A CONSTRUCTIVE APPROACH

FOR

CHINESE CHARACTER RECOGNITION

by

CHANG-LIN CHEN

A dissertation submitted to the Graduate Faculty in Computer Science in partial fulfillment of the requirements for the degree of Doctor Philosophy, The City University of New York

1995

This manuscript has been read and accepted for the Graduate Faculty in Computer Science in satisfaction of the dissertation requirement for the degree of Doctor of Philosophy.

Sept 11, 1995
Date

Michael Anshel
Professor Michael Anshel
Chair of Examining Committee

Sept. 13, 1995
Date

Stanley Habib
Professor Stanley Habib
Executive Officer

Professor Izidor Gertner

Professor Dipak Basu

Professor Tiah-Lih Tang

Supervisory Committee

THE CITY UNIVERSITY OF NEW YORK

ABSTRACT

A CONSTRUCTIVE APPROACH FOR CHINESE CHARACTER RECOGNITION

by

CHANG-LIN CHEN

Adviser: Professor Michael Anshel

This paper presents a constructive approach for solving problems associated with inputting Chinese characters for the recognition processing by digital computers. The idea presented here is "Construction Pattern Recognition (CPR)", which is essentially different from the existing Chinese character inputting methodologies. With the CPR, a Chinese character is treated as a graph rather than as a character of a language. Its construction pattern could be encoded by the "Construction Pattern Internal Coding System (CPICS)" and presented in a grid form with the "equivalence classes" as well as the matrix representation. The CPR method is developed based upon human learning and recognition of Chinese characters, applying the theory graph to form the body of the CPR algorithm. A proposed computer neural network will be presented as an upgradable learning and recognition algorithm for the future research on the CPR method.

PREFACE

In 1986, I visited several famous Chinese computer R&D centers, such as: (1) The Wang Laboratories, Inc. in Taiwan, (2) the Telecommunication Laboratories, Ministry of Communications, (3) the Chinese Character Analysis Group, Council for Cultural Planning and Development, Executive Yuan, and (4) the Computer Center of Institute of Information Science, Academia Sinica. Although they all have worked on the computer processing of Chinese characters, none have developed a highly efficient technology. There are three issues obstructing the development of the Chinese computer:
(1) Different internal coding systems co-existed on the market, (2) too many input methods and hardware devices, and (3) recognition of an input character is a very difficult mission to the Chinese computers.

I also found that many research projects for Chinese character input centered around the traditional index approaches which may use radicals, phonetics or the number of strokes through key-touch devices. Unfortunately, the keyboard devices are originally designed for the English character input and are limited in dealing with those Chinese index approaches. Those methods create an extra learning cycle and add confusion to the original index system adapted by the educational system and current dictionaries.

Some new approaches which used the sequence of the strokes written on the electronic pressure board were created at that time. Since methods of writing the same character are vary among different people, the successful rate of computer recognition does not meet the needs of the market. Essentially, all of the input methods focused on the analysis of the language itself; they didn't concern themselves with useability. They tried to reassemble the characters to meet the capabilities of the computer. The users then must sacrifice their personal writing styles to follow up a new way.

Our new approach centers upon the actual process of learning used by the average Chinese people. Rather than use an artificial means to process a character, we focus here on the abstract graph of the character and its analysis. The "Construction Pattern Recognition (CPR)" provides many benefits to solve the input problem.

I was encouraged by Professor Jack K.T. Huang when I told him my idea. Professor Huang is the creator of the "Three Corner Method" which is the most popular input method in China and Japan at that time. He said: "Construction pattern recognition is definitely a new potential approach, it provides the Chinese computer a natural writing input which has been used by Chinese people for five thousand years." He

provided me a lot of research material for my first paper. With his recommendation, the paper was published on the Proceedings of the 50th Annual Meeting of the American Society for Information Science in 1987.

1987-1988, I was assisted by Professor Jacob Rootenberg in Queens college, CUNY to rewrite the paper and published it in the International Journal of Systems Science in 1990. Later, the research work ceased when Dr. Rootenberg passed away in 1991. Professor Anshel subsequently guided me to use the neural network as a learning tool to implement the CPR concept on the computer. In the meantime, I further developed the CPR concept to establish the "Construction Pattern Internal Coding System (CPICS)" which can be used to encode the character. The CPICS can also adapted as an input method by the keyboard or by the electronic board.

I believe the CPR is one of the keys to solve the problem of computer recognition of the Chinese character. It may be developed for the other languages too, because the essentials of learning a language are the same among different people. The CPR approach can be easily implemented as a commercial product. This paper only presents the fundamental concept and study.

To

Hsioh-Ling, my eternally missed father,

Mei-Fan, my dear mother,

Edward, my son,

and

Judy, my lovely wife

ACKNOWLEDGMENTS

This study was encouraged by Prof. Jack Kai-tung Huang. His valuable advice, and particularly the reference materials he provided are gratefully acknowledged. The author especially thanks Professor Jacob Rootenberg, whose untimely death was a great loss to all. Professor Michael Anshel, who picked up where Professor Rootenberg left off, has been a constant source of guidance and assistance. Appreciation also extends to the editors, Dr. Joseph S. Fulda and David H. Wohl. Without their assistance, the completion of this paper would not be possible.

TABLE OF CONTENTS

Abstract iv

Preface v

Acknowledgement ix

List of Tables xii

List of Figures xiii

1 Introduction 1

2 Development of Computerized Chinese Characters 4

2.1 Background of Chinese Character 4
2.2 Coding System of Chinese Characters 6
2.2.1 External Coding System 7
2.2.2 Internal Coding System 8
2.3 Input Methods of Chinese Characters 9
2.3.1 One Key to One Chinese Character 10
2.3.2 Several Keys to One Chinese Character 10
2.3.2.1 New Primitive Element System 11
2.3.2.2 Phonetic System 12
2.3.2.3 Numeric Coding System 13
2.4 Ideal Input Methods 14

3 Fundamental of Construction Pattern Recognition 17

3.1 Characteristics of Chinese Characters 17
3.2 Learning of Chinese Characters 19
3.3 Human Learning and Recognition 21
3.3.1 Pattern Interpretation 22
3.3.2 Top-Down decomposition 24
3.4 Construction Pattern Recognition for Chinese Characters 25
3.4.1 Construction of Chinese Characters 26
3.4.2 Construction Pattern 27
3.4.3 Isomorphic Construction Pattern 29

4 Representation of the Chinese Construction Pattern 33

4.1 Defining Construction Pattern 33
4.1.1 Vertex Strategy 34
4.1.2 Equivalence Classes 35
4.2 Modeling the Traditional Array-Based Method 37

4.2.1 Representation of Construction Patterns Using Matrices 38
4.2.2 A Unique Array-Based Representation: Resolution into Equivalence Classes 39
4.2.3 Construction Pattern Internal Coding System, CPICS 41
4.2.4 Advantage of the CPICS 42

5 Implementation of the CPR Method 45

5.1 Segmentation 46
5.2 Partition 49
5.3 Pattern Recognition 53
5.4 Benefits of Using CPR Method 61

6 Future Research: A Neural-Net Learning and Recognition Algorithm 65

6.1 Why the Neural Network? 66
6.2 Basic Architecture of the CPR Neural Network 68
6.3 Conversion of a Matrix-Based Construction Pattern into a CPR Neural Network 70
6.4 Targets of CPR Neural Network 73
6.5 Implementation of the CPR Neural Network 74
6.5.1 Preparing the CPR Neural Network Data 75
6.5.2 Trial Testing and Output Interpretation 87

7 Conclusion 93

Appendix 1 96

Reference 97

LIST OF TABLES

1	A comparison coding table among CPICS and TCM (Three Corner Method), CCDC (Chinese Code for Data Communications), BIG-5, CCCII (Chinese Character Code for Information Interchange), and LCS (LIU'S Coding System).	43
2	Comparison between a digital computer and a neural network.	66
3	3a is a training data file for Figure 28a. 3b is a testing data file for Figure 28b.	77
4	4a is a training data file for Figure 28a. 4b is a testing data file for Figure 28b.	86
5	Results of testing data as shown in Figure 28b presented to the CPRNN.	89
6	Results of testing data as shown in Figure 29b presented to the CPRNN.	91

LIST OF FIGURES

1 Chinese characters have not constrained the length, directions and positions of their strokes. 18

2 The pattern interpretation affects the pattern matching and causes the pattern re-construction. It is also affected by the existing knowledge database. 23

3 An example of top-down decomposition. 24

4 Different Chinese writings for the same character denoting "I" in English. 27

5 An example of isomorphic construction patterns. h and h' are the edge functions that associates the endpoints, u and v, of edge e in G, and f(u), f(v) of edge g(e) in G'. 30

6 Using isomorphic construction pattern to examine construction patterns of a character. 31

7 Using the strategy of vertices to describe a construction pattern of Chinese character "I". 35

8 Degree-based equivalence classes of the vertices found in Chinese characters. 36

9 Matrix representation of a Chinese character. 38

10 A unique array-based representation of a Chinese character and its associated unique code of CPICS. 42

11 A flow chart -- The CPR method for the Chinese character recognition. 45

12 A process to substitute segments for points and arcs of a Chinese character "success". 46

13 A process to squeeze strokes of a character into a segmented construction. 48

14 The comparison between partitions of the CPR method and the Three Corner Coding method. 52

15 The conversion from a character into its array-based representation and CPICS matrix. 53

16 Frame analyses for some Chinese characters. 54

17 Structure analyses for one of frame types of a three-frame category. 56

18 Comparison of 18a, all characters beginning with 山 and 8b, just those with 4 frames. 58

19 A tree diagram demonstrates the searching path from the number of frames, the type of frames, the sepcific sequence of frames to the CPICS codes. 59

20 Process of frame pattern recognition and structure pattern recognition. 60

21 Character "A" in 21a has two vertices in "Class 3"; the vertex of "Class 2" is temporarily ignored here. 21b and 21c are the same characters as 21a, 21d is theoretically same as 21a, but is not necessarily recognized by people. 61

22 In contrast to 64x64 and 24x24 martices, the CPR method saves substantial amounts of memory space. 63

23 Traditional computer image processing. 65

24 A three-layered, feed-forward, fully interconnected neural network. 68

25 Detail of processing unit j of the hidden layer. 69

26 Converting a grid construction pattern of the Chinese character "heaven" into a matrix used to build a CPRNN. 70

27 The learning procedure of a neural network with back-error propagation. 72

28 The first experiment of the CPR neural network. 28a is a training data, 28b is a set of testing data which are the same chracters as 28a. 76

29 The second experiment. 29 is a set of training data which are different characters, 29b is a set of testing data. 80

30 Using the testing data to modify the network until both training and testing data can be identified by the network. 87

CHAPTER 1

INTRODUCTION

The Chinese written system is ideographic. Each character is composed of a number of differently shaped lines called "strokes" [22] within an imaginary square. Characters or ideographs which have a pictorial property represent units of meaning called morphemes.

In this thesis, we present the Construction Pattern Recognition ("CPR") system to process Chinese characters. This research represents substantial improvements over our joint work with Dr. Jacob Rootenberg [2]. While there have been many advances in pattern recognition technology since the appearance of the optical character recognition, our basic approach and its enhancements provide many unique capabilities not found among the others.

Current input devices, such as keyboards, electronic digitized board, image scanner and so on, are all designed for the English language. For Chinese ideographs, it is very difficult to computerize their writing method. More than 25% of the total population of the world are currently using Chinese characters, but do not have an easily accessible input

methodology. Effective processing of Chinese characters is clearly a project with many benefits for the Chinese people, as well to the entire computer science field.

To date, the electronic digitized board has been the most successful in emulating Chinese handwritten characters [32]. The development of the CPR process to recognize a pictorial writing or various writings for the same character present significant advantages over the digitized board.

Due to the inherent complexities of the Chinese alphabet, it should be noted that there are no systems available at this time which provide 100% accuracy in character recognition. With this in mind, we discuss the estimated expected error ratios, and possible ways to improve accuracy in future research.

In this paper, the objective is to let the computer to recognize Chinese characters while ignoring their shapes, sizes or relative stroke positions. We present an effective recognition algorithm to process Chinese characters.

Detailed studies of the physical input devices can be found in references [12,28,32]. The traditional linguistic analysis of Chinese characters are covered in references [3, 8,12,16,18,20,21,22,24,28,29,30,31]. For current, as well as

future input schemes, the key to success is the representation itself. The choice of an efficient representation of Chinese characters will always provide smooth interface capabilities to cope with the hardware input devices.

CHAPTER 2

DEVELOPMENT OF COMPUTERIZED CHINESE CHARACTERS

2.1 Background of Chinese Character

The written language is central to China's culture. It's characters act as a communication protocol to connect all the Chinese people, threading together five thousand years' history and culture.

The first stage of the development of Chinese written characters was known as the oracle bone inscriptions (called **Chia-Ku-Wen**, "shell-and-bone script"), dating from the Shang dynasty (18th-12th century BC) which is the first dynasty to leave historical records in China. Subsequent to Chia-Ku-Wen were: **Chin-Wen** ("metal script"), and **Ku-Wen** ("ancient script") found in Late Shang dynasty, **Ta-Chuan** ("large seal") in the Chou dynasty (1111-225 BC), and **Hsiao-Chuan** ("small seal") in Chin dynasty (221-206 BC). In the period of the Chin dynasty, a very important writing equipment, the "brush pen", was invented and widely used. This new tool generated a different way to write Chinese characters and caused the shape and stroke of the character to begin to change.

With the introduction of the brush pen, a new style of written characters, **Li-Shu**, became a major script in the Han

dynasty (206 BC-220 AD), after that, **Hsing-Shu** ("running style"), **Tsao-Shu** ("grass style"), and **Wei-Pei** followed. Ultimately, **Kai-Shu** prevailed in the Sui Tang dynasty (605-905 AD). **Kai-Shu** finally became a regular style of writing system used by all Chinese in almost 1400 years. **Kai-Shu** is also known as modern Chinese writing, and has not changed since the first century of the Christian Era.

The composition of Chinese characters is based on the "Six Principles" (called **Liu-Shu**) that were interpreted by Hsu Shen [16] in the Later (Eastern) Han Dynasty (25-167 AD). These principles elucidate the relationships among the shapes, sounds and meanings of Chinese characters. The contribution of Hsu to the Chinese written language is in his creation of 540 radicals, used to organize the Chinese characters into groups.

Further simplification and reduction to 214 radicals were instituted by Mei Yung-Tso in his master work, **Tzu-Hui** [24]. This 214 radical index system was later adopted by the biggest Chinese dictionary, **Kang-Hsi Tzu-Tien**, which collected 49,188 characters [30] and was compiled during the reign of Kang-Hsi emperor in the Ching dynasty (1644-1911 AD). Today, most dictionaries still use the same 214 radical system.

Observing the development of the Chinese written language, we may surprisingly find that the most of the

changes between different writing styles are in their strokes' shapes due to the various writing equipments. The basic arrangement of strokes was still maintained as the language evolved. It is an exciting discovery presented in this paper as evidence to support the fundamental relevancy of the CPR: Chinese characters can recognized by some features which are essentially from the pictorial graphs (or logograph). In spite of the change of writing tools, or the improvement of printing (the earliest printing discovered in China by the end of the 2nd century AD), the features of Chinese characters have been maintained since then.

In this paper, we show how these features can be used as a key to redefine the old Chinese language for computer processing.

2.2 Coding System of Chinese Characters

To computerize the Chinese characters, the coding system must be established. Here we separate the coding systems into two categories: Coding schemata designed for human interfaces can be named with a general term, "External Coding Systems". This is relative to the other term, "Internal Coding Systems", which is used for the computer internal processor functions and database management.

2.2.1 External Coding System

The external coding systems currently in use today are culminations of the various schemes in use prior to the computer age. Many linguists tried to group Chinese characters by the radicals (roots), the number of strokes, or other aspects. The approaches have been:

(1) Radical System. Grouping characters by radicals, or some primitive elements of characters.

(2) Stroke Number Coding System. Grouping characters in the ascending order according to the number of strokes.

(3) Telegraph Coding System. Characters are represented by a 4-unit code group.

(4) Frequency Coding System. Characters are coded by the frequency of individual character in use.

(5) Phonetic Symbol System. Using 37 phonetic symbols and 4 tonal marks to group characters.

(6) Four Corner Coding System. Using a 4-digit number to represent four corners of a character.

(7) Three Corner Coding System. An extended version of the Four Corner Coding System. It improves the situation of repeatedly using one number by several different characters in the Four Corner Coding System [30].

2.2.2 Internal Coding System

The Chinese language computer was developed shortly after the modern English language computers. Since the American Standard Code for Information Interchange (ASCII) had used 1 byte (8-bits) to define the characters, providing 256 distinct values per byte, is sufficient for English but not for the Chinese written language, the Chinese computer must extend the 1-byte ASCII coding schema to the 2-byte or 3-byte coding schema.

The internal coding system assigns each Chinese character a unique code which is a superset of the current ASCII characters. Initially, a 2-byte coding system is sufficient to temporarily satisfy the need of dealing with the frequently used characters (around 21,840). With increased demand, an advanced coding structure can cover existing 49,188 characters. A 3-byte coding system can easily meet this requirements.

(1) 2-byte Internal Coding System. Adopted by some

popular internal coding systems such as: IBM5550, Big-5, Wang, E-Ten, General, Kung-Hui, and Han-Yin coding systems.

(2) 3-byte internal coding system: Using a three [7-bit] byte vector to construct a 3-dimensional (94x94x94) coding space. It provides a total of 821,748 positions [17]. Invented by CCCII [4,5], and adopted REACC [26], and others.

2.3 Input Methods of Chinese Characters

The current Chinese input methods may be divided into three distinct categories: Key-touch keyboard input, Pen-base pressure board input, and Image-OCR scanner input. The most successful and popular one of the above three categories is the keyboard input, which is based on a "spelling" approach.

The keyboard input approach depends on the type of the keyboard devices which are: Chinese keyboard, English keyboard, and Chinese/English combined keyboard - each of which has, based on its own features, advantages and disadvantages. So far, the use of the English 101-key keyboard has become the mainstay in developing the Chinese input method. Since the development of input methodologies is tied with the physical input devices, the keyboard input approaches may be categorized into two groups listed as follows.

2.3.1 One Key to One Chinese Character

The design of a Chinese keyboard is based on a simple principle, namely one key to one Chinese character. This restriction invariably leads to an enormous sized keyboard. The early input methods dating back to the mechanical keyboard and early electronic keyboard were naturally affected by the mechanical aspects of the traditional Chinese typewriter and the various printing machines.

If we follow the Chinese printing history, the Chinese character is cast in a square-lead-type design, and printed as an unit whenever it is required to be printed. This method is simple and without the coding schema, but the problem is that a huge amount of single character keys are needed on the keyboard. This, in turn, requires users to resort to a long-term training in memorizing the position of a specific character key. Later, an improved version incorporating eight characters on one key and one optional key was added. The capacity of the keyboard has increased, but its size, in spite of its reduction, was still very large. The one, F6801A Kanji keyboard, produced by Tatung [28] is of this type.

2.3.2 Several Keys to One Chinese Character

The problem of how to reduce the physical size of the

Chinese keyboard has always been a topic that draws great interest and attention. A great deal of research has been centered around the idea that can be stated simply as several keys to one Chinese character - the so-called "spelling method". According to this method, a list of Chinese "alphabets" that may be radicals, roots, or certain primitive elements must be established.

2.3.2.1 New Primitive Elements System

The traditional radicals are usually used as indexes. Radicals are not primitive elements of Chinese characters, therefore, they can not be applied to "spell" a Chinese character. In order to achieve a "spelling method", one can either expend the numbers of 214 radicals, so as to get enough resolution to enable the spelling of Chinese characters, or build up a new primitive elements system and, reduce the number of 214 radicals. Some new radical systems are introduced as follows:

(1) 64-key System. Developed by Wang Laboratories, Inc. Using the combination of some traditional radicals and new elements [28].

(2) 610-character System. Developed by The Computer Center of The Department of Budgets, Accounting, and

Statistics of The Executive Yuan, ROC [28].

(3) 515-character System: Developed by The National Chiao-Tung University, ROC [28].

(4) Tsang-Chieh System: Developed by 0,1 Technology Co., using 24 prime radicals and 72 assisting elements [29].

In this category, there are many other systems which have tried to a create more human-like approach to make the input method faster and easier. However, no matter how they break Chinese characters into primitive elements, they also break the traditional radical system which is used in the educational system. This inevitably causes people to get confused and spend extra time to learn new methods.

2.3.2.2 Phonetic System

The phonetic systems include the Wade-Giles system, the Yale system, the Pin-Yin system, and the Chu-Yin Fu-Hao system [3]. The way to use the phonetic system is to define 41 keys of the English keyboard as the primitive elements with which users can easily "spell" a Chinese character.

This system requires the user to be familiar with the phonetic input system. For example, the Chinese characters

must be correctly pronounced and, then, the phonetic alphabets must be correctly spelled. It is not easy to suit different dialects spoken in various regions in China. In addition, many different Chinese characters have the same sound, or the same character has several different sounds that also make the phonetic system complicated to use.

2.3.2.3 Numeric Coding System

The restriction of keyboard input device is the last coding system mentioned here which just needs 10 numerical keys on the keyboard. The early Telegraph Coding System is for professional users, and requires considerable training. The latest and more efficient is the Three Corner Coding System, with its chart of 100 fundamental symbols. One can directly convert a Chinese character into a set of three 2-digit numbers.

Certainly, the Three Corner Coding System also requires practice and memorization. However, It provides users a workable way to encode a Chinese character by visualizing the character. It doesn't specifically define the keys on the keyboard, nor require the user to learn a totally new radical system.

2.4 Ideal Input Methods

Generally speaking, all the current keyboard input methodologies have good prospects to computerize the Chinese written language. They have tried their best to achieve targets: (1) Fast input method, minimizing the numbers of key-strokes to make a character, (2) less rules of memorizing and learning, easy-to-use for all the people, and (3) compatibility with English input at the same time.

By carefully investigating the various keyboard input approaches, we can find that some common problems are going to limit the development of keyboard input approaches.

(1) They are not the original ways the Chinese have used. Both of Chinese writing and index methods are not applied to the keyboard input approaches. Users must depress several English or numerical keys, under certain rules, in order to be able to compose one Chinese character, instead of directly "writing" it.

(2) They are indirect methodologies. To implement the keyboard input approaches, the keyboard must be able to convert sequences of keys to radicals. It takes time to learn and to manipulate the conversion [8].

(3) They conflict with the current language education in school. For educated people, the best way is to directly apply what they have learned from the school, and then use the same rules to input characters into computers. Any new approaches may contradict his previous learning.

(4) They reconcile themselves to the device of keyboard. The keyboard input approaches are designed with the computer in mind, not with the user's convenience as a prime consideration.

(5) They can not freely use alphabetic keys to assemble a character. The user can not just uses the key which are defined as primitive elements of Chinese characters to assemble or create a character. The way to make a character must be pre-loaded into the computer [2].

These five issues expose the weak points of the keyboard input approaches. It is our opinion that the ideal input method should match the nature of the Chinese "written" language. For the future, The input method should return to the original "writing" method instead of the key-touch method. The concept of a keyboard input approach could be an optional solution, but absolutely not a primary scheme to computerize the Chinese language.

In the following chapters, a totally different input method, the CPR method, is presented. It could be adapted by all kinds of input devices including punch-in keyboards. It makes the Chinese input method more natural and considerably expandable.

CHAPTER 3

FUNDAMENTAL OF CONSTRUCTION PATTERN RECOGNITION

To develop a natural input method for Chinese characters, the computer should be capable of recognizing Chinese characters just like Chinese people do. Therefore we discuss the methods Chinese people use to learn and recognize Chinese characters. Finally, we will explain how the construction pattern recognition is determined and how it works.

3.1 Characteristics of Chinese Characters

A language, in general, consists of sentences, which are made up of words, which in turn are constructed from characters. A character can be considered as a symbol, and the geometric aspects of the symbol are discussed here as a graph.

Chinese characters are unlike the linear character strings in English. An English word is viewed as a set of characters (alphabet) arranged on a linear space (line), and its length is one attribute of the language. A Chinese character has a square design with an imaginary boundary, its pictorial presentation is normally composed of 1 - 30 strokes which are composed of three basic building elements: line segments, arcs and points. The drawing of a Chinese character is performed by organizing these three basic elements inside

an imaginary square boundary. To further examine the difference between Chinese characters and English characters, please refer to appendix 1.

Chinese characters have not constrained their strokes. The drawing of line segments, arcs and points could be in various lengths, directions and positions. For example, the Chinese character "dog" is composed of 4 strokes shown as Figure 1. Figure 1a is a regular writing, Figure 1b has a slightly different appearance but is still treated as the same as the regular one.

1a 1b

Figure 1. Chinese characters have not constrained the length, directions and positions of their strokes.

To learn English is to memorize the alphabetic arrangements which are spelled, more or less, phonetically. Learning Chinese characters may be called "pictorial recognition". To learn a Chinese character is to learn three different things together: The way to draw strokes, the

meaning of a set of strokes (radical), and its associated pronunciation which is not phonetic.

Some characters have the same sound, but with different meanings and strokes. Some characters have the same strokes, but with the different meanings and sounds. Some characters have the same sound and meaning, but differ in their strokes (called "variant forms" [7]). In each case, these will be differentiated by the context in which they appear. Based on the principles given in **Liu-Shu**, Chinese people treat a character as a picture which is the combination of its sound, shape and meaning. From these fundamental components, many derivatives of the initial context occur. It may be one reason why the introduction of new Chinese words is less frequent than it is in English. There are many dialects and different systems of interpreting the meaning of a character, but these don't affect the strokes of a character which can be recognized by all Chinese.

3.2 Learning of Chinese Characters

Generally speaking, A Chinese person who starts his education in kindergarten and continues though college will learn some 4,000 - 5,000 characters (for a discussion of the 4,808 most frequently used characters see [4].) In other words, a Chinese college student should have learned around

4,000 - 5,000 pictorial or graph patterns. There are roughly more than 50,000 characters in existence today. It is both important and interesting to find out how Chinese people can memorize so many pictorial characters in their life, and how they can recognize different handwritten forms from the same regular characters.

When a Chinese child starts to learn his first word, the teacher always gives him a simple character with less strokes. In school, the sequence of drawing strokes is the first lesson for a child to learn a word. Today, some optical character recognition systems equipped with pen-based input devices [32] are using this approach to recognize a handwritten character. However, the sequence of drawing strokes does not make any difference for the result of a complete drawing. The sequence of strokes won't determine the pictorial pattern.

When children learn a character, they don't only learn the drawing of strokes, they assimilate the relationships among strokes. How does a child actually memorize those relationships? Normally, he must find a first stroke as a starting clue, then check the second stroke: Does it touch the first stroke? If it does, then how do they touch? If not, then what is the position the second stroke has relative to the first stroke? Using this approach, the child will continue checking strokes in this manner until the last stroke.

After learning a character, a child will keep the stroke relationships in his memory. Then, he is going to learn a second character - one with different stroke relationships, and has to memorize the stroke pattern of the new character. Chinese characters are virtual square graphs, with each graph an independent picture. If he can't distinguish the first from the second character, he will look at both graphs again until he finds some features which can show him the difference between the first and second character. Following this, the child will continue to memorize characters till his vocabulary consists of between 4,000 and 5,000 characters.

The learning cycle for Chinese characters is very long. There is no shortcut that can be used to learn pictorial characters. The relationship among strokes can't be described by reading it out as can a word in English by spelling it out; it must be described by writing it out.

3.3 Human Learning and Recognition

For a given Chinese character, even though different people learn to recognize the same character via distinct learning methods, its original and rewritten forms are still recognized as the same character. This gives us a clue that there must be something existing in between the learning and recognition to consistently support a process of

visualization, comprehension, memorization, retrieval and comparison.

3.3.1 Pattern Interpretation

Interpreting a scene is a process of learning. Normally, a scene itself provides original patterns which may be interpreted as variety of patterns by different people. If the original patterns match the existing patterns, the interpretation could be directly done just through the visualization. Otherwise, it needs to apply prior knowledge and complicated procedures to construct the workable patterns. We will further explain this procedure in the following paragraphs.

(1) Direct-vision interpretation. When one is trying to recognize a character, he will first visualize the character. Whatever features he then captures will be interpreted as some type of patterns. He will recall whether he learned it before, using a matching procedure. If he can't find the answer, he will then visualize the character again and interpret another type of pattern, and he will keep matching patterns until something with which he is familiar is found. In this type of interpretation, there is no prior knowledge involved.

(2) Interpretation with prior knowledge. When one visualizes a character for the first time, he adapts the

existing pattern to interpret its scene which will then affect the construction of its pattern. Human construction of a pattern is always based on experience from previous learning. Initially, one constructs a pattern only by what he sees, that may be or may not be affected by prior knowledge. If the pattern matching fails, it will cause the pattern to be re-interpreted and thus, pattern re-constructing and re-matching. Through several rounds of repeating this procedure, the final pattern will be compromised in between the original pattern and existing patterns. Figure 2 illustrates the relations among the pattern interpretation and its associates.

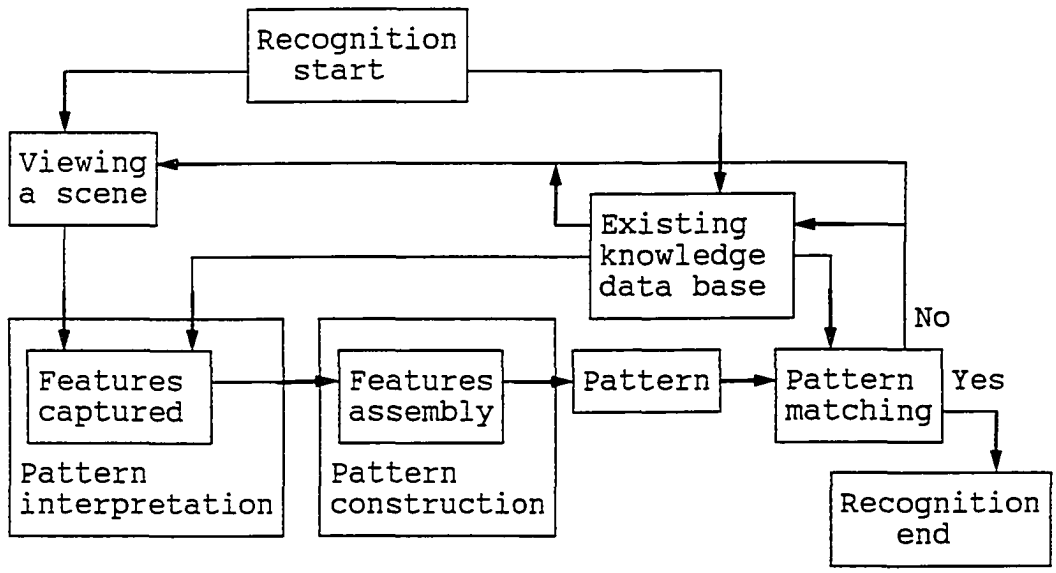
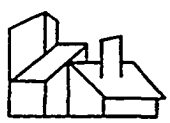


Figure 2. The pattern interpretation affects the pattern matching and causes the pattern re-construction. It is also affected by the existing knowledge database.

3.3.2 Top-Down Decomposition

When seeing a house, the first thing which attracts the human eyes may be its shape, color or style. If he fixes his view on the house longer, he may find more details: material, structure, size and so on. The last thing he may watch, if he wants, may be the lines, intersections, curvature and angles, etc. which are the primitives of the scene.



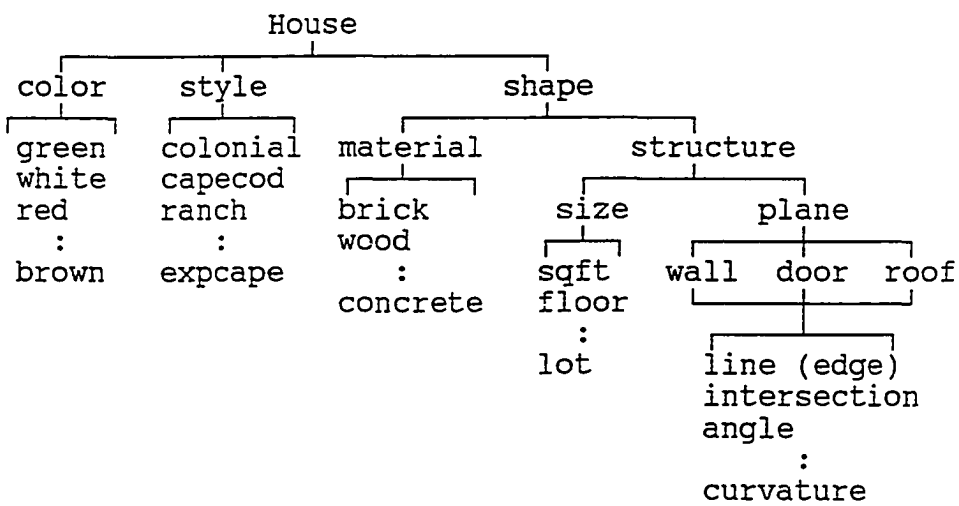


Figure 3. An example of top-down decomposition.

Figure 3 illustrates a top-down decomposition [15,33]. Computer recognition processing is a bottom-up composition.

When computer views the house through a CCD camera, it captures the image through the lenses and converts the image to a grid picture, each grid cell is represented by a set of bits which store the data of different scales of shades or colors. The computer vision collects the data which all stems from the primitives on each grid cell.

Much research has been done on assembling the information of grid cells to recompose lines or "abstracting" edges [12,33]. Many artificial intelligence projects also have developed algorithms on pattern recognition, trying to put lines, arcs, and points with different degrees of directions together. there is no doubt that with new technologies, reconstructing the primitive information to the original scene will be not difficult.

3.4 Construction Pattern Recognition for Chinese Characters

We will present a CPR (Construction Pattern Recognition) method which is modified on human top-down approach to directly represent a scene (i.e. Chinese characters) to a computer, then, let computer recognize the scene. A proposed neural network will be also presented as an implementation of the CPR method.

As we mentioned before, since the way the humans learn

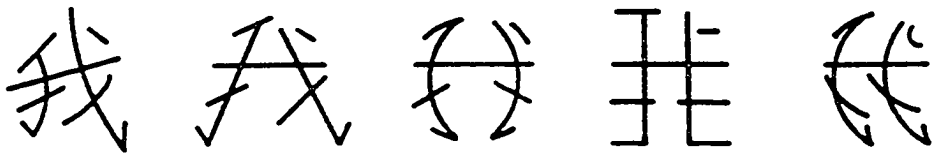
and recognize, they can't process all the detailed information captured by the eyes in a momentary glance. The brain won't attempt to capture a whole scene at first glance, therefore, multiple interpretations of its construction are the only way to grasp the essence of the representation of the scene. When a child learns Chinese characters, he finds it most difficult to memorize all the details of the strokes. He probably just captures stroke relationships in a character, then re-uses those relationships to re-write or recognize a character.

3.4.1 Construction of Chinese Characters

In this paper, a construction of a Chinese character is defined as a set of relationships of strokes. These relationships, or arrangements of strokes, can be treated as stroke patterns, and will not be affected by the lengths, degrees, and the relative positions of strokes which may be therefore adjusted by the drawer without changing the construction of a character. Similarly in English, the construction could be treated as a set of relationships on the alphabet. The appearance of each letter may be variant, but the relationships among them never change.

Let's take an example shown as Figure 4 to illustrate the construction of a Chinese character. Figure 4a presents a regular writing of a Chinese character which means "I" in

English. Figure 4b - 4e are different ways of "drawing" Figure 4a. All of these writings can be recognized as the same character because of the invariability of their "constructions".



4a 4b 4c 4d 4e

Figure 4. Different Chinese writings for the same character denoting "I" in English.

We should ask: From Figure 4a to 4e, what features enable a person to determine them as the same construction? Apparently, for Chinese, these features are the key to define the learning of the strokes' arrangement, that will support different people to interpret a uniform construction which can be commonly used to recognize the above five constructions as the same characters.

3.4.2 Construction Pattern

Traditional pattern recognition uses patterns stored in the computer to match the object. The pattern is an actual

image composed of pixels. The pixel which is obtained by the computer vision processing is just a set of digital signals without semantic content. Its image is a fixed pattern that cannot dynamically be assimilated by computers.

However, the traditional pattern approach doesn't work for human pattern recognition. Logically, ways of constructing patterns by biologic brain and computer devices must be different, therefore, patterns given to a computer must be different from patterns given to a human. Our target is to discover the patterns constructed by humans which are also suitable to computer processing.

Let's review the case of a child learning a Chinese character. After the child practices drawing a Chinese character several times, a complete image of the character will form in his brain. He may keep a clear picture of an ideograph in his brain, but later on, some information about strokes may be forgotten, and the pattern of construction of the character will help him to recognize and rewrite the character. If the child cannot maintain this construction pattern, then he will lose his ability of recognizing and rewriting this character. The pattern of construction must reach a certain threshold for learning and recognizing a Chinese character to take place. The concept of construction provides a very flexible dynamic pattern approach with the

following features:

When recognizing a Chinese character, only the construction pattern will be considered. This eliminates many factors of pattern recognition discussed in the literature such as avoiding the dead ends of dynamic pattern development in the traditional pattern recognition.

Construction patterns provides a new release of forming patterns. It can be used to explain how humans can recognize a given character, despite the fact that they can not copy it exactly. It will also explain how different persons can learn the same character in different ways, and still distinguish the given character from other characters.

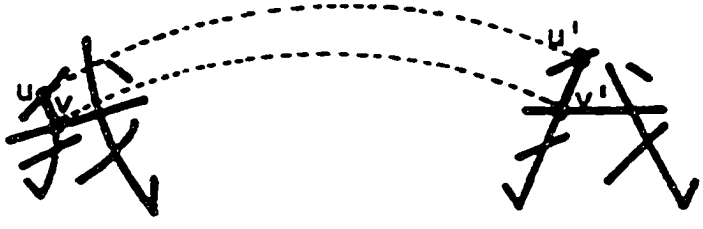
Finally, the concept of a construction pattern may be applied to other fields of recognition, not only that of Chinese characters. The requirement is to represent the concept of a construction pattern to the computer so that the character can be recognized.

3.4.3 Isomorphic Construction Pattern

A construction pattern may be treated as a physical or/and abstract feature of a character based on different people's point of views. However, every Chinese character must

have its own unique construction thus guaranteeing that different persons can capture the same stroke relationships and see multiple instances of the character as the same. There cannot be two characters with identical constructions, since, in that case, both will be understood as the same characters. This is called an isomorphism which is quite well known in the study of graph theory [13]. These isomorphism are used to provide the major enhancements contained in this thesis over our previous work [2].

For readers who do not know the Chinese written language, they may just treat the characters simply as graphs. Let's examine the same Chinese character used in Figure 4. Assuming Figure 4a and 4b are two given graphs G and G', shown as in Figure 5a and 5b, with their associated vertex set, V and V',



5a

Graph G, G(V,E)
h(e) = uv

5b

Graph G', G'(V',E')
h'(g(e)) = f(u)f(v)

Figure 5. An example of isomorphic construction patterns. h and h' are the edge functions that associates the endpoints, u and v, of edge e in G, and f(u), f(v) of edge g(e) in G'.

and edge sets, E and E', which are noted as G(V,E) and G'(V',E'). These two graphs are thought of as isomorphic if there are one-to-one correspondences, i.e. f: V $\subseteq$ V', and g: E $\subseteq$ E', where f is a vertex mapping function, and g is an edge mapping function, such that for every edge e between vertices u and v in G, there exists the corresponding edge g(e) between the corresponding vertices f(u) and f(v) in G'.

The concept of isomorphism can be used to examine the construction patterns of a character. For an example, Figure 6a presents a given regular writing of the character which is shown as Figure 4a. Figure 6b - 6e give some other writing forms, but only the one that appears in Figure 6d is the isomorphic graph to that of Figure 6a. Although their shapes are somewhat different from each other, they do mean the same, because their vertices and edges have exact one-to-one mapping relations.

6a 6b 6c 6d 6e

Figure 6. Using isomorphic construction pattern to examine construction patterns of a character.

The concept of isomorphism is good for examining the relative relations among construction patterns, however, it can not be used to represent a Chinese character. A construction pattern is the key to represent a character, but, what can we use for representing a construction pattern? There may be various ways to describe a specific construction pattern, for example five different formats of writing a character are presented in Figure 4. It may would take a large amount of time to describe them, if one didn't learn these characters before.

CHAPTER 4

REPRESENTATION OF THE CHINESE CONSTRUCTION PATTERN

The construction pattern approach that provides computers a new methodology for inputting a Chinese character is original from the human learning and recognition. For a person, the construction pattern is a set of relationships of strokes. In a computer implementation, these strokes' relationships must be first re-defined as some type of formats that can be interfaced with computers.

4.1 Defining Construction Pattern

In English, the primitive elements of words are the 26 letters, each of which has a unique look and sound (i.e., sign and name). In Chinese, the primitive elements are strokes which have no labels on them; when drawing a character, the length of a stroke is freely allowed to be adjusted, depending on the person's desires. Since there is no name for each stroke, the representation of the construction pattern is hard to be described by its size, shape, stroke sequence, or stroke arrangement.

When one writes a character, every time a new stroke is added to the previous drawing, the current construction pattern will be changed. As stated earlier, the basic elements

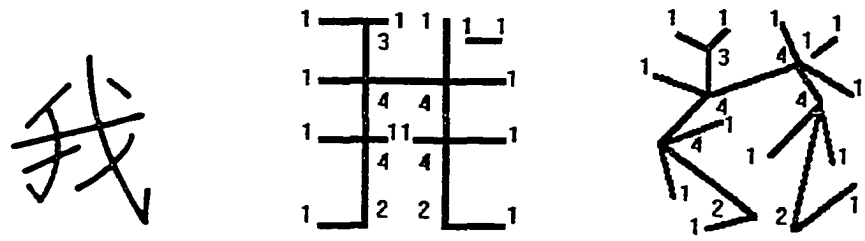
of the Chinese characters are lines, arcs, and points. In other words, the construction pattern is physically composed of only vertices (points) and edges (lines and arcs). A new added stroke will change the number of vertices and edges; there are many methods to analyze the change of a construction pattern. Here we show that the change can be simply described by the increase in the number of vertices to which the new stroke is incident.

4.1.1 Vertex Strategy

The way of using vertices to describe a construction pattern can be demonstrated by the following example. Define a vertex as the end-points of an edge, or intersection of edges. To identify a common construction pattern among different writings of single character, for example five different writings in Figure 4, it is necessary to select the features common to all. By carefully examining their strokes' arrangements, we discover that some specific types of vertices always occur at each writing. For strokes, since their appearance and length are variant, we couldn't find any common features among them. Therefore, in this paper, we determine to use only vertices instead of edges to represent a construction pattern of a Chinese character.

Figure 7 further illustrates the strategy of organizing

vertices as a construction pattern. Figure 4a is used again as Figure 7a, its construction pattern shown in Figure 7b, and counted with total 20 vertices. Using the properties, the number and the degree (Section 4.1.2), of these vertices, we can completely describe a construction pattern. In addition, with the properties of vertices, different people can learn to write their own characters which are still recognized as the same character. Figure 7c is an unusual construction pattern, a person may not figure out what it is, but with the same properties of 20 vertices, it can be easily recognized as the same character of Figure 7a.



7a 7b 7c

Figure 7. Using the strategy of vertices to describe a construction pattern of Chinese character "I".

4.1.2 Equivalence Classes

Several strokes and arcs may intersect or be adjacent at a vertex. This is known as the "degree" of the vertex. Suppose

there is a vertex v, the degree d(v) of which is the number of times v is used as an end-point of the incident edges. Through analysis of all the vertices and their associated degrees occurring in Chinese characters, we may separate the vertices into 7 equivalence classes as shown in Figure 8.

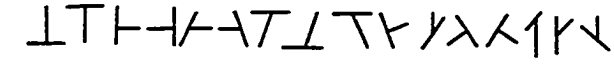
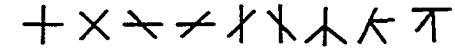
Name of class: Degree of vertex	Paradigm for equivalence class	Elements in the classes
0	•	Only one point: No intersection or adjacency exits.
1	—	— | / \
2	∨	
3	⋎	
4	+	
5	✳	✳
6	✳	✳

Figure 8. Degree-based equivalence classes of the vertices found in Chinese characters.

We assign the degree of a vertex as the name of its associated equivalence class. If d(v)=0 then the vertex is called a "point" which belongs to the "class 0". If d(v)=1 then the vertex which belongs to the "class 1" and its edge are together termed as a "segment". The equivalence classes

and their corresponding elements which are found in all of the Chinese characters are illustrated in the figure 8. Since Chinese characters have a wide tolerance with respect to stroke length, stroke direction, and stroke curvature, some of elements below appear different, but belong to the same equivalence class. For example, ㄴ , ㄱ , ㄏ and ㄱ are in the "class 2", i.e. the vertex is of degree 2.

4.2 Modeling the Traditional Array-Based Method

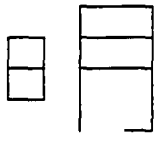
To determine construction patterns of Chinese characters involves analysis of the number and the type of its vertices. For example, Figure 7b can be described by {X,Y} where X is the number of vertices and Y is a tuple consisting of the degree of each of the X vertices. Therefore, its construction pattern can be represented as P(c)={20,[1,1,1,1,1,3,4,1,4,2, 1,4,1,4,2,1,1,1,1,1]}, where "P" is a CPR expression function, and "c" is a specific character, i.e. Figure 7a.

To simplify the representation of a construction pattern, we will use vertices which have degrees of at least 2 to represent the features of the construction pattern. The 1-degree vertex and the 0-degree vertex are ignored, except when the other features of the construction patterns are not sufficient to distinguish between characters. There are only 7 vertices with degree of 2, 3 and 4 shown in Figure 7b, which

will be considered as the elements of the construction pattern. Thus, the above representation can be re-described as P(c)={7,[3,4,4,2,4,4,2]}.

4.2.1 Representation of Construction Patterns Using Matrices

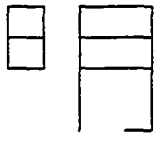
Actually, an array itself already indicates the number of vertices. Given only the degrees of the vertices, we can convert a construction pattern to a matrix representation. Figure 9a illustrates a character which means both "light" and "tomorrow" depending on the context. If we scan the character from the top-left corner to the top-right corner, and line-by-line from top-to-bottom, then we will obtain the matrix shown in Figure 9b.



9a

```
( 0, 0, 2, 2 )
( 2, 2, 3, 3 )
( 3, 3, 3, 3 )
( 2, 2, 0, 0 )
( 0, 0, 1, 2 )
```

9b



9c

```
( 2, 2, 2, 2 )
( 3, 3, 3, 3 )
( 2, 2, 3, 3 )
( 0, 0, 1, 2 )
```

9d

Figure 9. Matrix representation of a Chinese character.

The strategy used to scan a Chinese character is the central issue. The character in Figure 9a has two radicals, non-adjacent ideograms, and there is no rule in Chinese prescribing their relative height or how close they must be to each other. Moving one radical up or down, right or left, within a range of stroke relations does not affect the meaning. If we move the right radical up to the same height as the left radical as shown in Figure 9c, then the matrix in Figure 9b will be changed to the matrix in Figure 9d.

4.2.2 A Unique Array-Based Representation: Resolution into Equivalence Classes

Figure 9 demonstrates two different matrix representations for the same character. Similarly, if we move the right radical down to the bottom, or we scan the character with a different method, then we will obtain different matrices. A matrix representation is a good methodology for directly mapping a construction pattern into a grid format.

However, since Chinese characters are flexible in their appearances, a matrix representation can only reflect one of many isomorphic construction patterns of a character. Based on the previous discussion of isomorphism for a Chinese character, its construction pattern must be unique that guarantees the character can be recognized from the others. In Figure 9, 9a and 9c are the same characters, but their

associated matrices, 9b and 9d, are different. If we want to make 9a and 9c to have an identical representation, we should find their common parts with which an unique representation is formed.

Figure 8 shows 7 equivalence classes, but the "class 0" and "class 1" are ignored in this paper, hence only 5 classes are considered as the number of elements of an array. For a unique construction pattern, the number and the type of equivalence classes, and the number of vertices in each equivalence class must be different from that of other construction patterns.

For example, in Figure 9a, there are 7 vertices in "class 2" and 6 vertices in "class 3"; similarly for Figure 9c. No matter how these total 13 vertices are arranged, its matrix representation must have the same number and type of equivalence classes and corresponding number of vertices in each class. Let us use the array-based representation to demonstrate a construction pattern. There are 5 elements each of which has a pair of numbers in a array. The first number indicates the name of the equivalence class, the second number indicates the number of vertices in the equivalence class. Its expression can be described as $P(c)=\{[2,V_2],..,[E,V_E]\}$, where $E=2,..,6$ is the name of equivalence class; V_E is the number of vertices in E. Therefore, both Figure 9b and 9d can be

represented as P(c)={[2,7],[3,6],[4,0][5,0],[6,0]} or simply as an array: (2,7,3,6,4,0,5,0,6,0).

4.2.3 Construction Pattern Internal Coding System, CPICS

Since 2, 3, 4, 5, and 6 are in fixed positions, we can further simplify the array as $(V_2,..,V_E)$, where E=2,..,6 is the name of equivalence class, V_E represents the number of vertices in E. Therefore the array, (2,7,3,6,4,0,5,0,6,0), can be simplified as a 5-element array, (7,6,0,0,0). If we give each element a 2-digit variable which allows the number of vertices in each equivalence class from 0 to 99, then the 5-element array can be represented as a 10-digit number, 0706000000. An array, now as a numerical code, of a Chinese character must be an unique code. This coding system can be an internal coding system of Chinese characters, and can be named as a "Construction Pattern Internal Coding System (CPICS)". The CPICS can be demonstrated as shown in Figure 10 by using a unique array-based representation of a character shown as Figure 9a.

The following diagram represents one of the major enhancements of this current work over that of [2]. The transformation from the matrix to the unique array representation and subsequently the numerical code are the new results of this paper.

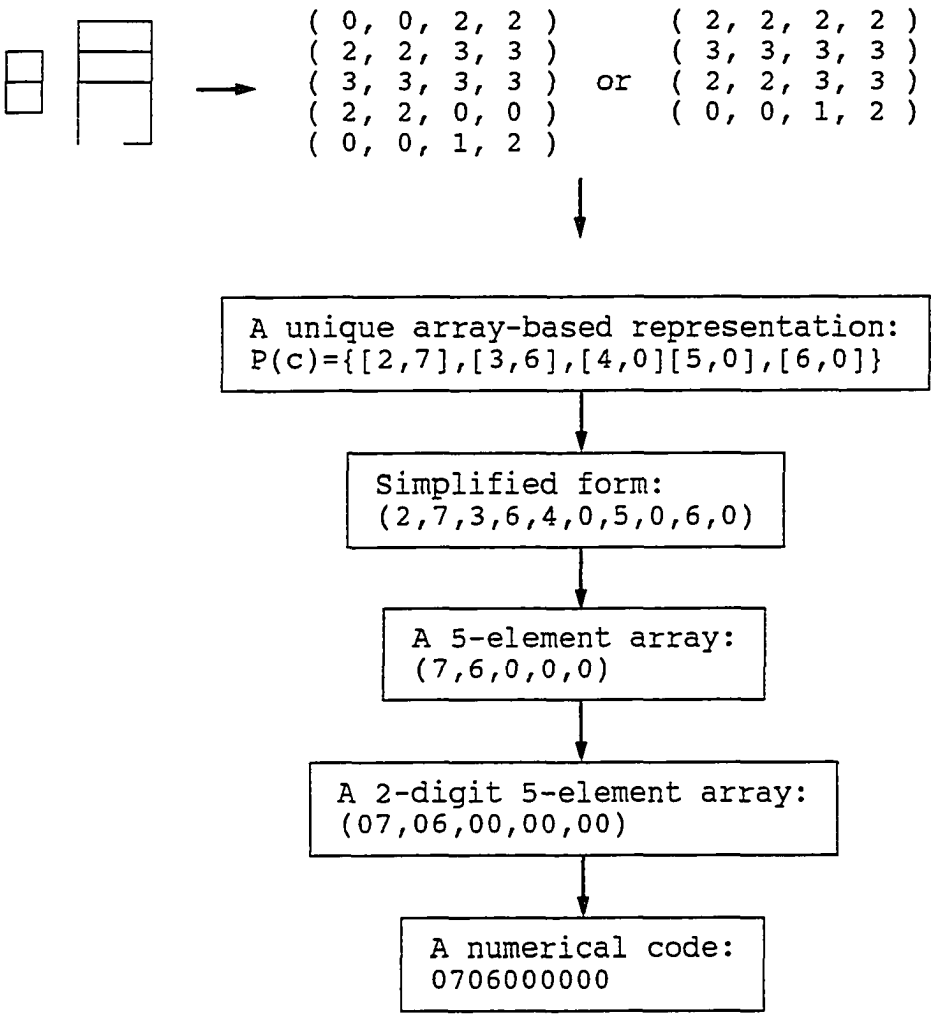


Figure 10. A unique array-based representation of a Chinese character and its associated unique code of CPICS.

4.2.4 Advantage of the CPICS

The CPICS assigns a code for each Chinese character which is based on its construction pattern. The codes may not be

continuous numbers in total. But each code must be unique. A comparison table of the CPICS with five other coding systems is listed as follows:

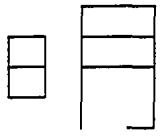
Chinese character	TCM	CCDC	BIG-5	CCCII	LCS	CPICS
	612600	2494	A9FA	214321	0252	0706000000

Table 1. A comparison coding table among CPICS and TCM (Three Corner Method), CCDC (Chinese Code for Data Communications), BIG-5, CCCII (Chinese Character Code for Information Interchange), and LCS (LIU'S Coding System).

The benefits of using the CPICS can be stated as follows:

(1) A natural representation to each Chinese character. The coding of each Chinese character is determined from its construction pattern. It is not an artificial coding system - the insertion and deletion of characters can be carried out at any place in the alphabet without affecting the operation of coding.

(2) It is compatible with all the current input approaches. The CPICS provides an efficient interface between an inputted character and computer storage devices. It can

work with all of the current input devices currently available on the market.

(3) It can be used to deal with the frame approach. The frame approach is important to divide a Chinese character into a set of frames. The CPICS can be used to represent each frame in a character as well as the entire character.

CHAPTER 5

IMPLEMENTATION OF THE CPR METHOD

The CPR implementation of a Chinese character by a computer may be theoretically divided into three parts: Segmentation, Partition and Pattern Recognition. A flow chart of the CPR implementation is shown in Figure 11. The CPR method allows Chinese characters to be sent to a processor in

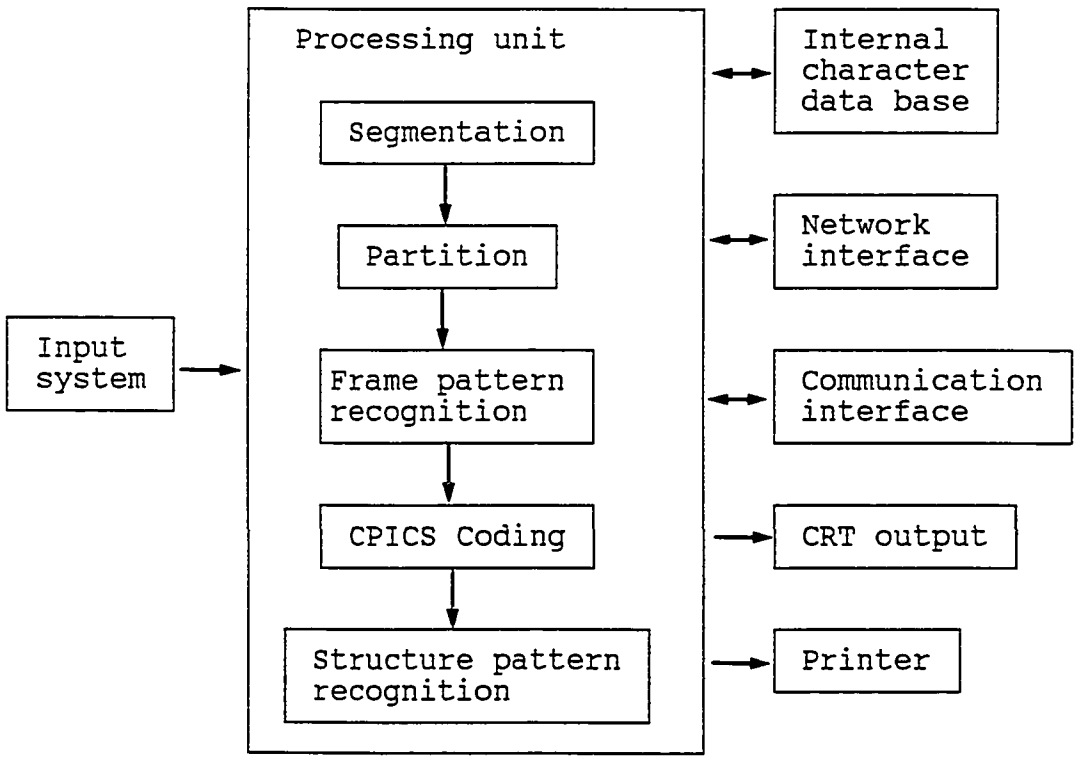


Figure 11. A flow chart - The CPR method for the Chinese character recognition.

any of the commonly available writing or printing styles. Although the actual technical aspects of the hardware of the input system are outside the general discussion of the CPR method, it is a relevant factor in helping explain the implementation aspects of this approach.

5.1 Segmentation

Segmentation is a process by which the segments (Section 4.1.2) are used to re-compose an input character. As stated earlier, the basic elements of Chinese characters are lines, arcs, and points. Segmentation uses segments to substitute for the above three attributes of lines, arcs, and points. Through the process, a character will be converted into its construction (skeleton) form.

12a 12b 12c 12d

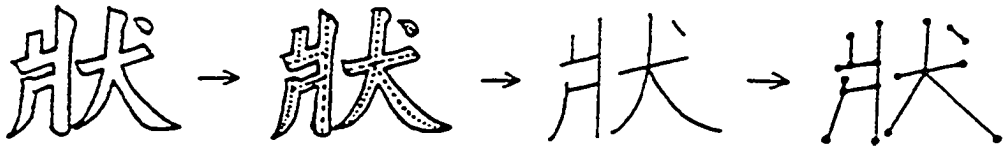
Figure 12. A process to substitute segments for points and arcs of a Chinese character "success".

One may notice that since the line which belongs to the "Class 1" requires no conversion, one only has to deal with points and arcs. A point is easily treated as the "Class 0". The substitution for an arc is determined by its end vertices. Figure 12 illustrates the process of the substitution.

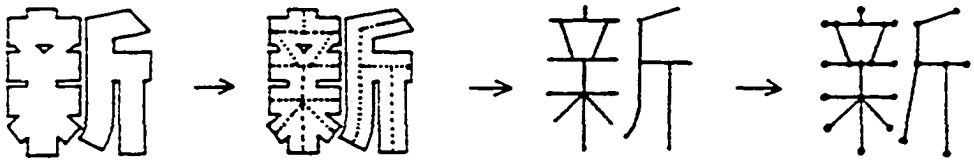
The Chinese character of Figure 12 which means "success" in English is an orthodox style of Chinese writing. Figure 12b substitutes dotted segments for the arcs in Figure 12a. The segmentation process is based on the ability to trace an arc from one vertex of the character to the next, then substituting a segment for an arc. The result from Figure 12b is shown in Figure 12c. The segmentation process thus allows the lengths and angles of the various segments in a given construction to be freely modified, while the substitution of arcs, points and lines will alter the representation. The segmentation process does not affect the construction when recomposing a character. Figure 12d illustrates this concept.

The second function of segmentation is to condense some special strokes that may appear in such cases as calligraphy. Segmentation "squeezes" the thick calligraphy stroke into a thin stroke line. Figure 13a is an orthodox calligraphy style of the Chinese writing. This character is then squeezed into a construction that highlights the central lines of the

strokes as illustrated in Figure 13b and 13c. If its construction includes arcs, then it will be segmented again and converted into a final segmented construction shown in Figure 13d. Figure 13e is the art-style representation of the Chinese writing; its construction is determined by the same method illustrated from Figure 13f to 13h.



13a 13b 13c 13d



13e 13f 13g 13h

Figure 13. A process to squeeze strokes of a character into a segmented construction.

Segmentation is used to adjust input characters, very much like a digital processor is used to convert an analogue signal into a digitized signal. When a processor accepts a character through an input system, no matter what the graph of the character is, segmentation will determine the substitution and converts it into a clear, segmented construction. The constructions of Chinese characters need to be characterized by some principles. The next section introduces the partition as a methodology that classifies constructions.

5.2 Partition

Partitioning is a process which divides a segmented construction obtained from segmentation into one or several disjoint parts. Essentially, each part is termed as a "frame" which, by itself, is treated as a finite connected undirected graph [13].

With partitioning, a processor can divide a graph into frames. Let a construction of a character be treated as a graph G(V,E). It consists of a set of vertices, $V=\{v_1,v_2,..,v_n\}$ and a set of edges $E=\{e_1,e_2,..,e_n\}$. Partitioning will distinguish the connected components and frames from the graph G(V,E). The connected component is a subgraph G'(V',E') of the graph G(V,E) which is not connected with other subgraphs; i.e. if $V' \subseteq V$, $E' \subseteq E$, then for every $x,y \in V'$, and $e \in E'$, there

exists an edge e connecting x and y.

In order to determine these connected components, Tremaux's Algorithm [13], a Depth-First Search technique, may be applied to scan the graph. It traverses all the vertices in the connected graph with no preplanning. The construction of a Chinese character may be a connected or disconnected graph. If the construction is not a connected graph, The algorithm will be repeatedly applied to the disjoint components to determine the frames of the construction.

Tremaux's Algorithm:

(1) Let s be a starting vertex in G(V,E)
(2) If d(s)=0 then go to (10)
(3) s <- v
(4) If all edges of v are marked then go to (9)
(5) Choose an unmarked edge of v, mark and trace it to the next vertex, v'
(6) If d(v')=1 then go to (8)
(7) v' <- v; go to (4)
(8) Back trace the previous v; go to (4)
(9) If there is an edge just traced in, then go to (8)
(10) Mark a frame G'(V',E'); stop

The partition process may start in any one of the vertices, v, on the construction, G(V,E). First, choose one of edges incident to v, then mark this edge and trace along to its next vertex, v'. Repeat this routine until d(v')=1 is encountered. Back track along the edge which was just traced to the previous vertex, then keep trying its incident edges with the same manner until every edge is marked. Then back

track to its previous vertex and execute the same process. Finally, the starting vertex is met again while all its incident edges are traced already, and the process halts. These traversed vertices and edges compose a frame G'(V',E'). The partition algorithm is described as follows.

The algorithm forms one frame of the construction per cycle. If G'(V',E')=G(V,E), then the partition operation is terminated; otherwise, the process is repeated and will continue with any vertex as the starting vertex of the remaining graph, G(V,E)-G'(V',E'), to determine the second frame G"(V",E"). This process is iterated to search for frames until G(V,E) is completely divided. The algorithm is only initially applied to determine the connected components for the frame analysis. There are features which may be implemented in hardware design not discussed here which makes the partition process simpler.

The first and most popular partition approach used to convert a character into frames was developed by the Three Corner Coding Method [18]. There are other similar methods [8,21] such as the Tsang-Chieh system based on the concept of frame. These methods, however, have been designed as users' tools. The user determines the necessary frame analyses for the processing by the computer. From the previous discussions, it is apparent that an ideal inputting method should be

designed with the computer as the main device to carry out the analysis in total, and thus will leave the end user without additional work. The partition process gives the computer this capability. The difference in separating a construction between the CPR method and the Three Corner Coding Method, is shown in Figure 20.

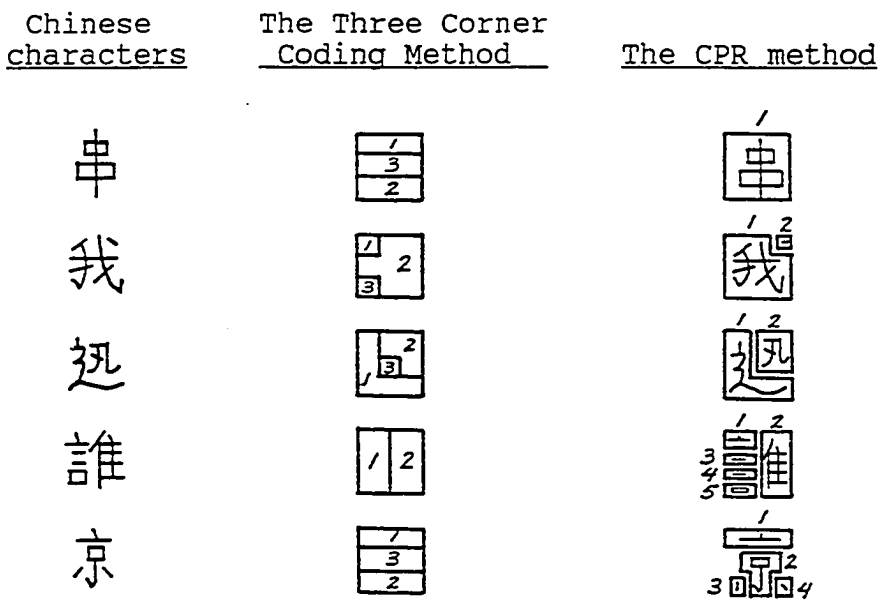


Figure 14. The comparison between partitions of the CPR method and the Three Corner Coding method.

The construction of a Chinese character may be divided into one or more frames. If each frame is represented by an array or a CPICS code, then frames of the construction will be represented by a set of arrays or CPICS codes. This concept

converts a construction into a matrix for ease in computer processing. Figure 15 illustrates the conversion of a segmented character into its matrix representation.

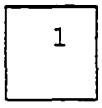
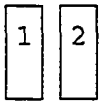
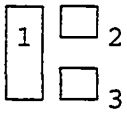
Segmentation	Partition	Arrays	CPICS matrix
		1-((2,2,4,2))	(0300010000)
		Total:	(0300010000)
		1-((2,3,2,0),(2,3,2,0)) 2-((2,3,3,0),(2,3,3,2))	(0402000000) (0304000000)
		Total:	(0706000000)
		1-(2 , 0) 2-((2,3,2),(2,3,2)) 3-(0 , 0)	(0100000000) (0402000000) (0000000000)
		Total:	(0502000000)

Figure 15. The conversion from a character into its array-based representation and CPICS matrix.

5.3 Pattern Recognition

The process of Pattern Recognition in the CPR method includes two steps: (1) The frame pattern recognition which determines the number and type of frames to be dealt with, and (2) the structure pattern recognition which determines, in essence, the content of the frame. After the processes of segmentation and partition, a frame analysis is applied to

start the pattern recognition process, by comparing the number of frames and the patterns of frame types stored in the computer memory. This process is called frame pattern recognition.

When the construction of a Chinese character is converted into its associated frames, the frame type within an imaginary square which bounds the frames of the character construction is one of the major points that distinguishes it from all the other frames. A preliminary frame analysis is shown in Figure 16. The examples in Figure 16 are of the one-frame, two-frame, and three-frame category. Frames like these, and other higher order frames, are built as an index table of patterns of frame types, and stored in the computer memory. Figure 16 indicates

Chinese characters	Frame type	Qty of frames	Chinese characters	Frame type	Qty of frames
天 夫		1	凶		2
昌 字		2	下 卞		2
林 岫		2	因 叉		2
成 我		2	川 捫		3
夙 向		2	三 享		3
匠 匣		2	畾 蟲		3

Figure 16. Frame analyses for some Chinese characters.

some types of the frame arrangements that exist, and is by no means an exhaustive representation. The above mentioned frame types are determined by the partitioning process. This is very much different from some other frame recognition approaches that were intuitively developed by adapting a process from the human point of view.

After the frame pattern recognition has been executed, the structure pattern recognition follows. This process performs a structure analysis which can be treated as a branch of a frame analysis. The structure analysis deals with the structure (i.e. arrays or CPICS code); namely the content of each frame.

Generally, the order of writing Chinese strokes begins from the upper left corner or the upper center position. This principle is useful when designing the procedure for executing the partition. It is also useful in the frame building phase of a Chinese character, the left frame, top frame, or outmost frame is always established first. This approach makes the frame pattern recognition and the structure pattern recognition convenient methods of comparison. Using this idea to build a "good order" of the frame arrangements will save time during the execution phase of the structure pattern recognition. For example, some frame patterns just corresponds exactly to one character. If this kind of pattern of frame is

chosen, one needs to check only the structure of the left, top, or outmost frame. Since the special structure of the ordering of the frames and the contents of the frame determine its corresponding character, it is not necessary to perform further comparisons.

Figure 17 gives an example of a three-frame category. This may, in turn, be further divided into many different frame types, one of which is chosen here to illustrate the process of the structure pattern recognition.

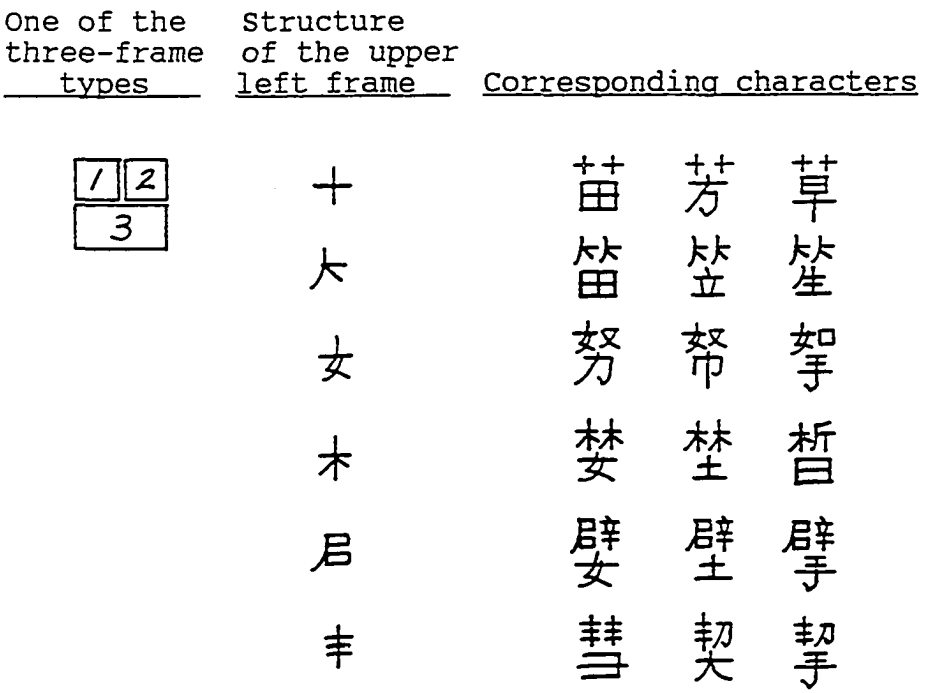


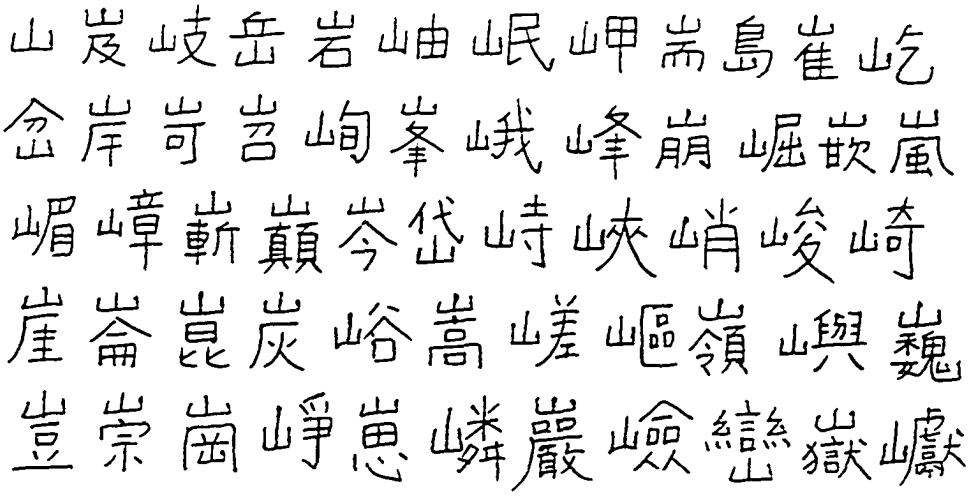
Figure 17. Structure analyses for one of frame types of a three-frame category.

The structure of the upper left frame of Figure 17 is compared first with a pattern stored in memory. Since there are some possible characters that may be matched based on the position of its corresponding contents, one must continue the search by now checking the upper right frame. If this upper right frame is matched as well, and has multiple alternative words, it is necessary then to continue by going to the third frame. A good approach for the software design should alternately use frame pattern recognition and structure pattern recognition. The example depicted in Figure 17 shows some structure analyses and their corresponding characters of the upper left frame. The several characters shown belong to one of the frame types of the three-frame category.

For many Chinese characters, the first frame determines the character and scanning the remaining frames is unnecessary. If the first frame doesn't suffice to determine the character, the second frame is scanned, and so on until the character is recognized. Although a character may not be recognized after scanning some number of frames, it may become known how many frames the character contains without having scanned the entire character. This greatly reduces the search space after a given frame or frames has been identified.

As an example, consider characters beginning with 山 -- of which there are 57 characters, but only 10 characters so

beginning with exactly 4 frames, as shown in Figure 18. The general strategy of only partially scanning a character not only saves time spent matching characters in different equivalence classes, but also substantially helps recognizing characters where some of the frames are only partially legible.



18a

18b

Figure 18. Comparison of 18a, all characters beginning with 山 and 18b, just those with 4 frames.

Eventually, all the existing categories and their corresponding characters are developed and stored as a complete table that includes all the Chinese characters, and

is stored in the system's memory. Note that the pattern used in the CPR method is that of the construction type. These patterns are of frame types (arrangement) or frame structure (content). They may be stored in the memory in the form of matrix. The complete table is coded in CPICS, and is compatible with the existing input systems. Structure pattern

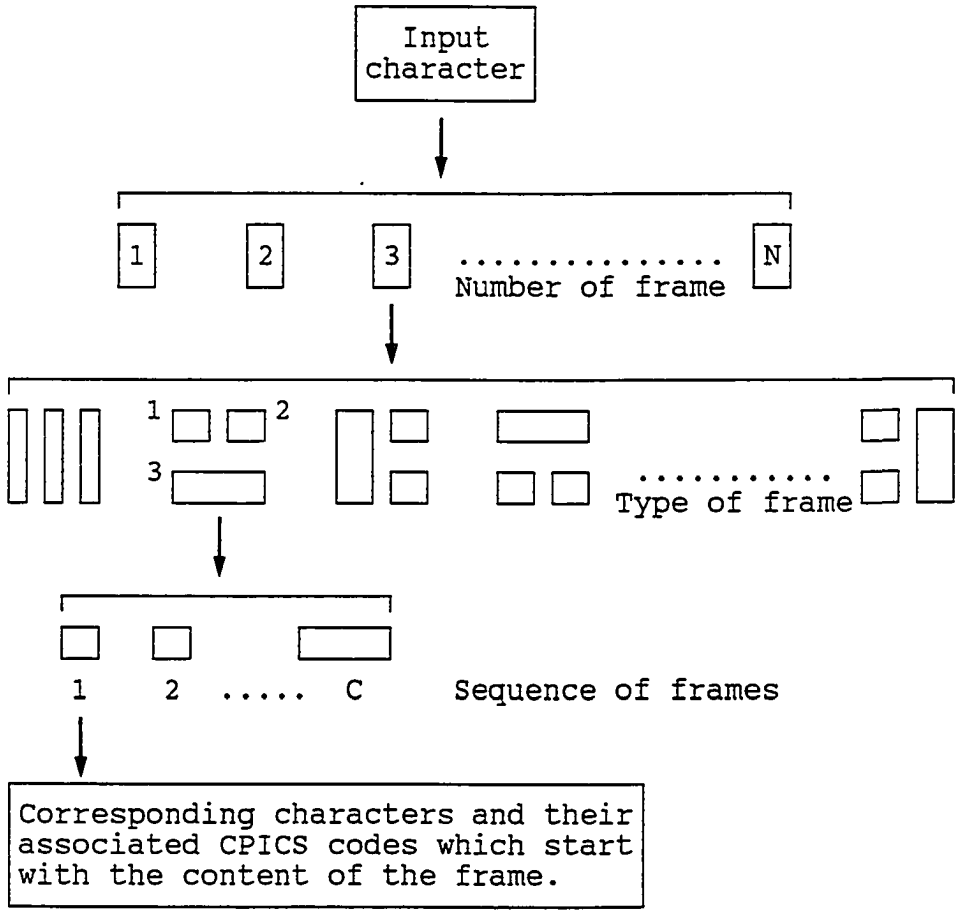


Figure 19. A tree diagram demonstrates the searching path from the number of frames, the type of frames, the specific sequence of frames to the CPICS codes.

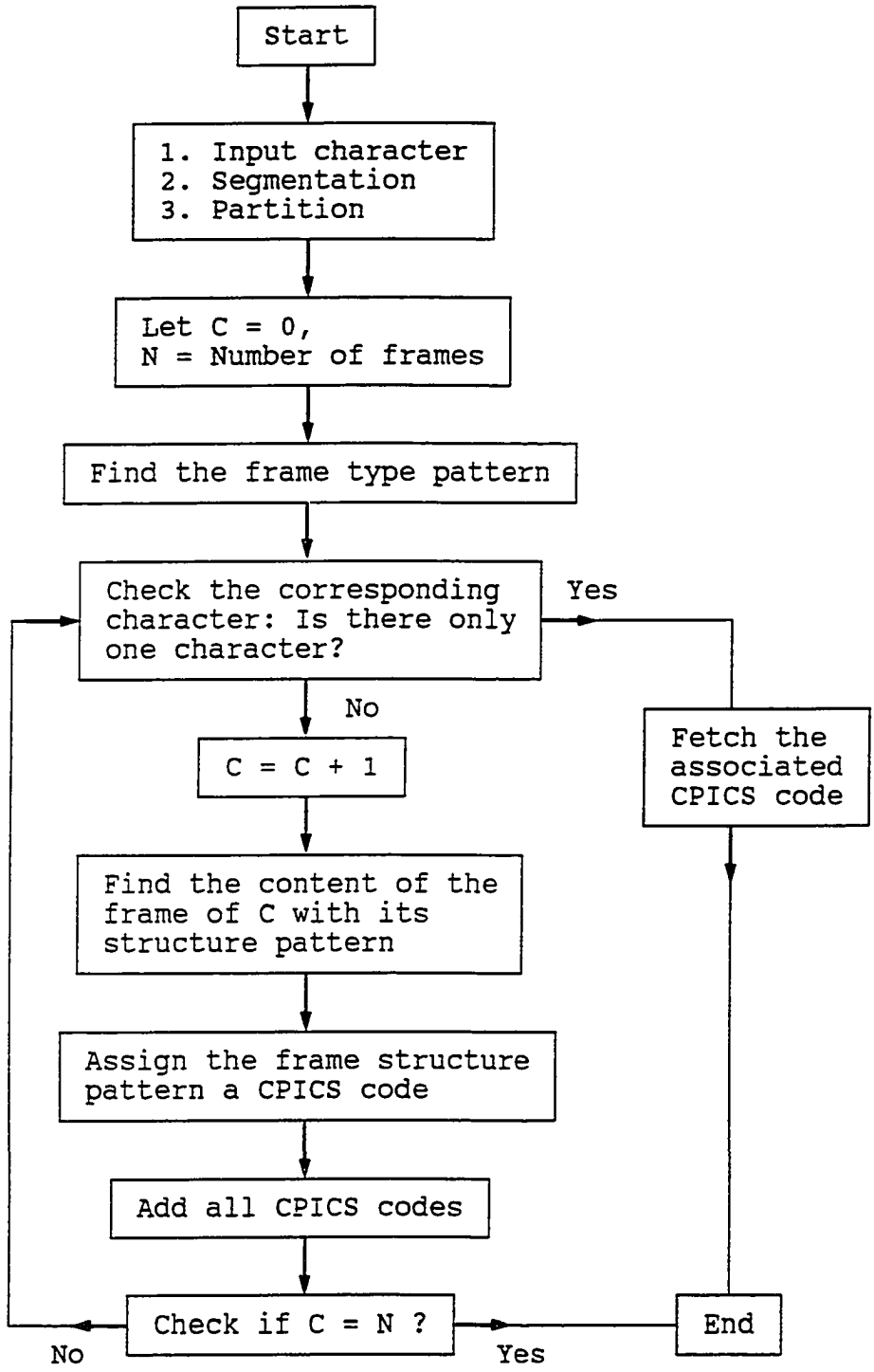


Figure 20. Process of frame pattern recognition and structure pattern recognition.

recognition is the last step of the CPR implementation. The process of the pattern recognition is shown by the Figure 19 and the flow chart scheme depicted in Figure 20.

5.4 Benefits of Using CPR Method

The CPR method is a new approach which can be used in different aspects of dealing with the Chinese characters. Besides the advantages stated in Section 4.2.4, some of the dominant advantages of the CPR method are:

(1) Friendly to the end users. Users are able to use the inputting method without any complex learning of rules and/or extensive training.

(2) Flexible to different languages. It accepts all kinds of characters such as the English character "A" shown in Figure 21, in various strokes, sizes, styles of writings, and patterns.

(3) Allowing different sequences of writing strokes. Unlike some handwritten recognition systems where the sequence of writing strokes must be in some specific order to determine the character. The CPR method allows users to write the character in any sequence of strokes.

21a 21b 21c 21d

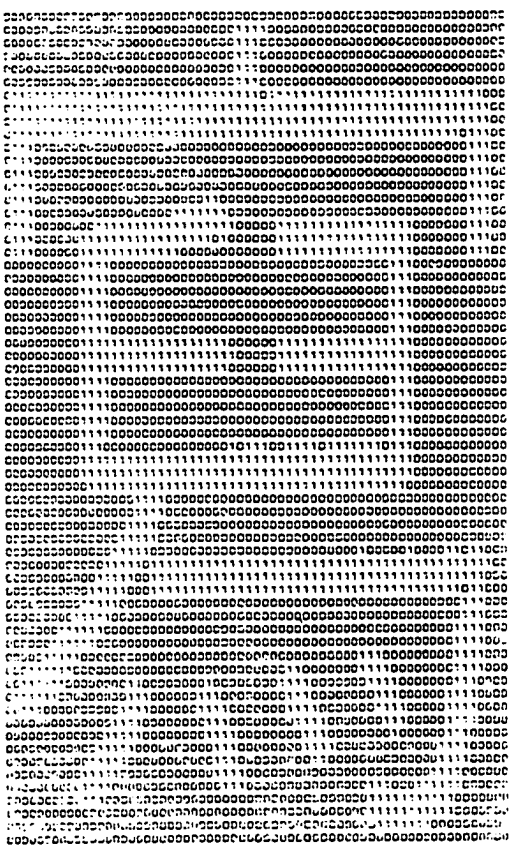
Figure 21. Character "A" in 21a has two vertices in "Class 3"; the vertex of "Class 2" is temporarily ignored here. 21b and 21c are the same characters as 21a, 21d is theoretically same as 21a, but is not necessarily recognized by people.

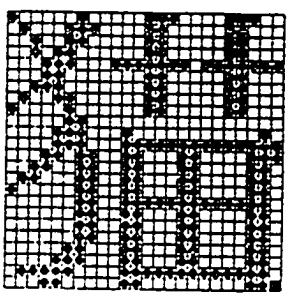
(4) Flexible with the appearance of writing strokes. It can easily recognize different writings of the same character, or a character from many different characters.

(5) Compatible with present inputting systems. It can be connected to different internal coding systems for storage, and to external coding systems for user's inputting, as well as outputting equipment.

(6) Working with a multi-input system. It allows the English keyboard to be used together with other auxiliary equipment; and it does not affect the present computer systems.

(7) Small memory space occupied. Since the CPR method only uses the construction pattern which consists of the number and type of vertices, which may be a vector or a number of the CPICS, it requires relatively small amounts of memory instead of the huge requirements of imaging files or bit





CPICS representation:
(0805000000)

CPICS representation:
(0505040000)

Figure 22. In contrast to 64x64 and 24x24 matrices, the CPR method saves substantial amounts of memory space.

mapping. Figure 22 shows two different matrix representations of 64x64 ("write" in English, [12]) and 24x24 ("cat" in English, [8]) as a comparison with the CPR.

(8) Speeding the recognition processing. The CPR method uses the number and type of the frames and can save a lot of time in matching patterns.

(9) More natural and efficient. It provides a better way to configure a Chinese character. For example, it does not need memorization of any charts such as in the Three Corner Coding system. It processes a natural construction pattern of a character. Any one can use it for any character input.

(10) More tolerance on character recognition. Using the CPR neural-net algorithm, the signal to noise ratio of the image of an inputted character can be high and nevertheless maintain high performance recognition.

(11) Potential for future development of handwritten character recognition. Since the CPR approach can tolerate various size, style, and appearances of the characters, as well as the noise of the data, it is a very effective approach to deal with handwritten characters.

CHAPTER 6

FUTURE RESEARCH: A NEURAL-NET LEARNING AND RECOGNITION ALGORITHM

It is currently impossible to program a computer to do what the human brain can do. Even Patrick Winston, the head of MIT's AI Laboratory, has stated that he has seen disappointingly little progress in standard AI approaches over the last few years [14]. Humans can teach a computer logical rules to deal with imaging data, but how can humans program a computer to recognize the dynamic features of an image? This field is clearly in its infancy stage.

Figure 23 illustrates traditional computer image processing which, however, doesn't work as well as the brain's visual capacities do. The "scene" is the Chinese ideogram for the character "First." The data captured by the scanning devices will be pre-processed through the Expert System

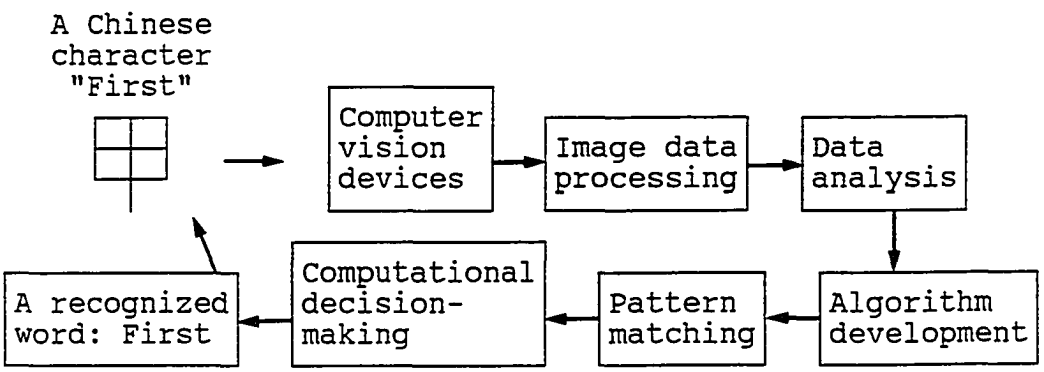


Figure 23. Traditional computer image processing.

approaches before being sent into data analysis. The whole procedure is very cumbersome and slow.

6.1 Why the Neural Network?

Standard digital computer	Neural Network
1. Does only one thing at a time.	1. Does many things at the same time.
2. Operates only as explicitly instructed	2. Discovers new strategies without being told how to do so.
3. Has a distinction between stored data and operations that manipulate that information.	3. Stores the information in a distributed fashion which can be shared among neuronal pathways.
4. Processes all data that its visual devices capture.	4. Uses or ignores some of the information at its disposal intentionally.

Table 2. Comparison between a digital computer and a neural network.

Table 2 is a summary of some of the advantages of a neural network over a standard computer [19]. The neural network is a different approach from standard logical coding. An artificial neural network has the following features which are analogous to those of the human brain:

(1) A neural network allows error and ambiguity to coexist with the correct data and has much more error tolerance than any other computational methodology.

(2) The output of a state function of a neural network

falls into the gray zone between black and white [0,1] similar to those of fuzzy set architectures.

(3) A neural network has a unique way of learning. It doesn't learn from a series of structured instructions or from a set of rules, but from examples. After learning, the output, although neither exactly the same each time, nor completely accurate, is still very precise and accurate.

(4) A neural network consists of nodes and connections joining nodes with weights on the connections. The concepts and terms used to describe the neural network are very similar to those used to describe the human brain: neurons, neuronal firing rates, connection, connection weight corresponding to brain cells, activation, synapses, synaptic strength.

We do not know how the biological neural network works internally. Sometimes, it is difficult to trace what the artificial neural network actually does. For example, if the size of the neural network is too large, the weight calculation will be too complicated to trace without great effort. Artificial neural networks have the performance which is closest to the brain, compared to other current computer architectures. We therefore adopt the artificial neural network as a tool to emulate the human brain.

6.2 Basic Architecture of the CPR Neural Network

Here, we present the design and implementation of CPR using a simple neural network model. In our CPR neural network ("CPRNN"), the strength of an impulse is given by the associated weight on each inter-connection ("synapse"). Figures 24 and 25 show a basic structure of the CPRNN which has three layers of processing units [9,14,25,27].

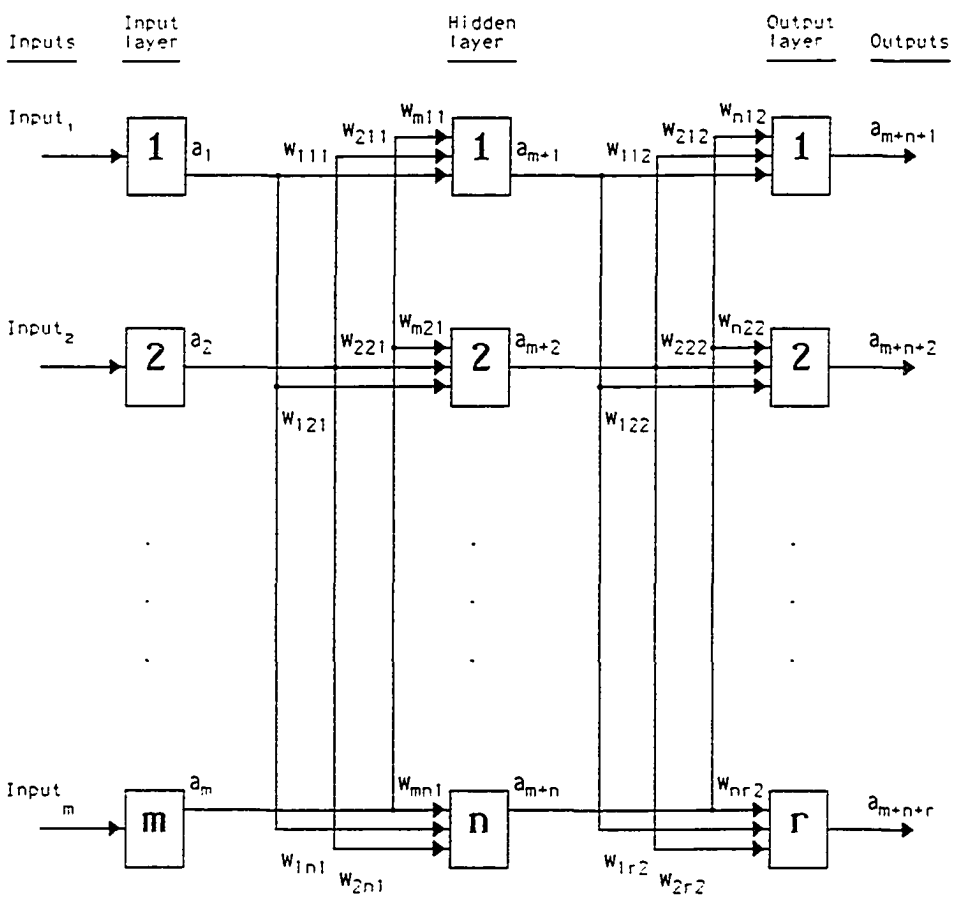


Figure 24. A three-layered, feed-forward, fully interconnected neural network.

The first consists of input units. The second is a hidden layer, i.e. it is not visible from the outside, but rather simply receives input from the input layer and transmits the output to the output layer. The last is the output layer. Each processing unit has forward connections with all units of the succeeding layer. Each interconnection has an associated weight $W_{i,j,k}$, where i and j are the number of processing units in two succeeding layers. K is 1 if the two layers are the input layer and the hidden layer, and k is 2 if the two layers are the hidden layer and the output layer. Each processing unit has an output a_p ($1 \leq p \leq m+n+r$) which is weighted, as discussed, giving $\sum_{i \in P} W_{i,j,k} a_p$, (j is fixed throughout the summation; i.e., it is a parameter) which then is input to a sigmoid transfer function f.

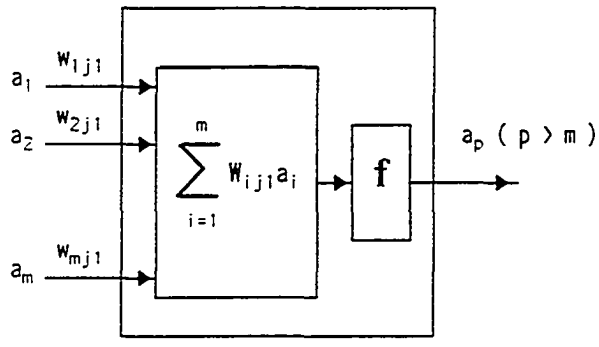
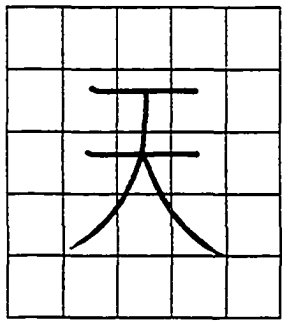


Figure 25. Detail of processing unit j of the hidden layer.

6.3 Conversion of a Matrix-Based Construction Pattern into a CPR Neural Network

In Chapter 4, we discussed a matrix-based approach to the representation of a Chinese ideograph. If we adopt this approach (shown in Figure 9), we can convert the resulting matrices to a CPRNN. There is a direct way, using a grid, to input the construction pattern to the network. Figure 26a shows the Chinese character "heaven" represented in a 5x5 grid. Figure 26b gives a matrix of its construction pattern. If the points of the equivalence classes 0 or 1 are found in the grid, then the value 0 is assigned; otherwise, the degree of the point associated with the remaining equivalence classes is assigned.



26a

0	0	0	0	0
0	0	3	0	0
0	0	5	0	0
0	0	0	0	0
0	0	0	0	0

26b

Figure 26. Converting a grid construction pattern of the Chinese character "heaven" into a matrix used to build a CPRNN.

To provide the capability of reading of the character represented by the above matrix by a CPRNN, the input layer will consist of 25 processing units, corresponding to the linearized matrix. The entire neural network is further described by the following:

Network Title:	Construction pattern recognition
Network Type:	Feed-forward topology
Training Algorithm:	Back-error Propagation
Transfer Function:	Linear/Sigmoid
Input Conversion:	Class 2 - 6
Network Size:	25 Processing units for input layer
	20 Processing units for hidden layer 1
	15 Processing units for hidden layer 2
	6 Processing units for output layer

When a CPRNN is given external input, updating of output values propagates forward from the input layer through each hidden layer to the output layer. An error function is needed for each processing unit, so that the neural network can learn to modify its weights in response to the input. The weight-correction process starts with the output units and propagates backward through each hidden layer to the input layer. This process minimizes the error between the current output and the target output by iteratively modifying the weights, until they converge as shown by the flow chart in Figure 27.

Each input unit has a simple linear correspondence to the value taken from the input matrix. The output of the sigmoid transfer function in each processing unit - except those of

value is then sent via the unit's interconnections to all units of the succeeding layer. The hidden layer works as a feature detector, with each of its processing units performing computations analogous to those shown in Figure 25.

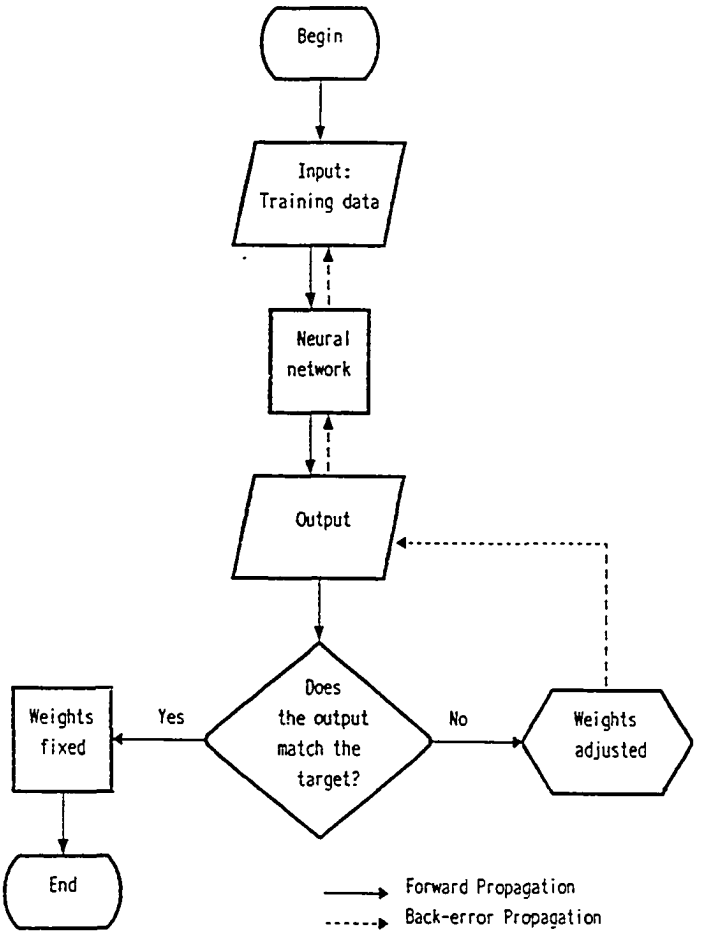


Figure 27. The learning procedure of a neural network with back-error propagation.

After setting up the specifications of a CPRNN, the next step is to choose a set of training data and a set of testing data, both of which include a sequence of pairs of the form {input, target output}. The convergence of weight factors depends heavily upon the correct choice of training data.

6.4 Targets of CPR Neural Network

The goal of the CPR neural network is to let the computer recognize and learn Chinese characters. More specifically, the handwritten characters our system can handle are those close to the printed character, which may even have some errors when compared with the standard printed character. If the handwritten character is too irregular (much like signatures in English), the recognition task is too sophisticated for our system.

Our neural network learning algorithm will be tested on two classes of cases.

(1) A CPRNN applied to one character with different appearances. The computer first learns a standard printed character. Then, we will input some handwritten characters with the same construction pattern as the standard, printed character, but with a different appearance, e.g., in size, position, and/or shape.

(2) A CPRNN applied to different characters. The computer first learns a given standard printed character. After learning this character, the trained CPRNN is given another standard, printed character to learn. After that, a third, and so on, until the CPRNN can learn many different characters.

The recognition of characters uses the above learning procedure (see Figure 27). Recognition of a character from a set of different learned characters is the basic task on which the performance of the algorithm is measured. If the CPRNN can achieve the targets, then we can consider it to be capable of implementing, like the human, additional recognition tasks.

6.5 Implementation of the CPR Neural Network

To reach the goal of building a CPRNN as an actual hands-on application, a specific simulation software package, DynaMind [11], is introduced as a tool here. In this paper, it is not intended to focus on all the aspects of the neural network application and the Chinese ideograph study. Through manipulating DynaMind, we can prove that the concept of the CPR and the learning of CPRNN are really workable.

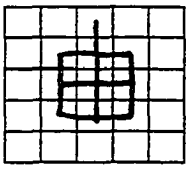
The implementation is carried out on PC-based devices. To speed up the CPRNN learning time, a PC with 486dx2/66Mhz CPU, fast 256K cache memory and Local Bus interface will be

adopted. We will also use the neural network setup which is described in section 6.3.

6.5.1 Preparing the CPR Neural Network Data

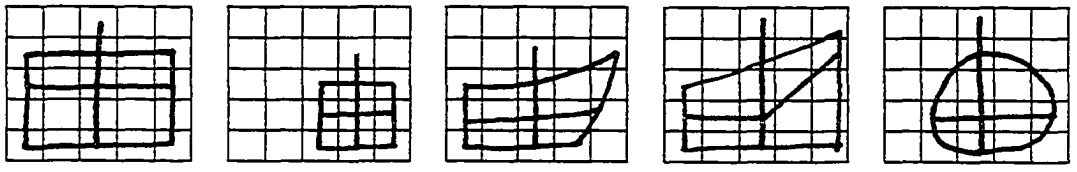
There are two sets of the training data will be used to carry out the targets of the CPRNN stated as the above Section 6.4. Figure 28 and Figure 29 show two sets of training data with matrices and their associated testing data sets with matrices. Figure 28a demonstrates a standard, printed character "reason" as the initial training data. There are 5 characters with the same construction pattern, but with different appearance, size and/or positions, which are used as the testing data shown as Figure 28b.

The way to manipulate the training and testing data on the DynaMind software is based on its design, the input data and target output data of the training data in Figure 28 may be organized as shown in Table 3. The training and testing data is of ASCII format which can be directly transformed by IO BUILDER (DynaMind input/output program) and input into the CPRNN. In Table 3, each input/output pair, IO pair, contains 28 elements. The first 25 floating point values are read in as the input associated with the IO pair name, following the pound sign "#" and a space, on the previous line, the next 3 floating point values are read in as the target output values.



0 0 0 0 0
0 2 4 2 0
0 3 4 3 0
0 2 3 2 0
0 0 0 0 0

28a



0 0 0 0 0 | 0 0 0 0 0 | 0 0 0 0 0 | 0 0 0 0 2 | 0 0 0 0 0
2 0 4 0 2 | 0 0 0 0 0 | 0 0 0 0 2 | 0 0 4 0 3 | 0 0 4 0 0
3 0 4 0 3 | 0 0 2 4 2 | 2 0 4 0 0 | 2 0 0 0 0 | 0 0 0 0 0
0 0 0 0 0 | 0 0 3 4 3 | 3 0 4 0 3 | 3 0 4 0 0 | 0 3 4 0 3
2 0 3 0 2 | 0 0 2 3 2 | 2 0 3 2 0 | 2 0 3 0 2 | 0 0 4 0 0

28b

Figure 28. The first experiment of the CPR neural network. 28a is a training data, 28b is a set of testing data which are the same chracters.

The IO sequence (its name preceeded by the "@" sign and a space) in the file of "pubtt.io" consists of five IO pairs: test1, test2, test3, test4 and test5. Each IO pair represents one of different writing formats of a Chinese character whose 5x5 matrix representations are described in Figure 28. The input values are scaled between 0 and 1. The target outputs are set up with the vale of "0.0 0.0 1.0" for this specific character.

```
* Input file: pubtn.io
* Network size: Input:25,
  Layer#1:15, Output:3
* Data set: Figure 28a
* Number of inputs
25
* Number of outputs
3

@ train1
# train1

0.0  0.0  0.0  0.0  0.0
0.0  0.2  0.4  0.2  0.0
0.0  0.3  0.4  0.3  0.0
0.0  0.2  0.3  0.2  0.0
0.0  0.0  0.0  0.0  0.0
0.0  0.0  1.0

* End of XOR file
```

3a

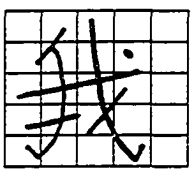
```
* Input file: pubtt.io
* Network size: Input:25,
  Layer#1:15, Output:3
* Data set: Figure 28b
* Number of inputs
25
* Number of outputs
3

@ test1
# test1

0.0  0.0  0.0  0.0  0.0
0.2  0.0  0.4  0.0  0.2
0.3  0.0  0.4  0.0  0.3
0.0  0.0  0.0  0.0  0.0
0.2  0.0  0.3  0.0  0.2
0.0  0.0  1.0

@ test2
# test2

0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.2  0.4  0.2
0.0  0.0  0.3  0.4  0.3
0.0  0.0  0.2  0.3  0.2
0.0  0.0  1.0

@ test3
# test3

0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.2
0.2  0.0  0.4  0.0  0.0
0.3  0.0  0.4  0.0  0.3
0.2  0.0  0.3  0.2  0.0
0.0  0.0  1.0

@ test4
# test4

0.0  0.0  0.0  0.0  0.2
0.0  0.0  0.4  0.0  0.3
0.2  0.0  0.0  0.0  0.0
0.3  0.0  0.4  0.0  0.0
0.2  0.0  0.3  0.0  0.2
0.0  0.0  1.0

@ test5
# test5

0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.4  0.0  0.0
0.0  0.0  0.0  0.0  0.0
0.0  0.3  0.4  0.0  0.3
0.0  0.0  0.3  0.0  0.0
0.0  0.0  1.0

* End of XOR file
```

3b

Table 3. 3a is a training data file for Figure 28a. 3b is a testing data file for Figure 28b.

Figure 29a demonstrates 20 printed characters as the training data. It trains the CPRNN 20 characters at one time, that saves a lot of time but by making no difference from the training of them one by one. Figure 29b shows another 20 characters as the testing data. Some testing data are designed to confuse the CPRNN, some of them have the same construction pattern but with different appearance, size or position, some of them are totally different characters.

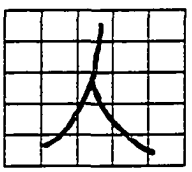
"I"



0 0 0 0 0
0 3 0 0 0
0 4 4 0 0
0 4 4 0 0
2 0 0 2 0

(1)

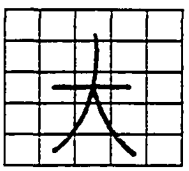
"Man"



0 0 0 0 0
0 0 0 0 0
0 0 3 0 0
0 0 0 0 0
0 0 0 0 0

(2)

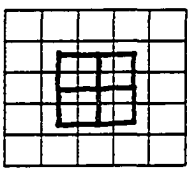
"Big"



0 0 0 0 0
0 0 0 0 0
0 0 5 0 0
0 0 0 0 0
0 0 0 0 0

(3)

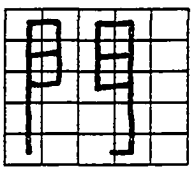
"Field"



0 0 0 0 0
0 2 3 2 0
0 3 4 3 0
0 2 3 2 0
0 0 0 0 0

(4)

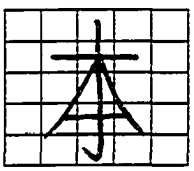
"Door"



2 2 2 2 0
3 3 3 3 0
3 2 2 3 0
0 0 0 0 0
0 0 0 2 0

(5)

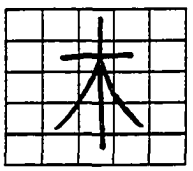
"Origin"



0 0 0 0 0
0 0 6 0 0
0 0 0 0 0
0 3 4 3 0
0 0 2 0 0

(6)

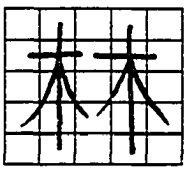
"Wood"



0 0 0 0 0
0 0 6 0 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0

(7)

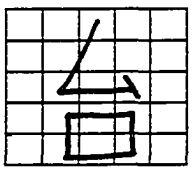
"Forest"



0 0 0 0 0
0 6 0 6 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0

(8)

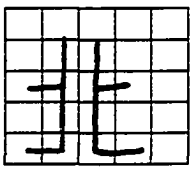
"Stage"



0 0 0 0 0
0 0 0 0 0
0 2 0 3 0
0 2 0 2 0
0 2 0 2 0

(9)

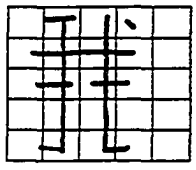
"North"



0 0 0 0 0
0 0 0 0 0
0 3 3 0 0
0 0 0 0 0
0 2 2 0 0

(10)

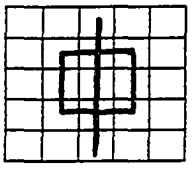
"I"



0 3 0 0 0
0 4 4 0 0
0 4 4 0 0
0 0 0 0 0
0 2 2 0 0

(11)

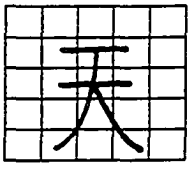
"Middle"



0 0 0 0 0
0 2 4 2 0
0 0 0 0 0
0 2 4 2 0
0 0 0 0 0

(12)

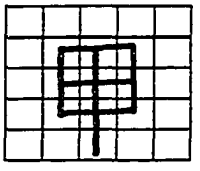
"Weather"



0 0 0 0 0
0 0 3 0 0
0 0 5 0 0
0 0 0 0 0
0 0 0 0 0

(13)

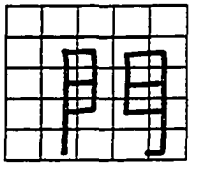
"First"



0 0 0 0 0
0 2 3 2 0
0 3 4 3 0
0 2 4 2 0
0 0 0 0 0

(14)

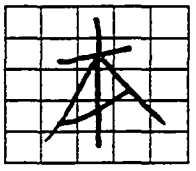
"Door"



0 0 0 0 0
0 2 2 2 2
0 2 3 3 2
0 2 2 2 2
0 0 0 0 2

(15)

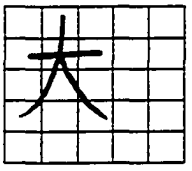
"Origin"



0 0 0 0 0
0 0 6 0 0
0 0 0 3 0
0 3 4 0 0
0 0 0 0 0

(16)

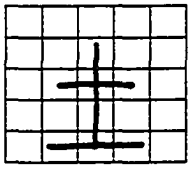
"Big"



0 0 0 0 0
0 5 0 0 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0

(17)

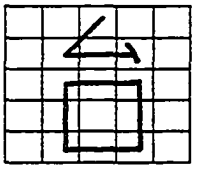
"Soil"



0 0 0 0 0
0 0 0 0 0
0 0 4 0 0
0 0 0 0 0
0 0 3 0 0

(18)

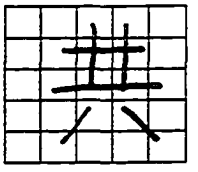
"Stage"



0 0 0 0 0
0 2 0 3 0
0 2 0 2 0
0 0 0 0 0
0 2 0 2 0

(19)

"Whole"

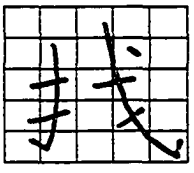


0 0 0 0 0
0 0 4 4 0
0 0 3 3 0
0 0 0 0 0
0 0 0 0 0

(20)

29a

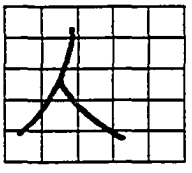
"Find"



0 0 0 0 0
0 0 0 0 0
0 4 0 4 0
0 4 0 4 0
0 2 0 0 2

(1)

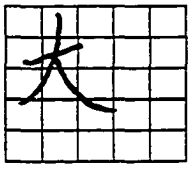
"Man"



0 0 0 0 0
0 0 0 0 0
0 3 0 0 0
0 0 0 0 0
0 0 0 0 0

(2)

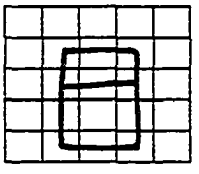
"Big"



0 0 0 0 0
0 5 0 0 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0

(3)

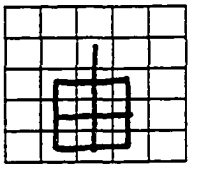
"Sun"



0 0 0 0 0
0 2 0 2 0
0 3 0 3 0
0 0 0 0 0
0 2 0 2 0

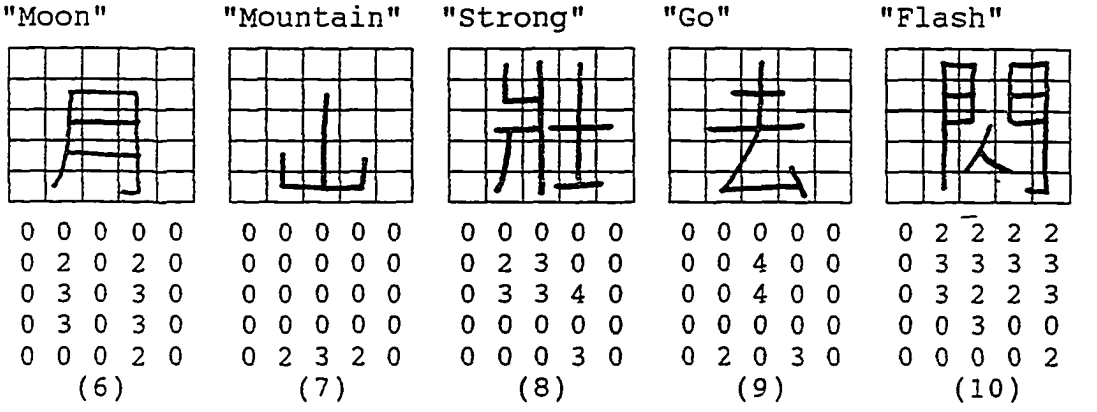
(4)

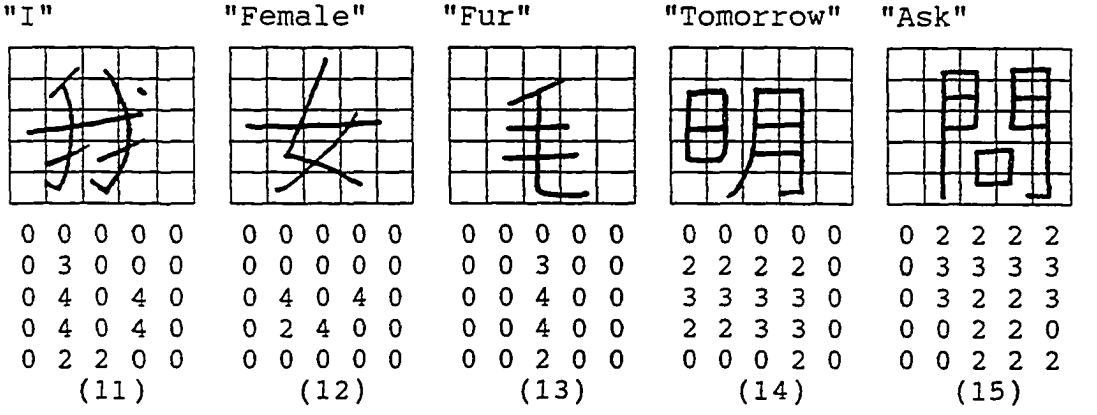
"Reason"

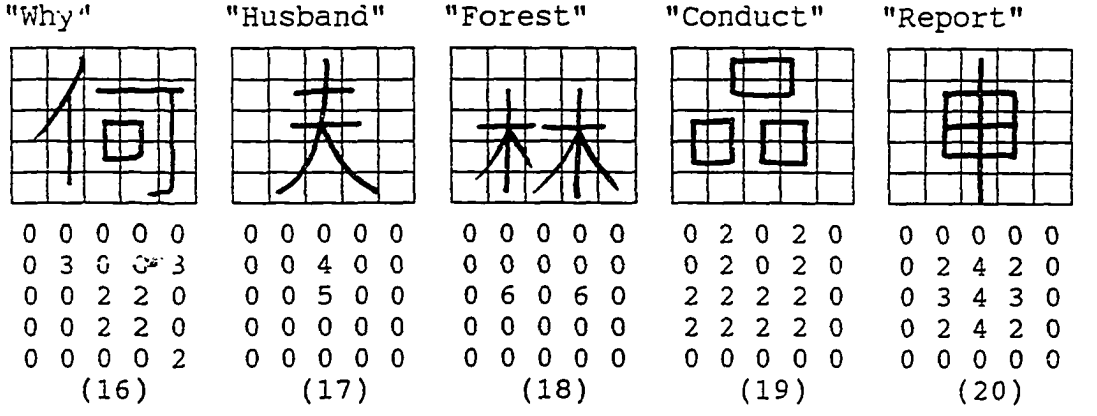


0 0 0 0 0
0 0 0 0 0
0 2 4 2 0
0 3 4 3 0
0 2 3 2 0

(5)







29b

Figure 29. The second experiment. 29a is a set of training data which are different characters, 29b is a set of testing data.

Similarly, the training and testing data of Figure 29 may be organized as shown in Table 4. Each Input/output pair contains 31 elements. The first 25 floating point values are read in as the input, the last 6 floating point values are read in as the target output values.

The IO sequence in the file of "pubtt41.io" as shown in Table 4b consists of 20 IO pairs. Each IO pair represents one writing format of a Chinese character whose 5x5 matrix representations are described in Figure 29b. The input values are scaled between 0 and 1. The target outputs are set up with specific vales for different characters.

```
* Input file: pubtn4.io
* Training data for 20
  different characters
* Network size: Input:25,
  Layer#1:20, Layer#2:15,
  Output:6
* Data set: Figure 29a
* Number of inputs
25
* Number of outputs
6
* 20 pairs of training
  data sets

@ trn0
# trn0

 0.0  0.0  0.0  0.0  0.0
 0.0  0.3  0.0  0.0  0.0
 0.0  0.4  0.4  0.0  0.0
 0.0  0.4  0.4  0.0  0.0
 0.2  0.0  0.0  0.2  0.0

 0.0  0.0  0.0  0.0  0.0
 0.1
```

```
@ trn1
# trn1

 0.0  0.0  0.0  0.0  0.0
 0.0  0.0  0.0  0.0  0.0
 0.0  0.0  0.3  0.0  0.0
 0.0  0.0  0.0  0.0  0.0
 0.0  0.0  0.0  0.0  0.0

 0.0  0.0  0.0  0.0  0.1
 0.0

@ trn2
# trn2

 0.0  0.0  0.0  0.0  0.0
 0.0  0.0  0.0  0.0  0.0
 0.0  0.0  0.5  0.0  0.0
 0.0  0.0  0.0  0.0  0.0
 0.0  0.0  0.0  0.0  0.0

 0.0  0.0  0.0  0.0  0.1
 0.1
```

```
@ trn3
# trn3

 0.0   0.0   0.0   0.0   0.0
 0.0   0.2   0.3   0.2   0.0
 0.0   0.3   0.4   0.3   0.0
 0.0   0.2   0.3   0.2   0.0
 0.0   0.0   0.0   0.0   0.0

 0.0   0.0   0.0   0.1   0.0
 0.0

@ trn4
# trn4

 0.2   0.2   0.2   0.2   0.0
 0.3   0.3   0.3   0.3   0.0
 0.3   0.2   0.2   0.3   0.0
 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.0   0.2   0.0

 0.0   0.0   0.0   0.1   0.0
 0.1

@ trn5
# trn5

 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.6   0.0   0.0
 0.0   0.0   0.0   0.0   0.0
 0.0   0.3   0.4   0.3   0.0
 0.0   0.0   0.2   0.0   0.0

 0.0   0.0   0.0   0.1   0.1
 0.0

@ trn6
# trn6

 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.6   0.0   0.0
 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.0   0.0   0.0

 0.0   0.0   0.0   0.1   0.1
 0.1

@ trn7
# trn7

 0.0   0.0   0.0   0.0   0.0
 0.0   0.6   0.0   0.6   0.0
 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.0   0.0   0.0

 0.0   0.0   0.1   0.0   0.0
 0.0

@ trn8
# trn8

 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.0   0.0   0.0
 0.0   0.2   0.0   0.3   0.0
 0.0   0.2   0.0   0.2   0.0
 0.0   0.2   0.0   0.2   0.0

 0.0   0.0   0.1   0.0   0.0
 0.1

@ trn9
# trn9

 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.0   0.0   0.0
 0.0   0.3   0.3   0.0   0.0
 0.0   0.0   0.0   0.0   0.0
 0.0   0.2   0.2   0.0   0.0

 0.0   0.0   0.1   0.0   0.1
 0.0

@ trn10
# trn10

 0.0   0.2   0.0   0.0   0.0
 0.0   0.4   0.4   0.0   0.0
 0.0   0.4   0.4   0.0   0.0
 0.0   0.0   0.0   0.0   0.0
 0.0   0.2   0.2   0.0   0.0

 0.0   0.0   0.0   0.0   0.0
 0.1

@ trn11
# trn11

 0.0   0.0   0.0   0.0   0.0
 0.0   0.2   0.4   0.2   0.0
 0.0   0.0   0.0   0.0   0.0
 0.0   0.2   0.4   0.2   0.0
 0.0   0.0   0.0   0.0   0.0
```

```
 0.0   0.0   0.1   0.0   0.1
 0.1

@ trn12
# trn12

 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.3   0.0   0.0
 0.0   0.0   0.5   0.0   0.0
 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.0   0.0   0.0

 0.0   0.0   0.1   0.1   0.0
 0.0

@ trn13
# trn13

 0.0   0.0   0.0   0.0   0.0
 0.0   0.2   0.3   0.2   0.0
 0.0   0.3   0.4   0.3   0.0
 0.0   0.2   0.4   0.2   0.0
 0.0   0.0   0.0   0.0   0.0

 0.0   0.0   0.1   0.1   0.0
 0.1

@ trn14
# trn14

 0.0   0.0   0.0   0.0   0.0
 0.0   0.2   0.2   0.2   0.2
 0.0   0.2   0.3   0.3   0.2
 0.0   0.2   0.2   0.2   0.2
 0.0   0.0   0.0   0.0   0.2

 0.0   0.0   0.0   0.1   0.0
 0.1

@ trn15
# trn15

 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.6   0.0   0.0
 0.0   0.0   0.3   0.0   0.0
 0.0   0.0   0.4   0.0   0.0
 0.0   0.3   0.0   0.0   0.0

 0.0   0.0   0.0   0.1   0.1
 0.0
```

```
@ trn16
# trn16

 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.5   0.0   0.0
 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.0   0.0   0.0

 0.0   0.0   0.0   0.0   0.1
 0.1

@ trn17
# trn17

 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.4   0.0   0.0
 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.3   0.0   0.0

 0.0   0.0   0.1   0.1   0.1
 0.1

@ trn18
# trn18

 0.0   0.0   0.0   0.0   0.0
 0.0   0.2   0.0   0.3   0.0
 0.0   0.2   0.0   0.2   0.0
 0.0   0.0   0.0   0.0   0.0
 0.0   0.2   0.0   0.2   0.0

 0.0   0.0   0.1   0.0   0.0
 0.1

@ trn19
# trn19

 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.4   0.4   0.0
 0.0   0.0   0.3   0.3   0.0
 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.0   0.0   0.0

 0.0   0.1   0.0   0.0   0.0
 0.0

* End of XOR file
```

```
* Input file: pubtt4.io
* Testing data for 20
  different characters
* Network size: Input:25,
  Layer#1:20, Layer#2:15,
  Output:6
* Data set: Figure 29b
* Number of inputs
25
* Number of outputs
6
* 20 pairs of testing
  data sets

@ test0
# test0

 0.0  0.0  0.0  0.0  0.0
 0.0  0.0  0.0  0.0  0.0
 0.0  0.4  0.0  0.4  0.0
 0.0  0.4  0.0  0.4  0.0
 0.0  0.0  0.0  0.0  0.2

 0.0  0.0  0.0  0.0  0.0
 0.1

@ test1
# test1

 0.0  0.0  0.0  0.0  0.0
 0.0  0.0  0.0  0.0  0.0
 0.0  0.3  0.0  0.0  0.0
 0.0  0.0  0.0  0.0  0.0
 0.0  0.0  0.0  0.0  0.0

 0.0  0.0  0.0  0.0  0.1
 0.0

@ test2
# test2

 0.0  0.0  0.0  0.0  0.0
 0.0  0.5  0.0  0.0  0.0
 0.0  0.0  0.0  0.0  0.0
 0.0  0.0  0.0  0.0  0.0
 0.0  0.0  0.0  0.0  0.0

 0.0  0.0  0.0  0.0  0.1
 0.1

@ test3
# test3

 0.0  0.0  0.0  0.0  0.0
 0.0  0.2  0.0  0.2  0.0
 0.0  0.3  0.0  0.3  0.0
 0.0  0.0  0.0  0.0  0.0
 0.0  0.2  0.0  0.2  0.0

 0.0  0.0  0.0  0.1  0.0
 0.0

@ test4
# test4

 0.0  0.0  0.0  0.0  0.0
 0.0  0.0  0.0  0.0  0.0
 0.0  0.2  0.4  0.2  0.0
 0.0  0.3  0.4  0.3  0.0
 0.0  0.2  0.3  0.2  0.0

 0.0  0.0  0.0  0.1  0.0
 0.1

@ test5
# test5

 0.0  0.0  0.0  0.0  0.0
 0.0  0.2  0.0  0.2  0.0
 0.0  0.3  0.0  0.3  0.0
 0.0  0.3  0.0  0.3  0.0
 0.0  0.0  0.0  0.2  0.0

 0.0  0.0  0.0  0.1  0.1
 0.0

@ test6
# test6

 0.0  0.0  0.0  0.0  0.0
 0.0  0.0  0.0  0.0  0.0
 0.0  0.0  0.0  0.0  0.0
 0.0  0.0  0.0  0.0  0.0
 0.0  0.2  0.3  0.2  0.0

 0.0  0.0  0.0  0.1  0.1
 0.1

@ test7
# test7

 0.0  0.0  0.0  0.0  0.0
```

```
 0.0   0.2   0.3   0.0   0.0
 0.0   0.3   0.3   0.4   0.0
 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.0   0.3   0.0

 0.0   0.0   0.1   0.0   0.0
 0.0

@ test8
# test8

 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.4   0.0   0.0
 0.0   0.0   0.4   0.0   0.0
 0.0   0.0   0.0   0.0   0.0
 0.0   0.2   0.0   0.3   0.0

 0.0   0.0   0.1   0.0   0.0
 0.1

@ test9
# test9

 0.0   0.2   0.2   0.2   0.2
 0.0   0.3   0.3   0.3   0.3
 0.0   0.3   0.2   0.2   0.3
 0.0   0.0   0.3   0.0   0.0
 0.0   0.0   0.0   0.0   0.2

 0.0   0.0   0.1   0.0   0.1
 0.0

@ test10
# test10

 0.0   0.0   0.0   0.0   0.0
 0.0   0.3   0.0   0.0   0.0
 0.0   0.4   0.0   0.4   0.0
 0.0   0.4   0.0   0.4   0.0
 0.0   0.2   0.2   0.0   0.0

 0.0   0.0   0.0   0.0   0.0
 0.1

@ test11
# test11

 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.0   0.0   0.0
 0.0   0.4   0.0   0.4   0.0
 0.0   0.2   0.4   0.0   0.0
 0.0   0.0   0.0   0.0   0.0

 0.0   0.0   0.1   0.0   0.1
 0.1

@ test12
# test12

 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.3   0.0   0.0
 0.0   0.0   0.4   0.0   0.0
 0.0   0.0   0.4   0.0   0.0
 0.0   0.0   0.2   0.0   0.0

 0.0   0.0   0.1   0.1   0.0
 0.0

@ test13
# test13

 0.0   0.0   0.0   0.0   0.0
 0.2   0.2   0.2   0.2   0.0
 0.3   0.3   0.3   0.3   0.0
 0.2   0.2   0.3   0.3   0.0
 0.0   0.0   0.0   0.2   0.0

 0.0   0.0   0.1   0.1   0.0
 0.1

@ test14
# test14

 0.0   0.2   0.2   0.2   0.2
 0.0   0.3   0.3   0.3   0.3
 0.0   0.3   0.2   0.2   0.3
 0.0   0.0   0.2   0.2   0.0
 0.0   0.0   0.2   0.2   0.2

 0.0   0.0   0.0   0.1   0.0
 0.1

@ test15
# test15

 0.0   0.0   0.0   0.0   0.0
 0.0   0.3   0.0   0.0   0.3
 0.0   0.0   0.2   0.2   0.0
 0.0   0.0   0.2   0.2   0.0
 0.0   0.0   0.0   0.0   0.2

 0.0   0.0   0.0   0.1   0.1
 0.0
```

```
@ test16
# test16

 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.4   0.0   0.0
 0.0   0.0   0.5   0.0   0.0
 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.0   0.0   0.0

 0.0   0.0   0.0   0.0   0.1
 0.1

@ test17
# test17

 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.0   0.0   0.0
 0.0   0.6   0.0   0.6   0.0
 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.0   0.0   0.0

 0.0   0.0   0.1   0.1   0.1
 0.1

@ test18
# test18

 0.0   0.2   0.0   0.2   0.0
 0.0   0.2   0.0   0.2   0.0
 0.2   0.2   0.2   0.2   0.0
 0.2   0.2   0.2   0.2   0.0
 0.0   0.0   0.0   0.0   0.0
 0.0   0.0   0.1   0.0   0.0
 0.1

@ test19
# test19

 0.0   0.0   0.0   0.0   0.0
 0.0   0.2   0.4   0.2   0.0
 0.0   0.3   0.4   0.3   0.0
 0.0   0.2   0.4   0.2   0.0
 0.0   0.0   0.0   0.0   0.0

 0.0   0.1   0.0   0.0   0.0
 0.0


* No more words (@)
  defined
* End of XOR file
```

4b

Table 4. 4a is a training data file for Figure 28a. 4b is a testing data file for Figure 28b.

6.5.2 Trial Testing and Output Interpretation

The target of the CPRNN is to learn the association between the input and the target output. When training a neural network with a given standard, printed character, the input data and the associated target output must converge. After the initial training process is completed, the testing data set will be used to check the performance of the trained network. If the network can not correctly recognize the testing character presented to it, then the network should be retained. To describe the process, a flow chart is demonstrated as the Figure 30.

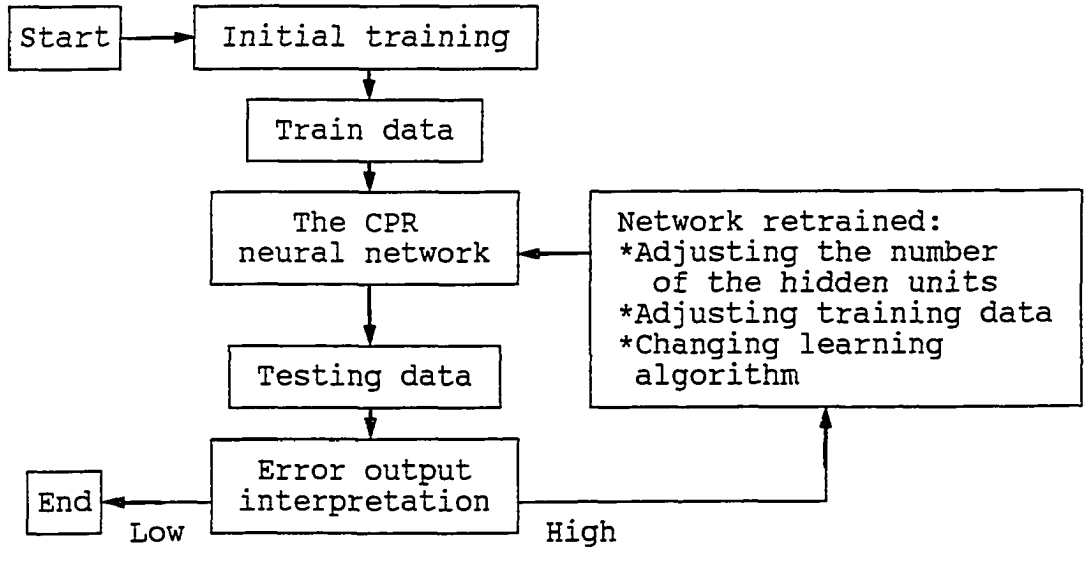


Figure 30. Using the testing data to modify the network until both training and testing data can be identified by the network.

Before starting the training to the CPRNN, the connection weights for each process unit have random values. The network error, E, is defined as the sum of squares of each output process unit's error which is between the actual outputs and the target outputs.

$$E = 0.5 \sum_{j=1}^{n} (\text{target output}_j - \text{actual output}_j)^2$$

During the training session, the error curve should slope down to zero, the network indicates a relatively low error as shown in Table 5. The output report is directly printed from the DynaMind software after 25000 cycles of the IO sequence. The sum of squares of each IO pair is 0.002261, the average sum of square of network error is 0.000377, therefore, the value, 0.986, of the right element of the output array could be interpreted as 1.0. The network has been trained to recognize five different written characters.

```
Sequential IO Presentation Dumped to Disk
Data from File: PUBTT.IO
Network File: PUB.NET
 Element     Sum Sqr.    Individual Relative
             Error       Error for Defined Outputs
        0  0.000228
Rel. Er:  UNDEF  UNDEF  -0.014
Output :   0.010  0.020  0.986
Target :   0.000  0.000  1.000

        1  0.000669
Rel. Er:  UNDEF  UNDEF  -0.014
Output :  -0.040  0.015  0.986
```

```
Target :    0.000   0.000   1.000

         2  0.000494
Rel. Er:   UNDEF   UNDEF  -0.014
Output :   -0.020   0.030   0.986
Target :    0.000   0.000   1.000

         3  0.000602
Rel. Er:   UNDEF   UNDEF  -0.014
Output :   -0.005   0.040   0.986
Target :    0.000   0.000   1.000

         4  0.000269
Rel. Er:   UNDEF   UNDEF  -0.014
Output :   -0.025  -0.000   0.986
Target :    0.000   0.000   1.000

Sum Sqr. Error 0.002261  Avg. Sum Sqr Error 0.000377
End of Seq.
```

Table 5. Results of testing data as shown in Figure 28b presented to the CPRNN.

Similarly, for the data as shown in Figure 29, the result of testing is shown as Table 6. It is one sample data of many trial tests performed. The snapshot error keeps changing but most of them are pretty low (less than one hundredth). The testing data: test1, test2 and test10 should match the training data: trn1, trn2 and trn10 respectively. The output error for the element 1, 2 and 10 in Table 6 are: 0.007903, 0.000000 and 0.016972, which are relatively low. The test2

```
Sequential IO Presentation Dumped to Disk
Data from File: PUBTT4.IO
Network File: PUB4.NET
 Element     Sum Sqr.    Individual Relative
             Error       Error for Defined Outputs
         0  0.034788
Rel. Er:   UNDEF   UNDEF   UNDEF   UNDEF   UNDEF  -4.140
```

```
Output :  -0.005  0.065 -0.124 -0.129  0.030 -0.314
Target :   0.000  0.000  0.000  0.000  0.000  0.100

        1  0.007903
Rel. Er:  UNDEF  UNDEF  UNDEF  UNDEF   0.974 UNDEF
Output :  -0.005 -0.090 -0.119  0.060  0.197 -0.110
Target :   0.000  0.000  0.000  0.000  0.100  0.000

        2  0.000000
Rel. Er:  UNDEF  UNDEF  UNDEF  UNDEF  -0.003 -0.003
Output :  -0.000 -0.000 -0.000 -0.000  0.100  0.100
Target :   0.000  0.000  0.000  0.000  0.100  0.100

        3  0.010027
Rel. Er:  UNDEF  UNDEF  UNDEF  -0.950 UNDEF  UNDEF
Output :   0.005  0.005  0.035  0.005  0.030  0.221
Target :   0.000  0.000  0.000  0.100  0.000  0.000

        4  0.044500
Rel. Er:  UNDEF  UNDEF  UNDEF   0.538 UNDEF   1.543
Output :  -0.000 -0.035  0.482  0.154 -0.085  0.254
Target :   0.000  0.000  0.000  0.100  0.000  0.100

        5  0.008011
Rel. Er:  UNDEF  UNDEF  UNDEF   0.194 -0.650 UNDEF
Output :  -0.010 -0.060 -0.178  0.119  0.035  0.090
Target :   0.000  0.000  0.000  0.100  0.100  0.000

        6  0.012477
Rel. Er:  UNDEF  UNDEF  UNDEF  -0.003 -0.750 -0.950
Output :  -0.000 -0.015  0.245  0.100  0.025  0.005
Target :   0.000  0.000  0.000  0.100  0.100  0.100

        7  0.082652
Rel. Er:  UNDEF  UNDEF  -0.850 UNDEF  UNDEF  UNDEF
Output :   0.005 -0.065  0.015  0.278 -0.015  0.638
Target :   0.000  0.000  0.100  0.000  0.000  0.000

        8  0.028213
Rel. Er:  UNDEF  UNDEF   2.584 UNDEF  UNDEF   2.185
Output :  -0.000 -0.030  0.358  0.178 -0.149  0.319
Target :   0.000  0.000  0.100  0.000  0.000  0.100

        9  0.020749
Rel. Er:  UNDEF  UNDEF   0.046 UNDEF  -0.251 UNDEF
Output :  -0.005 -0.080  0.105  0.114  0.075  0.323
Target :   0.000  0.000  0.100  0.000  0.100  0.000

       10  0.016972
Rel. Er:  UNDEF  UNDEF  UNDEF  UNDEF  UNDEF  -2.489
Output :  -0.000  0.095 -0.154 -0.080  0.030 -0.149
Target :   0.000  0.000  0.000  0.000  0.000  0.100
```

```
       11  0.005073
Rel. Er:  UNDEF  UNDEF  -0.251 UNDEF   0.829 -0.451
Output :  -0.005 -0.100  0.075  0.105  0.183  0.055
Target :   0.000  0.000  0.100  0.000  0.100  0.100

       12  0.030903
Rel. Er:  UNDEF  UNDEF   3.422  0.829 UNDEF  UNDEF
Output :  -0.000 -0.050  0.442  0.183  0.035  0.240
Target :   0.000  0.000  0.100  0.100  0.000  0.000

       13  0.005992
Rel. Er:  UNDEF  UNDEF   0.293  1.070 UNDEF   1.402
Output :  -0.000 -0.060  0.129  0.207 -0.020  0.240
Target :   0.000  0.000  0.100  0.100  0.000  0.100

       14  0.047660
Rel. Er:  UNDEF  UNDEF  UNDEF   1.449 UNDEF   4.850
Output :  -0.005 -0.095  0.144  0.245 -0.005  0.585
Target :   0.000  0.000  0.000  0.100  0.000  0.100

       15  0.031190
Rel. Er:  UNDEF  UNDEF  UNDEF  -1.200 -0.351 UNDEF
Output :   0.005  0.035  0.139 -0.020  0.065  0.388
Target :   0.000  0.000  0.000  0.100  0.100  0.000

       16  0.010961
Rel. Er:  UNDEF  UNDEF  UNDEF  UNDEF  -1.250 -1.500
Output :  -0.000 -0.000  0.105  0.129 -0.025 -0.050
Target :   0.000  0.000  0.000  0.000  0.100  0.100

       17  0.048382
Rel. Er:  UNDEF  UNDEF  -4.275 -0.800  1.496  2.799
Output :  -0.000 -0.020 -0.327  0.020  0.250  0.380
Target :   0.000  0.000  0.100  0.100  0.100  0.100

       18  0.007604
Rel. Er:  UNDEF  UNDEF  -1.100 UNDEF  UNDEF  -1.549
Output :  -0.010 -0.025 -0.010 -0.040  0.085 -0.055
Target :   0.000  0.000  0.100  0.000  0.000  0.100

       19  0.010649
Rel. Er:  UNDEF  -0.550 UNDEF  UNDEF  UNDEF  UNDEF
Output :   0.005  0.045  0.040  0.090 -0.010  0.226
Target :   0.000  0.100  0.000  0.000  0.000  0.000

Sum Sqr. Error: 0.464706  Avg. Sum Sqr Error 0.022129
End of Seq.
```

Table 6. Results of testing data as shown in Figure 29b presented to the CPRNN.

came out almost a perfect zero. These results indicate that CPRNN is able to recognize the 3 same characters from a group of 20 characters.

It is interesting that the results of different trial tests may not turn out to be same all the time, even using the same network set-up and training data. Before training, the connection weights had random values, the initial status is different every time. Sometimes, the neural network is not reliable, but without doubt, it is the closest solution to the human recognition.

The more training data is given to the neural network, the less amount of error appeared in the result. For a well trained neural network, a favorite path occurs, it tolerates errors and accepts different characters with different appearances. From the two experiments above, they show the performance of the CPRNN network could be very similar to human recognition. The artificial neural network is a potential candidate to implement the CPR method in the future. There are still more issues that need to be resolved. Further modifying the matrix-based representation, tuning up the learning algorithm, as well as the transfer function and so on will greatly enhance the CPRNN.

CHAPTER 7

CONCLUSION

In this paper, a useful model of human learning of the construction features of Chinese characters is presented first. A character of a language is a common symbol to the people who use it. Different people have different ways of learning the construction and recognition of a character, but they all finally can recognize the same character and its construction. Furthermore, humans first learn to recognize a printed character and later learn to recognize a handwritten character. These cases indicate that there must be some common, fixed components or patterns extracted from a image of a character to support recognition and learning of characters by people.

Chinese characters and English characters are totally different. This paper shows how the Chinese people use, learn, and recognize pictorial characters. Then, through a learning recognition model of pictorial Chinese ideographs, a concept of construction pattern recognition ("CPR") is formed and presented as a solution.

This paper creates a totally new concept to deal with the character recognition that is based on the graph theory and the computer processing. In the case of Chinese ideographs,

this paper has four separate modules.

(1) Representation of a Chinese character: Use the CPR method to represent a Chinese character with its unique construction pattern.

(2) Representing a construction pattern on a computer: Use a grid or matrix and the equivalence class of degree of points in a character to translate a construction pattern for use by a computer. This is the heart of our contribution. Neural networks are weak computers, but a proper representation, coupled with the abstracting and cooperative effects of the network, can become extremely fast, powerful, and accurate. This paper also presents a new CPICS method which is different from any current coding system to encode the Chinese characters.

(3) Implementation of the CPR method: For an input character captured through the computer vision, segmentation and partition are two steps before executing the CPR method, that determines the number and type of vertices as well as the number and type of frames. Finally, using the contents of the frame to recognize a character.

(4) Future research of the implementation on a computer: Use an artificial CPR neural network to learn and recognize

Chinese characters. The CPR neural network performs as a feature detector which captures the dynamic features of construction patterns. Using the CPR method, the computer can recognize Chinese characters with different appearances and can also distinguish ideographs for some characters from those for others.

The Chinese character is, [when generalized to ignore size, shape, and position of strokes], a fixed set of points and lines. If CPR neural networks can recognize Chinese characters, then they should be able to recognize similar types of pictographs, or more universally - any characters of any kind, giving this proposed work more general applicability.

For future development, the key is the construction pattern - the representation - while the neural network is merely a tool that today seems most appropriate for our task.

APPENDIX 1

Unlike English, Chinese characters are so-called square pictographs or ideographs. The following is shown as a comparative Chart of Chinese and English.

English	Chinese
1. Primitives:	
26 letters.	Strokes. line, segment, arc and point.
2. Properties of primitives:	
Each letter has its own symbol and name with sound.	Some generic names (sound) are used to describe strokes. Strokes are not described precisely in angle, direction and positions.
3. Arrangements of primitives:	
Linear arrangement of letters from left to right.	Strokes are arranged in a virtual square boundary.
4. Arrangements of characters:	
Line sequence of words from left to right.	Independent block, it can be written in any directions (from left to right or from right to left, top down or bottom up.)
5. Expression of a character:	
Can be expressed by either writing or reading its letter.	Only by writing its strokes.
6. Pattern for memorization:	
Order of letters.	Arrangement of strokes.
7. The way to memorize:	
Via the practice of reading, writing or watching.	Via the writing or watching (strokes can not be read.)

REFERENCES

1 Chen, Chang Lin., "A Proposed Chinese Input Method of the Computer: A Construction Pattern Recognition (CPR) Approach Based on Artificial Intelligence." Proc. 50th Annual Meeting of the American Society for Information Science (Medford NJ: Learned Information Inc. 1987) Vol. 24, pp. 36-45

2 Chen, Chang Lin and Rootenberg, Jacob., "New Artificial Intelligence Based Method of Inputting Chinese Characters for Computer Usage." International Journal of Systems Science (London: Taylor & Francis Ltd. 1990) Vol. 21, No. 1, pp. 157-174

3 The Chinese Character Analysis Group, A Comparative Study of Romanization Systems (Taipei, ROC: Council for Culture Planning and Development, 1985)

4 The Chinese Character Analysis Group, Chinese Character Code for Information Interchange (CCCII), Volume II (Taipei, ROC: Council for Culture Planning and Development, 1985) pp. 17

5 The Chinese Character Analysis Group, Chinese Character Code for Information Interchange (CCCII), Volume III (Taipei, ROC: Council for Culture Planning and Development, 1987)

6 The Chinese Character Analysis Group, The Database of Chinese Characters (Taipei, ROC: Council for Culture Planning and Development, 1983) pp. 1-4.

7 The Chinese Character Analysis Group, Variant Forms of Chinese Characters Code for Information Interchange (Taipei, ROC: National Central Library, 1982)

8 Chou, C.C., A Easy Learning of the Tsang-Chieh Input Method (Taipei, ROC: Sung-Kun Computer Publishing Inc., 1990)

9 Dayhoff, Judith E., Neural Network Architectures: An Introduction (New York: Van Nostrand Reinhold, 1990)

10 Chinese Code for Data Communications (Taipei, ROC: Directorate General of Telecommunications, Ministry of Communications, 1983)

11 DynaMind: User's Guide (South Pasadena, CA: NeuroDynamX, Inc. 1991)

12 Technical Reports of the Telecommunication Chinese Code for Data Communications (Taipei, ROC: Editorial Board of the Technical Report of the Telecommunication Laboratories, Directorate General of Telecommunications, Ministry of Communications, 1983)

13 Even, Shimon., Graph Algorithms (Potomac: Computer Science Press, Inc., 1979) pp.53-56

14 Gallant, Stephen I., Neural Network Learning and Expert System (Cambridge, Massachusetts: The MIT Press, 1993)

15 Gonzalez, Rafael C. & Thomason, Michael G., Syntactic Pattern Recognition; An Introduction (USA: Addison-Wesley Publishing Co., Inc., 1982) pp. 9

16 Hsu, Shen., Shue-Wen Chieh-Tsu (China: The Later Han Dynasty, 25-167 AD)

17 Huang K.T., Chang, C.T., Hsieh, C.C. Yang, C.C. and Tseng S.S., "A Multi-Lingual Coding System for Chinese, Japanese and Korean (CJK) data processing." ROC-Japan symposium on Information Management and Exchange: Present and Future (Taipei, ROC: 1986)

18 Huang, K.T., Chang, Y.W. and Hu. L.R., Training Manual for The Three Corner Coding Method (Taipei ROC: System Publication Co. Ltd., 1977)

19 Kosslyn, Stephen M. and Koenig, Olivier., Wet Mind (New York: The Free Press, 1992) pp. 18

20 Leung C.H., Cheung Y.S., and Wong Y.L., "A Knowledge-Based Stroke-Matching Method for Chinese Character Recognition" IEEE Transactions on Systems, Man, and Cybernetics (USA: 1987) Vol. SMC-17, No. 6, pp. 993-1003

21 Liu, In Mao., "Recognition of Fragment-delected Characters and Words" Computer Processing of Chinese and Oriental Language (Canada: 1984) Vol. 1, No. 4, pp. 276-287

22 Lunde, Ken., Understanding Japanese Information Processing (Sebastopol, CA: O'Reilly & Associates, Inc., 1993) pp. 26

23 Malcolm, Jr. Douglas R., Robotics: An Introduction (Boston, Massachusetts: Breton Publishers, 1985) pp. 295-302.

24 Mei, Yung Tso, Tsu-Hui (China: The Ming dynasty, 1368-1643 AD)

25 Nelson, Marilyn McCord and Illingworth, W.T., A Practical Guide to Neural Nets (Reading, Massachusetts: Addison-Wesley Publishing Co, Inc., 1991)

26 Network Development and MARC standards Office, USMARC Character Set: Chinese, Japan, Korean (Washington, DC: Library of Congress, 1986)

27 Simpson, Patrick K., Artificial Neural System: foundations, paradigms, applications, and implementations (New York: Pergamon Press, Inc., 1990)

28 Wei, Y., Research, Development and Evaluation Commission, Executive Yuan, a. The Development of the Chinese Computer in the Republic of China: present and Future (Taipei, ROC: Lu Fong Book Printing Co., 1979) pp 95; b. Ibid., pp91; c. Ibid., pp21; d. Ibid., pp.65.

29 0,1 Technology Co., The Third Generation of the Tsang-Chieh Chinese Character Input Method (Taipei, ROC: Chuan-Hua Technology publishing, 1985)

30 The Computerized Kanji Study of the Kyoto University (Taipei, ROC: National Science Concil, Executive Yuan, 1985) pp. 5, pp. 44-48

31 Pong, H.L., A Shaped and Phonetic Index Searching - A New Chinese Computer Input Method (Miao Li, ROC: Cheng Lien Typewriting Co., 1983) pp. 1-17

32 The Operation Manual of Ging-Ping Handwritten System (Shing-Chu, ROC: Ging-Ping Technology Co., 1983) pp. 1.1-1.11

33 Lee, C.S.G., Gonzalez R.C., Fu, K.S., (Silver Spring, MD: IEEE Computer Society Press, 1983) pp. 300-337