MULTICLASSIFIER NEURAL NETWORKS FOR HANDWRITTEN CHARACTER RECOGNITION

A Dissertation Presented to

The Faculty of the

Fritz J. and Dolores H. Russ
College of Engineering and Technology

Ohio University

In Partial Fulfillment

of the Requirement for the Degree

Doctor of Philosophy

by

Sin-Kuo Chai

March 1995

NOTE:

Text of this thesis was assembled from files left by the author after his graduation. The page layout may differ from the original document submitted to the Fritz J. and Dolores H. Russ College of Engineering and Technology

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Automated Handwritten Character Recognition

Automated handwritten character recognition has increasingly gained popularity in the area of pattern recognition in recent years. The need for fast processing of handwritten documents, due to the increasing amount of produced information, makes this area more important than ever. For example, the great demand for fast classification of letters by the post office requires a fast automated recognition system to do the task. Traditionally, the handwritten character recognition algorithms were implemented on a digital computer. However, since character recognition is computationally intensive, using the digital computer does not meet the demand of fast processing of the task. Digital computers are good at handling problems which are explicitly formulated, but handwritten character recognition is not such a problem. Since the advent of neurocomputing technology, great research effort has been devoted to using it to perform recognition tasks. The neural network is known for its ability of parallel computing since each neuron in the network is able

to perform local computations. Simulating the operations of the biological nervous system, the artificial neural networks can be used in problems which are not explicitly formulated, such the handwritten character recognition.

A number of research techniques for handwritten character recognition in various languages (e.g., [1], [2]) have been developed in recent years. Since features of the characters in a language may be quite different from those in other languages, methods for recognition depend on the languages involved. However, handwritten numerals are probably the most commonly used characters, and methods for recognizing them can be universally applied. On the other hand, since Arabic numerals are written by people in various cultures, their writing appearances vary over a large range. In other words, a numeral written by different people may look quite different and be confused with other numerals. This results in their partial inseparability, which means that different numeral classes may partly overlap each other in the feature space (see Fig. 1.1). Therefore, character recognition for handwritten numerals is a challenging task to tackle.

1.2 Conventional Pattern Recognition Approach

The conventional approach to the area of pattern recognition utilizes a digital computer to implement the pattern recognition algorithms. An object image is first

taken by a camera and then image digitization is performed. The process of digitization is to vertically and horizontally partition the image pixels, and assign a value to each pixel. The value assigned to a pixel of a monochrome image varies according to its brightness or gray level. The digitized image may need further processing using image processing techniques in order to perform the recognition task. To perform the pattern recognition, the digital computer must first learn to distinguish objects of all types based on a set of learning samples. After learning, the computer is ready to classify the input samples.

However, pattern recognition is a computationally intensive and time consuming task due to the vast amount of image data and large number of computation steps. Using the conventional approach always demands a very high speed computer or a parallel computer system to perform a satisfactory recognition.

1.3 Neural Network Approach

Development of the artificial neural networks is driven by study of biological neurons. The neurons in the human body are known by their large parallelism, which gives them the incredible capability of information processing. The number of neurons in a human brain is estimated at approximately in the order of 10^{11}, which makes the computational ability of the brain as high as 10^{14} interconnects per second

[3]. Biological neurons not only can process information concurrently, but also have the ability to accommodate new knowledge. Once trained, neural networks can recognize similar objects according to their learned knowledge. With the properties of computational parallelism and learning ability, the artificial neural networks are widely used in pattern recognition. Examples of the neural network approaches to pattern recognition are presented in [4] and [5].

However, the large number of neurons required and the sophisticated interconnections of the networks used for the recognition task complicate the hardware implementation. For example, a neural network called Neocognition developed by Fukushima, Miyake, and Ito [5] contains 14,529 neurons in the 9 neuron layers to perform the pattern recognition. The design challenge is to build pattern recognition systems with less complexity of connections and smaller number of neural elements while preserving the same performance.

1.4 Combinational Approach

The approach used in this research uses the advantages of the conventional pattern recognition and the neural network approaches to complete the task. Since an object recognition system consists of the learning and the classification stages, one of the stages can be completed by the conventional approach, and the other stage by the neural network. The learning process can be implemented by the conventional approach using a digital computer, even though the process may take a long simulation time. Once the learning is completed, the system is switched to the classification phase. For time invariant tasks, there is no need to involve neural networks in the learning stage so that the hardware complexity can be reduced.

On the other hand, with parallel computational ability, neural networks can perform efficient classification tasks. Hardware organization of a neural network depends on the learning results, and it is much simpler than that of a learning system. In addition to the classification task, in the feature based object recognition, a neural network has to perform the feature extraction. This complicates the hardware of the network even more. An illustration of a neural network feature extractor is shown in Fig. 1.2 [6]. In the combinational approach, the feature extraction can be performed on a digital computer and preprocessed data fed to a neural network. From the experimental results, the system developed in this research consumes the time in the

order of 10^{-2} seconds to perform the feature extraction of a 20×20 binary image. Therefore, for many applications, there will be no need to construct additional hardware for feature extraction.

1.5 Feature-based Recognition

Object recognition is generally performed on either the raw image in the image plane or on the feature representation in the feature space. In the earlier case, known as the low level image recognition, the system learns and recognizes an object according to the information given by all the pixels in the image plane. In an N×N image plane, the object is described by an image vector which consists of N^2 pixel values. The size of the image vector increases as the resolution of the object image increases. One of the drawbacks of this approach is a huge dimensionality which deepens the computational burden of the system. Moreover, the image vector of the shifted object image may be quite different from the original one. On the other hand, not all the pixels of the object image reveal crucial information of the object characteristics, and there is a large redundancy in the image vector.

The feature-based recognition uses only the information that best characterizes the object. It extracts the important information conveyed by some pixels and processes it to obtain the feature representation. The object in the image plane is then

represented by its feature vector in the feature space. In this case, the learning and the recognition is done in the feature space. Dimensionality of the input vector is greatly reduced, and the recognition can be invariant to some image transformations, such as image translation, rotation, and scaling, if the object features are properly selected.

A feature-based recognition system usually distinguishes objects by their shapes since they reveal the geometric attributes of the objects. Various methods for object recognition based on object features have been proposed [7]-[23]. Moments of the object image are often used as the feature representations. For example, Perantonis and Lisboa [7] proposed a moment-based method for feature representations. They used a finite number of Zernike moments to describe an object shape in the 2-D image plane. The Zernike moments are obtained by projecting the object image on a class of Zernike polynomials, which are orthogonal over the unit disk in the polar coordinates. In [8], Sardana *et al*. described a type of moments called edge standard moments for object contour description. These moments are found by normalizing the regular moments of the object contour so that they are invariant to transformations of scaling, rotation, and translation.

Using autoregressive (AR) models [9] to characterize object features is another approach in object recognition. An object in the 2-D image plane is sampled to obtain a 1-D discrete signal, which is to be modeled by an autoregressive process.

The coefficients of the AR model are then used for the contour description. Sekita, Kurita, and Otsu [10] suggested a complex AR model for object recognition. This method samples the contour to a number of points using an equal arc length. These points, represented in complex coordinates, are then fitted to an AR model to acquire an AR coefficient sequence, which is used for contour feature representation.

Fourier-based method is an alternative way to describe the object contour [11]-[13]. In their research, Zahn and Roskies [11] have proposed a type of Fourier descriptors for feature representations of the object shape. The object contour is represented as a parametric function of its arc length expressed by the accumulated direction change since the starting point. This function is then expanded in a Fourier series to obtain the Fourier descriptors for contour representations. In a method for Arabic character recognition, Mahmoud [12] used the Fourier descriptors as one of the feature representations. The contour of the object is represented by a periodic function which is obtained by tracing the contour in the 2-D plane. The Fourier descriptors, then, are the coefficients of the Fourier expansion of the function. In the algorithm developed by Taxt *et al.* [13], after approximation of a 2-D curve by B splines, the object contour is divided into a number of intervals, and the mean curvature in each interval is calculated. Together with these curvature values, the coefficients of the Fourier expansion of the object contour are used for the

descriptors of the shape.

A structural description of the object shape is also employed in some research [14]-[16]. In structural object recognition, the geometric structure of the object is described in terms of a number of its simple parts, such as line segments or arcs. In [15], Nishida and Mori described an algebraic description for the structural pattern recognition. The curve of the object contour is hierarchically decomposed into its primitive parts which are encoded to obtain primitive sequences. The primitive sequences, their connections, and the neighboring structures of a primitive sequence on the branch points are then used for the description of the object shape. In the method suggested by Rom and Medioni [16], each decomposed part is described by its symmetry axes, which are defined as the midpoint loci of the curve cross sections. The cross section is the line between two contour points forming a local symmetry. The symmetry axes and the relationships between each part serve as the description of object shape.

There is a number of research projects on the object representation which are based on the algorithms which sample the object contour and encode it to form a 1-D discrete signal or sequence [17]-[22]. For example, Bebibs and Papadourakis [21] used the centroidal profile function to express the object contour in a 1-D representation. The centroidal profile function is a function of distance from the

object boundary to the centroid whose coordinates are the ratio of first order moments to the zeroth order moments. In the discrete plane, this function is an ordered sequence in which each value is the Euclidean distance from a boundary pixel to the object's centroid. In the method suggested by Dinstein, Landau, and Guy [22], the object contour is detected by a number of 3×3 templates, and each contour pixel is assigned a record which contains the address of and the distance to its immediate neighboring contour pixel in clockwise direction. Both the address in each contour pixel and its length increment are then encoded in two sequences to represent the contour.

Another contour-based feature representation described by Lin, Dou, and Wang represents the object contour by an arc height function [23]. The height of an arc is defined as the length of the line segment from the midpoint of the arc's chord to the arc itself. The line segment is perpendicular to the chord of the arc. Tracing the contour once to find the height of each arc with the fixed length, starting from the current tracing point, results in a set of height values. This set of arc height values is used as the description of the object shape.

1.6 Clustering-based Learning

Clustering is a process that groups a set of data into a number of subsets such

that data in a subset are more similar to each other than to data from other subsets. Learning based on clustering groups a set of patterns from different object classes into clusters which can best represent these patterns. Various clustering techniques have been developed over the past decades.

One of the common approaches is k-means clustering algorithm [24], [25]. It begins with randomly selecting k samples as one-sample clusters and then assigning the rest of the samples to their nearest clusters. The nearest cluster to a sample is the one whose center is nearer to that sample than other cluster centers are. After all the assignments, the center of each cluster is to be recalculated. All the samples are then checked to see if they are in their nearest clusters due to the changes of the cluster centers. If their nearest clusters are changed, another assignment are applied to these samples. The check-and-assign procedure repeats until no new assignment is required, and the clustering is completed.

Agglomerative clustering is a class of hierarchical clustering techniques and has been employed by some researchers [26], [27]. Hattori and Torii [27] proposed nearest neighbor algorithms based on the agglomerative clustering. Initially, the number of clusters is first set to the number of learning samples, which means all the samples are singleton clusters at the beginning. Then, the nearest neighboring clusters are found and merged. The distance between two clusters is defined by the distance

between the nearest pair of data points from the two clusters, respectively. The process of cluster merging continues until the specified number of clusters is obtained. The merging of two clusters corresponds to linking them with an edge between their closest pair of data points. When the merging is completed, this results in a spanning tree with each data or node connected through a path to other nodes.

Clustering can be viewed as finding the optimal data partition under a minimum cost or error function. However, clustering can easily be trapped in local minima, which means that the process terminates at a point which is not the global minimum, and the clustering result is not optimal. To avoid this problem, simulated annealing ([28]) has been used in clustering. It simulates the process of annealing, in which metals are first heated up to a high temperature and then gradually cooled by lowering the temperature according to a specified schedule in order to obtain certain alloys. In the simulated annealing, the activation of the clustering is inversely proportional to the "temperature", according to the Boltzmann probabilities. At high temperatures, the memberships of the learning samples toward clusters are updated very frequently. This is to avoid the clustering residing at a local minimum. It is like an atom which is very unstable at a high temperature and is not trapped in a local minimum. By slowly decreasing the temperature, the clustering mechanism becomes stable and moves toward the global minimum. A well known neural network called

the Boltzmann machine [28] uses the simulated annealing to learn the weights of the network.

The methods described above are all in the category of "hard" or crisp clustering [29], in which a pattern can only be a member of one cluster. There are other clustering approaches based on fuzzy theory, in which an object sample belongs to more than one cluster according to the membership function [30]-[34]. The membership function indicates the degree of membership of a sample to each cluster. Kamel and Selim [31] described algorithms for clustering according to the fuzzy-c-means (FCM) algorithms. An FCM algorithm is an optimization approach to minimize an objective function, which is the sum of weighted variances between the samples and the cluster centers. In order to increase the speed of convergence, the membership grade for each sample and cluster center is checked and updated more often. The learning stops when the object function is at minimum. In the networks developed by Tsao, Bezdek, and Pal [32], the FCM algorithm is incorporated into the Kohenon clustering network (KCN) [33] to improve the performance of KCN. The learning rate for KCN updating at each iteration is evaluated by using the membership updating strategy in the FCM algorithm. Beni and Liu described a fuzzy clustering algorithm [34] which employed the maximum entropy principle to adjust the clustering process. The entropy in the communication areas is used to measure the

average information a group of data contains. In clustering, the entropy can be used to measure the learning effect of the system. At the beginning, the entropy is very high since the system is unlearned and tries to gain as much knowledge as possible. While the learning proceeds, the entropy gets smaller and smaller as the learning effect is reaching saturation. The entropy becomes zero at the end of learning.

The conventional clustering techniques are unsupervised learning techniques, which means no "teacher" during the learning. The system in unsupervised learning has no knowledge about the correct outputs of the corresponding input learning samples. In other words, without a priori probability for each object class, the clustering attempts to find the natural structure in the data in order to partition them into subclasses. This usually leads to a computationally intensive task due to the large number of iterations so that a good partition can be achieved. The problem of the convergence is also a great issue in the unsupervised learning, since the clustering may not converge to the global minimum. In addition, a number of clusters need to be provided to the clustering process by a human, and this may result in a misleading or an unnatural clustering.

In supervised learning, the learning process is under the supervision of a "teacher" that monitors the intermediate learning results. The learning samples associated with their states of nature or class labels are provided to the learning

process. This means that each learning example is provided with an input and its corresponding output, so that the learning effect can be evaluated for classification correctness. During learning, the system can be corrected if the output in response to an input sample is wrong. Learning is completed when the system produces the correct output to each learning sample. The time required for supervised learning is small compared to that for unsupervised learning since there is no waste of time in "guessing" what kind of adjustment needs to be made to obtain the desired results. On the other hand, clustering techniques greatly reduce the size of the data set by representing the learning samples with a small number of clusters. This reduces the computational complexity and eases the classification task in the recognition phase. Therefore, incorporating clustering techniques into supervised learning can result in accurate and efficient learning at a small computational cost.

In this research, clustering-based supervised learning techniques are developed, in which the clustering mechanisms are under supervision, so that each cluster contains a single class of samples. The learning techniques developed are described in Chapter 5.

1.7 Research Objective

As described in Chapter 2, automated handwritten character recognition is a highly challenging and difficult problem to tackle. The difficulties come from various writing styles, image transformations, image resolution, and noise in the image plane. To facilitate the solution of this problem, the objective of this dissertation is aimed at accomplishing two goals, addressed as follows.

A. Recognition Invariant to Writing Style and Image Transformations

The first goal is to develop various handwritten character recognition systems which perform the recognition task invariant to image scaling, translation, and rotation, as well as different writing styles. This objective is realized in three main stages: feature extraction, learning, and classification. The feature extraction stage is aimed at simple but effective extraction algorithms that fetch both global and local features of the object. The learning stage is focused on adaptation of the clustering techniques to the supervised learning, so that good learning results can be achieved. The classification stage is devoted to development of both single and multiple classifier systems in order to obtain good classification performances.

B. Development of Multiclassifier Systems

Each single classifier system has its strength and weakness in handwritten character recognition. Great effort made to optimize the performance of each stage in a single classifier system may not result in a great improvement in recognition by the whole system. To significantly enhance the recognition performance, the second goal of this dissertation is to develop multiclassifier systems which contain various single classifiers. The multiclassifier systems are to organize the developed single classifiers so that their weaknesses can be mutually compensated by the strength of each other. In this way, the classification decision of the multiclassifier can be made based on proper judgement, and the recognition rate can be greatly improved.

The rest of this dissertation is organized as follows. Chapter 2 gives an overview of the handwritten character recognition systems. The problems involved in handwritten character recognition are addressed. The feature extraction technique and the feature representations are described in Chapter 3. Chapter 4 discusses various similarity measures developed for the different feature representations. In Chapter 5, two learning processes based on supervised clustering techniques are described. A number of recognition strategies and classifier systems are depicted in Chapter 6. The experimental results are presented in Chapter 7. Finally, the

discussion and conclusions are given in Chapter 8.

CHAPTER 2

GENERAL DESCRIPTION OF HANDWRITTEN CHARACTER RECOGNITION

In this chapter, a general description of the handwritten character recognition is given. The problems encountered in handwritten character recognition are first identified and discussed. The organization of the system is then presented to generally describe the strategies to tackle these problems.

2.1 Problem Identification

The great variations in handwritten characters make the recognition task hard to be automated [35]. A handwritten character may appear differently from picture to picture due to the image transformations of size, orientation, or location. In addition, the diversity of writing styles results in vast variations in the appearances of characters. A handwritten character presented in the image plane should be correctly recognized by the system in spite of these circumstances. These problems are described in detail in the following sections.

2.1.1 Recognition in the Presence of Image Transformations

One of the recognition problems encountered due to image transformations is the scaling of the object in the image plane. This problem results from the change of size of the object image. As such a change is very common in handwritten documents, it should be overcome by the recognition system. For example, the three upper case A's with different sizes in Fig. 2.1(a) should be recognized as the same one. One way of dealing with this problem is performing an image normalization, so that all handwritten characters presented in the two dimensional image plane are of the same size before recognition. Another way is to extract from the character unique features which are invariant to scaling, so that the system responds strongly to the selected features, no matter what size the character is. In recognition system, once the features of a character are detected by the system, the character can be correctly identified. Fig. 2.1(b) illustrates the features which can be used to characterize the letter A independently of the scaling.

Though the methods described above can solve the scaling problem, they give no help in recognizing the rotated images. When the orientation of the letter A changes, as shown in Fig. 2.1(c), the features selected as parts of 2D image also change since they are not independent of rotation. Consequently, the response of the system to the character may be improper and may lead to a wrong recognition. In this case, normalization of image orientation can be applied so that all the

characters have the same orientation. In the orientation normalization, the elongation axis [36] of the object must be obtained in order to find the direction in which the object is elongated. The elongation axis of an object can be defined by the axis of least second moment, which is given by

$$L = \int\int_{R} r^2 f(x,y)dxdy \qquad \textbf{(2.1)}$$

where r is the perpendicular distance to the axis from the pixel at (x,y), f(x,y) is the characteristic function. The integral is taken over the whole object image region R. The function f(x,y) is zero for all the background pixels and is 1 for the object pixels. The axis to be found is the line for which the second moment is minimum. Fig. 2.2 illustrates the elongation axis of an object image. Orientation normalization is done by adjusting the elongation axes to a certain direction, so that all the characters can have the same orientation.

However, the normalization is based on the assumption that different images which represent the same object have similar elongation axes. While this is true for the images of some tools, this is not true for the case of handwritten characters because of the great variation of their shapes. This means that the elongation axis may not properly define the orientation of a handwritten character. Applying the orientation normalization on the elongation axes may not result in a reliable

recognition under the influence of image rotation.

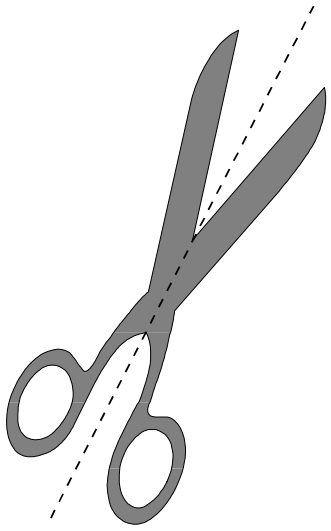

Fig. 2.2 A scissor and its elongation axis.

Another way to solve the orientation problem is using a vast amount of samples, which include characters of various orientations, to train the recognition system. This needs a huge data base and results in a large time overhead for training, testing, and even for recognition of a single character.

In addition to the problems described above, the change of the object

location in the image plane may also cause a false recognition. The location of a character image is important for the systems which perform the recognition on low level images, such as raw images. In this case, each pixel in the image plane is supposed to give some image information. For a 20×20 binary image, the image vector contains 400 elements each of which is a pixel value. Character image change in location results in a new image vector, which may be quite different from its original. In high level image recognition, the problem can be solved by edge detection techniques which detects the object boundary, and the system uses the object boundary for recognition task. However, the boundaries of handwritten characters are not all the same, and size normalization must be employed.

In summary, the problems resulting from image scaling, rotation, and translation are common and complicated in handwritten character recognition, and they can not be solved independently. Character features which are invariant to these image transformations are to be obtained in order to perform good recognition.

2.1.2 Recognition under the Variations of Writing Style

Great shape variations of handwritten characters due to different writing styles yield another recognition problem. Shape variations are the most important

reason why it is much harder to recognize handwritten characters than to recognize printed characters. Handwritten characters have a very wide range of variations from their prototypes. Different people writing the same character may produce incredibly different images. Fig. 2.3 illustrates some groups of handwritten characters which are quite different from each other. Thus, a handwritten character recognition system must have the flexibility to recognize the various writing appearances of the same character without losing the accuracy of distinguishing different characters which have similar appearances. This problem can be overcome by exploiting supervised clustering techniques for the learning process used in this research. Supervised clustering groups similar objects of the same type into clusters. All the characters in a cluster are of the same type. In other words, different writings of the same character may be grouped into several clusters, and the deviation between characters in a cluster is small. Different writings of a character can be clustered together if they are similar to each other, though quite different from their prototype. In this way, the system can be equipped with both the flexibility and the accuracy of correct handwritten character recognition. The supervised learning using clustering techniques is depicted in detail in Chapter 5.

2.2 Organization of Recognition System

The organization of the handwritten character recognition system is described in two phases, which are the learning and the testing (or classification) phases. These two phases are functionally different. The system first obtains the classification knowledge of handwritten characters of different types. Once the learning is done, the system switches to the testing phase to perform the recognition task.

2.2.1 The Learning Phase

The learning phase is composed of four stages: image segmentation, feature extraction, similarity measurement, and learning. The block diagram of the learning phase is shown in Fig. 2.4.

A. Image Segmentation

The term image segmentation in the areas of conventional image processing is used for the process that partitions an image into regions of different categories. In neural network applications, it is the task that extracts individual objects which are contained in an image, so that only one object is in a subimage. In other words, segmentation is to separate individual objects in an image from each other, so that the recognition system can process them individually. Images of handwritten

characters are obtained by applying image segmentation to picture images of survey sheets, as illustrated in Fig. 2.5(a). A threshold process then operates on each character image to obtain the binary image form in which the object pixels and the background pixels are assigned 1s and 0s, respectively, as shown in Fig. 2.5(b). The resolution of the binary image is 20×20, which means that a binary image is in a 20×20 matrix form. After the image segmentation, the binary images of objects are ready for feature extraction.

B. Feature Extraction

The developed system performs the recognition task based on the object features instead of the raw object images. This is due to the fact that object features provide distinguishing characteristics useful for character recognition. Employing critical object features results in more effective performance of the recognition process. In addition, the dimensionality of the feature representation is much smaller than that of the raw binary image. This greatly reduces the learning and the classification time [37]. Therefore, feature based object recognition has the advantages of effectiveness and efficiency.

The stage of feature extraction is to define the useful features and extract them from the binary image of the object. As discussed in section 2.1.1, the selected

features must provide the unchangeable information under image transformations. Also, the features must be easily extracted. When the feature extraction is done, the object is transformed from the two dimensional image plane to a vector representation in the feature space.

C. Similarity Measurement

The purpose of similarity measurement is to define how similar two objects are, based on their feature representations. In other words, it uses quantitative expressions, which are known as similarity measures, to describe the degree of analogy between objects. The similarity measure has to sharpen the likeness of the objects of the same type and deepen the differences between objects of different types.

D. Learning

The learning process considered in this dissertation is under supervision. This means that utilizing the similarity measurement techniques, the system learns to recognize handwritten characters under instructions and uses the given knowledge to incorporate similarity criteria. Learning is not to memorize all the learning samples with their class identifications. It is to accumulate the

classification knowledge under supervision, so that the system can not only identify the learning samples without errors, but also correctly recognize the object samples as different from the learning samples. Learning results in a number of clusters which serve as the prototypes of different classes of samples.

2.2.2 The Classification Phase

As shown in Fig. 2.6, the first three stages in the classification phase are the same as those in the learning phase. All the test images are to go through those three stages before they can be tested. The classification stage performs the recognition based on the knowledge from the learning phase. A test sample is compared to all the cluster centers generated in the learning phase, and the classification decision is made in accordance with the classification criteria. The decision made on a test sample is either to recognize or to reject this sample. The rejection is made when a sample can not be assigned to a unique object class. Otherwise, the classification is made. In multiclassifier systems, samples which are rejected by a single classifier may still be subject to the recognition process in other parts of the system.

In the following chapters, the principles and the algorithms of the feature extraction, the learning process, and the classification process are described in

detail.

CHAPTER 3

FEATURE EXTRACTION AND REPRESENTATIONS

The objectives of the feature extraction are to obtain critical features which best characterize the object in the image plane. In other words, it is desired to extract object features which can be used to distinguish objects of different types. Features that are common to different kinds of objects provide no important information for classification and are to be avoided. Due to the fact that the size, the location, and the orientation of an object may change when presented in various pictures, features which are invariant to scaling, translation, and rotation are the most desirable for object recognition. In this research, we take the shape information of the object as the fundamental feature since it possesses the desired invariant characteristics.

A feature representation is formed by directly organizing or further processing the extracted features in proper ways. After the process of obtaining feature representation, an object on the two dimensional image plane is transformed to its feature representation in the feature space. An object recognition is then performed on the feature representations of objects rather than on the raw images. In this dissertation, three types of feature representations are developed, which are

the angle sequence, vector contour representation [19], and Fourier transform representation.

3.1 Angle Sequence

On the two dimensional binary image plane, an object is composed of black pixels, and the background consists of all the white pixels. Fig. 3.1 illustrates the binary images of a numeral 2 and a numeral 3. The shape information of the object on the image plane can be obtained by finding the geometric relationships between the pixels on the contour (or the boundary). A boundary pixel is defined as the black pixel with at least one of its four sides adjacent to a white pixel. A pixel side is called a boundary side, if it is a side of a boundary pixel which is adjacent to a white pixel. In Fig. 3.2, the pixels which are marked by the letter B are boundary pixels, and the pixel sides which are pointed to by an arrow are the boundary sides. The geometric relationship between two adjacent boundary sides can be expressed in relative orientation. In this way, an object shape is described by a sequence of angle values, each of which represents the orientational relation between two adjacent boundary sides of a pixel or two pixels on the contour.

The sequence of angles of an image can be formed by arbitrarily selecting a boundary pixel side as the starting place and tracing each pixel along the contour

until the starting side is reached. Suppose that we have an object image with N boundary pixel sides. The sequence of angles formed after tracing the image is given by

$$A = \{ \theta_n \}, \; n = 1, \; 2, \; ..., \; N \qquad \textbf{(3.1)}$$

where θ_n is the angle between the n*th* and the (n+1)*th* pixel sides. θ_n can be categorized to three angle values equal to 0^o, 90^o, or -90^o, respectively. The angle 0^o means that there is no change of direction when tracing from a pixel side to its neighboring side. The 90^o indicates that the tracing direction changes by a positive right angle. This happens when tracing two pixel sides which belong two different pixels. On the other hand, -90^o results from tracing two adjacent sides of the same pixel. The three angles 0^0, 90^0, and -90^0 degrees can be represented by three integer numbers called angle values 0, 1, and -1, respectively, as shown in Fig. 3.3. The sequence consisting of these angle values is called the angle sequence, which is given by

$$\iota) = \begin{cases} 0, & for \; \theta_n = 0^o \\ 1, & for \; \theta_n = 90^o \\ -1, & for \; \theta_n = -90^o \end{cases} \qquad n = 1, \; 2, \; ..., \qquad \textbf{(3.2)}$$

Fig. 3.4 shows the angle sequence of the binary image of a handwritten numeral 2.

The angle sequence is circular. In the sequence, the last value, x(N), is the angle value from the last boundary side to the starting side. This is to say that by starting from a pixel side and traveling the object boundary a number of times, we will obtain a periodic sequence $x_N(n)$ with period N, which is

$$x_N(n) = x(n+kN),\ k = 0,\ \pm1,\ \pm2,\ \pm3,\ \ldots \quad \textbf{(3.3)}$$

As a result, two angle sequences which are obtained by tracing the same object image from different starting pixel sides are circularly identical. One sequence is a circular shift of the other. Suppose that the starting point of one sequence is a circular shift of the second sequence shifted by r points. Circularly shifting the first sequence r points in the opposite direction results in exactly the same sequence as the second one. Circular shift of an angle sequence corresponds to a rotation of its object image. The angle sequence is invariant to rotation, since the change of orientation of an object results only in a different starting point of the same angle sequence. It is also independent of translation, since the angle sequence depends only on the boundary pixels of the object, regardless of its location in the image plane.

3.1.1 Angle Sequence Extraction

As described above, the angle between two adjacent pixel sides can be categorized by three different angles, 0^o, 90^o, -90^o. As shown in Fig. 3.3, these angles are measured with respect to the north side of the lower-right pixel (pixel 4) in the 2x2 window. The three templates in Fig. 3.3 are the basic angle extraction templates used to fetch the angle value between two pixel sides. In the same manner, three more sets of the basic angle extraction templates are developed to extract the angle values for the south side of pixel 1, the west side of pixel 2, and the east side of pixel 3. These 12 angle extraction templates are shown in Fig. 3.5. According to the patterns of the angle extraction templates, only the boundary sides of the pixels on the contour are to be assigned angle values. The next neighboring boundary side of the current boundary side may be a side of the same pixel or of the adjacent pixel, depending on the local pattern of the window. For instance, the next neighboring side of the north side of pixel 4 in Fig. 3.3(a) is the north side of the pixel 3 while the west side of the pixel 4 in Fig. 3.3(c) is the next neighboring side of the north side of the same pixel. By moving a 2x2 window only once over the whole image and applying these templates at each window position, the angle sequence of the object image can be easily obtained.

When performing angle extraction of an object image, there is no need to distinguish whether or not a pixel is on the contour. After the process of angle

extraction, the boundary pixels will produce angle values while no angle values will be yielded by the non-boundary pixels. It is due to the fact that when applying the extraction templates to an object image, a boundary pixel at a certain window position will match one of the 12 template patterns, but a non-boundary pixel will not match any of the them at any window position.

The angle sequence can be used as a description of the object shape since it contains the geometric information of the object contour. It also can be further processed to obtain a vector contour representation described in section 3.2. The angle sequence contains detailed local information of the object shape, since it records the orientation relationships between adjacent boundary pixel pairs. On the other hand, the vector contour representation developed provides global information of the object shape. A local change of orientation affects slightly the vector contour representation of an object. Thus, the angle sequence is sensitive to local changes, while the vector representation is subjected to global deviation of the object shape. In order to take advantages of their strengths and compensates for the weakness of each form, both angle sequence and vector contour representation are used in this research for different recognition systems, and are combined to achieve good recognition results.

3.2 Vector Contour Representation (VCR)

In order to minimize the influence of local geometric changes on the object shape and to maintain invariance to the object size, the angle sequence is further processed to obtain the vector contour representation. First, smoothing procedure smoothes out the local noise, so that the sequence can more closely represent the object shape. Next, the integration procedure establishes the total variation of tracing directions. Finally, magnitude and length normalizations make the sequence independent of object size. The final form of this further processed sequence is called the vector contour representation.

3.2.1 Angle Sequence Smoothing

In the procedure of angle sequence smoothing, noise which is associated with local angle information is removed from the object contour function. The noise results from finite resolution of the pixel level representation, and from the associated coarse changes in the local angle information. For the example shown in Fig. 3.6, the angle sequence of a circle contains a number of local angles of 90^{o} and requires smoothing to obtain a closer shape approximation. The smoothing technique is achieved by applying a moving window of size h to the angle sequence. All the h consecutive angle values covered by the current window

position are averaged. The first value in the window is replaced by the averaged value. The smoothing process repeats as the window moves until all angle values are updated. Specifically, the averaged angle sequence of x(n) in equation (3.2) is given by

$$(n) = \frac{1}{h}\sum_{m=1}^{h} x(p_m), \qquad p_m = (m+n-1) \ MOD \qquad \textbf{(3.4)}$$

The way of calculating the p_m in the above equation is to retain the circularity for the smoothed sequence. To update the last (h-1) angle values in the angle sequence, the first (h-1) values need to be taken into account. For example, the last 3 values of an angle sequence of length 8 are to be updated by an averaging window of size 4 in the following manner providing the first angle value is x(0):

$$x(5) = \frac{1}{4}\,[\,x(5) + x(6) + x(7) + x(0)\,] \qquad (3.5a)$$

$$x(6) = \frac{1}{4}\,[\,x(6) + x(7) + x(0) + x(1)\,] \qquad (3.5b)$$

$$x(7) = \frac{1}{4}\,[\,x(7) + x(0) + x(1) + x(2)\,] \qquad (3.5c)$$

The larger the window size h (h < N), the smoother the angle sequence. However, if h is too large, the necessary local angle information will be lost, and the smoothed sequence can not properly represent the object shape. On the other hand, with a small h, the noise cannot be minimized.

3.2.2 Integration

Integration of the smoothed angle sequence results in the accumulated angle variation of the object contour. The angle variation observed at a point of the angle sequence indicates the cumulative angle change starting from the initial point. The

integral of the angle sequence can be carried out by summation. At any point, the integral is the sum of all the angle values from the first point to this point. Let y(n) denote the integral of the N-point smoothed angle sequence in equation (3.4). Then, y(n) is given by

$$y(n) = \sum_{m=1}^{n} \tilde{x}(m), \qquad n = 1, 2, ..., N \tag{3.6}$$

and

$$\begin{aligned} 0) &= x(0), \\ 1) &= x(0) + x(1) = y(0) + x(1), \\ 2) &= x(0) + x(1) + x(2) = y(1) + x(2), \\ &\quad . \ . \ . \ . \ . \ . \ . \ . \ . \ . \ . \ . \ . \ . \\ n) &= x(0) + x(1) + ... + x(n) = y(n-1) + x(n \end{aligned} \tag{3.7}$$

As an example, the original angle sequence, the average sequence, and the integrated sequence of a numeral 2 are shown in Fig. 3.7. The curve shown in Fig. 3.7(c) is actually the enveloping line of the integrated sequence.

3.2.3 Normalization

In order to measure similarities between objects with different sizes, their feature representations need to be normalized. It is due to the fact that different

objects may have different lengths of angle sequences, and the integral of an angle sequence may result in an angle variation larger than 2π. Therefore, the length normalization and the magnitude normalization must be applied to the angle sequences prior to their comparison. Length normalization makes all the sequences of the same length. Magnitude normalization keeps the total variation from the starting point to the end at 0, so that the circularity is preserved.

A. Magnitude Normalization

Let $y_m(n)$ denote the magnitude normalized angle sequence. The equation is given by

$$y_m(n) = y(n) \pm h \cdot \frac{n}{N}, \qquad n = 1, 2, ..., N \tag{3.8}$$

where y(n) is the integrated angle sequence in equation (3.6), h is the size of the smoothing window depicted in equation (3.4), and N is the number of elements in the angle sequence. The decision to perform either addition or subtraction in the equation (3.8) is according to the contour tracing direction. If the sequence results from tracing the object contour in a counterclockwise direction, addition is performed; otherwise, subtraction is performed.

After the magnitude normalization, the total variation after tracing back to

the initial point is zero, maintaining circular property of the angle sequence. The magnitude-normalized version of the signal in Fig. 3.7(c) is presented in Fig. 3.8(a).

B. Length Normalization

The length of a sequence is defined as the number of points (or elements) in the sequence. Length normalization is achieved by resampling the magnitude normalized sequences, so that all the sequences have the same number of elements. Resampling a sequence can be done by first finding a higher order polynomial which fits all the data points, and then resampling this curve to obtain a predefined number of points. However, obtaining a higher order polynomial to fit all the data points of the sequence results in a high computational cost and unreliable results due to roundoff errors [38]. To achieve good results of interpolations and resampling, the technique of piecewise interpolation is employed. Instead of fitting all the points of the sequence, the sequence is divided into a number of four-point intervals, and the points in each interval are approximated by a cubic Lagrange interpolating polynomial. The intervals are divided in such a way that each interval overlaps its immediate neighboring intervals by three points. Denote an interval by [n, n+3], which includes the points n, n+1, n+2, and n+3. Its preceding interval is [n-1, n+2], and its succeeding interval is [n+1, n+4].

The Lagrange interpolating polynomial $p_n(\tilde{n})$ for the interval [n, n+3] to fit the data points $y_m(n)$, $y_m(n+1)$, $y_m(n+2)$, and $y_m(n+3)$ is given by

$$y_m(n)g_0(\tilde{n}) + y_m(n+1)g_1(\tilde{n}) + y_m(n+2)g_2(\tilde{n}) + y_m(n+3$$
$$\sum_{i=0}^{3} y_m(n_i)g_i(\tilde{n}) \qquad \textbf{(3.9)}$$

where n = 1, 2, ..., N-1, n_i = (n+i) mode N, and ñ is continuous in [n, n+3]. The Lagrange fundamental polynomials of degree 3, $g_i(\tilde{n})$, are:

$$g_0(\tilde{n}) = \frac{[\tilde{n}-n][\tilde{n}-(n+1)][\tilde{n}-(n+2)][\tilde{n}-(n+3)]}{[n-(n+1)][n-(n+2)][n-(n+3)]}$$
$$= -\frac{1}{6}\sum_{j=0}^{3}[\tilde{n}-(n+j)] \qquad \textbf{(3.10)}$$

$$g_1(\tilde{n}) = \frac{[\tilde{n}-n][\tilde{n}-(n+1)][\tilde{n}-(n+2)][\tilde{n}-(n+3)]}{[(n+1)-n][(n+1)-(n+2)][(n+1)-(n+3)]}$$
$$= \frac{1}{2}\sum_{j=0}^{3}[\tilde{n}-(n+j)] \qquad \textbf{(3.11)}$$

$$g_2(\tilde{n}) = \frac{[\tilde{n}-n][\tilde{n}-(n+1)][\tilde{n}-(n+2)][\tilde{n}-(n+3)]}{[(n+2)-n][(n+2)-(n+1)][(n+2)-(n+3)]}$$
$$= -\frac{1}{2}\sum_{j=0}^{3}[\tilde{n}-(n+j)] \qquad \textbf{(3.12)}$$

$$g_3(\tilde{n}) = \frac{[\tilde{n}-n][\tilde{n}-(n+1)][\tilde{n}-(n+2)][\tilde{n}-(n+3)]}{[(n+3)-n][(n+3)-(n+1)][(n+3)-(n+2)]}$$
$$= \frac{1}{6}\sum_{j=0}^{3}[\tilde{n}-(n-j)] \qquad \textbf{(3.13)}$$

Substitute the equations (3.10), (3.11), (3.12), and (3.13) into equation (3.9), to obtain

$$= y_m(n_0)[-\frac{1}{6}\sum_{j=0}^{3}(\tilde{n}-n-j)] + y_m(n_1)[\frac{1}{2}\sum_{j=0}^{3}(\tilde{n}-n-j$$
$$y_m(n_2)[-\frac{1}{2}\sum_{j=0}^{3}(\tilde{n}-n-j)] + y_m(n_3)[\frac{1}{6}\sum_{j=0}^{3}(\tilde{n}-n-j \qquad \textbf{(3.14)}$$

$$p_n(\tilde{n}) = 6[-y_m(n_0) + 3y_m(n_1) - 3y_m(n_2) + y \qquad \textbf{(3.15)}$$

Applying equation (3.15) to all the (N-1) intervals results in a composite polynomial fitting all the data points $y_m(n)$'s from n=1 to N. The resulting polynomial is then sampled to obtain an equally spaced discrete sequence having a predefined number of points and retaining the characteristics of the sequence $y_m(n)$. Fig. 3.8(b) shows the result of the length normalization of the signal in Fig. 3.8(a). Normalizations are completed at this stage.

After smoothing, integration, and normalization, the angle sequence is

transformed to a curve which is called the vector contour representation (VCR). At this point, the contour of a pattern in the 2-D image plan is characterized by its VCR. All objects have their distinct VCRs with the same length. In training and classification, the similarity between two objects are measured by comparing their VCRs according to certain criteria.

3.3 Fourier Transform Representation (FTR)

The Fourier based methods for object contour representations are invariant to image rotation and translation [9], [12], [37]. They can also be independent of image scaling by length normalization. In this dissertation, the Fourier transform representation of an object is developed by taking the Fast Fourier transform of its angle sequence. Let x(n) be an angle sequence having N elements, the FTR of x(n) is given by

$$X(k) = \sum_{n=0}^{N-1} x(n)\ W_n^{\,k}\ , \qquad W_n = e^{-j\frac{2\pi}{N}n} \tag{3.16}$$

where k=0,1, 2, ..., N-1. In feature domain, objects are compared according to their FTRs. The Fourier transform representation is not scale invariant. If an object image is scaled M times (M>1), the resulting spectrum is M repetitions of the original FTR. Let x and x_s be the angle sequences of the original and the scaled

image, and X and X_s be their corresponding FTRs. Then the enlarged angle sequence $x_s(n_s)$, which contains MN elements, has the following property:

$$x_s(n_s) = \begin{cases} x(n), & if \quad n_s = M \cdot n \\ 0, & elsewhere. \end{cases} \tag{3.17}$$

According to (3.16), the FTR of the scaled image is given by

$$X_s(k_s) = \sum_{n_s=0}^{MN-1} x_s(n_s)\, W_{MN}^{k_s n_s} = \sum_{n_s=0}^{MN-1} x_s(n_s)\, e^{-j\frac{2\pi}{MN} k_s n_s} \tag{3.18}$$

where k_s=0, 1, 2, ..., MN-1. From (3.17) and (3.18), we obtain

$$X_s(0) = \sum_{n_s=0}^{MN-1} x_s(n_s) = \sum_{n=0}^{N-1} x(n) = X(0) \tag{3.19}$$

$$\begin{aligned} X_s(k_s) &= \sum_{n_s=0}^{(N-1)M} x_s(n_s) e^{-j\frac{2\pi}{MN} k_s n_s}, n_s = 0,\ M,\ 2M,\ \ldots,\ (N- \\ &= \sum_{n=0}^{N-1} x(n) e^{-j\frac{2\pi}{N} k n} = X(1), \quad n = \frac{n_s}{M} \end{aligned} \tag{3.20}$$

In the similar way,

$$X_s(2) = \sum_{n'=0}^{(N-1)M} x_s(n_s) e^{-j\frac{2\pi}{MN}2n'} = \sum_{n=0}^{N-1} x(n)\ e^{-j\frac{2\pi}{N}2n} = X(2 \qquad \textbf{(3.21)}$$

. . . . . .

$$;(N-1) = \sum_{n'=0}^{(N-1)M} x_s(n_s) e^{-j\frac{2\pi}{MN}(N-1)n'} = \sum_{n=0}^{N-1} x(n) e^{-j\frac{2\pi}{N}(N- \qquad \textbf{(3.22)}$$

$$= X(N-1)$$

The first (N-1) components of X_s are exactly the same as the whole spectrum of X. In the same manner we can show that the spectrum of X repeats (M-1) times:

$$\begin{aligned} N) &= X(0),\ X_s(N+1) = X(1),\ ...,\ X_s(2N-1) = X(N-1) \\ 2N) &= X(0),\ X_s(2N+1) = X(1),\ ...,\ X_s(3N-1) = X(N- \\ &\quad . \ . \ . \ . \ . \ . \ . \ . \ . \ . \ . \ . \ . \ . \\ X_s[(M-1)N] &= X(0),\ .\ .\ .,\ X_s(MN-1) = X(N-1). \end{aligned} \qquad \textbf{(3.23)}$$

As a result, the spectrum of the scaled image consists of M repetitions of the original spectrum. This is illustrated in Fig. 3.9, which shows the FTRs of the original object image and its enlarged (2 times larger) version. Note that Fig. 3.9 shows the magnitudes of the FTRs for simplicity. In the classification systems

developed, the samples are in the complex form of FTR. In the similarity evaluation of FTRs, the effect of image scaling must be considered so that the recognition is independent of the object's size. This is discussed and solved in Chapter 4. The FTRs of a numeral 3 and a numeral 5 are shown in Fig. 3.10.

CHAPTER 4

SIMILARITY MEASURES IN FEATURE SPACES

To measure how similar a group of objects is in the feature space is crucial in pattern recognition. Even when desired features are extracted, the recognition may still fail if the similarity between objects is not properly evaluated. A similarity measure must result in a great analogy between two patterns of the same type and a significant distinction between patterns from different classes [24].

The similarity measures developed are based on the distances between image patterns. Two patterns of different types should have a large distance from each other, while two patterns from the same object class should have a small distance to each other. However, there is no universal similarity measure which is adequate for all the different feature representations. Different feature representations may need different measures for similarity evaluation. The similarity measures used in this research for different feature representations are the Euclidean distance, correlation, and string matching cost.

4.1 Euclidean Distance as a Similarity Measure

Since an object presented in the image plane is transformed to a multi-

dimensional feature vector in the feature space, the Euclidean distance between two feature vectors can be used as a similarity measure. Let V_1 and V_2 be two VCRs. The Euclidean distance between them is given by

$$d_E(V_1,V_2) = \sqrt{\sum_{i=1}^{n} (v_{1i}-v_{2i})^2} \qquad \textbf{(4.1)}$$

where v_{1i} and v_{2i} are the components of V_1 and V_2, respectively, and n is the number of components in each vector. This distance is used particularly for measuring the similarity between VCRs but not for the original angle sequences, which are hereafter referred to as angle sequences. The reason is clear. The angle sequence contains detailed local information of the object shape, unlike the VCR, which provides the global information of the object. Since each value in the angle sequence represents the orientational relationship between two adjacent boundary pixels, a slight shift of the sequence may result in a great Euclidean distance to its original sequence. Therefore, in order to properly measure the similarities between angle sequences, another method is developed in section 4.3.

4.2 Correlation as a Similarity Measure

Correlation is another technique which can be used for similarity measurement since it shows how correlated a vector is to another [39]. The

correlation of two VCRs V_1 and V_2 is defined as

$$C(V_1, V_2) = \frac{\sum_{i=1}^{n} v_{1i} v_{2i}}{\sqrt{\sum_{i=1}^{n} v_{1i}^2} \sqrt{\sum_{i=1}^{n} v_{2i}^2}} = \cos\theta \qquad \textbf{(4.2)}$$

where θ is the angle between V_1 and V_2. It is clear that the value of correlation ranges from -1 to 1. The more similar the two VCRs are, the more positive the value of $C(V_1, V_2)$ is. The correlation technique actually evaluates how close the pointing directions of two vectors are in the feature space. If two patterns are similar, the pointing directions of their feature vectors must be very close, which means θ is small and the value of C is large.

4.3 Matching Cost as a Similarity Measure

The technique of string matching is exploited so that the matching cost of two angle sequences indicates the degree of analogy between them. In string matching developed in this research, the angle sequence is viewed as an ordered integer string consisting of 0, -1, and 1. The matching cost of two angle sequences is defined as the minimum degree of effort used to make one sequence exactly the same as the other.

Before describing this measure, a property that the angle sequence possesses must be first examined. Recall that in section 3.1, an angle sequence is circular. This means that the total angle change in tracing any object contour once is $\pm 2\pi$ (or ± 4 represented in angle value) no matter what size the object is. In other words, summing all the angle values in an angle sequence results in an angle value of ± 4. The sign of it depends on the tracing orientation. This property can be expressed mathematically. Let N_p and N_n be the numbers of positive and negative angle values, respectively. Then,

$$N_n = N_p + 4, \quad \sum_{n=1}^{N} x(n) = -4 \qquad \textbf{(4.3)}$$

where x(n) is the angle value of the n*th* element, and N is the total number of elements in the angle sequence. The negative sign in equation (4.3) indicates the tracing is in the counterclockwise direction. This property described by equation (4.3) can be illustrated by Fig. 4.1, in which there are two object images and their angle sequences are of different lengths. The equation (4.3) is satisfied by both sequences.

Due to the fact that two angle sequences most likely have different lengths, they are resampled before the matching can be performed. The process of resampling is done by magnifying the two sequences by different factors so that

they have the same length. Both sequences are resampled to (M×N) number of elements, where M and N are their lengths, respectively. Magnification of the angle sequence is accomplished by adding the same number of 0s between each two consecutive angle values. Let x(n) be an N-element angle sequence, where n=1, 2, 3, ..., N. Its resampled version is given by

$$x_r(m) = \begin{cases} x(n) \;\; when \;\; m = M \cdot n \\ \\ 0 \qquad otherwise \end{cases} \qquad \textbf{(4.4)}$$

where M is the magnification factor and m = 1, 2, 3, ..., (M×N). Magnifying the angle sequence is actually equivalent to enlarging its object image by putting the same number of black pixels between each pair of boundary pixels. This kind of resampling preserves the properties and characteristics of the original angle sequence. Fig. 4.2 shows the original and its resampled sequences alone with their images.

It appears that the resampling results in a huge sequence and greatly increases the number of nodes in the input layer of the neural network. However, it is not the case in the implementation. In fact, the angle sequence is not enlarged physically; only the address of each point is changed M times. In the matching algorithm, only the position of each nonzero angle value in the original sequence

is of interest. Specifically, only the addresses of +1s and -1s in the angle sequence are to be used for matching. This is further discussed in the description presented of angle sequence matching.

To match two angle sequences of the same length means to make them identical through local contour modification. The local contour modification is applied to the place where the mismatch of two angle sequence occurs. Let x_1 and x_2 be two angle sequences of the same length N, and $x_3(n)$ be their difference sequence, which is given by

$$= x_1 - x_2 = \{ x_1(n) - x_2(n) : n = 1, 2, ..., N \quad \textbf{(4.5)}$$

This is to say, that the sequence x_2 can be modified to x_1 by adding the difference sequence x_3. The condition $x_3(n) = 0$ means a match of $x_1(n)$ and $x_2(n)$ at the n*th* position. On the other hand, a nonzero element in x_3 indicates a mismatch of two angle values at that address. Either one or both of angle the sequences must be modified in order to eliminate the mismatch. However, the modification can not be done randomly. It must be done in such a way that the minimum modification cost can be achieved. To find out the extent of the modification needed, matching between positive elements and negative elements of x_3 must be obtained such that

$$\sum_{i=1}^{K/2} d_i^2$$

is minimized, where d_i is the address displacement between two elements of opposite signs in x_3, and K is the number of nonzero elements in x_3 and is given by

$$K = \sum_{n=1}^{N} |x_1(n) - x_2(n)| \qquad \textbf{(4.6)}$$

Matching of two opposite signed elements of x_3 may be either the case of matching two opposite signed elements of the same angle sequence, x_1 or x_2, or the other case of matching two elements of the same sign of two different sequences. For example, suppose that two nonzero elements $x_3(i)$ and $x_3(j)$ are to be matched, and $x_3(i) = -x_3(j) \neq 0$ ($i \neq j$). From equation (4.5), $x_3(i) = x_1(i) - x_2(i)$, and $x_3(j) = x_1(j) - x_2(j)$. This yields $x_1(i) - x_2(i) = -[x_1(j) - x_2(j)] \neq 0$, or

$$x_1(i) + x_1(j) = x_2(i) + x_2(j) \qquad \textbf{(4.7)}$$

From equation (4.7), if $x_2(i) = x_2(j) = 0$, then $x_1(i) = -x_1(j) \neq 0$. This is the case of matching two opposite signed elements of the same sequence. If $x_2(i) = x_1(j) = 0$, then $x_1(i) = x_2(j) \neq 0$, and this is the case of matching two elements of the same sign of different sequences. Any other possible combinations will fall into these two cases. Therefore, in the matching algorithm, the difference, x_3, does not need to be obtained. Instead, matching is done by matching the (1, -1) paris in each sequence

as well as matching the nonzero pairs $[x_1(i), x_2(j)]$, where $x_1(i) = x_2(j) \neq 0$ and i, j = 1, 2, ..., N. Since matching each pair results in an address displacement, the matching cost can be defined as the square of the address displacement. The total cost of matching two angle sequences is the sum of the costs of matching individual angle values. Consequently, matching two angle sequences is the task of finding out address displacement of each matching pair and obtaining the best match which minimizes the total cost. The algorithm of matching two angle sequences x_1, and x_2 is described as follows.

First, since the address displacement is greatly involved in string matching, the address of each nonzero element in both sequences is used to establish two address vectors for systematic matching. Address vector **A** consists of the subvectors $\mathbf{a}^+$ and $\mathbf{a}^-$, which contain the addresses of the positive values in x_1 and the addresses of the negative values in x_2, respectively. Vector **B** is composed of $\mathbf{b}^-$ and $\mathbf{b}^+$, which contain the addresses of the negative values of x_1 and the addresses of the positive values of x_2, respectively. By doing so, the case of matching two opposite signed elements in x_1 is equivalent to the case of matching their addresses, one of which is in $\mathbf{a}^+$, and the other is in $\mathbf{b}^-$. In a similar way, the matching of two positive element pairs $[x_1(i), x_2(j)]$ is the same as matching their addresses, which are in $\mathbf{a}^+$ and $\mathbf{b}^+$, respectively. Therefore, matching of x_1 and x_2 is actually

matching of the address vectors **A** and **B**, which are expressed as

$$A = \left[a^{+} \mid a^{-} \right], \quad B = \left[b^{+} \mid b^{-} \right]. \qquad \textbf{(4.8)}$$

The two address vectors, A and B, both have the same number of elements. To prove this statement, let N_{p1} and N_{n2} be the number of elements in $\mathbf{a}^{+}$ and the number of elements in $\mathbf{a}^{-}$, respectively. Similarly, let N_{n1} and N_{p2} be the number of elements in $\mathbf{b}^{-}$ and the number of elements in $\mathbf{b}^{+}$, respectively. Then, the number of the elements in **A** is $(N_{p1}+N_{n2})$, and the number of the elements in **B** is $(N_{n1}+N_{p2})$. From equation (4.3), we know that $N_{n1} = N_{p1} + 4$, and $N_{n2} = N_{p2} + 4$. Therefore,

$$N_{p1} + N_{n2} = (N_{n1} - 4) + (N_{p2} + 4) = N_{n1} + N_{p2} \qquad \textbf{(4.9)}$$

Equation (4.9) shows that vectors **A** and **B** are of the same dimension.

Next, the cost matrix **C** is to be found, in which each element c_{ij} is the square of the address displacement between the a_i and b_j of **A** and **B**, respectively. The dimension of **C** is $(N_{p1}+N_{n2})\times(N_{p1}+N_{n2})$. The element c_{ij}, which is given by

$$c_{ij} = (a_i - b_j)^2 \qquad \textbf{(4.10)}$$

is the cost of matching a_i with b_i. If x_1 and x_2 are identical, the cost matrix is

symmetric and the elements on the diagonal are all 0s since there is no mismatch and $c_{ij} = c_{ji}$, and $c_{ii} = 0$. On the other hand, the cost matrix for matching two different angle sequences is not symmetric, and not all c_{ii} are zero. With the individual matching costs provided by **C**, the best matching assignments are to be found in order to minimize the total cost T, which is given by

$$T = \sum_{i=1}^{L} C(i) \ , \qquad L = N_{p1} + N_{n2} \tag{4.11}$$

where C(i)s are the selected cost from each row so that T is minimum. Normalization can be applied to the minimum matching cost, T, so that it ranges from 0 to 1. The value 0 indicates two angle sequences are exactly the same while the value 1 indicates they are totally different. For an M-element angle sequence and an N-element angle sequence, the normalized matching cost is given by

$$T_N = \frac{T}{\left(\frac{MN}{2} \right)^2} \tag{4.12}$$

where (MN/2) is the maximum possible address displacement. The maximum possible address displacement in a sequence is its half length since the distance between two angle values is computed circularly.

The matching algorithm can be illustrated by the following example. Suppose there are two angle sequences

address (1) (2) (3) (4) (5) (6) (7) (8)

x_1 = [-1 0 -1 0 -1 0 -1 0], and

x_2 = [-1 1 -1 -1 0 -1 0 -1]

whose images are shown in Fig. 4.3. The address vectors for matching are **A**=[1 3 4 6 8], and **B**=[1 3 5 7 2]. The cost matrix is given by

$$\boldsymbol{C} = \begin{bmatrix} 0 & 4 & 16 & 4 & 1 \\ 4 & 0 & 4 & 16 & 1 \\ 9 & 1 & 1 & 9 & 4 \\ 9 & 9 & 1 & 1 & 16 \\ 1 & 9 & 9 & 1 & 4 \end{bmatrix} \qquad \textbf{(4.13)}$$

The minimum matching cost can be obtained by the following matching assignments:

(1 5), (2 2), (3 3), (4 4), (5 1).

These matching assignments correspond to matching the following pairs from angle sequences x_1 and x_2:

[x_2(1) x_2(2)], [x_2(3) x_1(3)], [x_2(4) x_1(5)], [x_2(6) x_1(7)], [x_2(8) x_1(1)]

or graphically

The minimum matching cost in this case is calculated as follows:

$$\begin{aligned} T &= c_{15} + c_{22} + c_{33} + c_{44} + c_{51} \\ &= 1 + 0 + 1 + 1 + 1 \\ &= 4 \end{aligned} \tag{4.14}$$

and from equation (4.12), the normalized cost is

$$T_N = \frac{4}{\left(\frac{8}{2}\right)^2} = \frac{1}{4} \tag{4.15}$$

The routine, written in Fortran code, to obtain the best assignments in the matrix **C** is developed in [40].

4.3.1 An Application of String Matching

The string matching technique developed above can be applied for various

recognition tasks based on the object shape. This section presents an example of recognizing puzzles which are quite similar to each other. In Fig. 4.4(a), an area is puzzled into eight regions, which are shown in Fig. 4.4(b). All of the puzzles' shapes are different but quite similar to each other, especially if they are arranged in the same orientation, as shown in Fig. 4.4(c). If all the parts are scattered around, it is not easy for a human to put them back to their correct positions in a short time. However, the recognition system using the developed string matching technique can do the job without any difficulty. In Fig. 4.5, the original positions of these puzzles are labelled with the letters from A to H, and all the puzzles are randomly labelled with numbers. The original regions and their corresponding puzzles are

(A, 3) (B, 5) (C, 2) (D, 1) (E, 7) (F, 4) (G, 8) (H, 6).

The system has no knowledge about the above original pairs in advance. In recognition, the system puts all the puzzles back to their original places correctly. This is demonstrated by the cost matrix and the matching assignments, which are shown in Table 4.1. Each element of the cost matrix, shown in Table 4.1(a), is the cost for matching a region with a puzzle. Reordering the columns of the cost matrix results in a symmetric matrix shown in Table 4.1(b). The values of all the elements on the diagonal are zero, each of which represents a perfect match between the region and the puzzle. Comparing the matching assignments in Table 4.1(c) with

the original pairs shows the matching results are totally correct.

4.4 Fourier Based Similarity Measure

FTRs of different object images cannot be directly compared since they may not have the same number of components. As described in Section 3.3, scaling an object image by n (n>1) is equivalent to repeating n times of its spectrum in frequency domain. Directly comparing their FTRs will result in a big difference. In this research, the similarity between two images is measured by evaluating the differences between the lower harmonics, instead of all harmonics, of their FTRs. This is due to the fact that the higher harmonics introduce higher noise, and including all the harmonics for comparison results in great inaccuracy. Two object images differing only by one pixel result in a small local difference between their angle sequences. However, this small local difference contributes to the whole spectrum in frequency domain, and is overly emphasized in high frequencies. This case is illustrated in Fig. 4.6, in which two similar objects have great differences in the high frequencies of their FTRs. The lower order harmonics show the similarity between two FTRs.

The distance, called error norm, developed to measure the similarity between two FTRs X_i and X_j having N_i and N_j elements, respectively, is given by

$$d_F(X_i,\ X_j)\ =\ \sqrt{\sum_{k=1}^{L} \|\ E_{i\ j}(k)\ \|^2} \qquad \textbf{(4.16)}$$

E_{ij} is the weighted error signal defined as

$$E_{i\ j}\ =\ \left\{ \frac{X_i(k)\ -\ X_j(k)}{k^{\alpha}} \right\},\quad k\ =\ 1,\ 2,\ ...,\ L \qquad \textbf{(4.17)}$$

where k^{α} is the weight, and α controls the weighing degree. The error signal is linearly weighted if $\alpha=1$, while it is nonlinearly weighted if $\alpha>1$. Weighing the error signal is needed to reduce the error introduced by the higher order harmonics. The number of harmonic components in each FTR to be compared is determined by

$$L\ =\ \frac{1}{\beta}\ \min\ (N_i,\ N_j)\ ,\quad \beta\ >\ 1. \qquad \textbf{(4.18)}$$

Only the first L harmonics of the Fourier spectrum are used for similarity measure. Fig. 4.7 shows the error rate from recognition of 700 handwritten characters under different values of α and β. The minimum error occurs when $\alpha \approx 1.5$ and β is 7 or 8. These optimized parameters were used to evaluate the error norm in the learning method on FTR as described in section 7.3.1.D.

In equations (4.16) and (4.17), the dc error (when k=0) is not taken into

account since it is always zero. Since from equation (4.3), for an N-element angle sequence, the sum of all the element is -4, then

$$X_i(0) = \sum_{m=0}^{N_{\alpha}-1} x_i(m) W_m^{\,0} = \sum_{m=0}^{N_{\alpha}-1} x_i(m) = -4 \qquad \textbf{(4.19)}$$

and

$$X_j(0) = \sum_{n=0}^{N_{\varpi}-1} x_j(n) W_n^{\,0} = \sum_{n=0}^{N_{\varpi}-1} x_j(n) = -4. \qquad \textbf{(4.20)}$$

which results in

$$X_i(0) - X_j(0) = 0 \qquad \textbf{(4.21)}$$

Similarity measurement for different kinds of object feature representations usually needs different methods. In this chapter, four different similarity measures are described. The Euclidean distance and the correlation technique, the matching cost, and the error norm are the similarity measures for VCRs, angle sequences, and FTRs, respectively. In the next chapter, various learning processes using different similarity measures are described.

CHAPTER 5

LEARNING TO CLASSIFY

5.1 Learning through Supervised Clustering

Learning to classify is a process of finding out how to distinguish patterns of various types based on a finite number of learning samples under the direction of the learning rules. The classification knowledge based on the learning samples continues to be accumulated until the learning effect of the classifier is satisfied, according to the learning rules. Different learning algorithms using clustering techniques have been proposed [41]-[47]. Conventional clustering techniques employ unsupervised schemes which group similar data into clusters. Based on the concept of unsupervised clustering, self-organized neural networks are developed. However, unsupervised learning suffers from the divergence problem and incorrect class assignments during clustering. On the hand, supervised clustering always converges and achieves proper data partition.

In this research, learning through supervised clustering [45]-[47] is developed. Each learning sample is provided with its state of nature (or class label). The mechanism of the supervised learning is shown in Fig. 5.1. During learning, the intermediate results are checked to see if the classifier is responding correctly.

The learning process terminates when the classifier is able to accurately classify all the learning samples. The learning effect of the classifier depends on the learning rule, which supervises the way of learning and determines the condition of termination.

Clustering is a process which groups similar patterns of the same type together into clusters, according to the similarity measures used, and represents each group of patterns with its cluster center. Due to the fact that the shapes of patterns in a class may deviate from each other quite a bit, patterns of the same type are grouped more likely into several clusters rather than into a single cluster. Thus, the clustering results in a small number of clusters which represent a vast amount of learning samples. In classification, a test sample is compared with all the cluster centers and identified with the most similar one. The clusters implemented in hardware are actually neurons of the neural network. An image received by the input layer is first transformed to its feature representation by the second layer and is sent to the next layer which contains all the neurons. Different neurons result in various responses to the input image, and these responses are fed to the following layers for classification.

Therefore, in clustering based learning, the learning rules results in a small number of clusters, which can best represent objects of different types. The pattern

classifier can then be realized in a simple hardware structure.

5.2 Maximum Region Clustering (MRC)

The maximum region clustering developed in this dissertation maximizes the size of each cluster with the constraint from other sample classes. Specifically, maximum region clustering tries to include as many samples of the same type as possible in a cluster without enclosing any alien sample. With the knowledge of class identification of each learning sample, the MRC technique groups the maximum number of the similar samples of the same type in a neighborhood of a cluster center. This is achieved by extending the radius of the cluster until an alien sample is encountered.

In learning, a sample is said to be active if it is neither a cluster center nor enclosed in any cluster. Inactive samples are removed from the rest of the learning process. As shown in Fig. 5.2, the clustering algorithm begins with randomly selecting an active learning sample as a cluster center. The radius of the cluster is equal to the distance to its nearest alien sample. All the active samples, whose distances to the cluster center are smaller than the radius, are assigned to this cluster and are deactivated. Once a cluster is established, a redundancy checking is applied to determine if there is any redundant cluster. If there is a cluster whose center is

enclosed in the current cluster, and whose radius is smaller, then this cluster is redundant and is eliminated. Samples from the redundant cluster are reassigned to the enclosing cluster if they are also enclosed in that cluster region. Samples which are not covered by the replacing cluster are reactivated and used in the next learning cycle. Fig. 5.3 visualizes the clustering algorithm in a two dimensional case. As stated before, the radius of a cluster is the distance to its nearest alien sample. For class j, the center of cluster 2 is inside the cluster 1, and $r < R$. Therefore, cluster 2 is removed, and the samples covered by both clusters are reassigned to cluster 1. The samples which are in cluster 2 but are not covered by cluster 1 are retained for the subsequent learning cycle. The learning cycle repeats until all the learning samples become inactive. The learning algorithm is summarized as follows.

1. Initialization.

   (a) set N the number of learning samples

   (b) set the number of clusters, $i = 0$.

   (c) set class index, $K = 1$ (the first object class to learn.)

2. Activate all the samples in class K.

3. Increment i, $i = i + 1$.

   Select an active sample in class K as a cluster center C_i, and

   deactivate this sample.

Find the distances to all the learning samples,

$$d(C_i, S_j),\ j = 1, 2, 3, ..., N.$$

4. Find the radius, R_i, of the cluster i.

$$R_i = \min \{ d(C_i, S_j) \},\ S_j \notin \text{class K}.$$

5. In class K, deactivate the samples S_j which satisfy the condition

$$d(C_i, S_j) < R_i,\ S_j \in \text{class K}.$$

6. Check for redundant clusters. Remove cluster m, if

$$d(C_i, C_m) < R_i, \text{ and } R_m < R_i.$$

7. Reactivate the samples previously in cluster m but not covered by the current cluster i.

8. If all samples in class K are deactivated, set K = K + 1, and

   go to step 9;

   else, go to step 3.

9. If $K \leq N$, go to step 2; else, terminate the learning process.

5.3 Accumulated Potential Clustering (APC)

The other supervised clustering technique developed in this research is accumulated potential clustering [46]. In APC, a cluster center is a potential generating center, which generates potential to all the samples not only of the same

type but also of other object types. Since an object class contains a number of cluster centers, the potential generated by a class means the sum of potentials generated by all the cluster centers in that class. During APC learning, all the samples accumulate different amount of potential generated by all the object classes. Potentials generated by different classes to a sample are not to be combined so that they can indicate how strongly the sample is affected by each class. After learning, the highest potential that the sample accumulates should come from its own class. Any accumulated potential generated by other classes should be smaller than that generated by the sample's class. In classification, a sample is classified to the class from which it receives the largest amount of accumulated potential.

5.3.1 Potential Function

The potential function of the feature space can be viewed as the potential of the electrostatic field [24]. A certain kind of sample in the feature space is analog to an electrical charge, located at the same position, which generates an electrostatic potential. In this algorithm, only the samples which are selected to be cluster centers carry charges and generate potential to all other samples. The ordinary samples, which are not the cluster centers, only receive the potential generated by the cluster centers. The concept of this technique is illustrated in Fig. 5.4, in which the peaks are the cluster centers which generate potentials to all the samples.

Let $p(S,C_i)$ denote the potential function which is generated by the center C_i to a sample S. The total potential the sample S receives from a class K, which has m cluster centers, is given by

$$P_K(S) = \sum_{i=1}^{m} p(S,\ C_i) \qquad \textbf{(5.1)}$$

The potential function $p(S,C_i)$ is inversely proportional to the distance between S and C_i. The farther away a sample is from a cluster center, the less potential it receives from this center. It is clear that the total potential, which S accumulates from class K, depends not only on the distances to the cluster centers but also on the number of cluster centers. Therefore, to protect its own samples from being attracted to other classes, the each class should establish enough cluster centers, while keeping the number of the cluster centers from being too large by selecting them at proper locations. This can be accomplished by employing good learning criterion and a learning algorithm.

5.3.2 Learning Criterion

In supervised learning, each learning sample is to be correctly classified

according to the learning criterion. This is done by first finding a sample's nearest class and then checking to see if it is the class where the sample belongs. If it is the case, the sample is correctly classified; otherwise, it is not correctly classified and is to be reused for subsequent learning. A class is said to be the nearest class to a sample if it produces the highest total potential to that sample, which is from

$$P_{\max}(S) = \max \{ P_K(S) \}, \quad K = 1, 2, ..., N \qquad \textbf{(5.2)}$$

where N is the number of object classes in the feature space. Therefore, the criterion of correctly distinguishing a learning sample is based on the potential that the sample collects from each object class. Once the highest potential comes from its own class, the sample is said to be correctly classified by the system.

However, it is possible for a sample that the highest potential is only slightly larger than the second highest potential. If the difference between the potential from the sample's nearest class and that from its next nearest class is insignificant, then the two classes are competing for this sample with an almost equal amount of energy, and there is no clarity about the sample's type. This means that the confidence to make a classification decision is very weak, and a test sample may not be correctly recognized. To strengthen the decision making confidence, a modification is made. Let $P_{next}(S)$ be the potential generated to the sample S by its

next closest class. The discriminant function, which can be used for learning criterion, is given by

$$D(S) = 1 - \frac{P_{next}(S)}{P_{\max}(S)} \qquad \textbf{(5.3)}$$

The value of D(S) ranges from 0 to 1, and it can be viewed as the confidence index. The larger the value, the higher the confidence. In order to obtain enough confidence, a threshold is set up during the supervised learning process. A sample is said to be properly classified only if the class of S is nearest to it, and if D(S) is larger than the threshold value. Let ω(S) be the class label of sample S, ϖ(S) the nearest class to S, and T the learning threshold. The learning criterion is as follows:

If ω(S) = ϖ(S) and D(S) > T, then S is correctly classified;

otherwise, more learning is required.

5.3.3 Learning Algorithm

The flow chart of this algorithm is shown in Fig. 5.5. A sample is said to be inactive if it is correctly distinguished from its alien classes or if it is a cluster center; otherwise, it is active. All the active learning samples are continuously used

in the learning process until they become inactive. Some of the active samples are selected as cluster centers to generate potential for their classes against the potential from the alien classes. Redundant clusters are detected and deleted during learning to keep the number of clusters at a minimum. The APC learning algorithm is described as follows.

1. Initialization:
   (1) Set a threshold value T;
   (2) Activate all samples;
   (3) Set all samples as non-cluster centers;
   (4) For each sample, set the potential from each class to zero.
2. If no more active samples, done;
   otherwise,
   (1) select an active sample S_i of class K, and set S_i to be a cluster center;
   (2) deactivate S_i since it is a cluster center;
3. Adding new cluster centers:
   (1) Generate potential from the cluster center S_i to all other samples.
   (2) For each sample S, add this potential to $P_K(S)$.

4. Check each sample S, except S_i, for potential change due to S_i:

   (1) find its two closest classes;

   (2) if the class of S is the same as its closest class and

      if D(S) > T,

         deactivate S and set it as a noncluster-center sample;

      else if S is not a cluster center,

         activate it for the next learning cycle;

      else deactivate it.

5. Check for redundant cluster centers:

   If any cluster center is to be removed, go to step 6;

   otherwise, go to step 2.

6. If the cluster center S_i of class K is to be removed:

   (1) Remove S_i;

      For each sample S, subtract the potential generated by S_i

      from $P_K(S)$.

   (2) Repeat step 4 (check all samples due to potential change

         by S_i).

   (3) Go to (1) if more cluster centers are to be removed;

      otherwise, go to step 2.

When the learning process is completed, each class will have a minimum number of cluster centers to represent it. These cluster centers are implemented as the neurons in the hardware. They should give strong response (high potential) to the similar testing samples and weak responses to other samples.

In order to visualize the effect of APC learning, the process is implemented in the 2-D case. The simulation is performed on three examples of different sample distributions on the two-dimensional feature plane. All the cases are linearly nonseparable. In the first example, which is shown in Fig. 5.6, there are two classes of samples, each of which contains 14 samples. The learning result is shown in Fig. 5.6(b). The brightest points are the cluster centers which generate potential to all the samples. The brighter the area is, the greater the amount of potential which the samples in that area receive. The darkest region is the boundary region in which D(S)'s of all the samples are equal to 0. This means that the samples in that region receive an equal amount of potential from the cluster centers of two dominating classes. Any samples falling in the boundary region cannot be classified to a unique class and will be rejected in the classification phase. The learning of this case results in 7 cluster centers.

Shown in Fig. 5.7(a) are the samples formed by adding another class of 14

samples to the plane from example 1. This is done to see how the presence of another class of samples affects the learning result. The result, which is shown in Fig. 5.7(b), reveals that insertion of the extra class changes the boundary shapes of the two previous classes. This results from the change of the potential distribution over the whole plane due to the introduction of the potential from the third class. Ten cluster centers are formed in this case, and locations of previously established cluster centers are different than in example 1.

In the first two examples, the density of sample distribution of one class is not much different from that of the other. In the third example, shown in Fig. 5.8, some classes have densely located samples while others do not. The samples of each class in this case are not well separated from one another. Each class contains 50 samples, and the number of cluster centers resulting from learning is 36.

The results of the above examples were obtained by performing APC learning with no threshold. The boundary region can be broader if a threshold value is set up during the learning, which means that a sample S is removed from learning only when D(S) is larger than threshold. The broader the boundary region, the more specific the learning which results in a more accurate but less flexible classification. Fig. 5.9 shows the results obtained by using a threshold value of 0.1 for the APC learning.

CHAPTER 6

PATTERN CLASSIFICATION

Pattern classification is based on the knowledge obtained from the learning process and on the classification criteria. The classification criteria are similar to those used in the supervised learning. The classifier will either make a classification decision on a testing sample or leave it undetermined. The classification performance of a pattern classifier relies greatly on the learning processes. Improvement can be made on the testing results of individual classifiers if a multiclassifier system is used, which organizes multiple classifiers in a certain way. The interconnections between classifiers in the multiclassifier system are built such that the individual classifiers' strengths are preserved, and their weaknesses are compensated. Both single classifier systems and multiclassifier systems are developed and described in this chapter, and their performances are discussed in Chapter 7.

6.1 Maximum Region Clustering (MRC) Classifier

The block diagram of this classifier is shown in Fig. 6.1. Binary images are input to the feature extraction stage to obtain their feature representation forms. The

next stage measures similarities between the input samples and cluster centers, and the final stage makes the classification decisions.

The classification rule is based on that of the learning process which employs the MRC technique. Recall that in MRC learning, every cluster contains the maximum number of learning samples that it can cover without including any alien sample. In classification, the distances between a test sample and all the cluster centers are computed. Classification decision can be made based on the nearest cluster center rule. According to this rule, the sample is categorized to the object type of its nearest cluster center. However, there are two disadvantages in using the nearest cluster center rule. First, consider the case that a test sample falls in the region that is outside all the clusters as shown in Fig. 6.2. This region is known as the region of decision boundary. In this case, the sample may be equally similar to the samples in various clusters which are close to it, and it is risky to say that the sample is most similar to its nearest cluster center. On the image plane, this may be the case where a numeral 2 with a long and curly tail is similar both to a numeral 2 and to a 3 as illustrated in Fig. 6.3.

The other disadvantage is when a test sample falls in a large cluster, and is outside a small cluster as shown in Fig. 6.4. The test sample is closer to the center of the small cluster than to the center of the large one. Apparently, the sample is

more similar to the samples in the large cluster than to those in the small one, since it is encompassed by the samples in the large cluster. Therefore, it is improper to identify the sample, according to the nearest cluster center rule, in case of a small cluster competing with a large one.

The above problems can be eliminated by using nearest cluster rule, instead of the nearest cluster center rule. According to nearest cluster rule, a sample is said to be most similar to the samples in the cluster which is nearest to it in the relative distance (or the normalized distance). The relative distance is obtained by normalizing the absolute distance by the radius of the cluster. Let $d(S,C_i)$ be the absolute distance from the sample S to the cluster center C_i, and let r_i be the radius of the cluster i. The normalized distance is $d(S,C_i)/r_i$. The shortest distance to a sample S, among the distances from all the cluster center C_i's, is given by

$$d_{\min}(S) = \min \{ d(S,C_i) \}, \quad i = 1, 2, ..., N \qquad \textbf{(6.1)}$$

where N is the number of clusters, and $d(S,C_i)$ is a similarity measure (Euclidean distance, the string matching cost, or Fourier-based similarity measure). Therefore, the cluster whose center is $d_{min}(S)$ away from S is the nearest cluster to S. In Fig. 6.4, the nearest cluster to S is the one which encloses it, which is cluster C_2, since

$$\frac{d(S,C_2)}{r_2} < \frac{d(S,C_1)}{r_1} \qquad \textbf{(6.2)}$$

It is obvious that the sample is in the region of decision boundary if $d_{min}(S)$ is larger than 1.

The classification rule then can be described as follows:

If $d_{min}(S) < 1$, then $S \in \omega_n$;

otherwise, S is rejected (or left undetermined)

where ω_n denotes the object class of the nearest cluster center. A certain threshold can also be set up to substitute for 1. Using a threshold value which is smaller than 1 results in more accurate classification results, since it rejects classifying the samples which are in a cluster but are far away from the center. This results in the rejection of a large number of samples during classification. On the other hand, applying a threshold value which is larger than 1 results in a small number of rejected samples, but the error rate may increase due to the classification of samples from the decision boundary.

The classification algorithm for MRC classifier is stated as follows.

1. Input a test sample S.

2. Compute the normalized distances from S to all the cluster centers

$$\frac{d(S,C_i)}{r_i}, \qquad i = 1, 2, ..., N.$$

C_i which are established after learning. That is, compute

3. Find

$$d_{min}(S) = \min \{ d(S, C_i)/r_i \}, \quad i = 1, 2, ..., N.$$

4. Make the classification decision.

If $d_{min}(S) < 1$, then $S \in \omega_n$;

otherwise, S is left undetermined.

6.2 Accumulated Potential Clustering (APC) Classifier

The classification rule of the APC classifier is quite different from that of MRC, just as their learning rules are different. In MRC learning, samples are removed from learning once they are enclosed by a proper cluster. In APC learning, a sample is removed from learning if it receives enough potential from its own class to keep it from being attracted to other classes. Due to the fact that a cluster center is a potential generating center, there is no explicit cluster region for each cluster center. The potential that an object class generates is actually the sum of potentials generated by all the cluster centers in that class. Exploiting the similar rule for the

classification, the APC classifier makes decisions based on the different amount of potentials that samples accumulate from all the object classes. Having found the largest and the second largest amount of potentials, a sample is classified according to its discriminant function, which is

$$D(S) = 1 - \frac{P(S)_{next}}{P(S)_{\max}} \qquad \textbf{(6.3)}$$

If D(S) is larger than a classification threshold, S is classified to the class from which it receives the maximum potential. Otherwise, S is left undetermined. The algorithm for the APC classifier is formulated as follows.

1. Input a test sample.
2. Compute the total potential received from each class ω_K:

$$P_K(S) = \sum_{i=1}^{m_K} p(S, C_i), \qquad K = 1, 2, ..., N.$$

   where m_K is the number of cluster centers in ω_K,

   $p(S, C_i)$is the potential generated by the cluster center C_i, and

   N is the number of classes.

3. Among the N potentials, find the maximum and the next maximum potentials, $P_{max}(S)$ and $P_{next}(S)$.

Identify the object class, ω_{max} which generates $P_{max}(S)$.

4. Compute the discriminant function,

$$D(S) = 1 - \frac{P(S)_{next}}{P(S)_{\max}}$$

5. Make the classification decision.

If D(S) > T, then S $\in \omega_{max}$

otherwise, S is left undetermined.

6.3 Multiclassifier Systems

The purpose of multiclassifier systems is to utilize the strengths of classifiers, while bypassing their weaknesses, so that the recognition rate can be maximized. In this research, we describe four different types of multiclassifier systems: cascaded, vote-to-decide, decision enhancement, and hierarchical learning systems.

6.3.1 Cascaded Multiclassifier System

The structure of the cascaded system is shown in Fig. 6.5. All the test samples are input to the classifier at the highest level. Each level contains only one

classifier which produces two kinds of outputs, namely, the classified and the rejected samples. Except for the top level, the inputs of the classifier at each level are the samples which were rejected by the preceding classifier.

In this system, the classification threshold of a classifier is always higher than that of the classifier at the next level. This means that the classifiers at higher levels have less classification flexibility to the samples which are less similar to their prototypes. In this way, the error rate at each level can be kept low, and a low misclassification rate of the whole system can be achieved.

6.3.2 Vote-to-Decide Multiclassifier System

In this system, shown in Fig. 6.6, classification decisions are made in accordance with the "majority opinions". Responding to a test sample, each classifier produces its own classification decision, which in this research is called the opinion. At this stage, no rejection is made, which means there is no classification threshold set up for any classifier. Any test sample, which is input to this system, will receive opinions from all the classifiers before it goes to the last stage for final decision. The sample will be categorized to an object class if the majority of the opinions are coincident, otherwise it will be rejected. The majority is defined as the number which is larger than a half. A sample is assigned to the

class to which the majority of the classifiers classify. The classification algorithm of this system can be organized as follows.

Let o_i be the classification opinion that classifier i gives for a sample S, and let Ω be the set of class labels. In the classification of handwritten numerals, Ω = { ω | ω is an integer, and $0 \le \omega \le 9$.} The opinion o_i is actually the object class to which S is suggested to belong, and $o_i \in \Omega$. For example, $o_i = 6$ means that sample S should be recognized as a numeral 6, in the opinion of classifier i. The input to the voting layer, then, is an opinion vector given by

$$O = \left[o_1,\ o_2,\ ...,\ o_M\right]^T \qquad \textbf{(6.4)}$$

where M is the number of classifiers in the system. The classification decision to sample S is made according to the following rule:

If K number of opinions, o_i's, in **O** agree to assign S to

class ω and K > M/2, then

S ∈ ω,

otherwise,

S is rejected.

In the realization of the system, various deviations from the classification criterion are also used and compared. The descriptions of the deviation and the results are

presented in the next chapter.

6.3.3 Confidence Enhancement Multiclassifier System

The structure of this multiclassifier system is organized as the input layer, the preclassification layer, the confidence enhancement layer, and the classification layer. As indicated in the Fig. 6.7, the input layer transforms the input image to its feature representation, and then sends it to the preclassification layer, which contains a number of classifiers. Instead of making classification decisions, each classifier generates a vector which contains the indices of decision confidences to all classes for the test sample. A sample's confidence index to a class indicates the degree of confidence to categorize the test sample to that class. The larger the index value, the stronger the classification confidence. The index vector generated by the classifier L can be expressed as

$$\Lambda_L = \left[\lambda_{L1},\ \lambda_{L2},\ \lambda_{L3},\ \ldots,\ \lambda_{LN}\right]^T \qquad \textbf{(6.5)}$$

The index vectors generated by all the classifiers are then output to the confidence enhancement layer, which sums up the indices in each dimension to produce a vector of combined indices:

$$\Lambda = \left[\lambda_1, \lambda_2, \lambda_3, \ldots, \lambda_4\right]^T,$$

$$= \sum_{L=1}^{M} w_L \lambda_{L1}, \quad \lambda_2 = \sum_{L=1}^{M} w_L \lambda_{L2}, \ \ldots, \ \lambda_N = \sum_{L=1}^{M} w_L \lambda \qquad \textbf{(6.6)}$$

where w_L is the weight for classifier L. Let λ_{max}, and λ_{min} be the maximum and the second maximum values among all the components of Λ. The decision function is then given by

$$D(S) = 1 - \frac{\lambda_{min}}{\lambda_{max}} \qquad \textbf{(6.7)}$$

The classification decision made by the output layer is in accordance with the following rule.

If D(S) is larger than the threshold, then $S \in \omega_i$;

otherwise, S is rejected.

This system actually amplifies the differences between indices so that the classification decision can be more precise. For example, if the difference between the highest index and the second highest index produced by a classifier is small, the decision which categorizes a sample to the class with the highest index is weak. By combining the values of indices generated by other classifiers, their differences may become bigger or smaller. In the former case, the classification decision will be made with larger confidence. In the latter case, the sample needs to be rejected, since the combined values of the indices show that it is unsafe to make a classification.

6.3.4 Hierarchical Learning Multiclassifier System

This is a compound system, which contains two classifier subsystems, the initial learning and further learning systems. Each system consists of its own learning and classification stages. However, the classification stage of the initial learning system does not make final classification decisions. Instead, it generates

other feature representations of images to be used as the inputs to the further learning system. Fig. 6.8 shows the system in the learning phase and the testing phase. In the learning phase, each classifier in the initial learning system produces a confidence index vector Λ_L, as the one in equation (6.5). All the Λ vectors from different classifiers are appended together, in order to form a hybrid vector Λ^H with M×N elements,

$$\Lambda^H = \left[\Lambda_1,\ \Lambda_2,\ \Lambda_3,\ ...,\ \Lambda_M\right]^T \qquad \textbf{(6.8)}$$

where M is the number of classifiers, and N is the number of classes. Each input image is represented by its Λ^H, generated by this subsystem. The Λ^H of an input image is taken as the second form of feature representation, which is to be sent to the further learning subsystem as an input.

The further learning subsystem can be either a single classifier or a multiclassifier system which takes Λ^H's as inputs for its learning stage to generate the learning results. Once the learning in this stage is completed, the whole system is ready for classification. As shown in Fig. 6.8(b), a test image goes through the first learned subsystem to obtain its final form of feature representation. The second learned subsystem then takes the feature representation for classification to determine which object class the input image belongs to. Since the second

subsystem can be any of the systems we developed above, the classification rule is the same as the one which is employed for the chosen system.

Having had the knowledge of distinguishing various object classes after learning, and having been equipped with the classification rules, the classifiers developed are ready to perform the classification on different objects. The next chapter describes computer simulation of different classifier systems, and presents the results of classification for handwritten character recognition.

CHAPTER 7

COMPUTER SIMULATION RESULTS

7.1 Computer Simulation Environment and Data Base

The neural network based handwritten recognition systems developed in this dissertation are simulated on a Sun Sparc 2 computer in order to evaluate their learning results and the classification performances. The feature extraction, learning, and testing processes of each system are implemented in C programs to run on the computer. The image samples for learning and classification used for the computer simulation are handwritten numerals from the data base established by the National Institute of Standards and Technology (NIST) in Washington, D. C. These handwritten numerals are obtained by collecting a number of survey sheets which are filled out by different people. Images of the survey sheets are then stored in a computer for an image processing to generate the binary images of the handwritten numerals. All the binary images of the handwritten numbers are in the 20×20 matrix from.

7.2 Simulation Results of Feature Extraction

The stage of feature extraction transforms the binary raw images to their feature representations. In other words, an image in the 20×20 matrix form whose elements are either 0 or 1 is transformed by the feature extraction stage into a vector of its feature representation.

In this research, three types of object feature representations are developed: the angle sequence, vector contour representation (VCR), and Fourier transform representation (FTR). The simulation results of this stage are to evaluate the efficiency of transformation. As shown in Table 7.1, the average simulation time consumed by the extraction of a single handwritten digit to form the angle sequence, VCR, and FTR is 0.028, 0.083, and 0.683 seconds, respectively. The time used to extract 700 samples is also indicated in Table 7.1. As expected, to form FTR takes the longest time due to the large number of computation steps needed for Fourier transformation.

7.3 Learning Results of Different Single Classifier Systems

7.3.1 MRC Learning

In MRC, the similarity measures used are the Euclidean distance, the vector correlation, the cost of string matching, and Fourier-based method. The learning results using these four different similarity measures are presented in Table 7.2.

A. Euclidean Distance as Similarity Measure

As described in section 4.1.1, the Euclidean distance between two VCRs V_1 and V_2 is given by

$$d_E(V_1,V_2) = \sqrt{\sum_{i=1}^{n} (v_{1i}-v_{2i})^2} \qquad \textbf{(7.1)}$$

where v_{1i} and v_{2i} are the components of V_1 and V_2, respectively, and n is the number of components in each vector. The number of learning samples used for the MRC learning is 700. As shown in Table 7.2, the learning results in 165 clusters which represent 10 object classes, digits 0 to 9. This number is 23% of the total number of the learning samples. Also, the number of clusters that each class has is also shown. The time taken for the learning is 2.2 minutes.

B. Correlation as Similarity Measure

$$C(V_1, V_2) = \frac{\sum_{i=1}^{n} v_{1i}v_{2i}}{\sqrt{\sum_{i=1}^{n} v_{1i}^2}\sqrt{\sum_{i=1}^{n} v_{2i}^2}} = \cos\theta \qquad \textbf{(7.2)}$$

The correlation of two VCRs V_1 and V_2 is defined as

where θ is the angle between V_1 and V_2. Results of MRC learning using correlation as similarity measure for VCRs are indicated in Table 2. Again, 700 samples are applied to the learning process which results in 182 clusters (26% of all samples) and takes 1.16 minutes of CPU time. We may conclude that both Euclidean distance and vector correlation produce similar results during learning and require similar simulation time.

C. Cost of String Matching as Similarity Measure

String matching is particularly useful to evaluate the degree of similarity between angle sequences. As stated in Chapter 4, the matching cost of two angle sequences is defined as the minimum degree of effort used to make one sequence exactly the same as the other. The detailed description of string matching is made in section 4.3. As shown in Table 7.2, MRC learning employing this technique as the similarity measure results in 143 clusters formed to represent 700 different learning samples from 10 classes. The ratio of the number of clusters to that of learning samples is 20.43%. The learning time is 19.18 minutes. This result indicates that matching cost gives slightly better results than Euclidean distance and vector correlation; however, learning time is significantly longer.

D. Fourier Based Similarity Measure

Similarity between two FTRs X_i and X_j, whose lengths are N_i and N_j, respectively, is evaluated by

$$(X_i, X_j) = \sqrt{\sum_{k=1}^{L} \left(\frac{X_i(k) - X_j(k)}{k^{\alpha}} \right)^2}, \quad L = \frac{1}{\beta}\min(N_i, \qquad \textbf{(7.3)}$$

In learning and testing, the values of α and β are set to 1.5 and 7 to obtain the best learning and classification results. MRC learning with 700 learning samples in FTR form generates 136 clusters and takes 1.23 minutes. Learning in this case is better than in other cases. The number of clusters in this case is smaller than in any other case, and the time required for learning is also little.

7.3.2 APC Learning

In APC learning, the similarity measurement is not the sample-to-sample based evaluation which is employed in the MRC learning, which is a traditional approach. The similarity measurement in APC learning is based on the relationship between a sample and each object class. A sample is said to be most similar to an object class if that object class generates a potential to the sample which is higher than a potential generated by any other object class. Thus, the similarity measure for

APC learning is based on the total accumulated potential.

As described in 5.3, the learning criterion is according to the discriminant function given by

$$D(S) = 1 - \frac{P_{next}(S)}{P_{\max}(S)} \qquad \textbf{(7.4)}$$

where S is a sample in the feature space, and $P_{max}(S)$ and $P_{next}(S)$ are the highest and the next highest potential S receives, respectively. The results of APC learning using 700 handwritten character images in both VCR and angle sequence forms are presented in Table 7.3. APC learning differs from MRC learning by the way similarity measures are used in the clustering process. In generating potential, we can use Euclidean distance, vector correlation, matching cost, and norm of error Fourier transform spectrum. This will result in learning techniques which correspond to MRC techniques. Learning with the APC technique using VCR as the feature representation results in 188 clusters in four days. In the case of angle sequence as the object feature, the number of clusters is 150. The time consumed in this case is 14 hours.

The learning results presented above characterize different learning processes applied to single classifier systems. The learning results of multiclassifier

systems presented together with the corresponding classification results are discussed in the following sections.

7.4 Classification Performances

The classification or the testing results are obtained from different systems which include the MRC, APC, cascaded, vote-to-decide, confidence enhancement, and hierarchical learning classification systems.

Classification performance of a classifier is evaluated by the recognition and the acceptance rates. There are two types of recognition rates: the raw recognition rate, R_{raw}, and the net recognition rate, R_{net}. They are given by

$$R_{raw} = \frac{N_c}{N_t} \times 100\% \quad \textbf{(7.5)}$$

$$R_{net} = \frac{N_c}{N_a} \times 100\% \quad \textbf{(7.6)}$$

where N_c, N_t, and N_a denote the number of correctly classified samples, the total number of test samples, and the number of accepted samples, respectively. R_{net} is actually the recognition rate resulting from classifying the accepted samples and is referred as to the recognition rate hereafter. The acceptance rate, A, is given by

$$A = \frac{N_a}{N_t} \times 100\% \qquad \textbf{(7.7)}$$

which indicates how many percents of the test samples are accepted for classification. The acceptance rate shows the classification flexibility of the recognition system to variations of samples.

7.4.1 MRC Classification

A. Classification of Characters in VCR Form

Table 7.4 shows the results from classification of test samples in their VCR form using Euclidean distance as the similarity measure. The recognition rate of classifying the 700 learning samples is 100% without any rejection. This means that the MRC algorithm properly clusters all the learning samples, such that no alien sample falls in a given cluster. Applying 700 test samples which are different from the learning samples results in the recognition rate of 94.65% at the 90.90% acceptance rate.

Classification, using correlation as similarity measure, results in a 89.57% acceptance rate with 92.50% of the accepted handwritten numerals correctly classified. The results are also shown in Table 7.4.

B. Classification of Characters in Angle Sequence Form

Classification of 700 test samples in the form of angle sequence is also tested. Matching cost is employed as the similarity measure in this classification. 90% of the test samples are accepted for classification, and 93.65% of them are correctly identified, as shown in Table 7.4.

C. Classification of Characters in FTR Form

Test samples in FTR forms are compared by using the Fourier based similarity measure, called error norm. As shown in Table 7.4, 92.14% of the test samples are accepted for classification. The acceptance rate of this method is higher than that of any other method. This reduces the gap between R_{net} and R_{raw}, which are 94.57% and 87.14%, respectively.

7.4.2 APC Classification

A. Classification of Characters in VCR Form

Results from testing 700 samples using an APC classifier with the Euclidean distance of VCR as the basis to evaluate potentials are shown in Table 7.5. Without thresholding, the net recognition rate is 87.27% with the acceptance rate as high as 99.86% (only one test sample is rejected). The net recognition rate increases to

94.64% in the case of a 0.05 threshold value, but the acceptance decreases to 80%.

B. Classification of Characters in Angle Sequence Form

APC classification for the characters in angle sequence form accepts 99.86% of test samples for identification. 90.99% of the accepted samples are correctly recognized, in the case of zero threshold. With the 0.015 threshold value, the recognition rate becomes 91.32%, and 95.44% of test samples are accepted for classification. The results are shown in Table 7.5.

7.4.3 Cascaded Multiclassifier System

This system employs three single classifiers which include two MRC classifiers performing on VCRs and angle sequences, respectively, and an APC classifier on VCRs. The system used for handwritten character recognition is shown in Fig. 7.1. As shown in Table 7.6, classification of 700 test handwritten numerals results in a 91.70% recognition rate and a 99.86% acceptance rate. The first stage accepts 89.71% of the test samples and correctly recognizes 94.43% of them. The rejected samples are applied to the next stage for classification. Up to the second stage, the total recognition rate becomes 93.38%, and the acceptance rate increases to 97.14%. The number of the accepted samples up to this stage rises from 628 to 680. Classifying the remaining 20 rejected samples by the third classifier results in

a final recognition rate of 91.70% and an acceptance rate of 99.86%. 699 samples are accepted for classification by the whole system, and only one sample is rejected.

Replacing the second classifier, MRC classifier for angle sequence, with the image matching (IM) classifier developed in [45], results in the performance shown in Table 7.7. With the 99.43% acceptance rate, 91.52% of the accepted samples are correctly recognized. The IM classifier performs the classification based on the raw images. The image is thinned and partitioned into several segments. Each segment of the original image contains one of the three basic features, which are the line, circle, and arc. The similarity between two images is measured by matching their segments and finding the total distance resulting from the best match. The IM classifier then employs the APC algorithm for the learning and the classification.

7.4.4 Vote-to-Decide Multiclassifier System

The vote-to-decide handwritten character recognition system is shown in Fig. 7.2. The system is composed of three classifiers which are the MRC classifier, the APC classifier, and the IM classifier. The MRC and APC classifiers in this system use the VCR as object feature representation. The classification opinions suggested by the individual classifiers are sent to the last stage for final classification decision. As shown in Table 7.8, at the 94.57% acceptance rate, the

recognition rate is 95.47%. The recognition rate and the acceptance of each individual classifier are also included for references.

7.4.5 Confidence Enhancement System

The system, shown in Fig. 7.3, contains two single classifiers which are the APC classifier operating on VCRs, and the IM classifier operating on raw images. Confidence index vectors generated by the two classifiers are combined for the last stage to make the classification decision. As shown in Table 7.9, 90% of the test samples are correctly classified, and all the samples are accepted in the case of no classification threshold. With a threshold value of 0.05, 95.50% of the samples are accepted for classification, and 93.19% of them are correctly recognized. Increasing the threshold value to 0.1 results in a 97.70% recognition rate and an 87.0% acceptance rate.

7.4.6. Hierarchical Learning Multiclassifier System

The organization of the learning and testing phases of the hierarchical learning multiclassifier system is shown in Fig. 7.4. The system uses two MRC classifiers and one hypercube (HP) [47] classifier in the initial learning stage. The classifier in the further learning uses the MRC algorithm. One of the MRC

classifiers in the initial learning operates on VCRs of the objects and uses the Euclidean distance as the similarity measure. The other MRC classifier operates on angle sequences and employs the string matching technique to measure the similarity.

The hypercube classifier, developed in [47], in the initial learning performs the learning using images in the binary form. In this approach, the entire image is described by a hypercube vector, which contains 0s, 1s, and don't cares. 0 and 1 represent the white and the black pixels, respectively, which contain the essential features. Pixels which provide no feature information are regarded as don't cares. The dimensionality of the hypercube is determined by the number of don't cares. Each hypercube in the feature space contains a number of similar samples. After learning, all the samples are assigned to different hypercubes which are represented by their cube centers. In classification, the test sample is checked to see if it falls in a hypercube. If this is the case, the sample is compared to the cube center by calculating the distance between them. The sample is identified with the cube center if the distance is within a predefined value.

A set of 700 learning samples is applied to each classifier in the initial learning stage. Each classifier establishes its own clusters to represent all the learning samples. After the initial learning is done, another 1000 learning samples

along with the previous 700 learning samples are applied to the initial learning stage. At this point, instead of learning, the learned classifiers in this stage produce their own confidence index vector for each learning sample. The three confidence index vectors for each sample are appended together to form a second form of feature representation of that sample. This feature representation for a sample is given by a 1×30 vector,

$$\Lambda^{H} = \left[\lambda_1,\ \lambda_2,\ \lambda_3,\ \ldots,\ \lambda_{29},\ \lambda_{30}\right]^{T} \qquad \textbf{(7.7)}$$

as described in section 6.3.4 of Chapter 6. The 1700 Λ^{H}'s are then applied to the classifier in the next stage for further learning.

In classification, a test sample is applied to the first stage of the system to obtain its Λ^{H} feature representation, which is input to the next stage for classification. Note that each classifier of the first stage generates a different feature representation for each test sample.

Learning using 1700 samples with a zero threshold results in 163 clusters on the second stage. The number of clusters increases to 231 in the case of learning with a 0.1 threshold value. Recognition of 700 test samples with a zero threshold produces a 93.09% recognition rate at the 97.14% acceptance rate. In the case of a 0.1 threshold, 95.98% of the samples are correctly classified among the 92.43%

of the accepted samples. In this hierarchical system, the single classifier which gives the highest recognition rate is the MRC classifier using VCRs. In this classifier used without threshold, 94.84% of the test samples are correctly identified, and 91.43% are accepted. With a 0.1 threshold value, a 97.47% recognition rate is achieved at the 79.00% acceptance rate. The testing results are summarized in Table 7.10.

In Chapter 8, comparison and discussion of the computer simulation results from different classifier systems are presented. Evaluation of system level solutions is given with general observations and recommendations for future research. Finally, the dissertation work is summarized and conclusions are made.

CHAPTER 8

DISCUSSION AND CONCLUSIONS

Several handwritten character recognition systems have been developed, and their performances have been simulated. Each system contains feature extraction, similarity measure, learning, and classification stages. The performance of each stage is affected by the operating efficiency of its preceding stages. For example, a good feature extraction will not result in proper learning if the similarity measure is poor. Good performance can not be achieved by the classification stage if the learning results are inadequate. Therefore, the analysis of the simulation results should be made on the performances of individual stages and the classification performance of the whole system.

8.1 Feature Extraction

Two types of feature representation developed in this work are the angle sequence representation and the vector contour representation. The angle sequence of an object reveals the geometric relationships between boundary pixel pairs. The vector contour representation, obtained by further processing the angle sequence, describes the total angle variation of a complete contour tracing. As shown in Table

7.1 of the previous chapter, the average time for feature extraction of each representation is small. This means that, with a little computational cost, a raw image can be transformed into its feature representation which provides critical object information for the following stages. With little cost, not only is the dimensionality reduced from 400 to 120, but the useless information is bypassed as well. Consequently, the system takes less time than the one which operates on raw images in learning and classification.

8.2 System Learning

The supervised learning processes for the developed systems are based on the supervised clustering techniques. Clustering process groups similar patterns into clusters according to the similarity measures used. As described in Chapter 7, different learning processes with different similarity measures were tested.

In the MRC method, the learning process which takes VCRs or FTRs as object feature representation is more efficient than the process which takes angle sequences as representation. The learning process using Euclidean distance as the similarity measure is 8.6 times faster than the one using matching cost (2.2 min. / 19.18 min.), and learning using correlation is 17 times faster (1.16 min / 19.18 min.). Using FTRs for learning takes 15.6 times shorter than using angle sequences

whose similarity is measured by string matching technique. This is due to the fact the string matching of angle sequences needs more time to compare the individual angle values between two angle sequences in order to obtain the best match.

As far as the number of clusters is concerned, learning using FTRs gives the smallest number of clusters among the four. It forms 136 clusters to represent the 700 learning samples. Learning using angle sequences generates 143 clusters, and learning using VCRs with Euclidean distance and correlation as similarity measures produces 165 and 182 clusters, respectively. The smaller number of clusters to represent all learning samples indicates better learning ability by methods based on angle sequences. This is a desired feature with respect to hardware costs. However, the learning results also need to be evaluated by the performance of the classification stage.

In the APC method, whose learning results are shown in Table 7.3 of Chapter 7, the learning time of each process is much longer than the MRC learning. This is because every time a cluster center is established, the potential distribution in the feature space is also changed, and the membership of each learning sample must be reevaluated. This reevaluation of memberships due to the potential distribution change demands a larger number of iterations. This results in a longer time for the APC learning process to converge. However, the learning time is not

of great importance for the whole recognition system, since the system only goes through the learning once. Once the learning is done, the system does not learn any more, and it switches to the classification to recognize all the input samples. Like the MRC learning, the APC learning with angle sequences results in a smaller number of clusters than those with VCRs. The number of clusters formed in the learning depends also on the threshold set up for the process. The larger the threshold value, the larger the number of clusters. This is because the larger the threshold value, the stronger the internal similarity of a cluster, and the more clusters needed to cover all the learning samples in the feature space.

A remark worth being made is that both the APC and MRC learning can be used as fuzzy learning. In the APC method, the potential generated by each object class to a sample is accumulated and recorded. The potential function can serve as the membership function in the fuzzy method. In other words, the accumulated potential from each class can be interpreted as the degree of membership of the sample in that class. In a similar way, the similarity measures used in MRC learning can also serve as the membership function in fuzzy algorithms. In this sense, various fuzzy algorithms can be applied to the APC and MRC learning.

8.3 Classification

The classification results reflect the learning effect and show the performance of the whole handwritten character recognition system. As described in section 7.4, classification performance of the classifier can be evaluated by the recognition and the acceptance rates given by equations (7.1), (7.2), and (7.3).

In evaluating the performance of a pattern recognition, the higher these three rates, the better the system. Ideally, a system rejects no test sample and recognizes all the test samples, which means

$$R_{raw} = R_{net} = A = 100\% \quad \textbf{(8.1)}$$

Practically, a trade-off must be made between these factors. That is to say, with a reasonable acceptance rate, a system can give a recognition rate as high as possible, and R_{raw} must be closed to R_{net}. A good system, therefore, is expected to meet the following condition:

$$R_{raw} \sim R_{net}, \quad R_{net} \sim 100\%, \quad A \sim 100\% \quad \textbf{(8.2)}$$

8.3.1 Single Classifier Systems

In the MRC classifier, a test image is to be compared with all the clusters to see which cluster it most parallels. The sample is then categorized to the object

class of that cluster. In the APC classifier, a test image is classified according to the different amounts of potential it accumulates from each object class which contains a number of potential generating centers, the clusters. It is classified to the object class that gives it the highest amount potential among all the classes.

Table 8.1 includes the results from testing different classifiers for comparison. The MRC classifier which performs recognition on VCRs using Euclidean distance gives the highest recognition rate, 94.65%, and a 90.90% acceptance rate. The MRC classifier recognizing test samples which are in FTR form produces a slightly smaller recognition rate, 94.57%, but a higher acceptance rate, 92.14%. The APC classifier, running on the VCRs with a 0.05 threshold value, gives a 94.64% recognition rate and a lower acceptance rate, 80%. On the other hand, while the APC classifier performing on the angle sequences with no threshold has a 88.98%, it produces a very high acceptance rate, 99.86%.

Among the developed single classifier systems, the MRC system for FTRs and the APC system for angle sequences are considered the best two classifiers. The former recognizes 94.57% of the accepted samples, which are 92.14% of the total test samples. The latter gives a smaller recognition rate (91.32%) and a higher acceptance rate (95.43%). It is desired that both the net recognition and acceptance rates be high. A high recognition rate means the system can recognize most of the

accepted samples, and a high acceptance rate reduces the gap between R_{raw} and R_{net}. However, it makes the recognition task more difficult if more test samples are accepted for classification. If some highly distorted handwritten characters which cannot even recognized by human eyes are accepted for recognition, the recognition rate will not be high. On the other hand, a high recognition rate becomes meaningless if it is achieved under a low acceptance rate. Under this consideration, the APC classifier for angle sequence can be considered better than the MRC classifier for FTRs since the acceptance rate is high enough, and the gap (4.18%) between R_{raw} and R_{net} is closer than that (7.43%) in the MRC system.

8.3.2 Multiclassifier Systems

The recognition performance of the cascade multiclassifier systems is shown in Table 8.2. Two systems with different single classifiers in the second classification stage are tested. They all show the improvement on the raw recognition acceptance rates over the single classifier systems. In Table 8.2(a),

$$R_{raw} : 84.71\% \rightarrow 90.71\% \rightarrow 91.57\%$$

$$A \quad : 89.71\% \rightarrow 97.14\% \rightarrow 99.86\%$$

R_{raw} is greatly improved, and is very close to R_{net} (only 0.13% lower) with the cost of a 2.73% drop of R_{net}. A is lifted up to 99.86%. In fact if the second stage is the

final output of the system, the performance of this cascade system is still good. At the 2nd stage, the acceptance rate is still very high (97.14%), and R_{raw} (90.71%) is close to R_{net} (93.38%). The performance of the system with IM classifier in the second stage gives a similar result, as shown in Table 8.2(b).

In this system, the raw recognition and acceptance rates are increased by 6.86% and 10.15%. This is a significant improvement, which shows that this multiclassifier system utilizes the proficiency of the individual classifiers to accomplish a higher performance in handwritten character recognition.

As for the vote-to-decide multiclassifier system, the performance is also better than that of any single classifier system. In Table 8.3, the classification results of each single classifier as well as the whole system are included. R_{raw} is lifted up from 87.14% to 90.29%, and R_{net} also increases from the highest value among the three (94.65%) to a even higher value, 95.47%. The results show that even though two of the single classifiers (APC and IM) have much less net recognition rates than MRC has, R_{raw} does not drop but is improved. On the other hand, the three R_{raw}'s are close to each other, but the resulting R_{raw} is much higher. The acceptance rate of the whole system is better than that of MRC and IM, though it drops from 99.86% to 94.57%, which is still in a good range.

The computer simulation results show that the vote-to-decide system is

worth being developed since it increases the both raw and net recognition rates. It is obvious that the APC classifier in this system compensates the other two single classifiers in the acceptance rate so that the whole system reaches a good acceptance rate. If the correlations between the single classifiers can be evaluated so that only uncorrelated single classifiers are incorporated into the vote-to-decide system, the recognition performance can be improved.

The classification results of the remaining multiclassifier systems are summarized in Table 8.4. The classification performance of the confidence enhancement systems are not as good as those described above, though they are better than single classifier systems. The same is true in the system which is the combination of the vote-to-decide and the confidence enhancement systems. The classification performance of the hierarchical learning multiclassifier system is close to the performances of the cascaded and vote-to-decide systems. At zero threshold, the system produces a 90.43% raw recognition rate, a 93.09% net recognition rate, and a 97.14% acceptance rate. If there are enough single classifiers, the network can be organized to more hierarchical levels, and the learning can be tuned at each level. In this case, the final learning effect can be expected to be much improved.

From the computer simulation results, thresholding controls the flexibility

of the recognition systems. High thresholding causes clustering groups samples with small variance. This results in more clusters after learning since clusters reject including learning samples with large variance, and more clusters need to be formed to cover all the learning samples. In testing, samples are likely to be rejected for classification if they are far away from the clusters in the feature space. As a result, R_{net} is increased since once samples are accepted, they are very likely to be correctly recognized. On the other hand, R_{raw} is decreased since more test samples are rejected. Normally, there are some handwritten characters which are highly confusing with other characters due to bad writing. In this case, it is good to leave them undetermined rather than to recognize them. This is done by thresholding as long as the acceptance rate is not affected too much.

8.4 Conclusions

Automated handwritten character recognition is a very difficult problem to tackle. It demands fast and accurate processing of the considerably different handwritten characters. In this research, the developed handwritten character recognition systems use the conventional pattern recognition techniques for learning and the neural networks for classification. The learning process employing conventional pattern recognition techniques requires a little computational cost, but

greatly reduces the complexity of the neural network hardware. The clustering-based supervised learning is very efficient due to its small computational complexity. The number of iterations required for the optimal supervised clustering is small in comparison to the unsupervised clustering. The clustering looks for the best partitions in each object class so that each cluster contains only the samples of the same class. The clustering is completed when each object sample is assigned to a proper group of its own type. This is not the case in unsupervised clustering, in which data partitioning must be readjusted very frequently to obtain an optimal clustering based on the similarity criterion used. This usually requires a large number of iterations to reach the convergence, and there is no guarantee that each cluster contains no alien object patterns since the class labels of all the learning samples are unknown.

In the testing phase, the neural network is used as the classifier so that its parallelism and computational ability are fully utilized. All the neurons serving as the cluster centers respond concurrently to the sample which is identified with the class of the neuron that gives the strongest response. This results in an efficient and fast process. Therefore, it is expected that the handwritten character recognition systems which combine the conventional pattern recognition and neural network approaches will be very useful.

The two goals as the research objective have been achieved. The first goal, recognition invariant to writing style and image transformations, is accomplished by extracting useful object features with proper similarity measures used in learning processes. The angle sequence is invariant to image scaling, translation, and rotation, as proven in Chapter 2 and illustrated with the puzzles in Chapter 4. The similarity measures for different feature representations to be used in learning and recognition are well developed. The two supervised clustering techniques developed, MRC and APC, generate a relatively small number of cluster centers to represent all the learning samples. This results in a fast classification in the testing phase.

The second goal, development of multiclassifier systems, is achieved by organizing the developed single classifiers in different systems. From the computer simulation results, the multiclassifier system successfully lifts up the recognition rate and keeps the acceptance rate high. In this sense, multiclassifier systems are promising organizations for neural networks in the area of object recognition. Every single classification system has its limit in handwritten recognition. Only the system with multiple classifiers can achieve a higher recognition rate at a high acceptance rate.

Therefore, with a good design, a multiclassifier system can be expected to

produce a near 100% raw recognition rate. In such a system, there are three conditions to be satisfied. First, the learning and classification algorithms used in different single classifiers should be as much uncorrelated to each other as possible in order to have the effect of mutual compensation. This means that it is hard for one of the single classifiers to make a classification decision, the object can still be correctly recognized if it is not confusing to other classifiers. In this way, the recognition performance of the whole system can be better than any of the single classifiers in the system. To fulfill this second condition, mathematical tools are needed to evaluate the independency of individual classifiers' learning. The second condition is that the single classifiers in the system must be of the same level of recognition performance (high R_{raw}) so that their "opinions" can be considered equally important. This means that the overall classification is made based on single classifiers with the same recognition ability but from different points of view. If it is not the case, the opinions must be weighted so that the final classification decision will not be misled by the classifiers with less recognition ability.

The final condition is the hardware consideration. The hardware cost depends on the size of each single classifier. It is reflected by the number of clusters and the computational ability required for each cluster center. Since each cluster center evaluates the similarity to a test sample, it needs physical space for the

computation hardware. Therefore, the hardware of each single classifier should be kept simple. In addition, single classifiers in the multiclassifier system should be organized in a hierarchical structure so that the system can be expanded in a most cost effective fashion. This means a multilayer neural network recognition system can be built based on the single layer hardware with a minimum cost. Satisfying these three conditions, the multiple classifier system is expected to produce satisfactory results in the area of handwritten character recognition.

REFERENCES

1. S. J. Smith, M. O. Bourgoin, K. Sims, and H. L. Voorhees, "Handwritten character classification using nearest neighbor in large database," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 16, no. 9, pp. 915-919, September 1994.

2. S.-W. Lee and H.-H. Song, "Optimal design of reference models for large-set handwritten character recognition," *Pattern Recognition*, vol. 27, no. 9, pp. 1267-1274, 1994.

3. M. Chester, *Neural Networks: a tutorial*, Prentice-Hall, Englewood Cliffs, New Jersey, 1993.

4. C. Yuceer and K. Oflazer, "A rotation, scaling, and translation invariant pattern classification system," *Pattern Recognition*, vol. 26, no. 5, pp. 687-710, 1993.

5. K. Fukushima, S. Miyake, and T. Ito, "Neocognition: a neural network model

for a mechanism of visual pattern recognition," *IEEE Transaction System, Man, and Cybernetics*, vol. 13, no. 5, pp. 826-834, September 1993.

6. H. Guo and S. B. Gelfand, "Classification tree with neural network feature extraction," *IEEE Transaction Neural Networks*, vol. 3, no. 6, pp. 923-933, November 1992.

7. S. J. Perantonis and P. J. G. Lisboa, "Translation, rotation, and scale invariant pattern recognition by high-order neural networks and moment classifiers," *IEEE Transaction Neural Networks*, vol. 3, no. 2, pp. 241-251, March 1992.

8. H. K. Sardana, M. F. Daemi, and M. K. Ibrahim, "Global description of edge patterns using moments," *Pattern Recognition*, vol. 27, no. 1, pp. 109-118, 1994.

9. A. K. Jain, *Fundamentals of digital image processing*, Prentice-Hall, Englewood Cliffs, New Jersey, 1989.

10. I. Sekita, T. Kurita, and N. Otsu, "Complex autoregressive model for shape

recognition," *IEEE Transaction Pattern Analysis and Machine Intelligence*, vol. 14, no. 4, pp. 489-496, April 1992.

11. C. T. Zahn and R. Z. Roskies, "Fourier descriptors for plane closed curves," *IEEE Transaction Computers*, vol. c-21, no. 3, pp. 269-281, March 1972.

12. S. A. Mahmoud, "Arabic character recognition using Fourier descriptors and character contour encoding," *Pattern Recognition*, vol. 27, no. 6, pp. 815-824, 1994.

13. T. Taxt, J. B., Ólafsdóttir, and M. Dæhlen, "Recognition of handwritten symbols," *Pattern Recognition*, vol. 23, no. 11, pp. 1155-1166, 1990.

14. T. S. Shepherd, W. Uttal, S. Dayanand, and R. Lovell, "A method for shift, rotation, and scale invariant pattern recognition using the form and arrangement of pattern-specific features," *Pattern Recognition*, vol. 25, no. 4, pp. 343-356, 1992.

15. H. Nishida and S. Mori, "Algebraic description of curve structure," *IEEE*

Transaction Pattern Analysis and Machine Intelligence, vol. 14, no. 5, pp. 516-533, May 1992.

16. H. Rom and G. Medioni, "Hierarchical decomposition and axial shape description," *IEEE Transaction Pattern Analysis and Machine Intelligence*, vol. 15, no. 10, pp. 973-981, October 1993.

17. L. Gupta, M. R. Sayeh, and R. Tammana, "A neural network approach to robust shape classification," *Pattern Recognition*, vol. 23, no. 6, pp. 563-568, 1990.

18. G. Cortelazzo, G. A. Mian, G. Vezzi, and P. Zamperoni, "Trademark shapes description by string-matching techniques," *Pattern Recognition*, vol. 27, no. 8, pp. 1005-1018, 1994.

19. J. A. Starzyk and S. Chai, "Vector contour representation for object recognition in neural networks," *IEEE International Conference on Systems, Man, and Cybernetics*, vol. 1, pp. 399-404, 1992.

20. L. Gupta and M. D. Srinath, "Invariant planar shape recognition using dynamic alignment," *Pattern Recognition*, vol. 21, no.3, pp. 235-239, 1988.

21. G. N. Bebis and G. M. Papadourakis, "Object recognition using invariant object boundary representations and neural network models," *Pattern Recognition*, vol. 25, no. 1, pp. 25-44, 1992.

22. I. Dinstein, G. M. Landau, and G. Guy, "Parallel (PRAM EREW) algorithms for contour-based 2D shape recognition," *Pattern Recognition*, vol. 24, no. 10, pp. 929-942, 1991.

23. Y. Lin, J. Dou, and H. Wang, "Contour shape description based on an arch height function," *Pattern Recognition*, vol. 25, no. 1, pp. 17-23, 1992.

24. R. Duda and P. Hart, *Pattern Classification and Scene Analysis*. Wiley-Interscience, New York, 1974.

25. S. I. Gallant, *Neural Network Learning and Expert Systems*, MIT Press, Cambridge, Massachusetts, 1993.

26. T. Kurita, "An efficient agglomerative clustering algorithm using a heap," *Pattern Recognition*, vol. 24, pp. 205-209, 1991.

27. K. Hattori and Y. Torii, "Effective algorithms for the nearest neighbor method in the clustering problem," *Pattern Recognition*, vol. 26, no. 5, pp. 741-746, 1993.

28. D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, "A learning algorithm for Boltzmann machines," *Cognitive Science*, vol. 9, pp. 147-169, 1985.

29. Y.-H. Pao, *Adaptive Pattern Recognition and Neural Networks*, Addison-Wesley, Reading, Massachusetts, 1989.

30. K. P. Chan and Y. S. Cheung, "Clustering of clusters," *Pattern Recognition*, vol. 25, no. 2, pp. 211-217, 1992.

31. M. S. Kamel and S. Z. Selim, "New algorithms for solving the fuzzy clustering problem," *Pattern Recognition*, vol. 27, no. 3, pp. 421-428, 1994.

32. E. C.-K. Tsao, J. C. Bezdek, and N. R. Pal, "Fuzzy Kohonen clustering networks," *Pattern Recognition*, vol. 27, no. 5, pp. 757-764, 1994.

33. T. Kohonen, *Self-organization and Associative Memory*, 3rd ed. Springer, Berlin, 1989.

34. G. Beni and X. Liu, "A least biased fuzzy clustering method," *IEEE Transaction Pattern Analysis and Machine Intelligence*, vol. 16. no. 9, pp. 954-960, September 1994.

35. Laurene Fausett, *Fundamentals of Neural Networks*, Prentice-Hall, Englewood Cliffs, New Jersey, 1994.

36. Berthold Klaus Paul Horn, *Robot Vision*, MIT Press, Cambridge, Massachusetts, 1986.

37. Arun D. Kulkarni, *Artificial Neural Networks for Image Understanding*, Van Nostrand Reinhold, New York, 1994.

38. R. L. Burden and J. D. Faires, *Numerical Analysis*, PWS-KENT Publishing Company, Boston, Massachusetts, 1993.

39. R. C. Gonzales and P. Wintz, *Digital Image Processing*, Addison-Wesley, Reading, Massachusetts, 1987.

40. G. Carpaneto and P. Toth, "Solution of the assignment problem [H]," *Association for Computing Machinery Transaction on Mathematical Software*, vol. 6, no. 1, pp 104-111, March 1980.

41. B. Jeon and D. A. Landgrebe, "Fast Parzen density estimation using clustering-based branch and bound," *IEEE Transaction Pattern Analysis and Machine Intelligence*, vol. 16, no. 9, pp. 950-954, September 1994.

42. S. Banerjee and A. Rosenfeld, "Model-based cluster analysis," *Pattern Recognition*, vol. 26, no. 6, pp. 963-974, 1993.

43. K.-R. Hsieh and W.-T. Chen, "A neural network model which combines

unsupervised and supervised learning," *IEEE Transaction Neural Networks*, vol. 4, no. 2, pp. 357-360.

44. K. Rose, E. Gurewitz, and G. C. Fox, "Constrained clustering as an optimization method," *IEEE Transaction Pattern Analysis and Machine Intelligence*, vol. 15, no. 8, pp. 785-794, August 1993.

45. J. A. Starzyk and Y.-W. Jan, "Algorithm and architecture for feature extraction in image recognition," *Proceedings of 26th southeastern Symposium on System Theory*, pp. 448-452, March 1994.

46. J. A. Starzyk and S. Chai, "Supervised learning with potentials for neural network-based object recognition," *Proceedings of 26th Southeastern Symposium on System Theory*, pp. 192-196, March 1994.

47. J. A. Starzyk and N. Ansari, "Feedforward neural network for handwritten character recognition," *IEEE International Symposium on Circuits and Systems*, vol. 6, pp. 2884-2887, 1992.

APPENDIX

ORGANIZATION OF COMPUTER PROGRAMS

The organization of all the computer programs developed in this dissertation are described in this section. Most of the programs are written in C codes. For string matching, an assignment program [40] written in Fortran is included. Matlab is also employed to perform FFT for Fourier-based feature representation and similarity measure. The home directory is /home/bobcat/project.

1. README file:

Description of the procedures of compiling and running different programs. This file also includes brief description of different classifier systems and their computer simulation programs.

2. Directories

Directories at the first level include:

1. Image/
2. Feature/
3. Single/

4. Multi/

3. Directory Image (Image/)

(1) img700l: 700 binary images for learning.

(2) img700t: 700 binary images for testing.

(3) img1700h: 1700 binary images for hierarchical learning.

4. Directory Feature (Feature/)

(1) angle.c: extracts the angle sequences of objects.

(2) vcr.c: extracts the angle sequence and forms the vector contour representation (VCR).

(3) ftr.m: a matlab program which takes FFT of angle sequences.

5.Directory Single (Single/)

This directory has three subdirectories: MRC, APC, and Function.

(1) Single/MRC/: contains MRC classifier programs using different feature representations.

* mrc_learna.c: MRC learning using angle sequences.

  mrc_classa: MRC testing using angle sequences.

* mrc_learnf.c: MRC learning using FTRs.

  mrc_classf.c: MRC testing using FTRs.

* mrc_learnv.c: MRC learning using VCRs.

  mrc_classv.c: MRC testing using VCRs.

(2) Single/APC/: contains APC classifier program

* apc_learna.c: APC learning using angle sequences.

  apc_classa.c: APC testing using angle sequences.

* apc_learnv.c: APC learning using VCRs.

  apc_classv.c: APC testing using VCRs.

(3) Single/Function/: contains function for string matching.

* assct.f: assignment program written in Fortran [40].

* cost.c: program to find the minimum cost of string matching.

6. Directory Multi (Multi/)

This directory contains programs for multiclassifier systems organized in four subdirectories: CAS, VOTE, CE, and HIER.

(1) Multi/CAS/: program for cascaded multiclassifier system.

* cas_class.c: testing program.

(2) Multi/VOTE/: program for vote-to-decide multiclassifier system.

* vote_class: testing program.

(3) Multi/CE/: program for confidence enhancement multiclassifier system.

* ce_class: testing program.

(4) Multi/HIER/: programs for hierarchical learning multiclassifier system.

* hier_learn.c: learning program.

* hier_class.c: testing program.

CHAI, SIN-KUO Ph.D. March, 1995

Electrical and Computer Engineering

Multiclassifier Neural Networks for Handwritten Character Recognition

(199pp.)

Director of Dissertation: Janusz A. Starzyk, Ph.D.

Handwritten character recognition is a highly challenging area in the field of pattern recognition. In performing recognition, any single classifier system has its strengths and weaknesses. The objective of this dissertation is to develop multiclassifier systems which utilize the combined strength of several classifiers to make a significant improvement in recognition over the single classifiers. The multiclassifier systems developed were cascaded, vote-to-decide, confidence enhancement, and hierarchical learning systems. In each multiclassifier system, the single classifiers contained their own feature extraction, similarity measure, learning, and classification stages.

Feature extraction extracted object features and formed feature representations. Three feature representations were developed, which were the angle sequence, vector contour representation (VCR), and Fourier transform representation (FTR). To evaluate the similarity of objects in different representation forms, measures based on Euclidean distance, vector correlation, string matching cost, and Fourier transform were developed.

For learning, two supervised clustering techniques were developed: maximum region clustering (MRC) and accumulated potential clustering (APC). The MRC learning maximized the clustering regions by including as many samples of the same type as possible in each cluster without enclosing any alien sample. In APC learning, the feature space was viewed as an electrostatic field in which each cluster served as a potential generating center. Each object class established the minimum number of cluster centers necessary to protect its members from being attracted to other classes.

In the classification stage, the MRC classifiers identified a test sample with the class of its nearest cluster center. The APC classifiers assigned a test sample to the object class which attracted it the most. In a multiclassifier system, the final classification decision was made based on the individual decisions made by the single classifiers. The rules of making a final classification decision depended on the multiclassifier system and differed from system to system.

All the developed single classifier and multiclassifier systems have been simulated on a Sun computer, and the results were presented. The computer simulation results showed that the multiclassifier system significantly improved the recognition performance.

Approved: _____

Signature of Director