Multi-font Alpha-numeric Recognition using Multi-layer Neural Networks with a Rejection Function

Keiji YAMADA
C&C Information Technology Research Laboratories, NEC Corporation
1-1, Miyazaki 4chome, Miyamae-ku, Kawasaki, 216, Japan
yamada@pat.cl.nec.co.jp

Abstract

A new method for pattern recognition using multi-layer neural networks with a rejection function is described. The system has two components: (1) STELA (STandstill Evading Learning Algorithm), a new learning algorithm that extends back propagation by adding a technique to avoid learning standstill states, and (2) the use of a rejection function which estimates the uncertainty of the classification, and rejects the pattern as unclassifiable if the network response is too uncertain. The ability of the method to recognize deformed and noisy patterns is demonstrated in the domain of multi-font alpha-numeric character recognition. The learning algorithm achieves perfect recognition for the training data. In order to realize a rejection function, an expression for recognition uncertainty is proposed, and a notion of consistency for this function is defined. Using consistency of uncertainty as a criterion, network responses to patterns that make the uncertainty expression inconsistent cause the pattern to be rejected. Without rejection, STELA can achieve about a twelfth as high an error rate for a multi-font alpha-numeric recognition as a typical back propagation. Furthermore, when STELA and back propagation are augmented by a rejection function, STELA needs to reject only half as many patterns as a network trained by back propagation in order to achieve a 0% error rate. Finally, a conventional rejection method is compared to the one derived here, and found to require twice as many rejections to achieve the same error rate.

Keywords: Multi-layer neural network, Character recognition, Rejection function, Multi-font alpha-numerics.

1 Introduction

There has been considerable recent interest in the use of Multi-Layer Neural Networks (MLNN) and back propagation learning for pattern recognition[1]-[8], in particular for character recognition. In some cases, the image data is presented without preprocessing, in others, features are extracted from raw image data and presented to the network. In both cases, the MLNN works as a template matching classifier that has good pattern recognition accuracy when pattern variation is smaller within classes than between classes.

Pattern recognition performance should be discussed from two view points, tolerance for unlearned geometrical deformation and tolerance for meaningless noise superimposed on patterns. Handwritten digit recognition is often discussed as an example where new deformations should be correctly classified. Usually, many kinds of deformed digits are learned and then a digit pattern with a shape which has never been learned must be recognized. As discussed in [8], a simple MLNN often does not perform correctly on novel shapes, even when all of the training patterns can be recognized. For this task, an MLNN should have a special structure so that spatial fluctuation of local features might be absorbed.

Multi-font character recognition can be viewed as an example of meaningless noise superimposed upon normal patterns. Here, character patterns of all the fonts are learned previously and noisy input patterns should be recognized accurately. An MLNN is expected to have a good performance for such a task.

For most practical applications, however, it is important for the classifier to signal when it cannot classify an input with certainty. The goal of this paper is to show how a network can achieve perfect performance on a training set and then achieve a 0% error rate (misclassification rate) on novel patterns by rejecting some as unclassifiable. Back propagation learning cannot be used to achieve perfect performance on the training set, since it often reaches what may be termed a "learning standstill state" (different from a local minima - see below). We have proposed an improved learning algorithm called STELA (STandstill Evading Learning Algorithm) which avoids this state. Second, when an input pattern is too noisy to be recognized exactly, the recognition process should not be forced to output an answer. The answer is uncertain. In the case where the answer is too uncertain, the recognition process should output a rejection signal rather than a category name. This rejection function increases the reliability of the pattern recognition system.

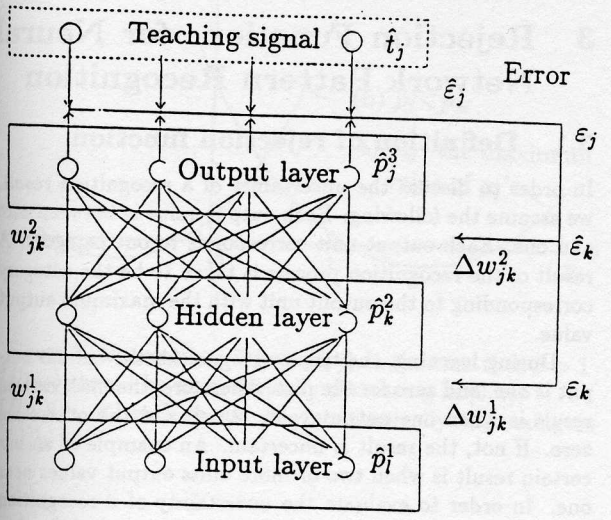

Figure 1: Multilayer neural network conceptual diagram

This paper describes multi-font alpha-numeric recognition performance of an MLNN with consideration of the two kinds of functions above mentioned. At first, the STELA method is briefly described. Next an uncertainty expression is defined and criteria for consistency of the behavior of this expression are proposed. Finally, experimental results of using the learning algorithm and the rejection function for pattern recognition of multi-font alpha-numerics are presented which support the utility of these mechanisms.

2 Improved Back Propagation Algorithm

2.1 Multi-layer neural network

The typical neural network used for pattern recognition is a three-layer network as shown in Figure 1. The pattern to be classified is presented as input, activation propagates forward through the network, activating the output units. There is an output unit for each category. The output unit with the highest activation is taken as the category to which the pattern belongs.

The feedforward process and the learning process are described below. A unit's activation value is the result of passing the weighted sum of its inputs through a nonlinear sigmoid function. Thus the output value $\hat{p}_j^i$ on the j-th unit on the i-th layer is given as

$$\hat{p}_j^i = f(\sum_{k=1}^{N_{i-1}} w_{jk}^i \hat{p}_k^{i-1} + \theta_j^i) \ , \tag{1}$$

where $\hat{p}_k^{i-1}$ is an output value on the k-th unit on the $(i-1)$-th layer and w_{jk}^i is a weighting value between $\hat{p}_j^i$ and $\hat{p}_k^{i-1}$. θ_j^i is a bias term for the j-th unit on the i-th layer. $f(x)$ is a sigmoid function, denoted as

$$f(x) = \frac{1}{2}(1 + \tanh \frac{x}{u_0}) \ , \tag{2}$$

where u_0 is a parameter for controlling the sigmoid function slant. An output value from a sigmoid function is between one and zero.

During learning, weights are changed according to the back propagation learning algorithm [10], which minimizes mean-squared error. Output unit values are calculated according to the feedforward process. The error at the output is computed as a difference of presented and expected signal for each unit on the output layer (the m-th layer):

$$\hat{\varepsilon}_j^m = \hat{t}_j - \hat{p}_j^m \ , \tag{3}$$

where $\hat{t}_j$ is a target value. A back propagated error value is calculated as

$$\varepsilon_j^i = \frac{2}{u_0}\hat{\varepsilon}_j^i \hat{p}_j^i (1 - \hat{p}_j^i) \tag{4}$$

and

$$\hat{\varepsilon}_k^{i-1} = \sum_{j=1}^{N_i} \varepsilon_j^i w_{jk}^i \ . \tag{5}$$

Weights and bias terms are corrected by:

$$\begin{aligned} \Delta w_{jk}^i &= \alpha \varepsilon_j^i \hat{p}_k^{i-1} \\ \Delta \theta_j^i &= \beta \varepsilon_j^i \ , \end{aligned} \tag{6}$$

where α and β are learning coefficients.

Using equations (4),(5),(6) repeatedly, weights at lower layers are updated.

2.2 The STELA Algorithm

In preliminary experiments using back propagation learning, the network often reached a learning standstill state [9]. In this state, weights do not change, as if the network were in a local minimum, even when the $|\hat{\varepsilon}_j^m|$'s are large. This state, however, is not a local minimum. The standstill state is due to the slope term in the back propagation Eq.(4). When a unit is near its maximum or minmum value, ($\hat{p}_j^i$ is near one or zero), the slope is near zero, so the error value ε_j^i in Eq.(4) is near zero and the weight change term Δw_{jk}^i is near zero. Weights are not corrected in this state. This also occurs if all $\hat{p}_k^{i-1}$'s are near zero. We term such a state a learning standstill.

In order to evade the learning standstill state, the slant parameter is controlled. That is, u_0 in Eq.(2) is enlarged so that output values are far from one and zero, as shown Figure 2. This can be performed by STELA (STandstll Evading Learning Algorithm) [8]. The algorithm is described below.

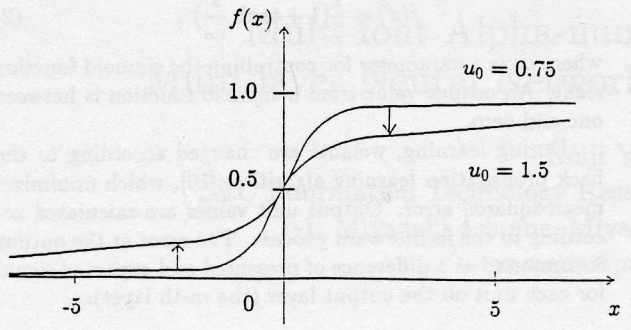

Figure 2: Slant controll using sigmoid slant paramete

(1) Detecting a learning standstill state

A1. A "serious" error is judged using the condition

$$|\hat{\varepsilon}_j^m| > \delta \tag{7}$$

A2. A network weight change is too small if:

$$\sum_{j,k} |\Delta w_{jk}^i| < \eta \tag{8}$$

In order to reduce the amount of computation, we actually use the following condition instead of A2:

$$\sum_{j,k} |\hat{p}_k^{i-1} \hat{p}_j^i (1 - \hat{p}_j^i)| < \xi \tag{9}$$

This clearly captures the zero-slope and the zero-output conditions characteristic of the learning standstill state.

(2) Evading a learning standstill state

A3. If the conditions in Eqs.(7) and (9) are satisfied, u_0 is enlarged and all output values are calculated again. Otherwise, go to A5.

A4. The judgements are again carried out from A1.

A5. Weighting values are corrected according to Eqs.(4),(5), and (6). After that, the initial value for the slant is set again to u_0.

3 Rejection Function for Neural Network Pattern Recognition

3.1 Definition of rejection function

In order to discuss the uncertainty of a recognition result, we assume the following: Each output value is between zero and one. Each output unit corresponds to one category. A result of the recognition process is taken to be the category corresponding to the output unit with the maximum output value.

During learning, the teaching signal for the correct output is one, and zero for the rest. Therefore, the most certain result is when one output value is one and the others are zero. If not, the result is uncertain. An example of an uncertain result is when two or more units output values near one. In order to evaluate the uncertainty of a recognition result, the following expression for uncertainty is defined.

$$R = \sum_i (1 - p_i) p_i + \beta (\sum_i p_i - 1)^2 , \tag{10}$$

where p_i is an output value of the i-th unit on the output layer.

This function is non-negative when the $p_i's$ are between 0 and 1, and when β is non-negative. The left-hand term is minimized when all units are near values of 1 or 0. The right hand term is minimized when all of the output units' activations sum to 1.

If the uncertainty is larger than a threshold value, the recognition result should be rejected because it is not reliable. However, we will use a slightly different criterion for rejection.

3.2 Consistency of the uncertainty

The stability of the uncertainty degree defined above is discussed, appropriate parameter values of β in Eq.(10) are derived, and a rejection threshold is estimated.

We would like the parameter β to be such that the following consistency criteria are obeyed.

Criterion 1 Uncertainty should decrease as the maximum-valued output, p_M, increases:

$$R(p_1, \cdots, p_M, \cdots, p_N) > R(p_1, \cdots, p_M + \Delta p, \cdots, p_N)$$

Criterion 2 When p_j is not the maximum value, uncertainty should decrease as p_j decreases:

$$R(p_1, \cdots, p_j, \cdots, p_N) > R(p_1, \cdots, p_j - \Delta p, \cdots, p_N)$$

where Δp is a small positive value.

The consistency of the uncertainty defined in Eq.10 is assured if it satisfies these criteria.

In order to consider the behavior of the uncertainty degree R concerning an output value, p_i, Eq.(10) is arranged as follows:

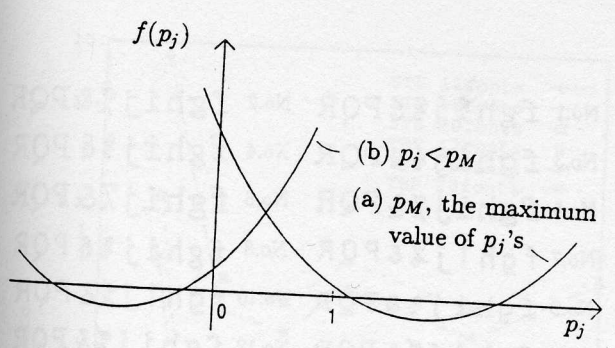

Figure 3: Uncertainty degree $f(p_j)$ in the case that $\beta > 1$ ($f(p_M)$ should be monotonically decreasing in $[0,1]$). ($f(p_j)(p_j < p_M)$ should be monotonically increasing in [0,1]).

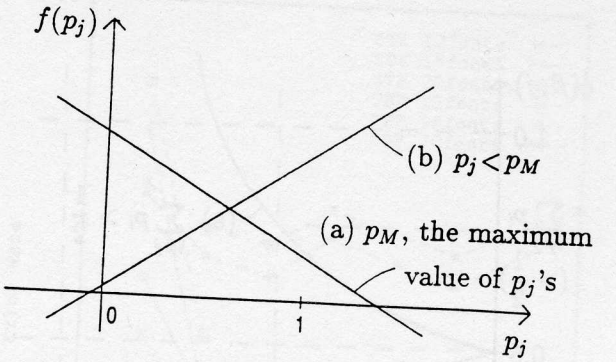

Figure 4: Uncertainty degree $f(p_j)$ in the case that $\beta = 1$ ($f(p_M)$ should be monotonically decreasing in $[0,1]$). ($f(p_j)(p_j < p_M)$ should be monotonically increasing in [0,1]).

$$R = f(p_j) = (\beta - 1)p_j^2 + (1 + 2\beta D)p_j + C + \beta D^2, \quad (11)$$

where $C = \sum_{i \neq j}(1 - p_i)p_i$ and $D = \sum_{i \neq j} p_i - 1$.

This equation will be discussed in the following three cases.

i) $\beta > 1$

If $j = M$, that is, p_j is the maximum value, $f(p_M)$ should decrease as p_M increases. Therefore, the x-coordinate of the minimum of the quadratic function must be larger than one, as shown in Figure 3(a). By taking the partial derivative of f with respect to p_j, setting it to negative, we can derive the following condition:

$$\sum_{i \neq M} p_i < \frac{1}{2\beta} . \quad (12)$$

If $j \neq M$, a following condition is derived from Criterion 2 as:

$$\sum_{i \neq j} p_i > 1 - \frac{1}{2\beta} . \quad (13)$$

ii) $\beta = 1$

In this case, f is linear, (see Figure 4). If $j = M$, the following condition is derived as above:

$$\sum_{j \neq M} p_i < \frac{1}{2} . \quad (14)$$

And the condition for $p_j (\neq p_M)$ is :

$$\sum_{i \neq j} p_i > \frac{1}{2} \qquad (j \neq M). \quad (15)$$

iii) $0 < \beta < 1$

The preceding analysis is difficult to carry over to this case. $f(p_j)$ is shown in Figure 5. The coordinate q of the

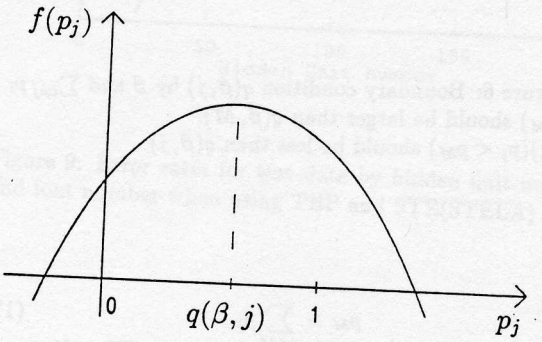

Figure 5: Uncertainty degree $f(p_j)$ in the case that $0 < \beta < 1$
($f(p_M)$ should be larger than $q(\beta, M)$.
($f(p_j)(p_j < p_M)$ should be less than $q(\beta, j)$.

peak along the p_j axis is :

$$q(\beta, j) = \frac{1 + 2\beta(\sum_{i \neq j} p_i - 1)}{2(1 - \beta)} . \quad (16)$$

Satisfaction of the criteria depends on whether p_j value is larger than $q(\beta, j)$. When $q(\beta, j)$ is regarded as a function of β, rough shapes of $q(\beta, j)$ are shown in Figure 6. Criteria 1 and 2 yield the conditions: $p_M > q(\beta, M)$ and $p_j < q(\beta, j)$ $(j \neq M)$, respectively.

Let us discuss p_M with respect to β and $\sum_{j \neq M} p_j$ in referencing Figure 6.

If β is near zero, p_M must be larger than 0.5. If β is a large value near one, a condition is given as $\sum_{j \neq M} p_j < \frac{1}{2}$. This includes the case that $\beta \geq 1$. If β is about 0.5, p_M must satisfy the following condition:

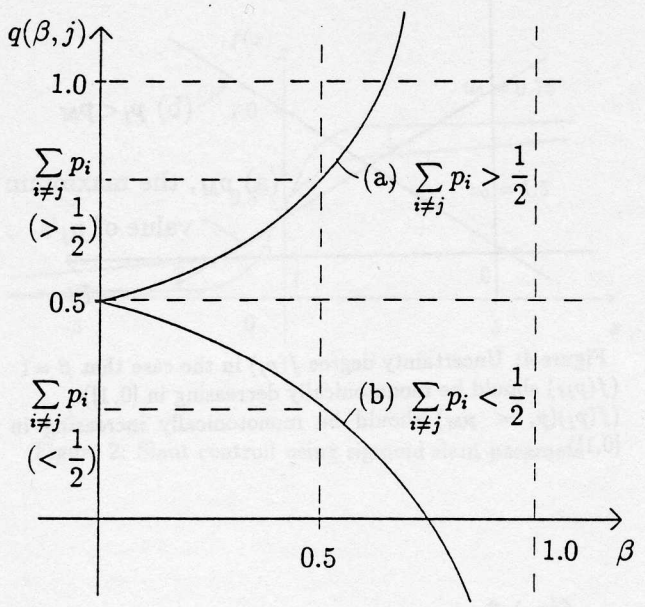

Figure 6: Boundary condition $q(\beta, j)$ by β and $\sum_{i\neq j} p_i$
$(f(p_M)$ should be larger than $q(\beta, M)$.
$(f(p_j)(p_j < p_M)$ should be less than $q(\beta, j)$.

$$p_M > \sum_{j\neq M} p_j \ . \tag{17}$$

On the other hand, p_j $(j \neq M)$ must satisfy $p_j < q(\beta, j)$. If β is near zero, p_j must be smaller than 0.5 independently of a value of $\sum_{i\neq j} p_i$. If β is about one, the condition for satisfying Criterion 2 is that $\sum_{i\neq j} p_i > \frac{1}{2}$. If β is about 0.5, the condition is that:

$$p_j < \sum_{i\neq j} p_i \qquad (j \neq M) \ . \tag{18}$$

This is always satisfied because $p_j < p_M \leq \sum_{i\neq j} p_i$.

Based on the above discussion, the case that β is about 0.5 is better than the cases that β is near zero or larger than about one, because the consistency criteria are satisfied under less strict conditions than in the other cases. In the following sections, β is set to be 0.5.

The boundary condition in the case that $\beta = 0.5$ is given from Eq.(17) as:

$$p_M = \sum_{j\neq M} p_j \ . \tag{19}$$

The substitution of Eq.(19) into Eq.(10) yields

$$BC = \frac{1}{2} + {p_M}^2 - \sum_{j\neq M} {p_j}^2 \ . \tag{20}$$

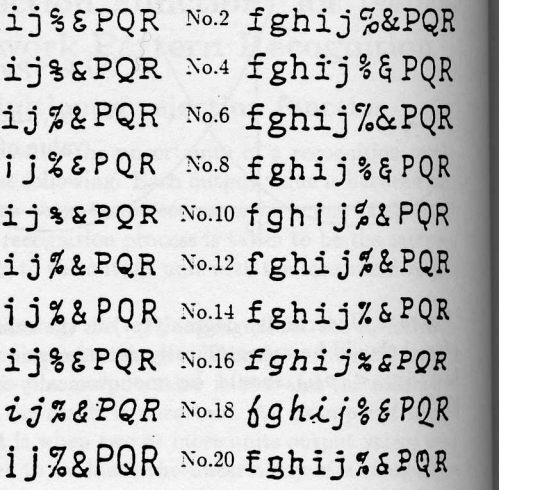

Figure 7: Samples of 20 fonts
(No.1 ~ No.12 are used for 12 font recognition)
(No.1 ~ No.16 are used for 16 font recognition)
(No.1 ~ No.20 are used for 20 font recognition)

The sum of the second term and the third term gives a small positive value. Any value of R below .5 is thus guaranteed to be consistent. Therefore, R is constrained to be smaller than $\frac{1}{2}$ and the rejection threshold of R is set to be smaller than 0.5 .

4 Experiments of Muli-font Alpha-numeric Recognition

4.1 Alpha-numeric characters and neural network model for experiments

In order to experiment on alpha-numeric recognition using MLNN (Multi-layer neural network) with a rejection function, twenty kinds of alpha-numeric fonts are used, as shown in Figure 7. The number of categories is 74. The same category name is given to a pair which cannot be classified, such as O(ou) and 0(zero), l(el) and 1(one), l(el) and I(ai without serif), and '(apostrophe) and ,(comma). Twelve fonts from No.1 through No.12 in Figure 7 are used for twelve font recognition experiments. Sixteen fonts through No.16 and twenty fonts through No.20 are used for the sixteen font recognition and twenty font recognition. Ten characters of each category and each font are typed and scanned with a 240dpi density scanner. Half of them are used for training and the other are used for recognition tests. A scanned binary character image is transformed into a 9 by 11 pixel gray level image using a Gaussian filter. In these experiments, a size normalization is not operated.

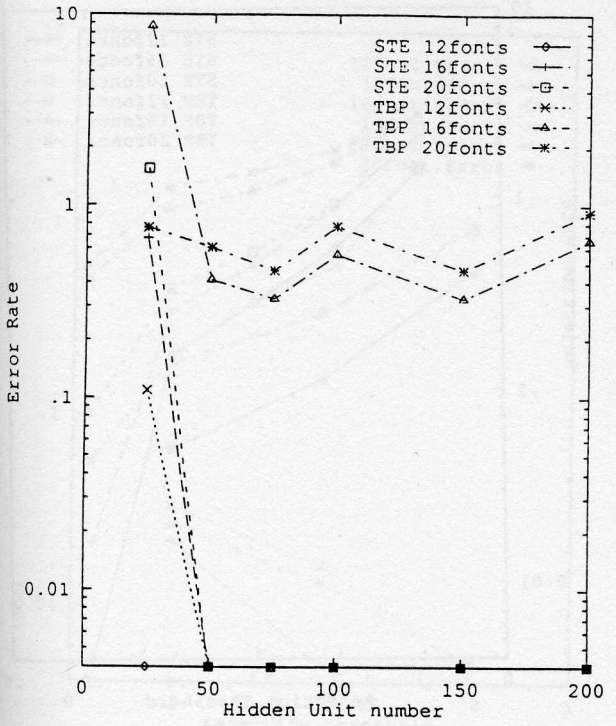

Figure 8: Error rates for training data by hidden unit number and font number when using TBP and STE(STELA)

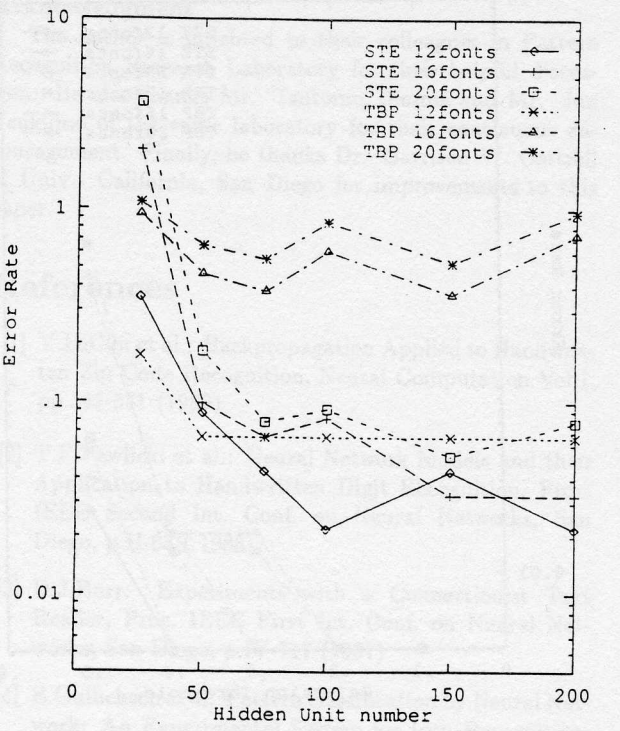

Figure 9: Error rates for test data by hidden unit number and font number when using TBP and STE(STELA)

As a neural network model, a globally connected feed-forward model with one hidden layer is used. From results of preliminary experments, a globally connected model and a locally connected model were found to have similar recognition performances for multi-fonts character recognitions. A locally connected model is a model whose hidden unit is connected to units in a local area of an input layer. The number of hidden units is 25, 50, 75, 100, 150, and 200. TBP (typical Back Propagation) and STELA (STandstill Evading Learning Algorithm) are used for training.

4.2 Recognition experiments without rejection

The recognition rate for the training data and the test data are investigated in each case of twelve fonts, sixteen fonts, and twenty fonts.

The results are shown in Figure 8 and Figure 9. In a case of twelve font trained by TBP, all of the traing data are exactly recognized. However, in cases that more fonts are trained by TBP, all of the trained data cannot be recognized. On the other hand, in a case of every font set trained by STELA, all of the trained data can be recognized. This means that STELA can evade a learning stand still to finish a training when a neural network has hidden units enough to represent input information.

When twenty-five hidden units are used, neither of TBP and STELA can recognize all of the training data. In this case, the recognition rate by STELA is worse than that by TBP. This is because STELA will not let a neural network state stay in a local minimum and will continue to work till finding a solution where all training data can be recognized.

In the case of TBP, error rates for test data are almost the same as those for training data. They are about 0.5%. In these errors, 'm', 'g', and 's' are mis-recognized as 'w', '9', and '5'. And many other characters are not recognized. In the case of STELA, the error rates are about 0.04% and much lower than those of TBP. A misrecognized character is 'i' that is recognized as '1', because a dot in 'i' is connected to a line owing to a noise.

4.3 Recognition experiments with rejection

The rejection rate are examined using Eq.(10) with $\beta = 0.5$. As a rejection threshold value, 0.1, 0.3,and 0.5 are used. The error rates by the threshold values and the rejection

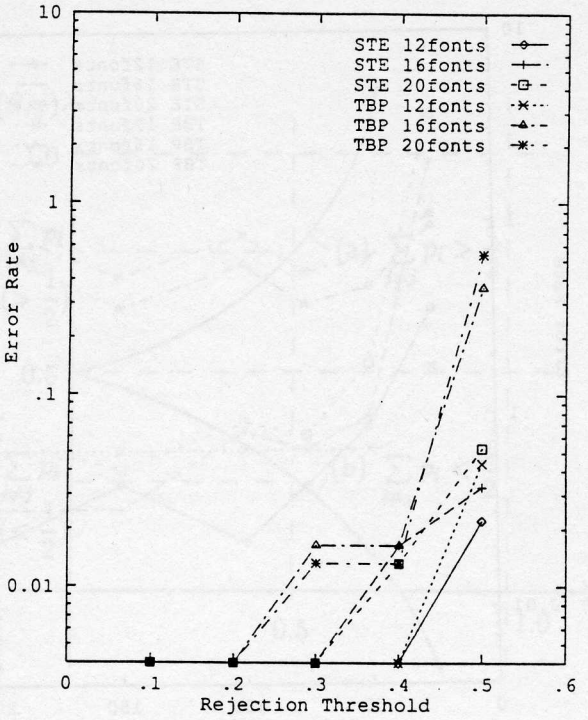

Figure 10: Error rates for test data by rejectoin threshold values when using uncertainty degree

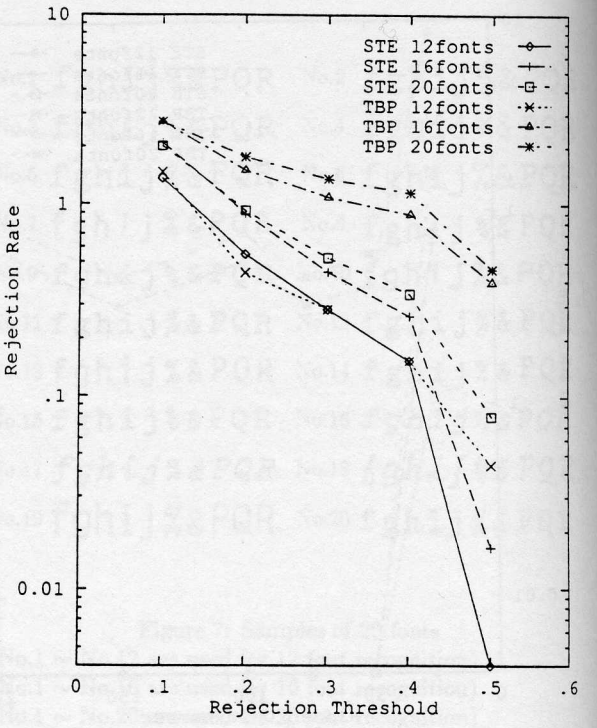

Figure 11: Rejection rates for test data by rejectoin threshold values when using uncertainty degree

rates by the threshold values are shown in Figure 10 and Figure 11.

In the sixteen font recognition, STELA and TBP achieve the no-error recognition when threshold values are 0.3 and 0.2, respectively. In the twenty font recognition, almost the same results are obtained.

Under the condition where an error rate is zero, a rejection rate of STELA is about 0.5% and one of TBP is about 1%. Totally, a rejection rate of TBP is larger than that of STELA. This shows that STELA can perform a learning with less ambiguity than TBP.

The performance of the proposed rejction method is compared with that of a conventional method. As a conventional rejection method, the method using the difference between the primary candidate and the second candidate is used. In this method, the maximum value and the second maximum value are selected among output values. Then the difference between them are calculated. If the difference is smaller than a threshold value, a recognition result is rejected.

Figure 12 shows error rates and rejection rates for each font set in the case of using STELA. Under the zero error rate condition, the rejection rates for sixteen fonts and twenty fonts are about 1% to be twice as large as those by the proposed rejection method.

This conventional method uses only the primary candidate and the second candidate and ignores the others. However, the third candidate and the fourth candidate must influence the assurance of a recognition result often. For example, assume that the maximum value is 0.8 and the second value is 0.5, the assurance in the case where the third value is 0.2 should be better than that in the case where it is 0.5. Therefore, the proposed method can reduce a rejection rate under the condition that an error rate is zero by means of calibrating a rejection threshold value because it uses all the output values.

5 Concluding Remarks

This paper has described a multi-font alpha-numeric recognition using a multi-layer neural network. As a learning algorithm, TBP (typical back propagation) and STELA(STandstill Evading Learning Algorithm) are used. And it has proposed the formula to define the uncertainty of recognition results and the rejection method using the uncertainty degree. The rejection method is compared with a conventional rejection method.

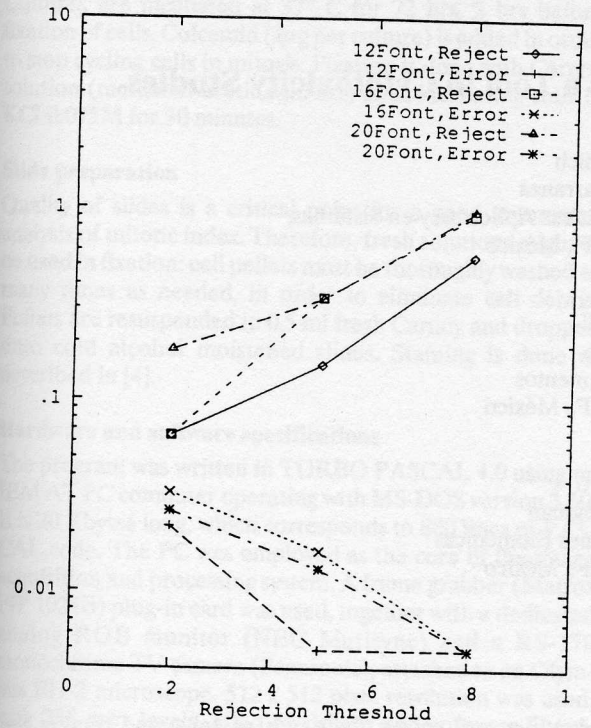

Figure 12: Error rate and rejection rate by threshold value when using the differeence between the primary candidate and the second candidate

In a case where TBP is used for learning, all of a training data set cannot be recognized. STELA can achieve 100% recognition rate for a training set and a twelfth as high a error rate for a test data set as TBP. These results mean that it is necessary for a multi-font character recognition that a training set should be recognized completely. Furthermore, the minimum hidden unit number in the cases where STELA can recognize all of training data is nearer to the optimal number than that of TBP.

Rejection experiments showed that under the condition where error rate is 0%, the rejection rate by STELA is about 0.5% and half of the rejection rate by TBP. This means that STELA has the ability to reduce the ambiguity of the discrimination function around category boundaries.

The rejection rate by the proposed method is half as high as that by a conventional rejection method. It is not only by using the maximum value and the second value but also by using the other values that the proposed rejection method can decrease a rejection rate without increasing a error rate.

This rejection method is applicable to other kinds of pattern recognition. It will be applied to a hand written digit recognition technique, in which error rate should be reduced to be as small as possible.

Acknowledgment

The author is indebted to their colleagues in Pattern Recognition Research Laboratory for their helpful discussion. He also thanks Mr. Tsutomu Temma and Mr. Jun Tsukumo of the same laboratory for their continuous encouragement. Finally, he thanks Dr. Garrison W. Cottrell of Univ. California, San Diego for improvements to this paper.

References

[1] Y.LeCun et al.: Backpropagation Applied to Handwritten Zip Code Recognition, Neural Computation Vol.1, pp.541-551 (1989)

[2] T.F.Pawlicki et al.: Neural Network Models and their Application to Handwritten Digit Recognition, Proc. IEEE Second Int. Conf. on Neural Networks, San Diego, p.II-63 (1988)

[3] D.J.Burr: Experiments with a Connectionist Text Reader, Proc. IEEE First Int. Conf. on Neural Networks, San Diego, p.IV-717 (1987)

[4] E.Gullichsen et al: Pattern classification by Neural Network: An Experimental System for Icon Recognition, Proc. IEEE First Int. Conf. on Neural Networks, San Diego, p.IV-725 (1987)

[5] T.Kohonen: The Neural Phonetic Typewriter, IEEE Computer, vol.21, No.3, pp.11-22 (1988)

[6] S.E.Troxel et al.: The Use of Neural Networks in PSRI Target Recognition, Proc. IEEE Second Int. Conf. on Neural Networks, San Diego, p.I-593 (1988)

[7] A.Rajavelu, et al.: A neural Network Approach to Character Recognition, Neural Networks, Vol.2, pp.387-393 (1989)

[8] K.Yamada, et al.: Handwritten Numeral Recognition by Multi-layered Neural Network with Improved learning Algorithm, Proc. IJCNN 89, pp.II259-II266 (1989)

[9] K.Yamada: Improved Learning Algorithm for Multilayer Neural Networks and Handwritten Numeral Recognition, NEC R&D, No.98, pp.81-88 (1990)

[10] D.E.Rumelhart et al.: Parallel Distributed Processing, Vol.1, p.322, The MIT Press (1986)