PRECLASSIFICATION OF CHINESE CHARACTERS BY HIERARCHICAL NEURAL NETWORKS

Young-Sheng Chen and Wen-Hsiang Tsai
Institute of Computer and Information Science
National Chiao Tung University
Hsinchu, Taiwan 300, Republic of China

ABSTRACT

A hierarchical neural network system for the preclassification of handwritten Chinese characters is proposed. The architecture of the preclassification system includes multiple Kohonen networks which perform a modified winner-take-all learning algorithm. Two files with 400 and 1,000 character classes were tested. A 95.5% average rate of correct classifications has been achieved.

Key Words: preclassification, Kohonen networks, winner-take-all algorithm.

1 Introduction

About one fourth of the people in the world read and write Chinese every day, and it is important to develop a more comfortable user interface that makes people easier to use a Chinese computer. The goal is to provide a more effective way to input Chinese to computers instead of input by keyboard. Recently, the neural network has been employed in the development of Chinese character recognition systems in order to take the advantages of the characteristics of parallel processing in neural networks. Several well known models have been used in the study of character recognition, such as neocognitron [1], the Hopfield network [2], the Hamming network [3], the Kohonen feature maps [4], etc. In this paper, a hierarchical neural network system composed of multiple Kohonen networks [5] for preclassification of handwritten Chinese characters is proposed. The goal of the system is to preclassify a Chinese character into a number of character groups so that a detailed matching subsystem with less complexity can be implemented for further recognition of the characters in each group.

In the remainder of this paper, the proposed approach is described in Section 2. Some experimental results are shown in Section 3, followed by conclusions in Section 4.

2. Proposed Approach for Preclassification

2.1 Basic concepts

There are about 5401 daily-used Chinese characters, and each Chinese character stands for a class which contains several samples in our database. Because of the large vocabulary set of Chinese characters, it is a good idea to use a tree classifier to yield a number of character sets, each with a small number of characters, before detailed matching when we are developing a recognition system. This problem is called preclassification and is a type of clustering problem. To solve this problem, we propose a

Kohonen neural network which implements a training algorithm extended from [6]. The algorithm is called the modified winner-take-all learning algorithm in this study.

The modified winner-take-all algorithm has the following advantages over the traditional winner-take-all algorithm: (1) it can break the local minimum that the first clustering problem may encounter; (2) its new criterion can reduce the number of overlapping clusters that the traditional algorithm cannot fulfill; (3) the initial weights can be assigned arbitrarily before training the network.

Suppose that there are N Chinese character classes and each class has P samples to be trained. Each sample has n features, so each sample can be represented by Equation (2.1) below, where i is the class number and k is the sample number in the ith class:

$$\mathrm{x}_{i,k} = \left(x^i_{k,1},\ x^i_{k,2},\ \ldots,\ x^i_{k,n}\right). \tag{2.1}$$

Because the samples in each class may finally be assigned to more than one cluster in the clustering process, we represent the number of classes which belong to both cluster-i and cluster-j as an entity w_{ij} of $\mathbf{W}$ where $\mathbf{W}$ is an $\mathrm{m}\times\mathrm{m}$ matrix, and represent whether the center of class-i belongs to cluster-j or not by a binary value v_{ij} of $\mathbf{V}$ where $\mathbf{V}$ is an $\mathrm{N}\times\mathrm{m}$ matrix. Notice that the center of each class belongs to exactly one cluster. The values v_{ij} and w_{ij} have some properties described by Equations (2.2) and (2.3) below:

$$\sum_{j=1,\ldots,\mathrm{m}} v_{ij} = 1 \text{ and } \sum_{i=1,\ldots,\mathrm{N}} \sum_{j=1,\ldots,\mathrm{m}} v_{ij} = \mathrm{N}; \tag{2.2}$$

$$\sum_{j=1,\ldots,\mathrm{m}} w_{ij} \geq 1 \text{ and } \sum_{i=1,\ldots,\mathrm{N}} \sum_{j=1,\ldots,\mathrm{m}} w_{ij} \geq \mathrm{N}. \tag{2.3}$$

2.2 Detailed description of proposed approach

There are two goals in our study. One is to determine a better distribution of class centers. The other is to decrease the number of overlapping clusters (called overlap number henceforth). In the following, we introduce a new criterion and four operations to equalize the distributions of the clusters.

In the modified winner-take-all learning algorithm, we use the criterion COST, defined as follows:

$$\mathrm{COST} = \left[\mathrm{k}^2 - \sum_{j=1}^{\mathrm{m}} \sum_{l\in \mathrm{A}_j} \left(\bar{\mathrm{w}}_j^{\mathrm{t}} \tilde{\mathrm{x}}_l\right)\right]; \tag{2.4}$$

where k is a constant; $\tilde{\mathrm{x}}_l = \mathrm{k}\cdot\dfrac{\left(\frac{1}{\mathrm{P}}\sum_{k=1}^{\mathrm{P}} \mathrm{x}_{l,k}\right)}{\left\|\frac{1}{\mathrm{P}}\sum_{k=1}^{\mathrm{P}} \mathrm{x}_{l,k}\right\|}$ is the normalized center of class-l, P is the number of training samples in each class; $l \in \mathrm{A}_j$ represents that an input to the network by at least one feature vector in class-l makes the jth output neuron the greatest value; $\bar{\mathrm{w}}_j = \begin{bmatrix} w_{j1} & w_{j2} & \ldots\ldots & w_{jN} \end{bmatrix}^{\mathrm{t}}$ is the normalized weight vector connected to output neuron-j, so that $\|\bar{\mathrm{x}}_l\| = \|\bar{\mathrm{w}}_j\| = \mathrm{k}$. Notice that COST = COST1 + COST2 where

$$\mathrm{COST1} = \sum_{i,j=1}^{i=N,j=m} v_{ij}\left(\mathrm{k}^2 - \bar{\mathrm{w}}_j^{\mathrm{t}}\bar{\mathrm{x}}_i\right) \quad \text{and} \quad \mathrm{COST2} = \sum_{i,j=1}^{i=N,j=m} \left(w_{ij} - v_{ij}\right)\left(\mathrm{k}^2 - \bar{\mathrm{w}}_j^{\mathrm{t}}\bar{\mathrm{x}}_i\right). \quad (2.5)$$

The term COST1 is the traditional distance criterion which measures the distribution of class centers in a clustering result, and the term COST2 reflects the total overlap numbers of the result. Based on this analysis, our goal is to get the smallest COST.

In addition, there are four operations used in our modified algorithm. They are CREATE, DELETE, SEPARATE, and LUMP. The LUMP and DELETE operations each decrease the number of output neurons by 1, and SEPARATE and CREATE each increase the number of clusters by 1. The decisions are based on the following rules.

If w_{ij} between cluster-i and cluster-j is large and there is a large number of classes whose centers are in cluster-i and cluster-j, then the CREATE operation is used. As shown in Figure 2.1, all the weights connected to the output neuron y_{u+1} are assigned values which comprise the center of this newly created cluster. The candidate of cluster pair (r,s) for doing the CREATE operation is chosen by calculating $\mathrm{f_C}(j,k)$ described by Equation (2.6) for each distinct pair of clusters (j,k) and selecting the pair of clusters which has the largest value:

$$\mathrm{f_c}(j,k) = \alpha_1^{\mathrm{c}}\left(w_{jk} - \bar{p}\right) + \alpha_2^{\mathrm{c}}\left(w_{jj} + w_{kk} - 2\bar{n}\right) \quad \text{where } j \neq k, j,k = 1,2,\ldots,\mathrm{m}; \quad (2.6)$$

where $\bar{n} = \dfrac{1}{\mathrm{m}}\sum_{j=1}^{\mathrm{m}} w_{jj}$ is the average number of classes whose centers belong to a cluster,

and $\bar{p} = \dfrac{2}{m(m-1)} \sum_{j,k=1,\ j<k}^{m} w_{jk}$ is the average number of classes in which there are samples belonging to more than one cluster.

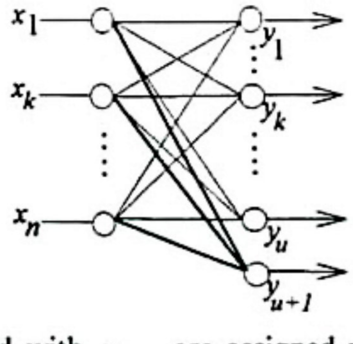

Figure 2.1: Weights connected with $y_{(u+1)}$ are assigned values, so there are total (u+1) clusters.

We then use cluster-*r* and cluster-*s* to compute $\tilde{w}_{u-1}$. Notice that $\tilde{w}_{u+1}$ corresponds to the center of newly created clusters, as shown in Figure 2.2.

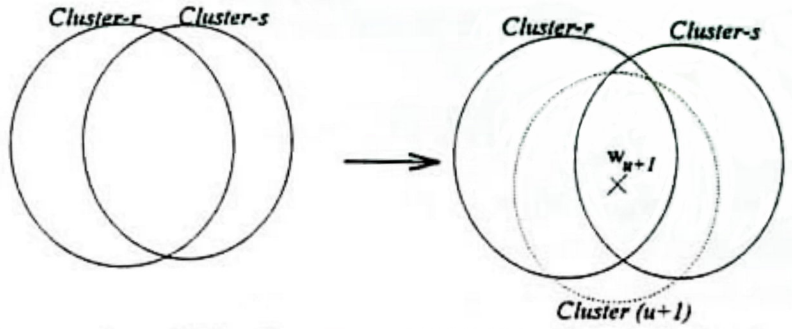

Figure 2.2: The meaning of the CREATE operation: the circle in dashed lines is the newly created cluster. The cross point is the center of this cluster.

If there is a large number of class centers in a cluster and many of them have samples belonging to other clusters, then the SEPARATE operation is used, and a new cluster is created as in the case of the CREATE operation. The candidate *j* for the SEPARATE operation can be chosen by computing the function value described by Equation (2.7) for each cluster j and by selecting the one which has the largest value:

$$f_s(j) = \alpha_1^s\left(w_{jj} - \bar{n}\right) + \alpha_2^s \sum_{l=1,l\neq j}^{l=m} w_{jl} \quad \text{where } j = 1,\ldots,m. \tag{2.7}$$

Notice that $\tilde{w}_{(u+1)}$ corresponds to the center of newly created clusters. See Figure 2.3 for an illustration of the meanings of $\tilde{w}_{(u+1)}$ and $\tilde{w}_j$.

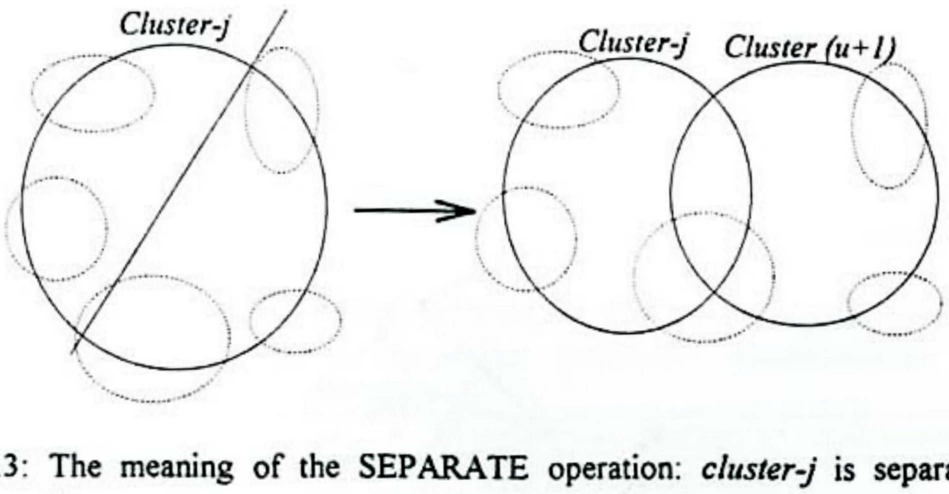

Figure 2.3: The meaning of the SEPARATE operation: *cluster-j* is separated into *cluster-j* and *cluster-(u+1)*. $\tilde{w}_j$ corresponds to the center of *cluster-j* and $\tilde{w}_{(u+1)}$ corresponds to the center of *cluster-(u+1)*.

If w_{ij} is large and there is a small number of class centers belonging to cluster-i and cluster-j, then we use the LUMP operation. The operation assigns the value of 0 to all weights connected to an output neuron, as shown in Figure 2.4. It takes place between a pair of clusters (i, j) which satisfies

$$\mathrm{f_l}(i,j) \ge \mathrm{f_l}(r,s) \quad \text{for every } r,s = 1,\dots,\mathrm{m} \text{ and } r \ne s; \tag{2.8}$$

$$\mathrm{f_l}(r,s) = \alpha_1^{l}\left(2\bar{n} - w_{jj} - w_{kk}\right) + \alpha_2^{l}\left(w_{jk} - \bar{p}\right) \quad \text{where } j \ne k;\ j,k = 1,\dots,\mathrm{m}. \tag{2.9}$$

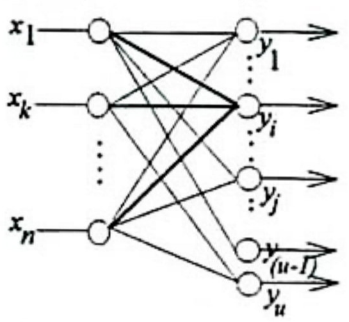

Figure 2.4: The following description lumps cluster-i and cluster-j. Notice that $\tilde{w}_i$ is updated, $\tilde{w}_j$ is replaced by $\tilde{w}_u$, and y_u is marked inactive, so there are totally (u-1) clusters left.

If there is a cluster with a small number of class centers belonging to it, but many of them have samples belonging to other clusters, then we use the DELETE operation, as shown in Figure 2.5. The DELETE operation takes place at cluster-j if

$$\mathrm{f_d}(j) \ge \mathrm{f_d}(k) \quad \text{for every } k = 1,\dots,\mathrm{m}; \tag{2.10}$$

$$\mathrm{f_d}(k) = \alpha_1^{d}(\bar{n} - w_{kk}) + \alpha_2^{d}\sum_{l=1, l\ne k}^{l=\mathrm{m}} w_{kl} \quad \text{where } k = 1,\dots,\mathrm{m}. \tag{2.11}$$

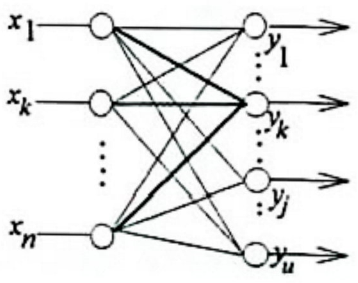

Figure 2.5: $\tilde{w}_j$ is replaced by $\tilde{w}_u$, and y_u is marked inactive, so there are totally (u-1) clusters left.

Suppose that there are N classes to be trained and each class contains P training samples. The proposed modified winner-take-all algorithm is executed on a Kohonen network which has n input and (m+3) output neurons, and the goal of this learning algorithm is to classify the N classes into m clusters. Also, define the function δ as follows:

$\delta(\text{CREATE}) = \delta(\text{SEPARATE}) = 1$; $\delta(\text{LUMP}) = \delta(\text{DELETE}) = -1$. At last, denote class-$i$ by C_i which contains training samples $c_{i1}, \ldots, c_{iP}$.

Modified-winner-take-all-algorithm.

1. Initialize all weights of the Kohonen network with 0, and COST with 0.0.
2. Calculate the center of each class C_i and normalize each vector c_{ij} for every i=1, ..., N and j=1, ..., P.
3. Select m vectors randomly from the N normalized class centers, and assign these vectors to m weights connected to the first m output neurons.
4. Execute the winner-take-all learning algorithm on the network using the N*P training samples.
5. Randomly select a sequence of the four operations introduced previously, which are represented by $\{s_1, \ s_2, \ \ldots.., \ s_x\}$. The sequence must satisfies the constraints $\sum_{i=1}^{x} \delta(s_i) = 0$ and $x \le 6$. Sequentially apply these operations to the network.
6. Calculate COST, compare it with the previous COST value, and save the weights which have the larger corresponding COST value.
7. If there is no change of the value of COST in a certain number of iterations, then stop; else go to Step 4.

We have designed some experiments to prove that the modified winner-take-all algorithm is feasible for our applications. From the database of the Industrial Technology Research Institute (ITRI), we select 60 categories of Chinese characters. In the first three experiments, 8 samples were randomly chosen from each category to get totally 480 samples in our training set. At first, each sample was processed through the feature-extraction stage to get a feature vector of dimension 144. The network has 144 input neurons and 8 output neurons; that is, it is desired to classify these samples into 8 clusters. The training parameters in each training process were the same. The initial weights were selected in several ways and the experimental result is shown in Table 2.1.

Table 2.1: Statistical results of three experiments. The initial weights of the network come from 8 of the 60 class centers. (Method 1: winner-take-all. Method 2: The modified winner-take-all.)

Example #	Initial weight	Method 1	Method 2	# of Iterations
1	1, 2, 3, 4, 5, 6, 7, 8	849.238165	603.669661	16
2	6, 9, 10, 21, 22, 31, 32, 58	874.338785	613.63219	26
3	3, 4, 5, 6, 7, 8, 9, 10	843.196740	633.527163	20

We conclude from table 2.1 that the modified winner-take-all learning algorithm gets smaller values of COST.

Another experiment was performed using the same categories of characters but 30 samples rather than 8 samples are chosen for each category. Our goal is to prove that the method can reduce the average overlap number between clusters. Let w_{ij} and w'_{ij} be the numbers of classes in which there are at least one sample belonging to cluster-*i* and cluster-*j* after the winner-take-all and modified winner-take-all algorithms are performed, respectively. From the result, we conclude that the average overlap number of the winner-take-all algorithm is 21.6 and it is reduced to 12.46 by our method.

The above comparison reveals that adopting the proposed algorithm is helpful in developing our system, and the proposed network is feasible to be constructed hierarchically.

3. Experimental Results

Experiments have been conducted to examine the performance of the proposed approach. Two files were tested in the experiments. For the first file, totally 400 classes of Chinese characters were selected from the handwritten Chinese character image database, version 1 (CCL-HCI1), of the Computer & Communication Laboratories of the ITRI which contains 5401 character classes and each class includes 250 variants arranged according to writing qualities. 10 samples were randomly chosen from each character class for training the preclassification network. For the second file, totally 1000 classes of Chinese characters were selected from CCL-HCI1, and 12 samples randomly extracted from each class were used for training the preclassification network.

For the first file, the 400 character classes are classified into 6 categories at first, and then each category are classified to 5 categories. Thus, there are two layers in the preclassification network. To compute feature vectors, the summation of the membership function values of all the extracted feature points in each grid is obtained as a feature value. And the combination of the feature values of all the grids in the horizontal and vertical planes is used as the feature vector for training and classification of the first layer. The combination of the feature values of all the grids of the four planes is used as the feature vector for training and classification of the second layer. According to the above illustrations, a character input to the hierarchical network is classified into one of the thirty clusters at the second layer.

For the second file, the 1000 character classes are classified into 12 categories at first, and then each category are classified to several categories, as shown in Table 3.1. Thus, there are two layers in the preclassification network. The combination of the horizontal and vertical grid frames is used as the feature vector for training and classification in the first layer. And the combination of the grid frames of the four orientations is used as the feature vector for training and classification in the second layer. A character input to the hierarchical network is thus classified into one of the 130 clusters at the second layer. We have experimented with 10 sets of character classes and the rates of correct classifications are shown in Table 3.2. The average rate is about 94%.

Table 3.1: (a) The number of classes of each of the 12 clusters at the first layer. (b) The number of clusters each of the 12 clusters are classified into.

Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5	Cluster 6
228	461	264	322	220	334
Cluster 7	Cluster 8	Cluster 9	Cluster 10	Cluster 11	Cluster 12
345	227	259	358	192	318

(a)

Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5	Cluster 6
8	18	8	16	8	12
Cluster 7	Cluster 8	Cluster 9	Cluster 10	Cluster 11	Cluster 12
12	8	8	12	8	12

(b)

Table 3.2: The average rate of correct classifications in 10 tested sets. The average number of classes in each cluster is 87.138.

	Set 1	Set 2	Set 3	Set 4	Set 5
average rate %	94.2	93.1	94.4	94.3	95.9
	Set 6	Set 7	Set 8	Set 9	Set 10
average rate %	92.8	93.8	93.4	94.4	93.2

4. Conclusions

In this study, a new learning algorithm which can be implemented on a Kohonen neural network was developed . Under this approach, the local minimum problem encountered by the conventional winner-take-all learning algorithm can be avoided, and the overlap ratio can be reduced. We have experimented with two files. There are 400 character classes in the first file. The preclassification system achieved a 97.0% average rate of correct classifications with 2000 tested data. In the second file, there are totally 1,000 character classes and the average rate of correct classifications is 94% with 10000 tested data.

5. References

[1] K. Fukushima, "Neocognitron: a hierarchical neural network capable of visual pattern reognition," *Neural Networks*, vol. 1, pp. 119-130, 1988.

[2] J. J. Hopfield and D. W. Tank, "Neural computation of decisions in optimization problems," *Biological Cybernetics*, vol. 52, pp. 141-152, 1985.

[3] W. K. Chou and D. Y. Yun, "On hamming networks," *IJCNN*, vol. 2, pp. 594, 1989.

[4] T. Kohonen, "An introduction to neural computing," *Neural Networks*, vol. 1, pp. 3-16, 1988.

[5] J. M. Bishop and R. J. Mitchell, "Neural networks- an introduction," *IEE Colloquium on Neural Networks for Systems Principles and Applications*, pp. 11-13, Jan. 1991.

[6] Q. Zhang, Q. R. Wang and R. Boyle, "A clustering algorithm for data-sets with a large number of classes," *Pattern Recognition*, vol. 24, pp. 331-340, 1991.