COMPUTER PROCESSING OF CHINESE AND ORIENTAL LANGUAGES
Vol. 8, No. 1, June 1994, pp. 53–73 © CLCS, Inc.

Extracting Structural Features from Binary Images by Neural Networks

Rei-Yao Wu§ and Wen-Hsiang Tsai*

ABSTRACT

A neural network system for extraction of structural features from thinned binary images is proposed. The system consists of two neural networks, called the line separation network (LSN) and the feature extraction network (FEN). The LSN includes several neuron planes which classify input image lines resulting from image thinning into groups of lines in distinct directions. And the FEN constructs structural features from the lines with known directions. The LSN is a recurrent network which collects information about neighbors in wide areas by information propagation through iterations. So, even curvy lines like strokes in characters can be obtained. This in turn makes it easier for the FEN to extract features. Good experimental results show the effectiveness of the proposed neural network system.

Keywords: Neural network, feature extraction, line classification, binary image, thinning.

1. INTRODUCTION

For an image recognition application, proper selection of features is often one of the keys to the success of the recognition system. Intersection points, such as crosses, forks, corners, line ends, etc., in line images are useful for many applications, like optical character recognition, engineering drawing recognition, etc. They are especially suitable for the recognition of Chinese characters since most Chinese characters are composed of linear strokes.

Neural networks have been proved to be effective tools for pattern recognition [1-3]. In a neural network implementation of an image recognition system, the preprocessing phase is often ignored. That is, the features fed into the neural network are usually extracted by algorithmic approaches using computers, not by the neural network itself. Nevertheless, Fukushima [4] successfully derived features from binary images and recognized numeral characters using the neocognitron neural network. Graf et al. [5] used a VLSI architecture to extract features by multiplying a binary input image with a stored matrix of weights.

Extracting features by neural networks is efficient since the processing elements in a network work in parallel. In this paper, a new neural network scheme is proposed for extraction

---

Manuscript received on July 19, 1993; revised March 11, 1994. The authors are with the §Institute of Computer Science and Information Engineering, and *Department of Computer and Information Science, National Chiao Tung University, Hsinchu, Taiwan 300, Republic of China.

of the features of various line intersection and end points. The structural features which can be extracted by the proposed neural network system are shown in Fig. 1. Because all the feature points come from the intersections or ends of one or more lines, it makes sense to extract these features from lines in different directions rather than to extract them directly from original object images. Based on this principle, lines in different directions are extracted from input images first in the proposed approach. Desired structural feature points are then derived from the extracted lines.

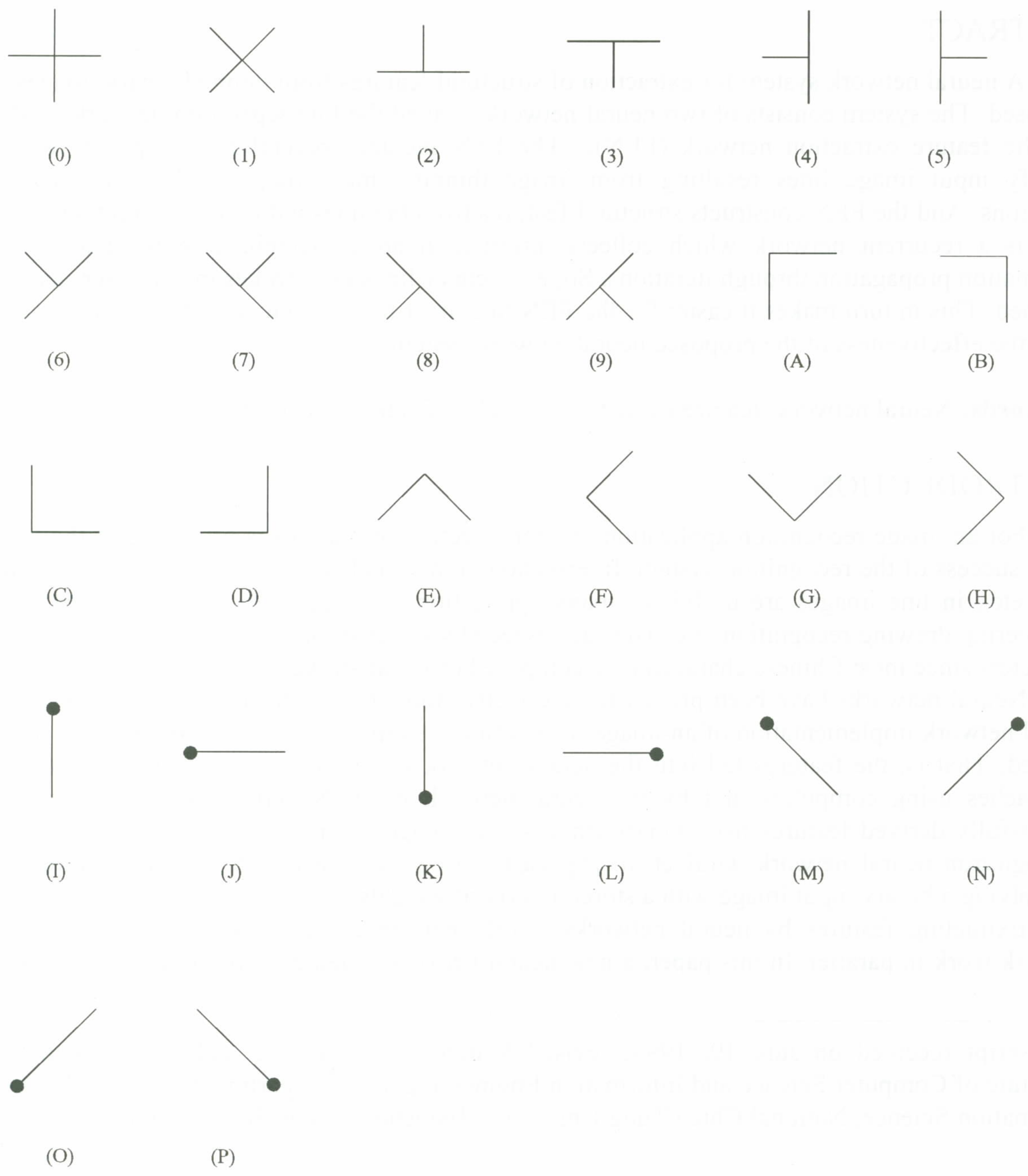


Fig. 1. The structural features which can be extracted by proposed neural network system.

One way of extracting a line from the image is to label each pixel simply by considering the condition of its local neighbors, as is done by the neocognitron [4]. A problem of this approach is that bad line continuation will be produced when pixels on a curvy line are labeled (i.e. a curvy line will be broken into several pieces after the extraction process). Low tolerance of noise or distortion in line images will result from this problem when the features are extracted. To avoid this problem, the proposed approach iteratively gather more global information to extract lines, in a manner like the relaxation algorithm [6]. Lines with good continuation (like those strokes in Chinese characters) can be obtained by this approach.

Each image processed by the proposed approach is a line pattern with single line width, which comes from the thinning result of an input binary image. A one-pass parallel thinning algorithm [7], which was proposed by the authors, shown to have good thinning effect, and implemented by a neural network [8], is used here to obtain thinned images.

As shown in Fig. 2, to systematically extract desired structural features, lines in four different directions are first identified by a neural network called line separation network (LSN). The lines derived by the LSN are then used to extract various structural features by a neural network called feature extraction network (FEN).

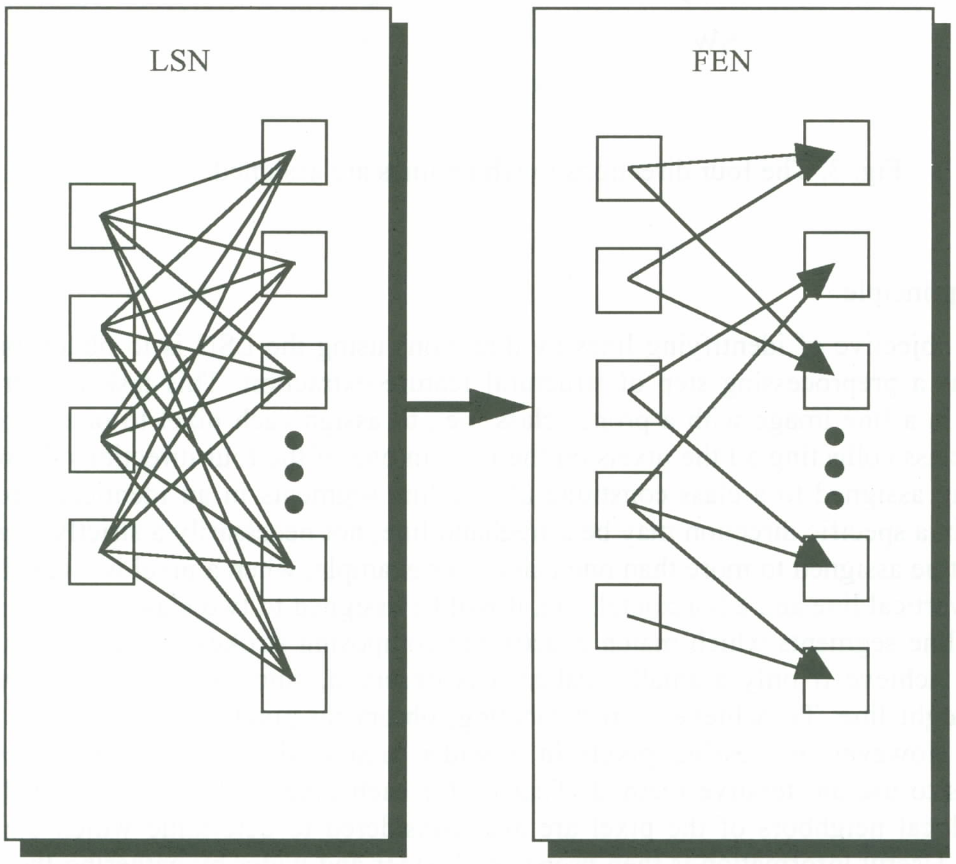


Fig. 2. Structure of the proposed neural network system.

The remainder of this paper is organized as follows. Section 2 describes the architecture of the LSN. Section 3 describes how to extract structural features by the FEN. Some examples of extracted lines and features are given in Section 4 to illustrate the effectiveness of the proposed networks. Finally, Section 5 includes some conclusion remarks.

2. LINE SEPARATION NETWORK (LSN)

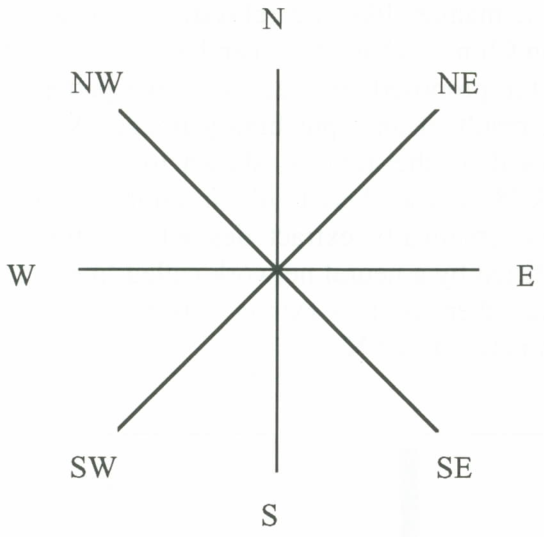


Fig. 3. The four directions to which lines are assigned.

A. Design principle

The objective of identifying lines by directions using the LSN is to obtain proper stroke segments as a preprocessing step of structural feature extraction. This task is merely to label each pixel in a line image with a proper class, i.e., to assign each pixel to one or more classes with each class collecting all the pixels on the lines in one of the four directions shown in Fig. 3. Pixels being assigned to a class constitute all the line segments in an identical direction. Note that a line in a specific direction may be a freehand line, not necessarily a strictly straight line. A pixel might be assigned to more than one class. For example, when a pixel is located at the cross point of a vertical line and a horizontal line, it will be assigned to two classes. An ideal case is to obtain the line segments which match exactly the composing strokes of a character. But this is difficult to achieve if only a small local area is observed, since a stroke is not necessarily a strictly straight line. To achieve correct labeling, observing pixels in a properly wider area is necessary. However, processing pixels in a wider area is difficult and inefficient. Another approach is to use an iterative method. That is, for each pixel to be labeled, besides the pixel itself, the local neighbors of the pixel are also considered to determine which class the pixel belongs to. Useful information is then propagated wider and wider by gathering the neighboring information iteratively to correct improper initial labeling. By this way, for example, pixels above and below a pixel on a vertical line encourage the pixel to be considered as one on the vertical line. The proposed LSN has this type of iterative pixel labeling capability.

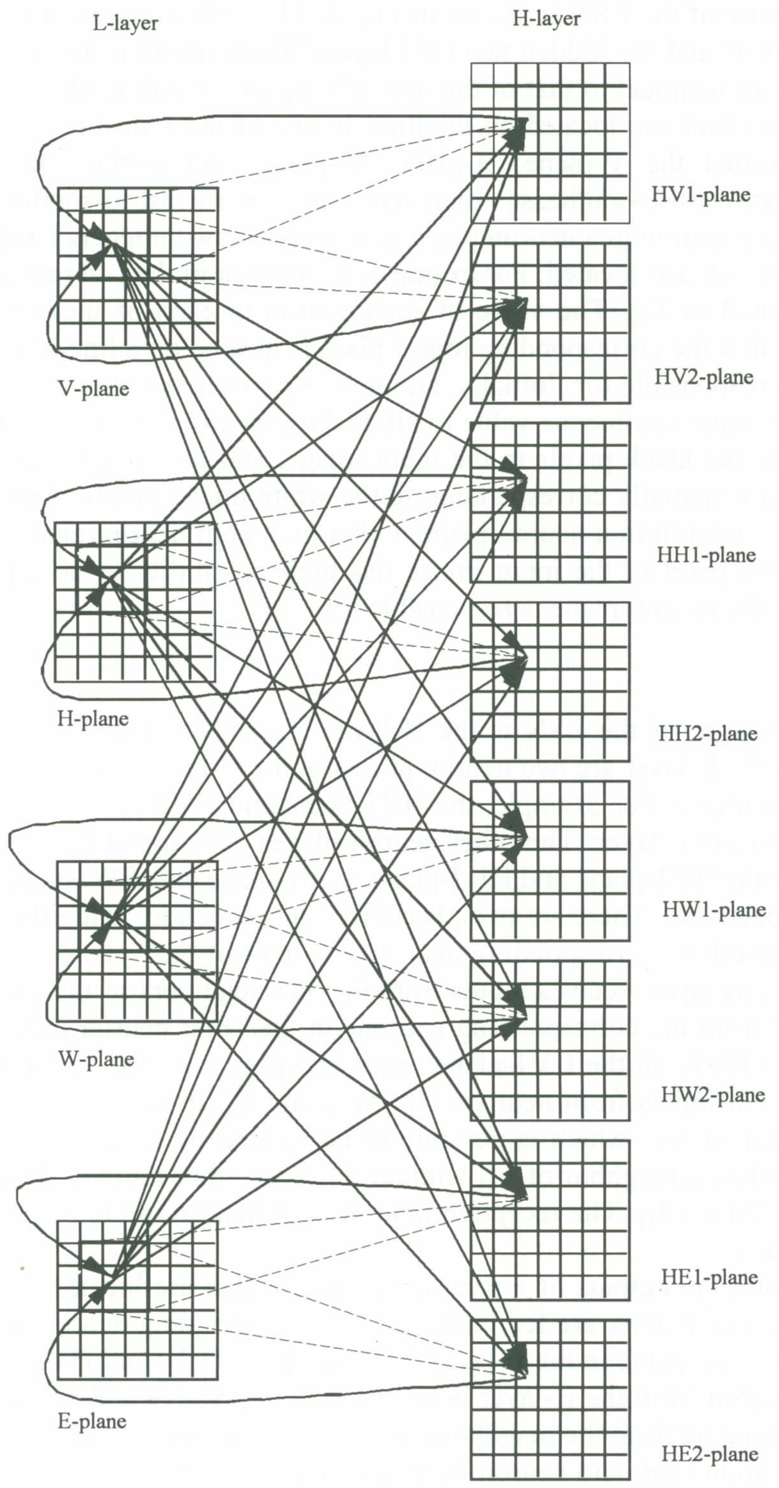


Fig. 4. Structure of the LSN.

B. L-layer

The structure of the LSN is shown in Fig. 4. The LSN consists of two layers, namely, the line layer (L-layer) and the hidden layer (H-layer). There are four neuron planes in the L-layer with their sizes all identical to that of the image being processed. Each of them is used to collect the black points which are located on the lines in one of the four directions is shown in Fig. 3. Let them be called the V-plane, H-plane, W-plane, and E-plane for identifying vertical, horizontal, northwest-to-southeast diagonal, and northeast-to-southwest diagonal lines, respectively. Each neuron in a neuron plane is indexed by two integers i and j to indicate the row and column in which it is located. For example, the neuron in the i-th row and j-th column in the V-plane is denoted as V_{ij}. The value of each neuron in each of these planes is a confidence measure of whether the corresponding image pixel is located on a line in the direction which the neuron plane is responsible for. Initially, an input line image is fed into these four neuron planes by assigning a unique confidence value to all of those neurons in each plane which are spatially corresponding to the black pixels in the input image, and the unique value of 0 to all of those neurons which are spatially corresponding to the white image pixels. After the operation of the LSN stabilizes, a neuron in a neuron plane with a high confidence value represents the fact that the corresponding pixel in the input image has high possibility to be a point on a line in the direction which the neuron plane is responsible for.

C. H-layer

The other layer of the LSN is the H-layer. As shown in Fig. 4, corresponding to each neuron plane in the L-layer are two neuron planes in the H-layer also with their sizes identical to that of the input image. For example, the HV1-plane and HV2-plane in the H-layer correspond to the V-plane in the L-layer. The other neuron planes corresponding to the H-plane, W-plane, and E-plane are the HH1-plane and HH2-plane, HW1-plane and HW2-plane, and HE1-plane and HE2-plane, respectively. There are two planes for each direction, and either of them is called the dual plane of the other. Corresponding neurons in a pair of dual planes are called dual neurons. Each neuron in this layer receives inputs from the 3x3 neighbors of the corresponding neuron in the L-layer and from the corresponding neurons in the other neuron planes of the L-layer. For example, neuron $HV1_{ij}$ in the HV1-plane receives inputs from the 3x3 neighbors of neuron V_{ij} in the V-plane and inputs from neurons H_{ij}, W_{ij}, and E_{ij} in the other neuron planes of the L-layer. A neighbor of V_{ij}, which is possible to be located on a vertical line, together with V_{ij} excites $HV1_{ij}$; while a neighbor of V_{ij} without this possibility inhibits $HV1_{ij}$. Neurons H_{ij}, W_{ij}, and E_{ij} also inhibit $HV1_{ij}$. The receptive field of each neuron in the other neuron planes can be described similarly.

Furthermore, the outputs of a pair of neurons, which are located respectively at identical positions in two dual planes, are fed back into the corresponding neuron which is located on the corresponding neuron plane in the L-layer. For example, $HV1_{ij}$ and $HV2_{ij}$ are fed back into V_{ij}. The role of a neuron M in the H-layer is to gather both positive and negative information about the confidence support from the neighbors of its corresponding neuron N in the L-layer and then feed this information back into neuron N, which, together with the information coming from the dual neuron of M, is used for the evaluation of the new confidence value of N.

D. Principle of confidence and connection weight setting

Observing the 3x3 local area around a pixel P_{ij} in the input image for the measurement of the possibility that P_{ij} is located on a line in one specific direction is the key to the evaluation of

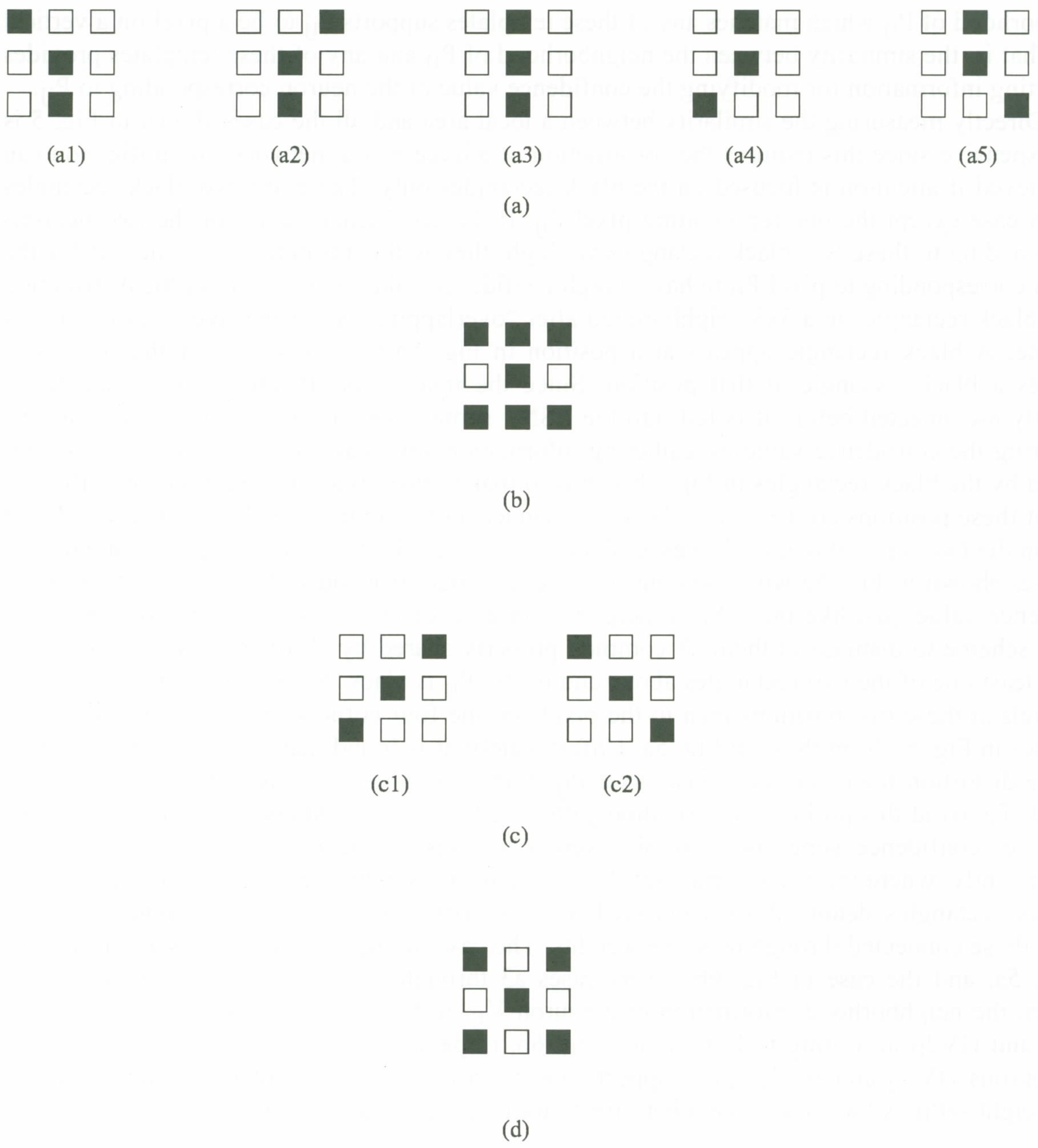


Fig. 5. Cases for the illustration of vertical line extraction. (a) The cases on a vertical line. (b) Combination of all the cases in (a). (c) Two cases not on a vertical line. (d) A third case not on a vertical line.

the previously-mentioned confidence value. In the following, confidence value modification for a pixel on a vertical line is discussed for the illustration of this concept. Fig. 5a shows all the cases in which a pixel P_{ij} may be located on a vertical line. In each case, a 3x3 local area around P_{ij} is shown, where each black rectangle denotes a pixel with a high confidence value to be on a vertical line, while each white rectangle denotes one of the other pixels. Case a3 in Fig. 5a is one which occurs on a straight vertical line. For the tolerance of handprinted vertical line distortion, four other cases (a1, a2, a4, a5) are also included. By treating these cases as templates, a neighborhood of P_{ij} which matches any of these templates supports P_{ij} to be a pixel on a vertical line. That is, the similarity between the neighborhood of P_{ij} and any of these templates provides supporting information for modifying the confidence value of the neuron corresponding to P_{ij}.

Directly measuring the similarity between a local area and all the cases shown in Fig. 5 is very expensive since this requires the construction of a huge neural network. Simplification can be achieved if attention is focused on the black rectangles only. There are two black rectangles in each case except the one representing pixel P_{ij}. If the confidence values of the two neurons corresponding to these two black rectangles are high, they will contribute strong support for the neuron corresponding to pixel P_{ij} to have a high confidence value. Fig. 5b shows the distribution of the black rectangles in a 3x3 neighborhood after "overlapping" all of the five cases of Fig. 5a into one. A black rectangle appears at a position in Fig. 5b if at least one of the five cases includes a black rectangle at that position. Since the image has already been thinned to be perfectly 8-connected before it is fed into the LSN, some cases in Fig. 5a will not coexist. So measuring the confidence value by gathering information simultaneously from all the positions marked by the black rectangles in Fig. 5b seems to make sense. But a related problem will arise if all of these positions are treated in the same manner. For example, see the neighborhood of a pixel on the two types of diagonal lines as shown in Fig. 5c. The two black neighbors in either of the cases shown in Fig. 5c will also contribute equally but erroneously to the evaluation of the confidence value, just like those black neighbors in each of the cases in Fig. 5a, if there is no proper scheme to distinguish them. A common property shared by all of the cases in Fig. 5a is that at least one of the two rectangles above and below P_{ij} is black. So, setting higher weights to the pixels at these two positions than to the pixels on the four corners can correctly distinguish the cases in Fig. 5c from those in Fig. 5a. Unfortunately, it is found that effective weight setting for line distortion tolerance will cause pixel P_{ij} in the case shown in Fig. 5d to be incorrectly labeled. To avoid this problem, information gathering from the neighbors is divided. As shown in Fig. 6, confidence supports from two weight settings of the neighborhood are gathered independently, where the black rectangles denote the neurons connected through strong weights, the gray rectangles denote those connected through weak weights, and the white rectangles denote those connected through negative weights. The case in Fig. 6a covers cases a1 through a3 in Fig. 5a, and the case in Fig. 6b covers cases a3 through a5 in Fig. 5a. By setting proper weights, the neighborhood information of a neuron V_{ij} in the L-layer is collected into neurons $HV1_{ij}$ and $HV2_{ij}$ according to Fig. 6a and Fig. 6b, respectively. A high value of either of the two neurons $HV1_{ij}$ and $HV2_{ij}$ then supports the increment of the confidence value of neuron V_{ij}. Weight settings for lines in the other directions are proceeded similarly.

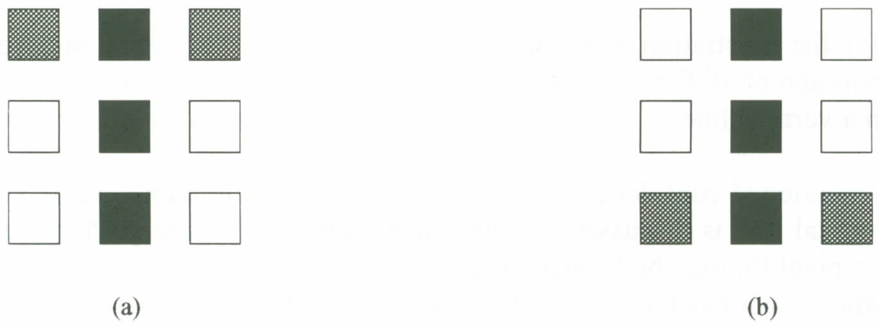
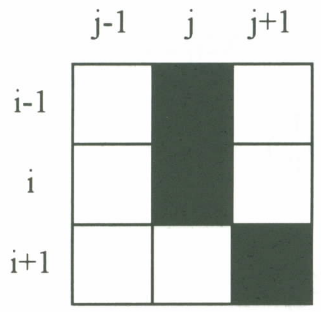


Fig. 6. Cases for vertical line extraction.

E. Detailed descriptions of weight setting

At the beginning, a neuron in the four neuron planes in the L-layer is initialized with a positive constant value c if the corresponding pixel in the image is black. Let X-plane be one of the planes in the L-layer, i.e., let X be one of V, H, W, and E. The connections from a neighbor N of a neuron X_{ij} in a X-plane in the L-layer to neurons $HX1_{ij}$ and $HX2_{ij}$ in the H-layer are assigned a strong weight W_s, a weak weight W_w, or an inhibitory negative weight W_h in the way discussed previously if both of the image pixels P_N and P_{ij} corresponding to N and X_{ij}, respectively, are black pixels in the input image. And the connections from neuron X_{ij} to $HX1_{ij}$ and $HX2_{ij}$ are assigned another strong weight W_c if P_{ij} is a black pixel. For example, assume that $P_{i-1,j}$, P_{ij}, and $P_{i+1,j+1}$ are black pixels, as shown in Fig. 7a, then the connection weights from neurons $V_{i-1,j}$, V_{ij}, and $V_{i+1,j+1}$ to neuron $HV2_{ij}$ are initialized to be W_s, W_c, and W_w, respectively, the weights from neurons $V_{i-1,j-1}$, $V_{i-1,j+1}$, $V_{i,j-1}$, and $V_{i,j+1}$ to neuron $HV2_{ij}$ are set to W_h, and the weights from neurons $V_{i+1,j-1}$ and $V_{i+1,j}$ to neuron $HV2_{ij}$ are set to 0, as shown in Fig. 7b.



(a)

	j-1	j	j+1
i-1	W_h	W_s	W_h
i	W_h	W_c	W_h
i+1	0	0	W_w

(b)

Fig. 7. Example of weight setting. (a) Local area around pixel P_{ij} in the image. (b) The weights from neurons around neuron V_{ij} to $HV2_{ij}$.

In addition to the inputs from the neighborhood of X_{ij}, neurons $HX1_{ij}$ and $HX2_{ij}$ also receive inputs from the corresponding neurons in the other neuron planes in the L-layer with negative connection weights. For example, neuron $HV1_{ij}$ receives inhibitory inputs from neurons H_{ij}, W_{ij}, and E_{ij}. These inhibitory inputs reduce the confidence value of a neuron in the L-layer if the pixel corresponding to the neuron is not located on a line in the correct direction. The value of a neuron in the L-layer will converge to either the high value 1 or the low value 0 after a sufficient number of iterations.

In summary, the net input $HX1^I_{ij}$ of a neuron $HX1_{ij}$ in the H-layer is

$$HX1^I_{ij} = \sum_{\substack{i-1\le k\le i+1 \\ j-1\le l\le j+1}} WX^1_{ijkl} * X^o_{kl} + \sum_{Y\neq X} W_b * Y^o_{ij} \qquad (1)$$

where X denotes one of the symbols V, H, W, or E; WX^{1}_{ijkl} is the weight (one of W_c, W_s, W_w, and W_h mentioned previously) from neuron X_{kl} to neuron $HX1_{ij}$; X^{o}_{kl} is the output of neuron X_{kl}; Y is one of the symbols V, H, W, or E, and is not X; W_b is the inhibitory connection weight from the corresponding neuron in each of the other neuron planes; and Y^{o}_{ij} is the output of Y_{ij}. The net input $HX2^{I}_{ij}$ of neuron $HX2_{ij}$ can be obtained similarly as follows:

$$HX2^{I}_{ij} = \sum_{\substack{i-1\le k\le i+1\\ j-1\le l\le j+1}} WX^{2}_{ijkl} * X^{o}_{kl} + \sum_{Y\neq X} W_b * Y^{o}_{ij} \tag{2}$$

where WX^{2}_{ijkl} is the weight from neuron X_{kl} to neuron $HX2_{ij}$.

The outputs $HX1^{o}_{ij}$ and $HX2^{o}_{ij}$ of neurons $HX1_{ij}$ and $HX2_{ij}$ are

$$HX1^{o}_{ij} = F_H(\, HX1^{I}_{ij}\,) \tag{3}$$

and

$$HX2^{o}_{ij} = F_H(\, HX2^{I}_{ij}\,), \tag{4}$$

respectively, where F_H is a nonlinear function described by

$$F_H(x) = \begin{cases} 1 & \text{if } x \ge 1; \\ 0 & \text{if } x \le 0; \\ x & \text{otherwise.} \end{cases}$$

Neuron X_{ij} in the L-layer receives inputs from neurons $HX1_{ij}$ and $HX2_{ij}$ in the H-layer. Its output X^{o}_{ij} is

$$X^{o}_{ij} = F_L\,(\, HX1^{o}_{ij}, HX2^{o}_{ij}\,), \tag{5}$$

where F_L is a function described by

$$F_L(x1, x2) = \begin{cases} x1 & \text{if } x1 \ge x2; \\ x2 & \text{otherwise.} \end{cases}$$

3. FEATURE EXTRACTION NETWORK (FEN)

A. Design principle

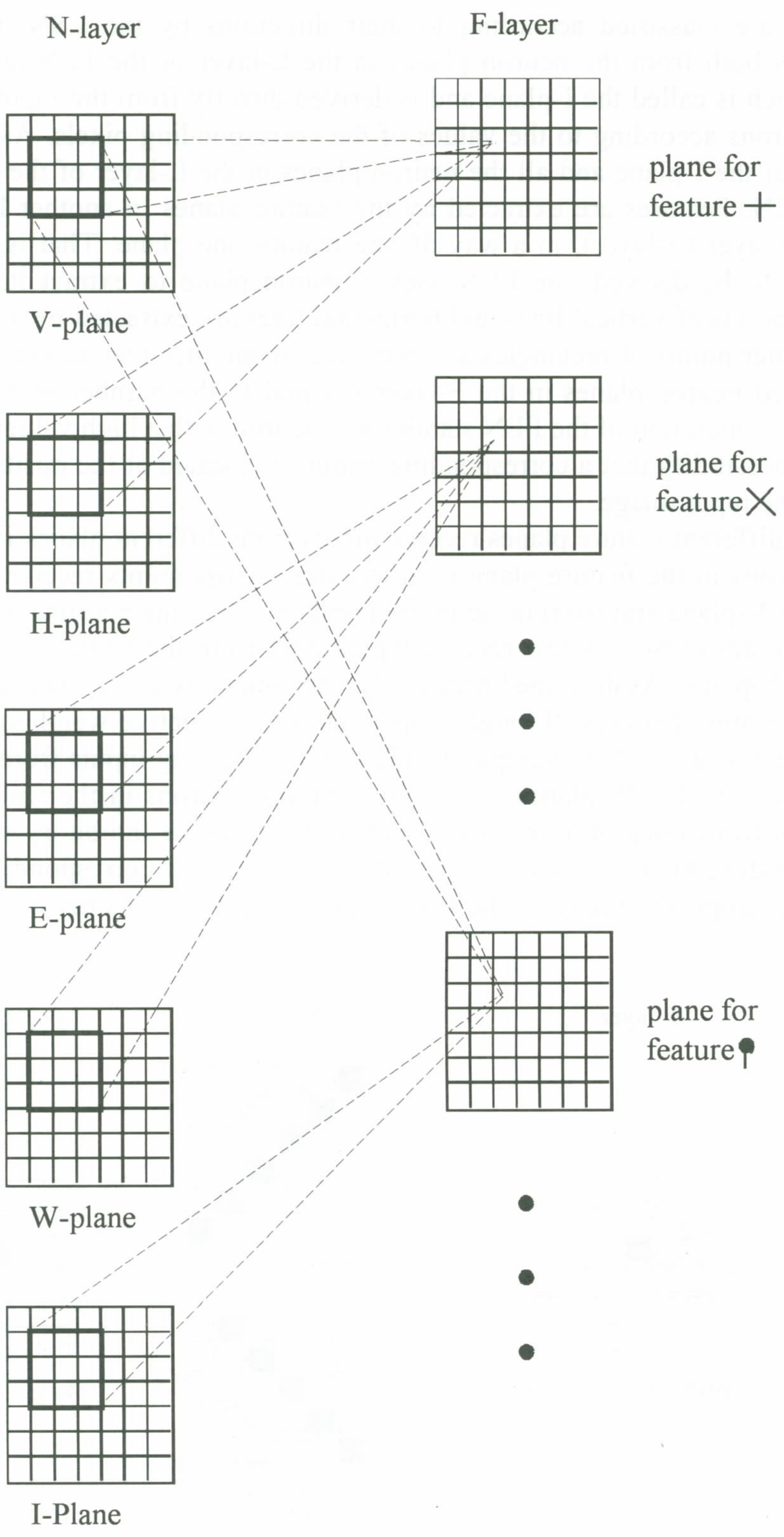


Fig. 8. Structure of FEN.

After lines are classified according to their directions by the LSN, the FEN extracts structural features both from the neuron planes in the L-layer of the LSN and from an image neuron plane, which is called the I-plane and is derived directly from the input image by setting the values of neurons according to the values of the corresponding pixels. As shown in Fig. 8, the combination of the I-plane and all the neuron planes in the L-layer of the LSN is called the N-layer in the FEN. Features are extracted by the feature planes in another layer of the FEN, called the feature layer (F-layer), in a way of one feature one plane. That is, for each type of structural feature to be derived, the FEN uses a neuron plane to extract it. Specifically, the orthogonal cross points of vertical lines and horizontal lines are extracted by a neuron plane, and the upper-left corner points of rectangles are extracted by another feature plane, and so on. The number of required neuron planes in the F-layer is equal to the number of feature types to be extracted. After the operation of the FEN stabilizes, a neuron with a high value in a feature plane implies the high possibility that a corresponding feature is located at the position corresponding to the neuron in the input image.

Neurons in different feature planes receive inputs from different planes in the N-layer. For example, the neurons in the feature plane used to extract cross points receive inputs both from the neurons in the V-plane and from those in the H-plane, while the neurons in the feature plane used to extract crosses of St. Andrew receive inputs both from the neurons in the W-plane and from those in the E-plane. As described previously, the I-plane is also included in the N-layer. It is used to inhibit simple features (through proper setting of negative weights) when a complex feature exists at that location. For example, in addition to receiving inputs from a line plane (i.e., from one of the V-, H-, E-, W-planes in the N-layer), the neurons in the feature plane used to extract end points from lines in a specific direction also receive inhibitory inputs from the I-plane to avoid yielding high values in the outputs of the neurons corresponding to fork points. Fig. 9 gives two examples to illustrate the feature extraction functions performed by the feature planes.

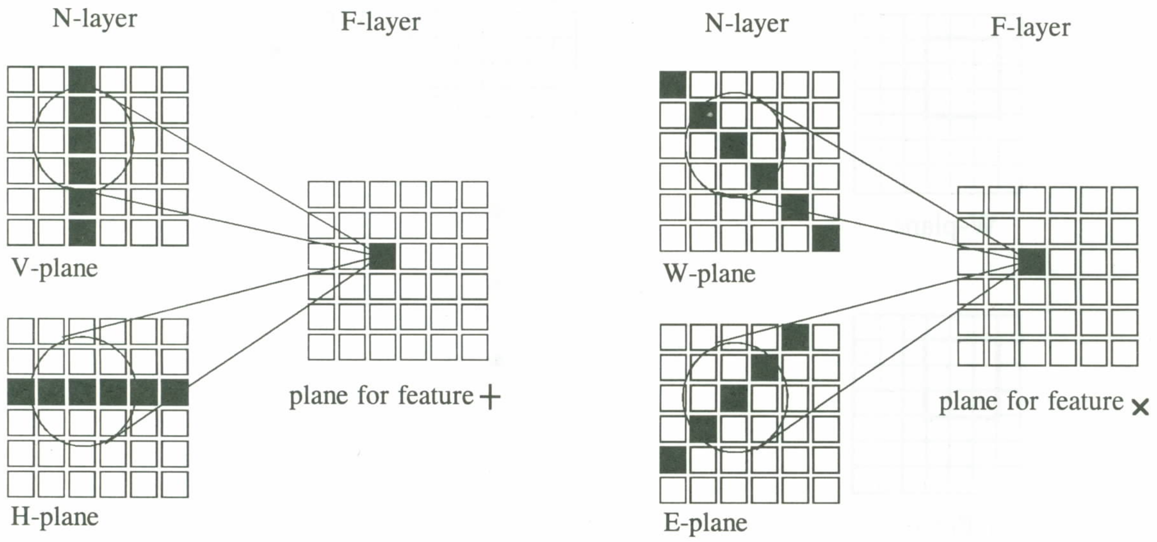


Fig. 9. Illustration of feature extraction. (a) Extraction of feature '+'. (b) Extraction of feature 'x'.

B. Connection weight setting

A neuron in a feature plane in the F-layer receives inputs from the 3x3 neighborhoods of the corresponding neurons in two or more neuron planes in the N-layer. For a neuron R_{ij} in a feature plane R in the F-layer which receives inputs from a set of neuron planes S in the N-layer, its net input R_{ij}^{I} is

$$R_{ij}^{I} = \sum_{X \in S} \sum_{\substack{-1 \le k \le 1 \\ -1 \le l \le 1}} WRX_{kl} * X_{i+k,j+l}^{o} \tag{6}$$

where X is a neuron plane in S; $X_{i+k,j+l}^{o}$ is the output of neuron $X_{i+k,j+l}$; and WRX_{kl} is the connection weight from neuron $X_{i+k,j+l}$ to neuron R_{ij}.

The output of neuron R_{ij} is

$$R_{ij}^{o} = F_r (R_{ij}^{I}) \tag{7}$$

where F_r is a nonlinear function described by

$$F_r(x) = \begin{cases} 1 & \text{if } x \ge 1; \\ 0 & \text{if } x \le 0; \\ x & \text{otherwise.} \end{cases}$$

In addition to the selection of the neuron planes in the N-layer which should be connected to each neuron plane in the F-layer, another important issue in the extraction of a specific type of feature is the setting of the connection weights from the neurons in the N-layer to the neurons in the F-layer. The output value of a neuron in a specific feature plane in the F-layer indicates the possibility that a corresponding feature point exists at that location, so all the neurons in a feature plane are treated identically except their locations. That is, the setting of the connection weights of the receptive fields is treated identically for all the neurons in a neuron plane in the F-layer. For each type of structural feature, the setting of the connection weights depends on the spatial location of the input neurons in the 3x3 receptive field as well as the plane they are located in. Actually, this weight setting reflects the importance of the existence of a line point at that location. Examples of the connection weights from neurons in a local area of the neuron planes in the N-layer to a neuron R_{ij} in the F-layer are shown in Table 1.

4. EXPERIMENTAL RESULTS

The proposed neural networks have been implemented by software simulation. And experiments have been conducted to see the effects of the proposed neural networks for line identification and feature extraction. Images fed into the networks are of dimension 32x32. The initial confidence value c is 0.25 for all neurons in the L-layer corresponding to the black pixels in the image. The weights from the L-layer to the H-layer are 0.55, 0.51, 0.25, -0.21, and -0.05 for W_c, W_s, W_w, W_h, and W_b, respectively. Fig. 10 shows the results of line classification for a Chinese character obtained by the LSN after 12 iterations. In the figure, symbol '.' marks the object skeleton point where the confidence value of its corresponding neuron is smaller than 0.1; the decimal symbol '1' shows the confidence value of a neuron for the value range 0.1 to 0.2,

Table 1. Weights from the N-layer to a neuron R_{ij} in the F-layer (the heading of each of columns 3 through 11 specify the indices of a neuron in the N-layer).

Feature	From	i-1,j-1	i-1,j	i-1,j+1	i,j-1	i,j	i,j+1	i+1,j-1	i+1,j	i+1,j+1
+	V	0.09	0.18	0.09	-0.18	0.18	0.00	0.09	0.15	0.09
	H	0.09	-0.18	0.09	0.15	0.18	0.18	0.09	0.00	0.09
×	W	0.20	0.10	0.00	0.10	0.10	0.10	0.00	0.10	0.20
	E	0.00	0.10	0.20	0.10	0.10	0.10	0.20	0.10	0.00
⊥	V	0.00	0.20	0.00	-0.20	0.20	-0.20	-0.20	-0.30	-0.20
	H	0.00	0.00	0.00	0.20	0.20	0.20	0.10	0.00	0.10
Y	W	0.20	0.00	0.00	0.00	0.20	-0.20	0.00	-0.20	-0.30
	E	0.00	0.00	0.20	0.00	0.20	0.10	0.20	0.10	0.00
┌	V	-0.50	-0.50	-0.50	-0.50	0.00	0.00	0.00	0.50	0.00
	H	-0.50	-0.50	0.00	-0.50	0.00	0.50	-0.50	0.00	0.00
∧	W	-0.30	-0.30	-0.30	-0.30	0.30	0.30	-0.30	-0.15	0.30
	E	-0.30	-0.30	-0.30	-0.15	0.30	-0.30	0.30	0.15	-0.30
⫯	V	0.00	0.00	0.00	0.00	0.50	0.00	0.25	0.50	0.25
	I	-0.75	-0.75	-0.75	-0.50	0.00	-0.50	-0.25	0.00	-0.25
•╲	W	0.00	0.00	0.00	0.00	0.50	0.25	0.00	0.25	0.50
	I	-0.75	-0.75	-0.50	-0.75	0.00	-0.25	-0.50	-0.25	0.00

symbol '2' for the range 0.2 to 0.3, and so on. It can be seen from the figure that line continuation is preserved for each line in each of the four directions. Also, high confidence values are possible to exist at the same position in two or more neuron planes. For example, the cross points have confidence values greater than 0.9 both in the V-plane and the H-plane. Presence of a feature can be obtained by thresholding the output values of the F-layer. Fig. 11 shows the features extracted from the neuron planes shown in Fig. 10. Instead of showing all the feature planes, a plane consisting of all the features is shown. In the figure, symbol '.' is used

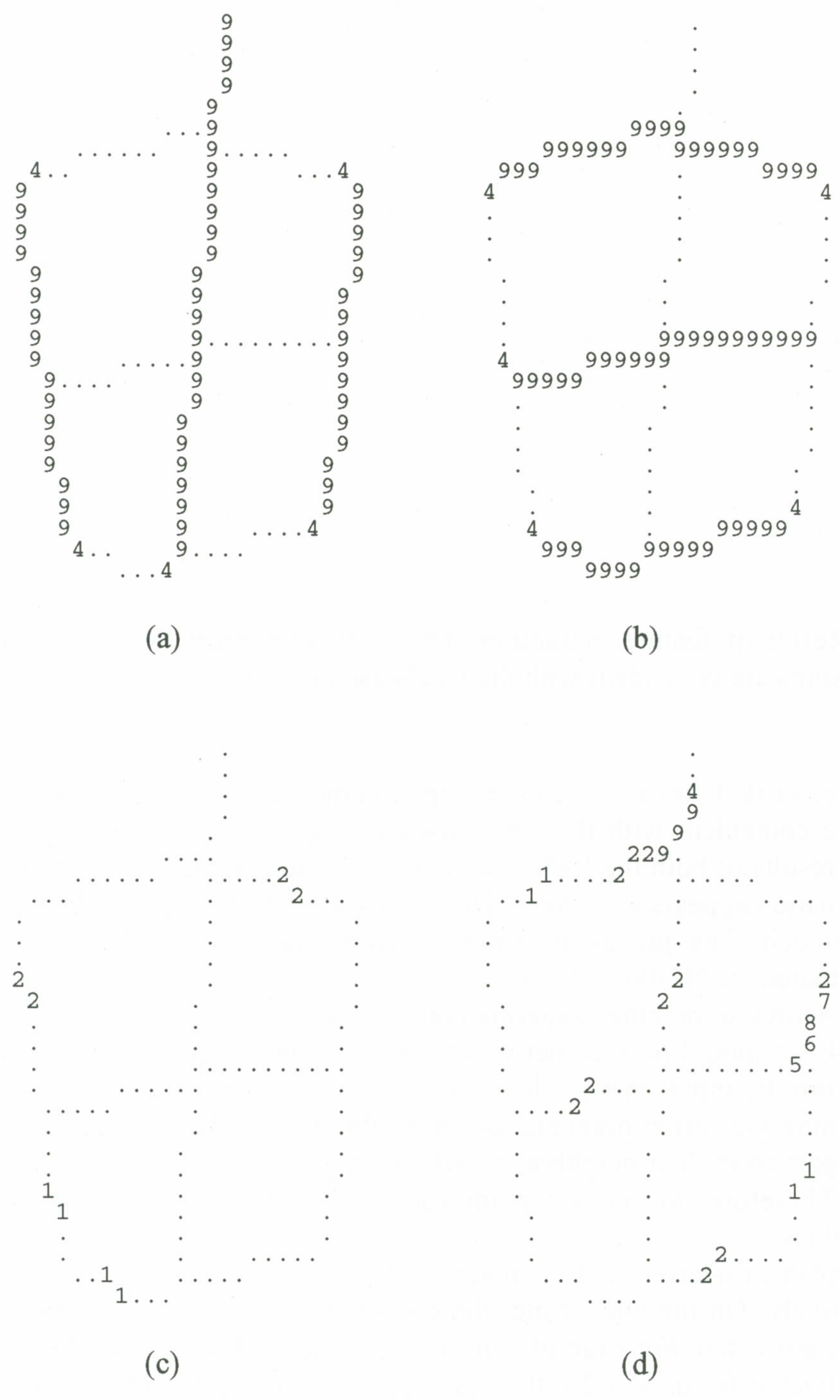


Fig. 10. Results of line separation. (a) Vertical lines obtained by the V-plane. (b) Horizontal lines obtained by the H-plane. (c) Diagonal lines obtained by the W-plane. (d) Diagonal lines obtained by the E-plane.

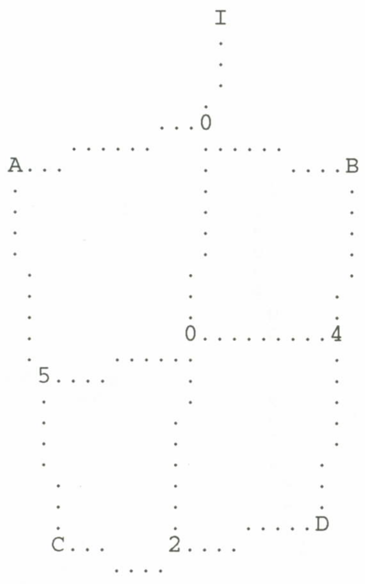


Fig. 11. Result of feature extraction. The alpha-numerical symbols used to mark feature points are coincident with the labels used in Fig. 1.

to mark the object skeleton again, and the alpha-numerical symbols are used to mark the feature points and are coincident with the labels used in Fig. 1. Fig. 12 and Fig. 13 show two other experimental results of both the LSN and the FEN. Note that feature points are marked correctly though deformation appears near them. This shows the effectiveness of the iteration nature of the proposed approach. The processing time for each character is 7.1 seconds when the neural system is simulated on PC 486 DX-33.

Fig. 14 shows some other experimental results of the FEN. The pixel marked by symbol "*" in Fig. 14 (v) should be a corner point, while the proposed neural system cannot extracted this feature from the input image. The occurrence of this error is due to the fact that the stroke segments forming the corner point are too short. Points on a short stroke segment cannot receive sufficient support from their neighbors to achieve high confidence values in any neuron plane in the L-layer. Therefore, the corner point cannot be extracted from the neighbors with low confidence values.

The number of neurons and the number of connections required in the LSN are $12N^2$ and $80N^2$, respectively. On the other hand, the size of the FEN depends on the number of features expected to be extracted. For example, in the experiments described in the previous section, the number of extracted features is 26; then the number of neurons and the number of connections required are $(26+5)N^2$ and $26\times18N^2$, respectively.

5. CONCLUSIONS

A neural network system for structural feature extraction has been proposed in this paper. The system consists of two neural networks, namely, the LSN and the FEN. The LSN classifies

```
        9
        9
        9   ....
     ...9...
  ...   9
        9
         9  ...4
       ..9..    9
    4..  9      9
   9     9      9
   9     9  ....9
   9   ..9..    9
   9...  9      9
   9     9 ...  9
   9   ..9.   .4    9
    4..  9          9
         9  ..9.....9
   ......9..  9     9
...           9
            ..9....
       .....  9    ..
 ......       9
        .     9
         ..  9
           . 9
             9
        .    9
         ..  9
           .3
```

(a)

```
        .
        .
        .   9999
     9999999
  999   .
        .
         .  9999
       99999    4
    999  .      .
   4     .      .
   .     .  99999
   .   99999    .
   9999  .      .
   .     . 999  3
   4   9999   99    .
    999  .          .
         .  999999999
   999999999  .     .
999           .
            9999999
       99999  .    99
 999999       .
        2     .
         77  .
           2 .
             .
        4    .
         99  3
           99
```

(b)

```
        .
        .
        .   ....
     .16....
  ...   8
        9
         9  ...1
       ...93    1
    ...  .      .
   .     .      .
   .     .  .....
   .   .....    .
   ....  .      .
   .     . ..2  .
   .   ....   2.     .
    ...  .          .
         .  .........
   .........  .     .
...           .
            .....14
       .....  .    41
 ......       .
        9     .
         99  .
           9 .
             .
        9    .
         99  .
           93
```

(c)

```
        .
        .
        .   2...
     1.....2
  ..2   .
        .
         .  2...
       2...2    .
    ..2  .      .
   .     .      .
   .     .  41...
   .   2..14    .
   ...2  .      .
   .     . 94.  1
   .   2..9   .1    .
    ..2  9          .
         8  .........
   2...16...  1     .
..2           4
            65.....
       2..15  .    ..
 .....2       .
        .     2
         ..  2
           . .
             .
        .    .
         ..  1
           .1
```

(d)

```
        I
        .
        .   ...L
     ...0...
  J..   .
        .
         .  ....B
       ..0..    .
   A...  .      .
   .     .      .
   .     .  ....4
   .   ..0..    .
   5...  .      .
   .     . ...  .
   .   ..0.   ..D   I
   C...  .          .
         .  ..3.....4
   ......2..  .     .
J..           .
            ..0....
       .....  .    .L
 J.....       .
        M     .
         ..  .
           P .
             .
        M    .
         ..  .
           ..D
```

(e)

Fig. 12. Features and lines extracted by the proposed approach. (a) Vertical lines obtained by the V-plane. (b) Horizontal lines obtained by the H-plane. (c) Diagonal lines obtained by the W-plane. (d) Diagonal lines obtained by the E-plane. (e) Features extracted by the FEN.

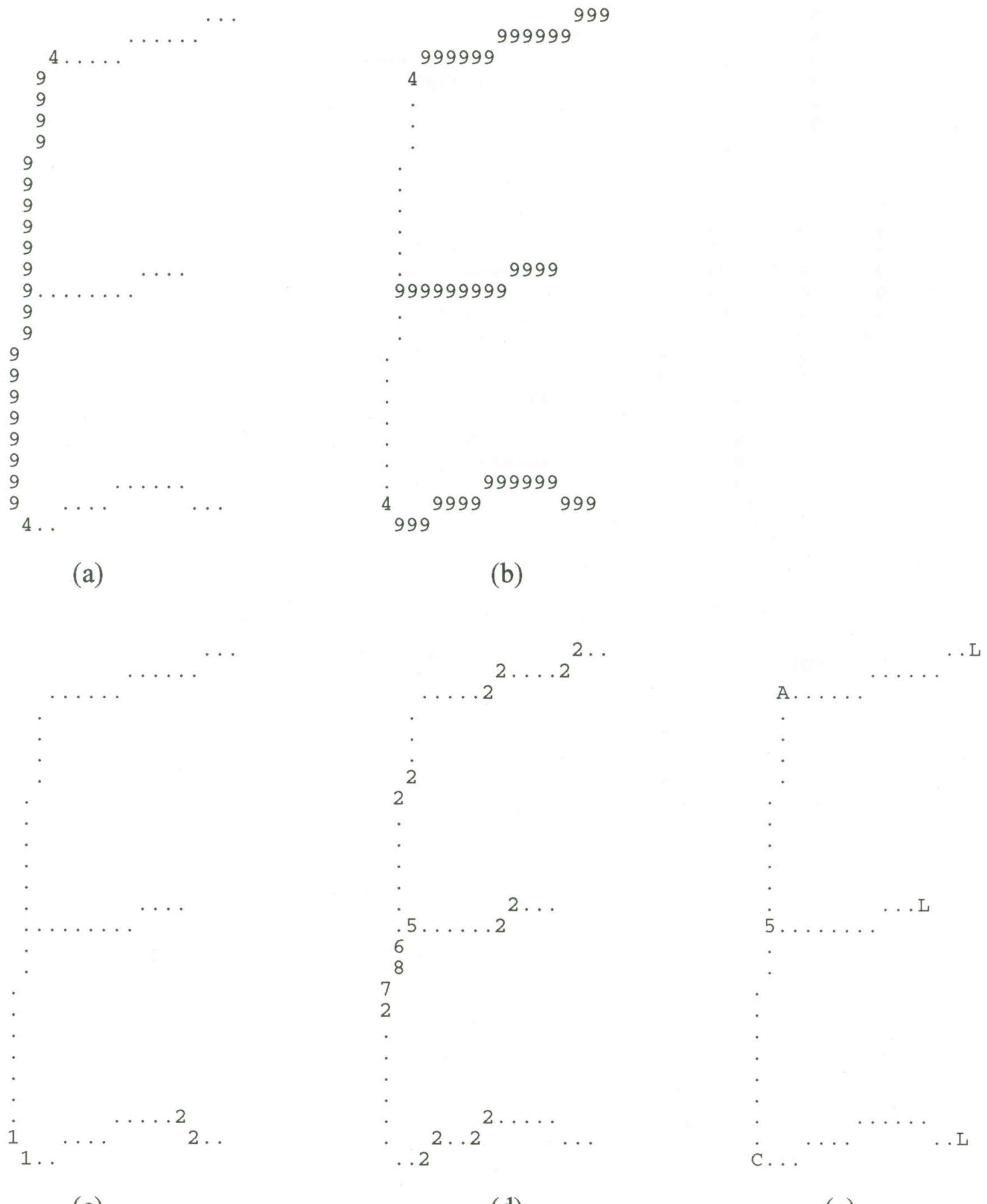


Fig. 13. Features and lines extracted from the English character "E". (a) Vertical lines obtained by the V-plane. (b) Horizontal lines obtained by the H-plane. (c) Diagonal lines obtained by the W-plane. (d) Diagonal lines obtained by the E-plane. (e) Features extracted by the FEN.

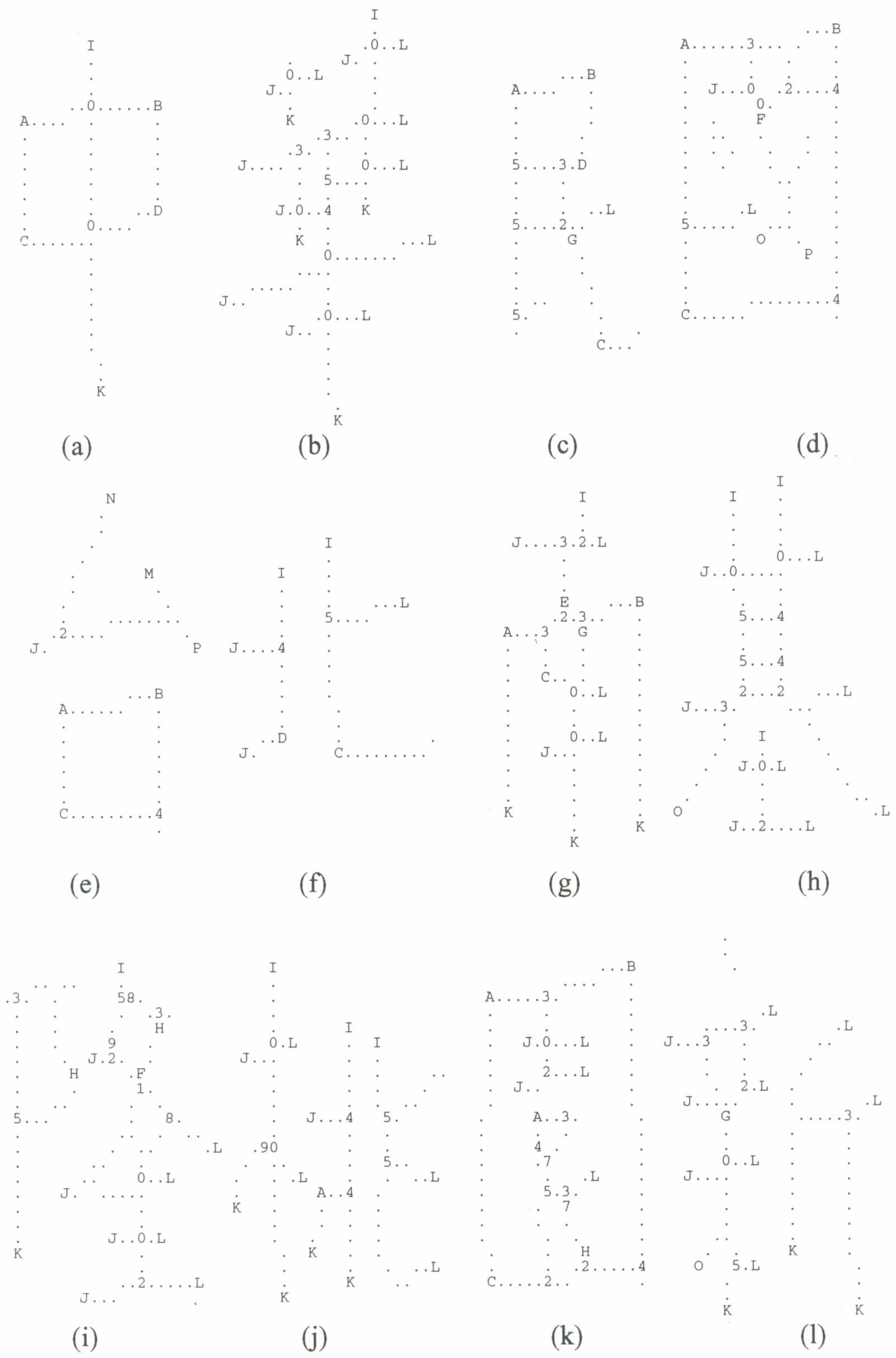


Fig. 14. Features extracted from Chinese characters.

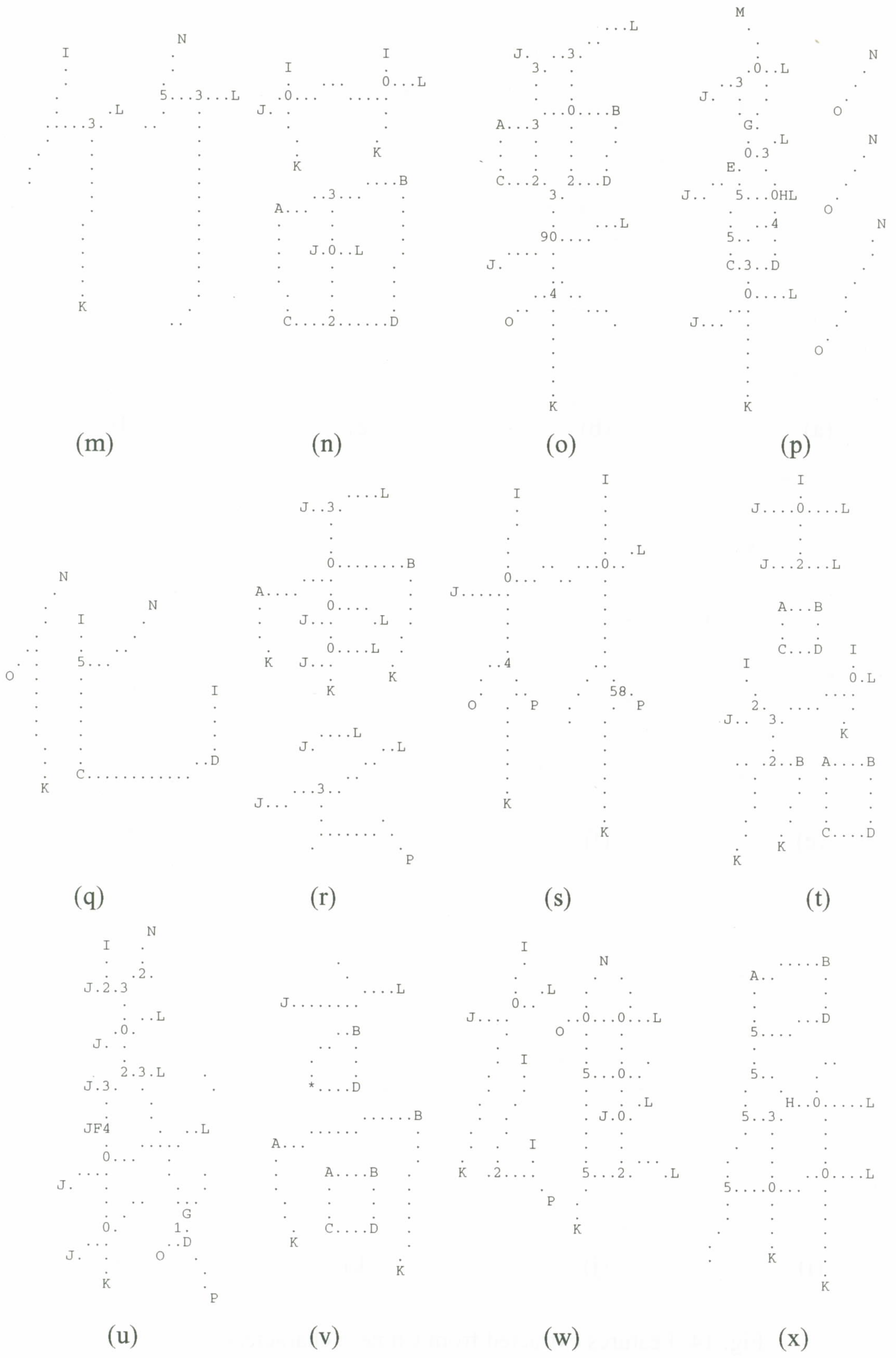


Fig. 14. Features extracted from Chinese characters (continued).

black pixels into different classes according to the directions of the lines they are located on. And the FEN extracts structural features from the results of the LSN.

Instead of deriving features directly from the image plane, the proposed scheme first identifies pixels on lines in specific directions using the LSN. Like the manner of a relaxation algorithm, the LSN propagates the supporting information along lines to tolerate line deformation. The intersection point of two lines can thus be successfully labeled by activating the neurons in the planes responsible for both lines regardless of the disturbance from each other. On the other hand, features cannot be correctly extracted by observing the local area around the feature points if deformation appears near the feature points. After the discovery of the directions of the lines in which pixels are located, this problem can be solved by considering the directions of the line patterns, and this is what the FEN performs. In short, the directions of lines derived by the LSN make it easier for the FEN to extract features.

Structural features are useful for further analysis or recognition of images, for example, for optical character recognition (OCR). More specifically, the extracted structural features is very useful for the recognition of Chinese characters since most Chinese characters are formed by line strokes which construct the structural features discussed in this paper. Recognition of image contents based on the structural features derived by this scheme is worthwhile for further studies. Furthermore, the lines extracted according to their directions by the LSN are also good features for image analysis. This is also a good topic for future investigations.

REFERENCES

[1] D. E. Rumelhart, and J. L. McClelland, Parallel Distributed Processing: Explorations in The Microstructure of Cognition, Volume 1, MIT Press, Cambridge, MA, 1986.

[2] G. A. Carpenter, and S. Grossberg, A Massively Parallel Architecture for A Self-organizing Neural Pattern Recognition Machine. *Computer Visions, Graphics, and Image Processing* 37, pp. 54-115, 1987.

[3] D. L. Reilly, L. N. Cooper, and C. Elbaum, A Neural Model for Category Learning, *Biological Cybernetics* 45, pp. 35-41, 1982.

[4] K. Fukushima, Neocognitron: A Self-organizing Neural Network Model for A Mechanism of Pattern Recognition Unaffected by Shift in Position, *Biological Cybernetics* 36, pp. 193-202, 1980.

[5] H. P. Graf, L. D. Jackel, and W. E. Hubbard, VLSI Implementation of a Neural Network Model, *IEEE Computer,* pp. 41-48, 1988.

[6] A. Rosenfeld, and A. C. Kak, Digital Picture Processing, Volume 2, Academic Press, New York, 1982.

[7] R. Y. Wu, and W. H. Tsai, A New One-Pass Parallel Thinning Algorithm for Binary Images, *Pattern Recognition Letters* 13, pp. 715-723, 1992.

[8] R. Y. Wu, and W. H. Tsai, A Single-Layer Neural Network for Parallel Thinning, Internal Journal of Neural Systems, Vol. 3, No. 4, pp. 395-404, 1992.