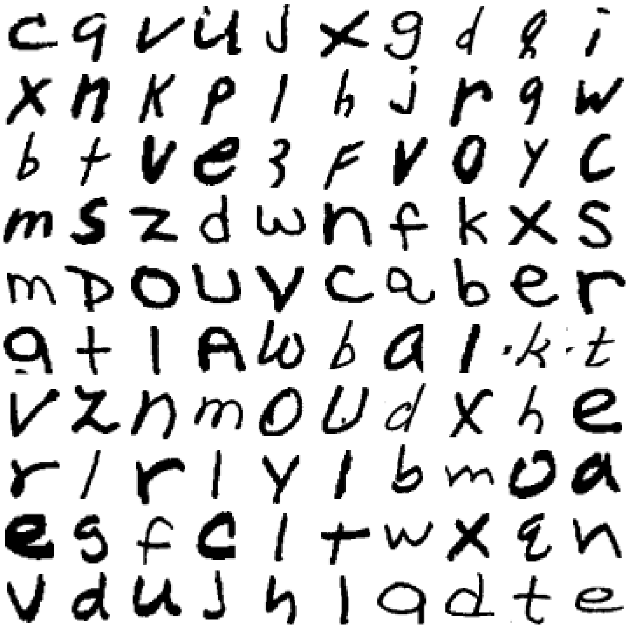


Fig 5.6 (c) Original images of lower case characters in testing set, No. 1001 to 1100

Recognition Result:

c	q	v	u	j	x	g	d	l	i
x	n	k	p	l	h	j	r	q	w
b	t	v	e	f	f	v	v	y	c
m	s	z	d	w	n	f	k	x	s
m	p	o	u	v	c	a	b	e	r
g	t	l	a	b	b	a	l	k	t
v	z	n	m	o	u	d	x	h	e
r	l	r	l	y	l	b	m	o	a
e	g	f	c	l	t	w	x	q	n
v	d	u	j	h	l	j	d	t	e

In this case there are 5 recognition errors in this table. Some images are segmented incorrectly and appear ambiguous. The three groups of sample images show that the recognition rates of lower case characters are largely dependent on writing styles and the performance of the segmentation method. Recognition of segmented characters has thus been well solved in this chapter. We now focus on word and sentence recognition in Chapter 6.

CHAPTER VI

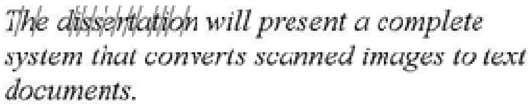
Experiments on Word and Sentence Recognition

6.1 Slant and Skew

6.1.1 Definition

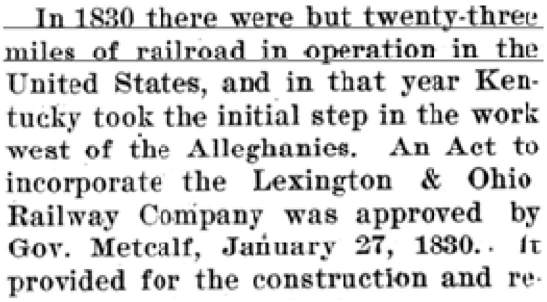
Slant is the predominant angle of the downward stroke to horizontal right direction in Western handwriting. Slant angle is usually less than 90 degrees, i.e., it is bent to the right. Left-handed writing is often accompanied by a slant angle larger than 90 degrees.

Skew means that the whole line of sentence is going up or down at a fixed slope. Whether the skew value is positive or negative depends on personal writing custom, or even the placement while scanning the handwritten images.



The dissertation will present a complete system that converts scanned images to text documents.

Example of slant



In 1830 there were but twenty-three miles of railroad in operation in the United States, and in that year Kentucky took the initial step in the work west of the Alleghanies. An Act to incorporate the Lexington & Ohio Railway Company was approved by Gov. Metcalf, January 27, 1830. It provided for the construction and re-

Example of skew

Fig. 6.1 Difference between slant and skew

6.1.2 Skew Removal

Skew is easier to be removed than slant. Here is a simple algorithm:

1) Generate several pairs of parallel lines with different slope, from -30 degrees to 30 degrees, with step change of 0.5 degree.

2) Each pair is composed of two parallel lines. One is at the upper bound, and the other is the lower bound. Make sure the whole sentence is located within the two lines.

3) Move the two lines as close as possible. They may touch but they may not cross any part of the sentence. Define *interval* to be the distance between the lines.

4) Find the shortest interval among the pairs. The lines of the pair are considered to be parallel to the sentence, and the skew of sentence is therefore equivalent to the slope of the lines. Then the given skew can be removed by vertical shearing.

6.1.3 Stroke-based Slant Removal

Slant is detected by the angle between downward strokes and horizontal positive axis. The challenge is to find the downward strokes.

In typography, each character is divided into three portions by two lines known as the mean line and the base line. The portion above the mean line is called ascender, the portion below the base line is called descender, and the portion between the lines is called median.

All the letters have medians, while only five letters have descenders, *g*, *j*, *p*, *q*, and *y*. It is clear that downward strokes appear both at the ascent and median portions.

If medians and descenders are discarded, it’s pretty easy to find the downward strokes from the remaining ascenders. The algorithm developed for this purpose is described below:

1) Make sure the skew has been removed. Discard (fill with blank) the bottom 2/3 portion of the line.

2) Generate several thin windows with different slope, from -30 degrees to 30 degrees, with step of 0.5 degree. The width of the sliding window is set to 5, which is approximately the width of a stroke.

3) Move each window from left to right, and make slope convolution.

4) Define *squaresum* to be the sum of the square of the slope convolution result. The simple sum is always equal to the total pixel value of the image and contains no information. *Squaresum* is applied here to emphasize non-uniformity.

5) Find the largest *squaresum* among the sliding windows. The slant of the characters is equivalent to the slope of the window. Then the given slant can be removed by horizontal shearing.

6.2 Algorithm of Segmentation

6.2.1 Type of segmentation

Segmentation is one of the major problems faced when dealing with handwritten word recognition. The accurate recognition of segmented characters plays the most important role in word recognition. Upper case characters are easy to segment, so only lower case characters are discussed in this section.

There are several different types of segmentation methods that are available in the literature. [Verma, 2003] gave a contour code based method for segmentation. [Wang and Jean, 1994] used another neural network approach to segment characters.

In this dissertation, a rule-based method is proposed instead of using an adaptive algorithm, because rules are easy to be described and implemented. The method is called circle-based segmentation.

6.2.2 Circle-based Segmentation

After the effects of slant and skew are removed, all the baselines are horizontal and segmentation lines are vertical. Some lower case characters have a circle inside, such as *a*, *b*, *d*, *g*, and *o*. There are two rules that are used for the circles:

1) No character has more than one circle with different horizontal position.

2) The circles will never be broken when the words are being segmented into separate characters.

Thus some non-break intervals and must-break intervals can be derived according to the rules. Some possible trials can be made on the intervals that don't belong to any of the types above. The trial points are selected at the position where the connection between two adjacent characters has the least width.

After segmentation, the neural network described in Chapter 6 can be used to recognize the separated characters. Then the recognized characters are used to reconstruct words.

6.3 Dictionary-based Correction

After recognition, the result has to be checked before it is output. First, we check if the recognized word is in a pre-defined dictionary. If yes, the word is considered correct. Otherwise, it is replaced with the closest word in the dictionary.

6.3.1 Word Matching and Edit Distance

Microsoft WORD gives a perfect example of word matching. Each time when there is a typographical error, WORD will check for the closest word to offer it as a suggestion. If there is only one closest word, the typographical error is changed automatically; otherwise WORD underlines the incorrect word with a red line and gives a list of suggested words. The suggested words are given according to their edit distance, from closer to further.

The edit distance between two words is the number of operations required to transform one of them to the other. The allowed operations include: insert, delete, and replace.

The classic algorithm to calculate the edit distance between two words is dynamic programming. The time complexity is of the order $O(ab)$, where a and b are the length of the words, respectively. However, there are hundreds of thousands of words in the dictionary, plus their variations. It is impossible to calculate the edit distance between the recognized word and all the words in the dictionary, because the overall complexity is now $O(abn)$, where n is the number of words in the dictionary. A more advanced algorithm is thus needed in this case.

6.3.2 BK-Tree

The BK-Tree was first proposed in [Burkhard and Keller, 1973]. [Baeza-Yates and Navarro, 1998] gives an efficient algorithm with BK-Tree to find all the words whose edit distance to the given word is at most *k*. The complexity is $O(akn^{\beta})$, where β is a constant which is between 0.6 and 0.7.

In this dissertation, only one recognition error is allowed so *k* is always 1. It usually takes less than 0.01 second to find the closest word in the dictionary of 100,000 words, or returns null if there's no such word.

In case of failure to match a word in the dictionary, the original word is required to be segmented and recognized again.

6.3.3 Dependency of Segmentation and Recognition

For word or sentence recognition, segmentation happens before character recognition. However, the recognition result may give feedback to segmentation. If the recognized word is not in the dictionary or there is no 'closest' word in the dictionary by editing up to one (1) character, the recognized word is considered to be incorrect and the original word need to be segmented and analyzed again. Therefore segmentation and recognition are dependent on each other. Good recognition system helps improve the segmentation mechanism and vice versa.

6.4 Experimental Results

There are some discussions and experiments on recognition of words and sentences. The sentences to be recognized are within one line, without punctuations, and consist of lower letters only. Unlike segmented character recognition, no training set is needed for word or sentence recognition. 500 images of upper case words and 500 lower case words that provided by SD-19 database are used in this experiment. 500 handwritten words and 100 lines of sentences are also collected from lab mates. These databases are labeled as 1, 2, 3, and 4.

In Table 6.1, which shows the recognition results, full recognition rate means the rate of words that are fully recognized as correct. Partial recognition rate means the rate of characters that are partially recognized as correct. Dictionary correction actually helps to improve the recognition rate, because several words have only one misrecognized character. The recognized text is actually readable and understandable, although about 25% of the words need to be corrected according to the context. A sample image is provided in Figure 6.2. Note how the OCR performed so poorly.

Table 6.1 Recognition rates of different databases

Database Index	Database 1	Database 2	Database 3	Database 4
Segmentation Rate	98.8	80.2	89.2	92.3
Partial Recognition Rate	96.0	74.6	80.8	82.5
Full Recognition Rate	92.6	66.8	72.0	70.8
Dictionary Correction Recognition Rate	94.2	70.8	76.4	75.6

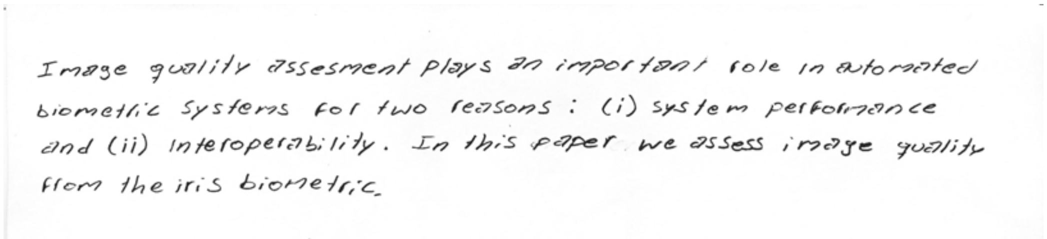


Image quality assesment plays an important role in automated biometric systems for two reasons : (i) system performance and (ii) interoperability. In this paper we assess image quality from the iris biometric

Original Image

```
lmaqe quality assessment plays an important role in automated
biometric systems for two yeqsoms il slsiem lerjormance
and innteroferability in this lafer we assess image quaqitf
from the iris biometric
```

Recognized Text

```
rr'7c'7'je '!jv~//Iy c7SSeSr?~-'?/ plC'J,Y':::' dr? r'~:::>O'7~J?/7/ role /-7 ~/Or-?r?.Ied

c,IDr-?ei;;·c. SyS/~r7..s ~or fWD rec7S0;?S : c.: Sysle;v7 pe(~oFr?c?/)ce

c70d (ii) II'7!e(operr:"7b,'/;ly. IrJ /h;s P<::'/,PeY" we d. SeS5 "r?C"'.j~ ,/u<::7/;Jy

Fr c1"""7 I he iT,·..s b'-o~e I,,'c
```

Recognized Result by OCR Software

Fig. 6.2 Sample of original image and recognized text

CHAPTER VII

Conclusion and Future Work

This dissertation presented a new system for handwritten text recognition based on an improved artificial neural network. The quartic neural network was compared to another two popular techniques. In general, it has the best recognition result and efficiency.

The semi-third order training method is considered to be the combination of the second order and the third order method, depending on the sign of the discriminant. It is one of the greatest advantages of the mean quartic error function, although it takes 50% more training time than the second order method. The semi-third order method helps to improve upper case character recognition but does not improve lower case character recognition. It shows more additional power for well-constrained data.

The bounded training method significantly accelerates the training process and gives better or at least equal results. The next-minimal training method is supposed to find more than one possible solution and avoid getting stuck in local minima. These two methods work well together and improve both the efficiency and accuracy of the artificial neural network.

Increasing the diversity usually improves the recognition rate of the artificial neural network. Two preprocessing methods proposed in this dissertation simulate various human writing styles and generate new characters from existing images. This process is similar to collecting more samples and increasing the training set.

Therefore the network produces a better result.

Character segmentation is one of the most important steps in handwriting recognition. Circle-based character segmentation is proven to be a feasible method, but there is enough room for improvement. Dictionary-based correction not only corrects recognition mistakes but also helps to find segmentation errors. It is very effective in English word recognition.

In future research more rules will be added to improve character segmentation. The recognizable text will also be extended from lines to paragraphs and pages. The system will understand punctuations, numbers and symbols in the text. Finally, cameras instead of scanners will be used to capture images, and the recognized text will be read aloud for the convenience of visually impaired users, as it was the initial motivation of this dissertation.

LIST OF REFERENCES

Adjouadi, M., and Wang, L., “Automated Book Reader for Persons with Blindness”, *US Provisional*, Patent Application Serial No: 60/987, 673, Date Filed: Nov 13, 2007.

Al-Jawfi, R., "Handwriting Arabic Character Recognition LeNet Using Neural Network", *International Arab Journal of Information Technology*, Vol. 6, No. 3, pp. 304-309, July 2009.

Amin, A., and Fischer, S., “A document skew detection method using the Hough Transform”, *Pattern Analysis and Applications*: 3 (3), pp. 243-253, Sept. 2000.

Baeza-Yates, R., and Navarro, G., “Fast approximate string matching in a dictionary”, *Proc. String Processing and Information Retrieval Symposium*, pp. 14-22, 1998.

Barakat, N. H. and Bradley, A. P., “Rule Extraction from Support Vector Machines: A Sequential Covering Approach”, *IEEE Transactions on Knowledge and Data Engineering*, Vol. 19, No. 6, June 2007.

Battiti, R., "First and Second-Order Methods for Learning: between Steepest Descent and Newton's Method", *Neural Computation*, Vol. 4, pp. 141-166, 1992

Blumenstein, M., Cheng, C. and Liu, X., “New Preprocessing Techniques for Handwritten Word Recognition”, *2nd IASTED International Conference Visualization, Imaging, and Image Processing*, Benalmadena, Malaga, Spain, September 9–12, 2002.

Blumenstein, M., Verma B., and Basli, H., “A novel feature extraction technique for the recognition of segmented handwritten characters”, *Proceedings of the Seventh International Conference on Document Analysis and Recognition*, Edinburgh, Scotland, pp. 137-141, 2003

Burges, C., “A Tutorial on Support Vector Machines for Pattern Recognition,” *Data Mining and Knowledge Discovery*, Kluwer Academic Publishers, 1998.

Burkhard, W. A. and Keller, R. M., “Some approaches to best match file searching”, *CACM*, 16(4):230-236, 1973.

Cheung, K., Talking Camera Aids Blind Persons, Kurzweil Educational Systems, Inc, http://www.digitalcamerainfo.com/content/Talking-Camera-Aids-Blind-Persons.htm, 2006

Ebrahimpour, R., Esmkhani, A., Faridi S., "Farsi handwritten digit recognition based on mixture of RBF experts", *IEICE Electronics Express*, Vol. 7, No. 14, pp. 1014-1019, July 25 2010.

Gader, P. D., Mohamed, M., and Chiang, J. H., “Handwritten Word Recognition with Character and Inter-Character Neural Networks”, *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics*, Vol. 27, pp. 158-164, 1997

Google Inc, “Detection of grooves in scanned images”, patent number 7508978, 2009.

Han, C., Adjouadi, M., Barreto, A., Rishe, N., and Andrian, J., “Improved Pyramidal Neural Network for Segmented Handwritten Characters Recognition”, *International Conference on Image Processing, Computer Vision, and Pattern Recognition*, pp. 695-699, 2009.

Ishitani, Y., “Logical Structure Analysis of Document Images Based on Emergent Computation”, *IEICE Trans. on Information and Systems*, (8), pp. 1831-1842, August 2005.

Khashman, A., “A Modified Backpropagation Learning Algorithm With Added Emotional Coefficients”, *IEEE Transactions on Neural Networks*, Vol. 19, No. 11, November 2008

Kimura, F., Kayahara, N., Miyake, Y. and Shridhar, M., “Machine and Human Recognition of Segmented Characters from Handwritten Words”, *4th International Conference on Document Analysis and Recognition*, Ulm, Germany, pp. 866-869, 1997.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. and Jackel, L. D., “Handwritten digit recognition with a back-propagation network”, in Lisboa P.G.J. (Eds), *Neural Networks*, current applications, Chappman and Hall, 1992.

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

Liang, J., DeMenthon, D., and Doermann, D., "Flattening curved documents in images", *International Conference on Computer Vision and Pattern Recognition*, San Diego, USA, pp. 338-345, June, 2005.

Nagy, G., “Twenty Years of Document Image Analysis in PAMI”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22 (1), pp. 38-62, Jan. 2000.

Pajdla, T., Werner, T., and Hlavac, V., “Correcting Radial Lens Distortion without knowledge of 3-D Structure”, *Technical report*, TR97-138, FEL CVUT, Prague, Czech Republic, 1997.

Phung, S. L., and Bouzerdoum, A., “A Pyramidal Neural Network For Visual Pattern Recognition”, *IEEE Transaction on Neural Networks*, Vol. 18, No. 2, March 2007

Pilu, M., "Undoing Paper Curl Distortion Using Applicable Surfaces", *International Conference on Computer Vision and Pattern Recognition*, Kauai, Hawaii, USA, pp. 67-72, 2001.

Pollard, S., and Pilu, M., "Building cameras to capture documents", *International Journal of Document Analysis and Recognition*, Vol. 7 (2), pp. 123-137, July, 2005.

Rao, S., “Electronic books: their integration into library and information centers”, *Electronic Library*, 23 (1): 116-140 2005.

Vass, G., and Perlaki, T., "Applying and Removing Lens Distortion in Post Production", *Colorfront Ltd.*, Budapest, 2003.

Verma, B., “Fast Training of Multilayer Perceptrons”, *IEEE Transactions on Neural Networks*, VOL. 8, NO. 6, November 1997

Verma, B., “A Contour Code Feature Based Segmentation For Handwriting Recognition”, *Proceeding of the Seventh International Conference on Document Analysis and Recognition*, ICDAR 2003.

Wang, J. and Jean, J., “Segmentation of merged characters by neural networks and shortest path”, *Pattern Recognition*, vol. 27, no. 5, pp. 649-658, May 1994.

Yamada, H. and Nakano, Y., “Cursive Handwritten Word Recognition Using Multiple Segmentation Determined by Contour Analysis”, *IEICE Transactions on Information and Systems*, Vol. E79-D, pp. 464-470, 1996.

Zant, T. V. D., Schomaker, L., and Haak, K., “Handwritten-Word Spotting Using Biologically Inspired Features”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 30, No. 11, November 2008.

VITA

CHANGAN HAN

1982	Born, Chengdu, China
2003	B.S., Electrical Engineering Beijing Normal University Beijing, China
2005	M.S., Computer Science Florida International University Beijing, China
2010	Ph.D. Candidate, Electrical Engineering Florida International University Miami, Florida

HONORS AND AWARDS

Florida International University, Research Assistant (2005 - 2010)

No. 6 in worldwide noisy iris recognition challenge Evaluation Part I (NICE.I), No. 1 in USA, 2008

PUBLICATIONS AND PRESENTATIONS

Publications

1. Changan Han, Malek Adjouadi, Armando Barreto, Naphtali Rishe, Jean Andrian: Improved Pyramidal Neural Network for Segmented Handwritten Characters Recognition. IPCV 2009: 695-699

2. Yu Chen, Malek Adjouadi, Changan Han, Armando Barreto: A New Unconstrained Iris Image Analysis and Segmentation Method in Biometrics. ISBI 2009: 13-16

3. Yu Chen, Jin Wang, Changan Han, Lu Wang, Malek Adjouadi: A robust segmentation approach to iris recognition based on video. AIPR 2008: 1-8

4. Yu Chen, Malek Adjouadi, Changan Han, Jin Wang, Armando Barreto, Naphtali Rishe, Jean Andrian: A highly accurate and computationally efficient approach for unconstrained iris segmentation. Image Vision Comput. 28(2): 261-269 (2010)

Submitted

1. Changan Han, Malek Adjouadi: Segmented Handwriting Characters Recognition With Mean Quartic Error Neural Network and Semi-Third Order Method, submitted to IEEE Transaction on Neural Network