A fuzzy classifier with ellipsoidal regions

Abe, Shigeo
Thawonmas, Ruck

(Citation)
IEEE transactions on fuzzy systems,5(3):358-368

(Issue Date)
1997-08

(Resource Type)
journal article

(Version)
Version of Record

(URL)
https://hdl.handle.net/20.500.14094/90000221

A Fuzzy Classifier with Ellipsoidal Regions

Shigeo Abe, *Senior Member, IEEE*, and Ruck Thawonmas, *Member, IEEE*

Abstract— **In this paper, we discuss a fuzzy classifier with ellipsoidal regions which has a learning capability. First, we divide the training data for each class into several clusters. Then, for each cluster, we define a fuzzy rule with an ellipsoidal region around a cluster center. Using the training data for each cluster, we calculate the center and the covariance matrix of the ellipsoidal region for the cluster. Then we tune the fuzzy rules, i.e., the slopes of the membership functions, successively until there is no improvement in the recognition rate of the training data. We evaluate our method using the Fisher iris data, numeral data of vehicle license plates, thyroid data, and blood cell data. The recognition rates (except for the thyroid data) of our classifier are comparable to the maximum recognition rates of the multilayered neural network classifier and the training times (except for the iris data) are two to three orders of magnitude shorter.**

Index Terms—**Blood cell classification, Fisher iris data, fuzzy classifiers, license plate recognition, membership function, neural networks, rule extraction, thyroid data, tuning.**

NOMENCLATURE

$\alpha_{ij}(>0)$	Tuning parameter for cluster ij.
$\beta_{ij}(l)$	Maximum value of $V_{ij}(\boldsymbol{x})$ which is smaller than $U_{ij}(l)$.
$\boldsymbol{c}_{ij}$	Center of cluster ij.
δ	Parameter to control the margin of α_{ij} setting.
$\mathrm{Dec}(l)$	Number of misclassified data that are correctly classified when the value of α_{ij} is in $(L_{ij}(l), L_{ij}(l-1)]$.
$d_{ij}(\boldsymbol{x})$	Weighted distance between $\boldsymbol{x}$ and $\boldsymbol{c}_{ij}$.
$\gamma_{ij}(l)$	Minimum value of $K_{ij}(\boldsymbol{x})$, which is larger than $L_{ij}(l)$.
$h_{ij}(\boldsymbol{x})$	Tuned distance.
$\mathrm{Inc}(l)$	Number of misclassified data that are correctly classified when α_{ij} is in $[U_{ij}(l-1), U_{ij}(l))$.
$K_{ij}(\boldsymbol{x})$	Upper bound of α_{ij} that makes misclassified $\boldsymbol{x}$ become correctly classified.
$L_{if}(\boldsymbol{x})$	Lower bound of α_{ij} to keep $\boldsymbol{x}$ correctly classified.
$L_{ij}(l)$	Lower bound of α_{ij} that allows $l-1$ correctly classified data to be misclassified.
$m_{ij}(\boldsymbol{x})$	Membership function of cluster ij for input $\boldsymbol{x}$.
$N_{\max}$	Upper bound of the number of data belonging to each cluster.
$N_{\min}$	Lower bound of the number of data belonging to each cluster.
N_{ij}	Number of data belonging to cluster ij.
Q_{ij}	$m \times m$ covariance matrix of cluster ij.
$U_{ij}(\boldsymbol{x})$	Upper bound of α_{ij} to keep $\boldsymbol{x}$ correctly classified.
$U_{ij}(l)$	Upper bound of α_{ij} that allows $l-1$ correctly classified data to be misclassified.
$V_{ij}(\boldsymbol{x})$	Lower bound of α_{ij} that makes misclassified $\boldsymbol{x}$ correctly classified.
X	Training data that are correctly classified using the set of fuzzy rules $\{R_{ij}\}$.
Y	Training data misclassified using the set of fuzzy rules $\{R_{ij}\}$.

Manuscript received March 15, 1996; revised October 1, 1996.
S. Abe is with the Department of Electrical and Electronics Engineering, Kobe University, Kobe, 657 Japan.
R. Thawonmas is with the Department of Computer Hardware, University of Aizu, Aizu-Wakamatsu, 965–80 Japan.
Publisher Item Identifier S 1063-6706(97)02836-1.

I. INTRODUCTION

MULTILAYERED neural network classifiers have a learning capability, but analysis of the trained classifier is difficult. To solve this problem, many types of fuzzy classifiers [1]–[5] with a learning capability have been proposed. In general, fuzzy regions which approximate class regions can be classified into 1) ellipsoidal regions [5]; 2) hyperbox regions whose surfaces are parallel to one of the input variables [1], [3]; and 3) polyhedron regions whose surfaces are expressed by a linear combination of input variables [4]. A typical classifier using ellipsoidal regions is the radial basis function classifier [5], which can be considered as both a neural network classifier and a fuzzy classifier.

There are two measures to evaluate a classifier: the training time and the generalization ability. The generalization ability is defined as the recognition rate of the data that are not included in the training data. In the following, we measure the generalization ability of a classifier by one set of the test data that are gathered independently from the training data.

The training time of the fuzzy classifier with hyperbox regions [3] is extremely fast since it is only necessary to calculate the minimum and maximum values of the training data in each input variable; the 100% recognition rate is achieved for the training data as long as there are no identical data in different classes. But, the major drawback of this classifier is that when the characteristics of the training data and the test data differ, the recognition rate of the test data is lower than that of the multilayered neural network classifier. To overcome this problem, we developed the fuzzy classifier with polyhedron regions, which are approximated by shifting the separation hyperplanes extracted from the trained multilayered neural network classifier [4]. The average recognition rate of this classifier for the test data was shown to be better than that of the neural network classifier. But, since this fuzzy classifier is based on the neural network classifier, its training is slow.

In this paper, we discuss a fuzzy classifier with ellipsoidal regions which will realize both high-speed learning and high-generalization ability. To improve ease of analysis of the classifier, it consists of only two layers: the input layer and the output layer, which consists of the fuzzy rules with ellipsoidal regions. We first divide a data set belonging to a class into several clusters. Then we define a fuzzy rule for each cluster, approximating each cluster by an ellipsoidal region with a center and a covariance matrix. The degree of membership of an input for a fuzzy rule is calculated as follows. If the input is at the center of the cluster, the degree of membership is one and as the input moves away from the center the degree of membership decreases. The idea of fuzzy rule tuning is as follows. We only tune the slope of the membership function. By decreasing the slope of the membership function of a fuzzy rule belonging to some class, the degree of membership for that class increases. Thus, the formerly misclassified data belonging to that class may be correctly classified, while the correctly classified data belonging to other classes may be misclassified. We calculate the net increase of the number of correctly classified data. Likewise, we calculate the net increase of the number of correctly classified data by increasing the slope. We then modify the slope so that the recognition rate of the classifier is maximized. In this way, we successively tune the slopes of all the fuzzy rules. We iterate the tuning until there is no improvement in the recognition rate of the training data. With this tuning, the recognition rate of the training data is monotonically improved. The fuzzy rule tuning that allows formerly correctly classified data to be misclassified automatically excludes outliers. We can similarly tune the slopes of the membership functions of the fuzzy classifiers with hyperbox regions [1], [3] and polyhedron regions [4].

In Section II, we describe the classifier architecture which consists of two layers. In Section III, we describe the clustering method in which the training data of each class are divided by the axis that minimizes the size of the existing regions of the clustered training data. In Sections IV and V, we discuss fuzzy rule extraction and fuzzy rule tuning. In Section VI, using the Fisher iris data, numeral data for vehicle license plate recognition, thyroid data, and blood cell data, we compare the performance of the proposed classifier with that of other fuzzy classifiers and the neural network classifier.

II. Classifier Architecture

Consider classification of an m-dimensional input vector $\boldsymbol{x}$ into n classes. Assume that class i $(i = 1, \cdots, n)$ is divided into several clusters ij $(j = 1, \cdots)$, where cluster ij denotes the jth cluster for class i. For each cluster ij, we define the following fuzzy rule:

$$R_{ij}: \quad \text{If } \boldsymbol{x} \text{ is } \boldsymbol{c}_{ij} \text{ then } \boldsymbol{x} \text{ belongs to class } i \tag{1}$$

where $\boldsymbol{c}_{ij}$ is the center of cluster ij. The membership function $m_{ij}(\boldsymbol{x})$ of (1) for input $\boldsymbol{x}$ is given by

$$m_{ij}(\boldsymbol{x}) = \exp(-h_{ij}^2(\boldsymbol{x})) \tag{2}$$

$$h_{ij}^2(\boldsymbol{x}) = \frac{d_{ij}^2(\boldsymbol{x})}{\alpha_{ij}} \tag{3}$$

$$d_{ij}^2(\boldsymbol{x}) = (\boldsymbol{x} - \boldsymbol{c}_{ij})^t Q_{ij}^{-1} (\boldsymbol{x} - \boldsymbol{c}_{ij}) \tag{4}$$

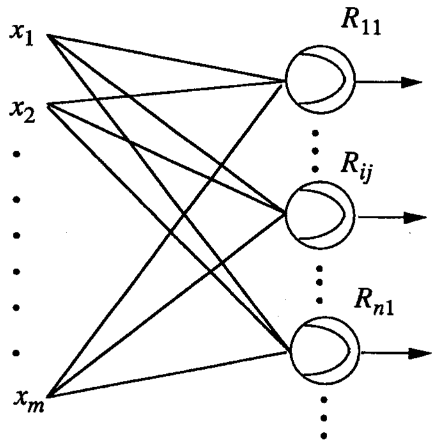


Fig. 1. Architecture of a fuzzy classifier.

where $d_{ij}(\boldsymbol{x})$ is the weighted distance between $\boldsymbol{x}$ and $\boldsymbol{c}_{ij} = (c_{ij,1}, \cdots, c_{ij,m})t$, $h_{ij}(\boldsymbol{x})$ is the tuned distance, $\alpha_{ij}(>0)$ is a tuning parameter for cluster ij, Q_{ij} is the $m \times m$ covariance matrix of cluster ij, the superscript t denotes the transpose of a matrix, and the superscript -1 denotes the inverse of a matrix. An increase of α_{ij} decreases the slope of the membership function $m_{ij}(\boldsymbol{x})$ or increases the value of $m_{ij}(\boldsymbol{x})$. And a decrease of α_{ij} increases the slope of $m_{ij}(\boldsymbol{x})$ or decreases the value of $m_{ij}(\boldsymbol{x})$. As is discussed later, by calculating the covariance matrix Q_{ij} using the training data, estimation of the covariance matrix [5] is not necessary.

Fig. 1 shows the architecture of the fuzzy classifier. For input $\boldsymbol{x}$, if the membership function $m_{kl}(\boldsymbol{x})$ is the largest, input $\boldsymbol{x}$ is classified into class k. The exponential function in (2) makes the output range of (2) lie in [0, 1]. Thus, if we classify input $\boldsymbol{x}$ using the input of the exponential function in (2), we need to find the smallest $h_{ij}(x)$. This is the simplest architecture that is conceivable.

III. Clustering

If we compare the multilayered neural network classifier and the radial basis function classifier, which has an additional layer with linear output functions in Fig. 1, we find the generalization ability of the former classifier is better than that of the latter, but the former requires more learning time. This is because the former classifier uses the sigmoid function, which works to separate the input space by a separation hyperplane formed by weights; the sigmoid function is a global function, while the Gaussian function used in the latter classifier is a local function. Thus, using the local function, the classifier outputs can easily fit the desired outputs of the training data. This leads to overfitting and worsens the generalization ability of the classifier. Thus, to obtain the generalization ability comparable to the neural network classifier, the region that the Gaussian function covers needs to be as large as possible. Therefore, when we divide the data belonging to the same class into clusters, the number of data belonging to a cluster should not be too small.

There are several clustering techniques [6]–[9]. Most of them are iterative methods; that is, they iterate the procedure until the clustering is converged. In addition, most of them do not have a mechanism to control the number of data belonging to a cluster. Thus, the number of data belonging to one cluster may be very small, while that of another cluster may be very large. If we determine the parameters in (2) to (4) using a relatively small number of data, we should not expect a good generalization ability. Thus, we need to cluster the training data for each class considering both the degree of data gathering and the number of data included in the cluster.

Here we discuss a simple clustering algorithm that divides the training data by one input axis at a time. As a measure of the degree of data gathering, we use the size of the hyperboxes that include the training data. Consider the training data with two-dimensional inputs, as shown in Fig. 2. In Fig. 2(a) and (b), the training data are divided by the x_1 axis and the x_2 axis, respectively. The total size of the hyperboxes that includes the training data in Fig. 2(b) is smaller than that in Fig. 2(a). Thus, we consider that dividing the training data by the x_2 axis is more favorable. But if the number of the data included in either class is smaller than the specified number we do not choose this axis for division. It may be better to divide the data by a hyperplane expressed by a linear combination of the input variables. But here, for simplicity, we do not consider this. In the following, we show the more detailed algorithm.

1) Let $N_{\max}$ and $N_{\min}$ be the upper and lower bounds of the number of the data belonging to each cluster. First, assume that each class has one cluster.
2) Select a cluster, e.g., cluster ij, whose number of data exceeds $N_{\max}$. Calculate the center $\boldsymbol{c}_{ij}$ by calculating the average values of the data belonging to the cluster

$$c_{ij,k} = \frac{1}{N_{ij}} \sum_{\boldsymbol{x} \in \text{cluster}\, ij} x_k \tag{5}$$

where N_{ij} is the number of the data belonging to cluster ij. The size of the two hyperboxes that include the training data belonging to cluster ij, when the training data are divided by the axis $x_k = \boldsymbol{c}_{ij,k}$, is given by

$$S_{ij,k} = \prod_{l=1}^{m} \left(\max_{\substack{\boldsymbol{x} \in \text{cluster}\, ij \\ x_k \geq c_{ij,k}}} x_l - \min_{\substack{\boldsymbol{x} \in \text{cluster}\, ij \\ x_k \geq c_{ij,k}}} x_l \right) + \prod_{l=1}^{m} \left(\max_{\substack{\boldsymbol{x} \in \text{cluster}\, ij \\ x_k < c_{ij,k}}} x_l - \min_{\substack{\boldsymbol{x} \in \text{cluster}\, ij \\ x_k < c_{ij,k}}} x_l \right). \tag{6}$$

We select such k that minimizes

$$\min_{\substack{k=1,\cdots,m \\ N_{ij,1} > N_{\min}, N_{ij,2} > N_{\min}}} S_{ij,k} \tag{7}$$

where $N_{ij,1}$ and $N_{ij,2}$ are the numbers of data satisfying $x_k \geq \boldsymbol{c}_{ij,k}$ and $x_k < \boldsymbol{c}_{ij,k}$, respectively. If there is no such k that minimizes (7) we do not divide cluster ij.
3) Iterate step 2) until there is no cluster whose number of data exceeds $N_{\max}$ and which is dividable.

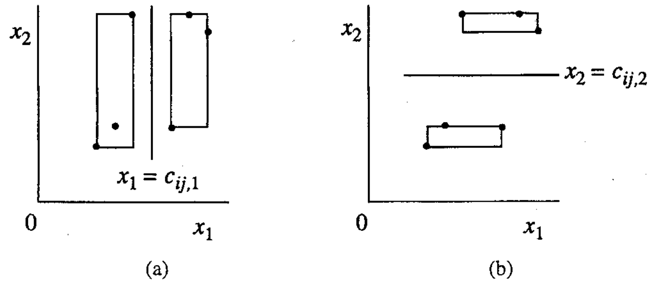


Fig. 2. Concept of clustering. (a) Division by $x_1 = \boldsymbol{c}_{ij,1}$. (b) Division by $x_2 = \boldsymbol{c}_{ij,2}$.

In our simulation discussed in Section VI, we set $N_{\max} = N_{\min}/2$. To determine the optimal $N_{\max}$, we need trial and error. But, as is shown in Section VI, for most problems, clustering of the training data is not necessary; that is, one cluster for one class.

IV. Fuzzy Rule Extraction

For each cluster, we define the fuzzy rule R_{ij} given by (1). At this stage, we approximate the fuzzy regions without considering the overlaps between the fuzzy regions of different classes. These overlaps are resolved by tuning the parameters α_{ij}.

First, for the cluster ij, we calculate the center $\boldsymbol{c}_{ij}$ using (5). Then, we calculate the covariance matrix Q_{ij} by

$$Q_{ij} = \frac{1}{N_{ij}} \sum_{\boldsymbol{x} \in \text{cluster}\, ij} (\boldsymbol{x} - \boldsymbol{c}_{ij})(\boldsymbol{x} - \boldsymbol{c}_{ij})^t. \tag{8}$$

If the covariance matrix Q_{ij} is singular, we set all the off-diagonal elements of Q_{ij} to zero so that Q_{ij} becomes regular. By making the covariance matrix diagonal, the principal axes of the associated ellipsoidal region are parallel to the input axes.

In conventional methods, the covariance matrices are estimated during training. But, since we know which training data are included in which cluster, we can calculate the covariance matrix for each cluster using the training data included in the cluster.

The fuzzy classifier with the ellipsoidal regions given by (1)–(5) and (8) is equivalent to the Gaussian classifier in which the probability distribution function for a cluster is calculated assuming that the training data belonging to that cluster obey the Gaussian distribution. But, since the data belonging to a cluster do not necessarily obey the Gaussian distribution, we need to tune the fuzzy rules so that the recognition rate of the training data is maximized.

V. Fuzzy Rule Tuning

A. Concept

Since the fuzzy rules are defined without considering the overlaps between classes, we need to tune the fuzzy rules to improve the recognition rates for both the training data and test data. In the following, we tune the fuzzy rules so that the maximum recognition rate is obtained for the training data.

But, since we tune the fuzzy rules using only the training data, this strategy does not necessarily lead to the improvement of the generalization ability, i.e., overfitting occurs, as is the case with the training of the neural network classifier. One way to solve this problem is to stop tuning when the recognition rate of the test data begins to decrease. But in our paper, to show the effect of tuning both for the training data and the test data we do not use this strategy; we tune the fuzzy rules until the recognition rate of the training data is maximized.

If we tune the centers and the covariance matrices of fuzzy rules we need to resort to the steepest descent method, which is very time consuming. Instead, we tune only one parameter for each fuzzy rule R_{ij}, i.e., α_{ij}, so that the recognition rate of the training data is maximized. If we increase α_{ij}, the degree of membership given by (2) increases, and if we decrease it, the degree of membership decreases. To explain the concept of tuning, we consider a two-class case with one rule for each class, as shown in Fig. 3(a). (In the figures that follow, instead of the Gaussian function, we use the triangular function as the membership function.) In this case, datum 1, belonging to class 1, is misclassified into class 2. This datum can be correctly classified if we increase α_{11} so that the membership function lies between the shaded regions without causing datum 2 to be misclassified. This can also be achieved when we decrease α_{21}.

Fig. 3(b) shows a more complicated situation. Datum 1 is correctly classified into class 2, while data 2, 3, and 4 are misclassified into class 2. If we increase α_{11} or decrease α_{21}, datum 1 is first misclassified, but if we allow datum 1 to be misclassified we can make data 2, 3, and 4 be correctly classified. Fig. 3(b) shows this when α_{21} is decreased so that the degree of membership for class 2 lies between the shaded regions. Then, by allowing one datum to be misclassified, three data are correctly classified, i.e., the recognition rate is improved by two data.

Our tuning algorithm determines, for each fuzzy rule R_{ij}, the optimum tuning parameter α_{ij}, allowing the data that are correctly classified before tuning R_{ij} to become misclassified after tuning R_{ij} as long as the recognition rate of the training data is improved. We call the update of all $\alpha_{ij}(i = 1, \cdots, n, j = 1, \cdots)$ one iteration of tuning and, if there is no improvement in the recognition rate for the two consecutive iterations or the recognition rate of the training data reaches 100%, we stop tuning. To allow the data that are correctly classified before tuning some fuzzy rule to be misclassified after tuning that fuzzy rule is, so to speak, to prevent the tuning process from leading to convergence to a local minimum. But, of course, since the tuning process is nonlinear, we cannot guarantee that this method always gives the optimal solution.

The special feature of the fuzzy rule tuning is that outliers [Datum 1 in Fig. 3(b)] are automatically eliminated by allowing the data that are correctly classified before tuning to be misclassified after tuning. The elimination of outliers was not considered in [1], [3], and [4].

In Section V-B, we calculate the upper bound and the lower bound of α_{ij} that allow the $l-1(\geq 0)$ data that are correctly classified to become misclassified and, in Section V-C, we

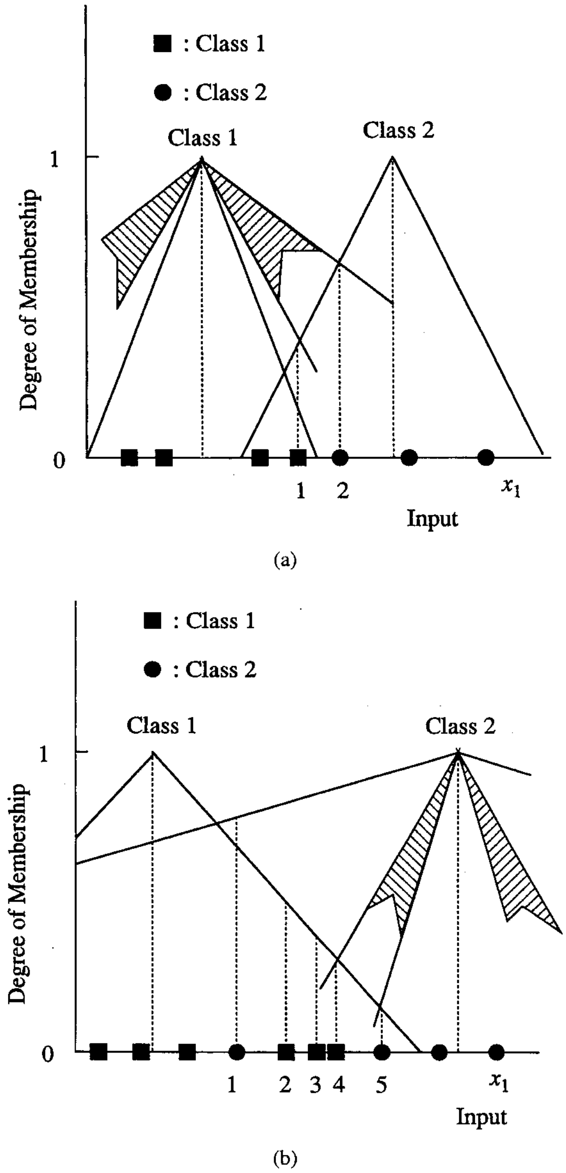


Fig. 3. Concept of tuning. (a) If the slope of the membership function for class 1 is decreased so that the resulting function lies between the shaded regions, datum 1 can be correctly classified. (b) If the slope of the membership function for class 2 is increased so that the resulting function lies between the shaded regions, datum 1 is misclassified, but data 2, 3, and 4 are correctly classified.

check how many data that are misclassified are correctly classified if α_{ij} is changed within the bounds calculated in Section V-B. Then in Section V-D, α_{ij} is determined so that the recognition rate of the training data is maximized.

B. Upper and Lower Bounds of α_{ij}

We calculate the upper bound $U_{ij}(l)$ and the lower bound $L_{ij}(l)$ of α_{ij} allowing the $l-1(\geq 0)$ data that are correctly classified to be misclassified. We divide a set of input data into X and Y, where X consists of the data correctly classified using the set of fuzzy rules $\{R_{ij}\}$ and Y consists of the misclassified data. Then, we choose $\boldsymbol{x}(\in X)$ that belongs to class i and that satisfies

$$h_{ij}(\boldsymbol{x}) \leq \min_{k \neq j} h_{ik}(\boldsymbol{x}). \tag{9}$$

If (9) does not hold, $\boldsymbol{x}$ remains to be correctly classified even if we change α_{ij}. If $\boldsymbol{x}$ further satisfies

$$h_{ij}^2(\boldsymbol{x}) = \frac{d_{ij}^2(\boldsymbol{x})}{\alpha_{ij}} < \min_{o \neq i} h_{op}^2(\boldsymbol{x}) \leq \min_{k \neq j} h_{ik}^2(\boldsymbol{x}) \tag{10}$$

there is a lower bound $L_{ij}(\boldsymbol{x})$ to keep $\boldsymbol{x}$ correctly classified (see Fig. 4(a)—in the text, we use the tuned distance $h_{ij}(\boldsymbol{x})$, but in the figure, we use the membership function $m_{ij}(\boldsymbol{x})$ for ease of understanding)

$$L_{ij}(\boldsymbol{x}) = \frac{d_{ij}^2(\boldsymbol{x})}{\min_{o \neq i} h_{op}^2(\boldsymbol{x})} < \alpha_{ij}. \tag{11}$$

If (10) is not satisfied, namely

$$h_{ij}^2(\boldsymbol{x}) = \frac{d_{ij}^2(\boldsymbol{x})}{\alpha_{ij}} < \min_{k \neq j} h_{ik}^2(\boldsymbol{x}) < \min_{o \neq i} h_{op}^2(\boldsymbol{x}) \tag{12}$$

α_{ij} can be decreased without making $\boldsymbol{x}$ become misclassified [see Fig. 4(b)].

Now, the lower bound $L_{ij}(1)$, which is defined as the lower bound that does not make any correctly classified data become misclassified, is

$$L_{ij}(1) = \max_{\boldsymbol{x} \in X} L_{ij}(\boldsymbol{x}). \tag{13}$$

To clarify the discussion, we assume that $L_{ij}(x)$ is different for different $\boldsymbol{x}$ .Then, (13) is satisfied by one $\boldsymbol{x}$. Similarly, $L_{ij}(2)$, which is defined as the lower bound that allows one correctly classified datum to be misclassified, is the second maximum among $L_{ij}(\boldsymbol{x})$ and is given by

$$L_{ij}(2) = \max_{\boldsymbol{x} \in X, L_{ij}(\boldsymbol{x}) \neq L_{ij}(1)} L_{ij}(\boldsymbol{x}). \tag{14}$$

In general,

$$L_{ij}(l) = \max_{\boldsymbol{x} \in X, L_{ij}(\boldsymbol{x}) \neq L_{ij}(1), \cdots, L_{ij}(l-1)} L_{ij}(\boldsymbol{x}). \tag{15}$$

In the similar manner that we determined the lower bound $L_{ij}(l)$, we can determine the upper bound $U_{ij}(l)$. We choose $\boldsymbol{x} (\in X)$, which belongs to class $o (\neq i)$. Let cluster op have the minimum tuned distance $h_{op}(\boldsymbol{x})$:

$$h_{op}(\boldsymbol{x}) = \min_{q} h_{oq}(\boldsymbol{x}). \tag{16}$$

Since the tuned distance $h_{ij}(\boldsymbol{x})$ is larger than $h_{op}(\boldsymbol{x})$, the upper bound $U_{ij}(\boldsymbol{x})$ of α_{ij} in which $\boldsymbol{x}$ remains correctly classified is (Fig. 5)

$$U_{ij}(\boldsymbol{x}) = \frac{d_{ij}^2(\boldsymbol{x})}{\min_{q} h_{oq}^2(\boldsymbol{x})}. \tag{17}$$

Now the upper bound $U_{ij}(1)$, which is defined as the upper bound that does not make any correctly classified data be misclassified, is

$$U_{ij}(1) = \min_{\boldsymbol{x} \in X} U_{ij}(\boldsymbol{x}). \tag{18}$$

Here, we also assume that $U_{ij}(\boldsymbol{x})$ is different for different $\boldsymbol{x}$; then, (18) is satisfied by one $\boldsymbol{x}$. Similarly, $U_{ij}(2)$, which is defined as the upper bound that allows one correctly classified datum to be misclassified, is the second minimum among $U_{ij}(\boldsymbol{x})$ and is given by

$$U_{ij}(2) = \min_{\boldsymbol{x} \in X, U_{ij}(\boldsymbol{x}) \neq U_{ij}(1)} U_{ij}(\boldsymbol{x}). \tag{19}$$

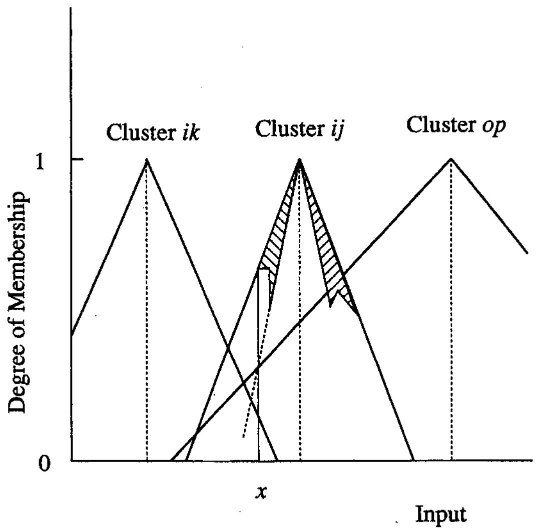
In general,

$$U_{ij}(l) = \min_{\boldsymbol{x} \in X, U_{ij}(\boldsymbol{x}) \neq U_{ij}(1), \cdots, U_{ij}(l-1)} U_{ij}(\boldsymbol{x}). \tag{20}$$

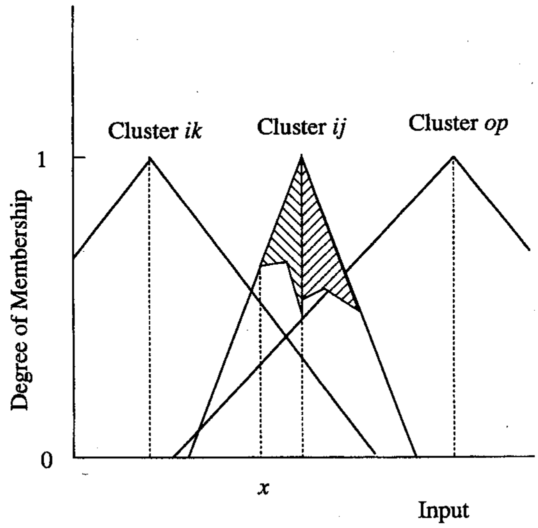
Thus, α_{ij} is bounded by

$$\begin{aligned} \cdots < L_{ij}(l) < L_{ij}(l-1) < \cdots < L_{ij}(1) < \alpha_{ij} < U_{ij}(1) \\ < \cdots < U_{ij}(l-1) < U_{ij}(l) < \cdots. \end{aligned} \tag{21}$$

If we change α_{ij} in the range of $(L_{ij}(1), U_{ij}(1))$, the correctly classified data remain to be correctly classified where (a, b) denotes an open interval and if we change α_{ij} in the range of $[U_{ij}(l-1), U_{ij}(l))$ or $(L_{ij}(l), L_{ij}(l-1)]$, the $l-1$ correctly



(a)



(b)

Fig. 4. Lower bound of α_{ij}. (a) If the slope of the membership function for cluster ij is increased so that the resulting function lies between the shaded regions, x is not misclassified. (b) Since $m_{ik}(x)$ is larger than $m_{op}(x)$, increase of $m_{ij}(x)$ does not cause misclassification of x.

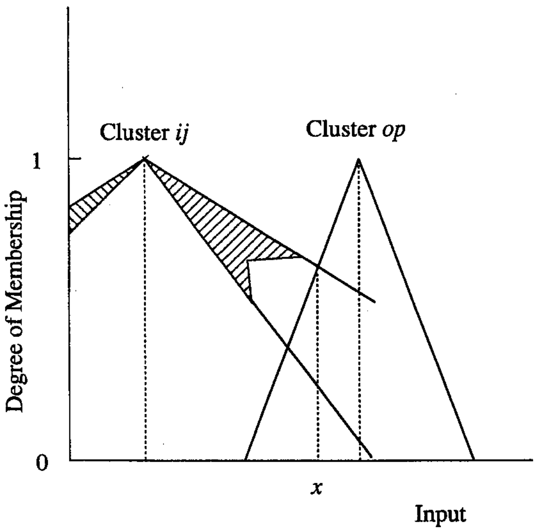


Fig. 5. Upper bound of α_{ij}. The slope of $m_{ij}(x)$ can be decreased in the shaded regions without causing misclassification of x which belongs to class o.

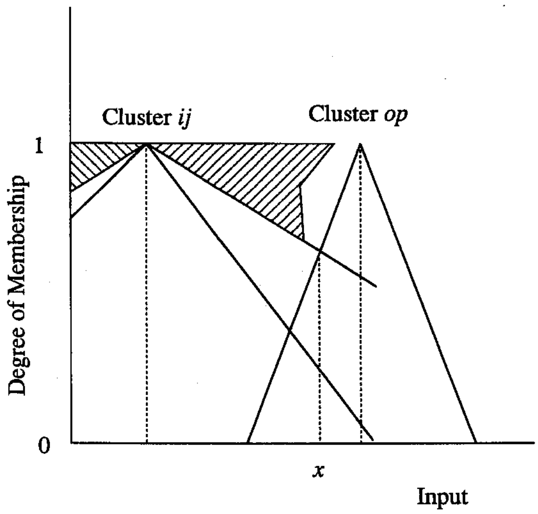


Fig. 6. Resolution of misclassification by decreasing the slope. If the slope of $m_{ij}(x)$ is decreased within the shaded regions, datum x, which belongs to class i, is correctly classified.

classified data are misclassified where $[a, b]$ denotes a closed interval.

C. Resolution of Misclassification by Changing α_{ij}

For $\boldsymbol{x}(\in Y)$ which is misclassified into class i or which belongs to class i but is misclassified into class o $(\neq i)$, we check whether it can be correctly classified by changing α_{ij}. First, we consider increasing α_{ij}. Let $\boldsymbol{x}$, which belongs to class i, be misclassified into class o. This datum can be correctly classified if

$$\alpha_{ij} > V_{ij}(\boldsymbol{x}) = \frac{d_{ij}^2(\boldsymbol{x})}{\min\limits_{p} h_{op}^2(\boldsymbol{x})} \tag{22}$$

irrespective of the values of $h_{ik}(\boldsymbol{x})(k \neq i)$ where $V_{ij}(\boldsymbol{x})$ is the lower bound of α_{ij} that makes the misclassified $\boldsymbol{x}$ correctly classified (see Fig. 6).

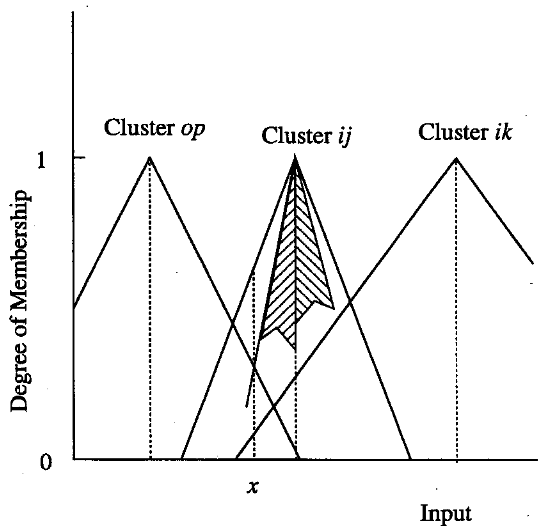


Fig. 7. Resolution of misclassification by increasing the slope. If $m_{ij}(x)$ is increased within the shaded regions, datum x, which belongs to class o, is correctly classified.

Let $\mathrm{Inc}(l)$ denote the number of the misclassified data that are correctly classified if we set the value of α_{ij} in $[U_{ij}(l-1), U_{ij}(l))$. We increase $\mathrm{Inc}(l)$ by one if $V_{ij}(\boldsymbol{x})$ is included in $(\alpha_{ij}, U_{ij}(l))$ and we define

$$\beta_{ij}(l) = \max_{V_{ij}(\boldsymbol{x}) < U_{ij}(l)} V_{ij}(\boldsymbol{x}). \tag{23}$$

If α_{ij} is set to be larger than $\max(\beta_{ij}(l), U_{ij}(l-1))$, $\mathrm{Inc}(l)$ data are correctly classified although the $l-1$ correctly classified data are misclassified.

Let $\boldsymbol{x}$, which belongs to class o, be misclassified into class i. Then, similar to the above discussions, we check whether $\boldsymbol{x}$ can be correctly classified by decreasing α_{ij}. First, the minimum tuned distance for class o should be the second minimum among n classes, namely, q in the following equation needs to be o:

$$\min_{k} h_{ik}(\boldsymbol{x}) < \min_{q \neq i} h_{qr}(\boldsymbol{x}). \tag{24}$$

Second, $h_{ij}(\boldsymbol{x})$ needs to be the minimum in class i, and the second minimum in class i is larger than the minimum tuned distance in class o

$$h_{ij}(\boldsymbol{x}) < \min_{p} h_{op}(\boldsymbol{x}) < \min_{k \neq j} h_{ik}(\boldsymbol{x}). \tag{25}$$

Then, the datum can be correctly classified if (Fig. 7)

$$\alpha_{ij} < K_{ij}(\boldsymbol{x}) = \frac{d_{ij}^2(\boldsymbol{x})}{\min\limits_{p} h_{op}^2(\boldsymbol{x})} \tag{26}$$

where $K_{ij}(\boldsymbol{x})$ is the upper bound of α_{ij} that makes misclassified $\boldsymbol{x}$ become correctly classified.

Let $\mathrm{Dec}(l)$ denote the number of the misclassified data that are correctly classified if we set the value of α_{ij} in $(L_{ij}(l), L_{ij}(l-1)]$. We increase $\mathrm{Dec}(l)$ by one if $K_{ij}(\boldsymbol{x})$ is included in $(L_{ij}(l), \alpha_{ij})$. We define

$$\gamma_{ij}(l) = \min_{K_{if}(\boldsymbol{x}) > L_{if}(l)} K_{ij}(\boldsymbol{x}). \tag{27}$$

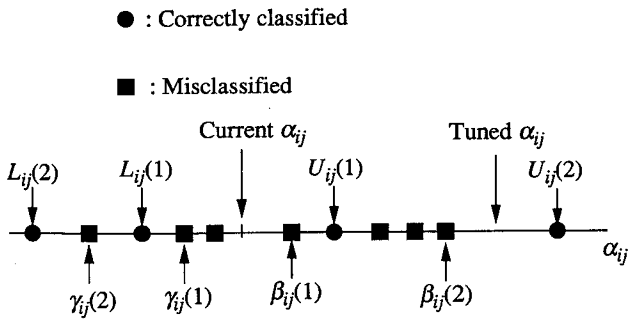


Fig. 8. Determination of tuned α_{ij}. If the current α_{ij} is modified to the tuned α_{ij} in $(\beta_{ij}(2), U_{ij}(2))$, one correctly classified datum is misclassified but four misclassified data are correctly classified.

If α_{ij} is set to be smaller than $\min(\gamma_{ij}(l), L_{ij}(l-1)), \mathrm{Dec}(l)$ data are correctly classified although the $l-1$ correctly classified data are misclassified.

D. Modification of α_{ij}

For $\mathrm{Inc}(l), l = 1, \cdots, l_M$, where l_M is a positive integer, we find l that satisfies

$$\max_l(\mathrm{Inc}(l) - l + 1). \tag{28}$$

Similarly, for $\mathrm{Dec}(l), l = 1, \cdots l_M$, we find l that satisfies

$$\max_l(\mathrm{Dec}(l) - l + 1). \tag{29}$$

If there are plural l's that satisfy (28) or (29), we chose the smallest l. First, we consider the case where (28) is larger than or equal to (29). If we increase α_{ij} so that it is larger than $\beta_{ij}(l)$ in $(\alpha_{ij}, U_{ij}(l))$, the net increase of the correctly classified data is $\mathrm{Inc}(l) - l + 1$. Thus, we set α_{ij} in $[\beta_{ij}(l), U_{ij}(l))$ as follows (see Fig. 8):

$$\alpha_{ij} = \beta_{ij}(l) + \delta(U_{ij}(l) - \beta_{ij}(l)) \tag{30}$$

where δ satisfies $0 < \delta < 1$. Here, $\beta_{ij}(l) \geq U_{ij}(l-1)$ holds, otherwise l cannot satisfy (28).

Likewise, if (28) is smaller than (29), we decrease α_{ij} so that it is smaller than $\gamma_{ij}(l)$ in $(L_{ij}(l), \gamma_{ij}(l)]$ as follows:

$$\alpha_{ij} = \gamma_{ij}(l) - \delta(\gamma_{ij}(l) - L_{ij}(l)). \tag{31}$$

The parameter δ is used to control the recognition rate of the test data (the recognition rate of the training data is the same irrespective of the value of δ). According to our experiments, the value of δ did not affect the recognition rate of the test data significantly, but a small value of δ sometimes gave a better recognition rate of the test data. Thus, in the experiments in Section VI, we used 0.1.

E. Tuning Algorithm

According to the above discussion, the tuning algorithm becomes as follows.

1) Set a positive number to parameter l_M, where $l_M - 1$ is the maximum number of misclassifications allowed for tuning α_{ij}, set a value in (0, 1) to δ in (30) and (31), and set the same positive initial value (usually 1) to α_{ij}. In our experiments $l_M = 10$ is sufficient.
2) For $\alpha_{ij}(i = 1, \cdots, n, j = 1, \cdots)$, calculate $L_{ij}(l), U_{ij}(l), \mathrm{Inc}(l), \mathrm{Dec}(l), \beta_{ij}(l)$, and $\gamma_{ij}(l)$ for $l = 1, \cdots, l_M$. Find l that maximizes (28) or (29), and change α_{ij} using (30) or (31).
3) Iterate Step 2) until there is no improvement in the recognition rate.

In the following, we count one execution of Step 2) as one iteration.

TABLE I
PERFORMANCE FOR THE IRIS TEST DATA

Classifier	No. Wrong	No. Rules	Time(s)
N.N.	2.2 (1–3)[1]	2 units	2
Hyperbox	2–6	17–5	<1
Ellipsoid	1(1)[2] → 2(0)[2]	3	2

()[1]: Minimum and maximum numbers of misclassified data
()[2]: Number of the misclassified training data

VI. Performance Evaluation

We evaluated the performance of the fuzzy classifier with ellipsoidal regions using the iris data [10], numerical data for license plate recognition [11], [12], thyroid data [13], and blood cell data [14], and compared the performance with that of the multilayered neural network classifier, the fuzzy classier with hyperbox regions [3], and the fuzzy classifier with polyhedron regions [4]. Unless otherwise stated, we set $\alpha_{ij} = 1, \delta = 0.1, l_M = 10$, and $N_{\min} = N_{\max}/2$. Except for the thyroid data, without clustering the training data belonging to a class, the recognition rates of the test data for the fuzzy classifier with ellipsoidal regions were comparable to the maximum recognition rates for the remaining classifiers. Thus, except for the thyroid data, we did not cluster the data belonging to a class and the number of fuzzy rules created was the number of classes. For evaluation of the fuzzy classifier with hyperbox regions, we used a 16-MIPS workstation. For all other evaluations, we used a 60-MIPS mainframe computer and the calculation times listed in the following tables are the CPU times.

A. Iris Data

The Fisher iris data [10] consist of 150 data with four input features and three classes. In our study, the training data were composed of the first 25 data of each class, while the test data were composed of the remaining 25 data of each class.

Table I shows the performance of the neural network classifier, the fuzzy classifier with hyperbox regions, and the fuzzy classifier with ellipsoidal regions. The three-layered neural network classifier with two hidden units was trained ten times using different initial weights distributed in [−0.1, 0.1]; for each training the number of epochs was 1000 and the learning rate was set to 1.0 with zero momentum. The minimum number of misclassified test data was one and their average number was 2.2. Using the fuzzy classifier with hyperbox regions, we previously found the minimum number of misclassified test data was two and the corresponding number of rules was 17 [3]. The left-hand side of the No. Wrong column (see Table I) for the fuzzy classifier with ellipsoidal

TABLE II
PERFORMANCE FOR THE NUMERAL TEST DATA USING 12 INPUT FEATURES

Classifier	Rate (%)	No. Rules	Time (s)
N.N.	99.41 (99.76–98.90)¹	6 units	79
Hyperbox	99.63	10	< 1
Ellipsoid	99.63 (99.75)² → 99.39 (99.88)²	10	4

()¹: Minimum and maximum recognition rates
()²: Recognition rate of the training data

TABLE III
PERFORMANCE FOR THE NUMERAL TEST DATA USING SIX INPUT FEATURES

Classifier	Rate (%)	No. Rules	Time (s)
N.N.	97.78 (98.66–96.83)¹	6 units	417
Hyperbox	98.05	16	< 1
Ellipsoid	98.29 (97.90)² → 98.41 (99.26)²	10	3

()¹: Minimum and maximum recognition rates
()²: Recognition rate of the training data

regions lists the numbers of the misclassified data for the test data and the training data without tuning fuzzy rules and the right-hand side lists the numbers of misclassified data for the test and the training data after tuning fuzzy rules. Without tuning, the number of the misclassified test data was one. With one iteration of tuning, all the training data were correctly classified, but the number of the misclassified test data was increased to two. Namely, overfitting occurred. Therefore, for the iris data, tuning was not necessary. The fuzzy classifier with ellipsoidal regions realized the best performance of the neural network classifier with comparable computation time.

B. License Plate Recognition System

The data used in this study were originally collected to develop a license plate recognition system [11], [12]. Numerals from 0 to 9 were considered. Each of these data consisted of 12 input features extracted from the images of running cars as taken by a television camera. There were 1630 data, which were divided into 810 training data and 820 test data.

Table II shows the results for the three-layered neural network classifier, the fuzzy classifier with hyperbox regions [3], and the fuzzy classifier with ellipsoidal regions. The neural network classifier with six hidden units was trained 100 times with different initial weights; the maximum number of epochs was set to 10 000.

For the fuzzy classifier with ellipsoidal regions, without clustering the number of fuzzy rules was ten. For the test data, the best recognition rate of 99.63% was obtained without tuning. This was the same recognition rate as the fuzzy classifier with hyperbox regions; it was comparable to the maximum recognition rate of the neural network classifier, while the computation time was 20 times faster.

To produce a difficult classification situation, we used only the first and eighth to twelfth features. Table III shows the results. The best recognition rate of 98.41% was obtained by tuning the fuzzy classifier with ellipsoidal regions. This was better than that of the fuzzy classifier with hyperbox regions; it was comparable to the maximum recognition rate of the neural network classifier and the computation time was more than one hundred times faster.

TABLE IV
PERFORMANCE FOR THE THYROID TEST DATA

Classifier	Rate (%)	No. Rules	Time (s)
N.N.	98.00 (98.48–97.78)¹	3 units	60.8 min
Hyperbox	99.15	10	<5
Ellipsoid	86.41 (86.77)² → 95.60 (96.02)²	3	25

()¹: Minimum and maximum recognition rates
()²: Recognition rate of the training data

TABLE V
PERFORMANCE OF THE FUZZY CLASSIFIER WITH ELLIPSOIDAL REGIONS FOR THE THYROID DATA BY CHANGING l_M

l_M	Init. (%)	Final (%)	Iterations	Time (s)
1	86.41 (86.77)	87.08 (87.67)	2	10
5	86.41 (86.77)	94.37 (94.99)	8	29
10	86.41 (86.77)	95.60 (96.02)	7	25
15	86.41 (86.77)	95.62 (96.05)	5	19
20	86.41 (86.77)	95.60 (96.05)	5	19

(): Recognition rate of the training data

TABLE VI
PERFORMANCE OF THE FUZZY CLASSIFIER WITH ELLIPSOIDAL REGIONS FOR THE THYROID DATA BY CHANGING $N_{\max}(N_{\min} = N_{\max}/2)$

N_{max}	No. Rules	Init.(%)	Final (%)	Iterations	Time (s)
–	3	86.41 (86.77)	95.60 (96.02)	7	25
1000	7	87.95 (88.04)	96.00 (96.92)	4	46
750	9	86.44 (87.35)	96.76 (97.14)	3	50
500	12	87.54 (86.90)	96.70 (97.59)	4	70
250	25	87.78 (87.04)	96.79 (98.01)	4	174

(): Recognition rate of the training data

C. Thyroid Data

The thyroid data classify input data, consisting of 21 features, into three classes. The training data and the test data consist of 3772 and 3428 data, respectively. The characteristics of the data are that the input features include 15 discrete features and more than 92% of the data belong to one class. Table IV shows the results. The three-layered neural network classifier with three hidden units was trained ten times; the number of epochs was 10 000 and the computation time was 60.8 min for each simulation, 608 min in total. The fuzzy classifier with hyperbox regions showed the best recognition rate. The recognition rate of the fuzzy classifier with ellipsoidal regions improved drastically by tuning, but it was less than those of the neural network classifier and the fuzzy classifier with hyperbox regions.

Table V shows the effect of l_M on the recognition rate without clustering. In the table, the Init. column lists the recognition rates of the test data and the training data without tuning fuzzy rules, and the Final column lists the recognition rates of the test and the training data after tuning fuzzy rules. For $l_M = 1$, the final recognition rate of the test data was 87.08% and there was little improvement, but when we increased l_M, the recognition rate was drastically improved. When l_M was equal to or larger than ten, the recognition

TABLE VII
PERFORMANCE FOR THE THYROID TEST DATA USING SIX CONTINUOUS INPUTS

Classifier	Rate (%)	No. Rules	Time (s)
N.N.	96.42 (96.53–96.38)^1	3 units	23 min
Hyperbox	97.11	44	<5
Ellipsoid	94.31 (94.88)^2 $\rightarrow$ 95.83 (96.34)^2	3	4

()^1: Minimum and maximum recognition rates
()^2: Recognition rate of the training data

TABLE VIII
PERFORMANCE OF THE FUZZY CLASSIFIER WITH ELLIPSOIDAL REGIONS FOR THE THYROID DATA BY CHANGING $N_{\max}$ ($N_{\min} = N_{\max}/2$, SIX CONTINUOUS INPUTS)

N_{max}	No. Rules	Init.(%)	Final (%)	Iterations	Time (s)
–	3	94.31 (94.88)	95.83 (96.34)	5	4
1000	8	85.91 (83.99)	96.32 (96.90)	5	10
750	9	85.71 (83.54)	96.47 (96.74)	4	10
500	13	83.46 (81.39)	96.44 (96.74)	4	12
250	5	89.93 (90.03)	96.44 (96.92)	3	6

(): Recognition rate of the training data

TABLE IX
PERFORMANCE FOR THE BLOOD CELL TEST DATA

Classifier	Rate (%)	No. Rules	Time (s)
N.N.	87.44 (90.46)^1	15 units	133 min.
Polyhedron	90.58 (91.68)^1	302	133 min.
Hyperbox	86.52	217	<5
Ellipsoid	87.45 (92.64)^2 $\rightarrow$ 91.65 (95.41)^2	12	29

()^1: Maximum recognition rate
()^2: Recognition rate of the training data

TABLE X
PERFORMANCE OF THE FUZZY CLASSIFIER WITH ELLIPSOIDAL REGIONS FOR THE BLOOD CELL DATA BY CHANGING l_M

l_M	Init.(%)	Final (%)	Iterations	Time (s)
1	87.45 (92.64)	87.77 (93.15)	2	15
5	87.45 (92.64)	90.32 (95.00)	4	30
10	87.45 (92.64)	91.65 (95.41)	4	29
15	87.45 (92.64)	91.39 (95.41)	3	26
20	87.45 (92.64)	91.55 (95.38)	5	34
25	87.45 (92.64)	91.29 (95.29)	4	28
30	87.45 (92.64)	91.55 (95.38)	4	31

(): Recognition rate of the training data

rates were almost the same. Thus, it was a good choice to set $l_M = 10$.

To improve the recognition rate of the fuzzy classifier with ellipsoidal regions, we clustered the training data. Table VI shows the results for $l_M = 10$. The maximum recognition rate of 96.79% for the test data was obtained when $N_{\max} = 250$, but it was still lower than that of the neural network classifier or the fuzzy classifier with hyperbox regions. The recognition rate of the training data for $N_{\max} = 250$ was 98.01%.

The poor performance of the fuzzy classifier with ellipsoidal regions indicated that the distribution of the data for each class was not Gaussian. This could be understood by the low recognition rates of 86.41% for the test data and 86.77% for the training data before tuning, as listed in Table IV. One of the reasons that the distribution of the thyroid data was not Gaussian was that 15 out of the 21 inputs were discrete. To eliminate the effect of discrete inputs, we compared the performance of the neural network classifier, the fuzzy classifiers with hyperbox regions and ellipsoidal regions only using the six continuous inputs that were the first and the 17th to the 21st inputs. Table VII shows the results for the neural network classifier, the fuzzy classifier with hyperbox regions, and the fuzzy classifier with ellipsoidal regions without clustering. The training conditions of the neural network classifier were the same as those with the 21 inputs. The recognition rates of the test data dropped by about 2%; this meant that the 15 discrete inputs worked to improve the recognition rates of the test data. For the fuzzy classifier with ellipsoidal regions, the initial recognition rates improved to 94.31% for the test data and to 94.88% for the training data, and the final recognition rates were a little better than those, as listed in Table IV, when the 21 inputs were used. Thus, for the fuzzy classifier with ellipsoidal regions, the discrete inputs did not work to improve the recognition rates at all.

Table VIII shows the performance of the fuzzy classifier with ellipsoidal regions when the six continuous inputs were used. The cases are the same with those listed in Table VI. The parameter l_M was set to ten. The reason that the number of the fuzzy rules for $N_{\max} = 250$ was five and less than that for $N_{\max} = 500$ was that the clusters with more than 250 training data were not divided because of the lower bound $N_{\min} = 125$. When the training data were clustered the recognition rates of the test data were comparable to those of the neural network classifier and the fuzzy classifier with hyperbox regions.

D. Blood Cell Data

The blood cell data consist of 3097 training data and 3100 test data. The blood cell classification involves classifying optically screened white blood cells into 12 classes using 13 features. This is a very difficult problem; class boundaries for some classes are ambiguous because the classes are defined according to the growth stages of blood white cells.

Table IX shows the results for the neural network classifier, the fuzzy classifier with polyhedron regions [4], the fuzzy classifier with hyperbox regions, and the fuzzy classifier with ellipsoidal regions. The neural network classifier with 15 hidden units was trained 25 times with different initial weights; the number of epochs was 15 000 which required 133 min of CPU time. The fuzzy classifier with polyhedron regions was derived from the neural network classifier. This derivation was done within a minute. The fuzzy classifier with polyhedron regions had the best recognition rate among the three classifiers. The recognition rate of the fuzzy classifier with hyperbox regions was 3–4% lower, but the fuzzy rule extraction was extremely fast. For the fuzzy classifier with ellipsoidal regions, the recognition rate of the test data, after fuzzy rule tuning, was comparable to the maximum recognition rate of the fuzzy classifier with polyhedron regions, while the computation time was more than 200 times faster and the number of rules was only one-twentyfifth.

Table X shows the performance of the fuzzy classifier with ellipsoidal regions when l_M was changed. The training data

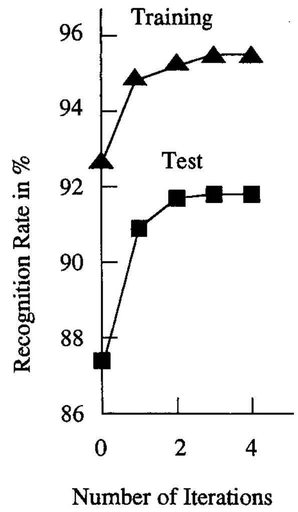


Fig. 9. Convergence process of the blood cell data ($l_M = 10$).

were not clustered; hence, the number of fuzzy rules was 12. The initial recognition rate of the test data exceeded that of the fuzzy classifier with hyperbox regions and the average recognition rate of the neural network classifier. When l_M was larger than five, the recognition rates of the test data were almost the same. Thus, in this case also, it was reasonable to set $l_M = 10$. Fig. 9 shows the convergence process when $l_M = 10$. At the first iteration, the recognition rates of both the training and test data were improved drastically and subsequent improvement by the following iterations was small.

To check the effectiveness of the nondiagonal covariance matrices, we evaluated the recognition rate of the test data with $l_M = 10$ and setting the off-diagonal elements of the covariance matrices to zero. The initial recognition rate was 80.10% and by tuning the recognition rate was improved to 84.19%. The recognition rate was less than 7% below that with the nondiagonal covariance matrices. This meant that the principal axes of the distribution of the blood cell data belonging to a class were not parallel to the input variables and this explained the reason why the recognition rate of the test data by the fuzzy classifier with hyperbox regions was not good, although the recognition rate of the training data was 100%—namely, overfitting occurred.

VII. Discussion

Except for the thyroid data, without clustering the training data, the fuzzy classifier with ellipsoidal regions we have proposed achieved a recognition rate of test data that was comparable to or better than the maximum performance obtained by the neural network classifier, the fuzzy classifier with polyhedron regions, or the fuzzy classifier with hyperbox regions. Especially for the blood cell data, the recognition rate of the test data outperformed that of the fuzzy classifier with hyperbox regions. For the fuzzy classifier with hyperbox regions, the recognition rate of the training data was always 100% if there were no identical training data in different classes. But since the principal axes of the distribution of the training data belonging to a class were not parallel to the input variables, a 100% recognition rate of the training data led to inferior recognition rate of the test data; namely, it caused overfitting. Although the recognition rate of the training data by the fuzzy classifier with ellipsoidal regions was 95.41% at best, as listed in Table X, the drop of the recognition rate of the test data was not significant when l_M was larger than one. As discussed in Section VI-D, this was because by the nondiagonal covariant matrices the robust classifier was created even when the principal axes of the distribution of the training data were not parallel to the input variables.

For the thyroid data, the fuzzy classifier with hyperbox regions outperformed the neural network classifier and the fuzzy classifier with ellipsoidal regions. The differences of the thyroid data from other data were that 1) most of the inputs were discrete; 2) more than 92% of the data belonged to one class and, hence, the numbers of data for the other two classes were small; and 3) the centers of classes were very close to one another. One of the reasons that the fuzzy classifier with ellipsoidal regions showed poor performance was that because of the discrete inputs, the distribution of the training data belonging to a class was not Gaussian. When we deleted the discrete inputs, the recognition rate of the test data was still slightly lower than that of the fuzzy classifier with hyperbox regions. For the fuzzy classifier with hyperbox regions, the recognition rate of the training data was always 100%, as stated above. For the thyroid data this worked to improve the recognition rate of the test data since characteristics of the training data and test data were similar. While for the fuzzy classifier with ellipsoidal regions, since the centers of the different classes were so close, the ellipsoidal regions of the different classes overlapped significantly; thus, the recognition rate of the training data was 96.92% at best when the six continuous inputs were used (see Table VIII).

From the above discussions, we note the fuzzy classifier with ellipsoidal regions will perform well when the distribution of the training data belonging to a class is Gaussian and it is especially advantageous over the fuzzy classifier with hyperbox regions when the principal axes of the distribution of the training data belonging to a class are not parallel to the input variables.

Development of a pattern classification system needs repetition of training, by changing the input variables to determine the necessary input variables for classification, changing the training data and the test data or gathering additional data to obtain high generalization ability, and even by changing classifiers since no single classifier can give high generalization ability for all classification problems. Thus, the training time of the classifier used in the development needs to be as short as possible. The training of the fuzzy classifier with ellipsoidal regions is slower than that of the fuzzy classifier with hyperbox regions, but much faster than that of the neural network classifier or the fuzzy classifier with polyhedron regions and the training time is not a problem. Thus, the fuzzy classifiers with ellipsoidal regions and hyperbox regions are suited for the development of large-scale classification problems.

VIII. Conclusions

We discussed a fuzzy classifier with ellipsoidal regions which has a learning capability. First, we divided the training data for each class into several clusters. Then, for each cluster, we defined a fuzzy rule with an ellipsoidal region around a cluster center. Then we tuned the fuzzy rules successively until there was no improvement in the recognition rate of the training data. We evaluated our method using the Fisher iris data, numeral data of vehicle license plates, thyroid data, and blood cell data. The recognition rates, except for the thyroid data, of our classifier were comparable to the maximum recognition rates of the neural network classifier, and the training times, except for the iris data, were two to three orders of magnitude shorter. The fuzzy classifier with ellipsoidal regions is especially suited when the distribution of the training data belonging to a class is Gaussian and the principal axes of the distribution of the training data belonging to a class are not parallel to the input variables.

ACKNOWLEDGMENT

The authors would like to thank Prof. N. Matsuda of Kawasaki Medical School, Japan, for providing the blood cell data and P. M. Murphy and D. W. Aha of the University of California at Irvine for organizing the data bases including the thyroid data. They would also like to thank the anonymous reviewers for their constructive comments.

REFERENCES

[1] P. K. Simpson, "Fuzzy min-max neural networks—Part 1: Classification," *IEEE Trans. Neural Networks,* vol. 3, pp. 776–786, May 1992.
[2] L.-X. Wang and J. M. Mendel, "Generating fuzzy rules by learning from examples," *IEEE Trans. Syst., Man, Cybern.,* vol. 22, pp. 1414–1427, Nov./Dec. 1992.
[3] S. Abe and M.-S. Lan, "A method for fuzzy rules extraction directly from numerical data and its application to pattern classification," *IEEE Trans. Fuzzy Syst.,* vol. 3, pp. 18–28, Feb. 1995.
[4] F. Uebele, S. Abe, and M.-S. Lan, "A neural network-based fuzzy classifier," *IEEE Trans. Syst., Man, Cybern.,* vol. 25, pp. 353–361, Feb. 1995.
[5] M. T. Musavi, W. Ahmed, K. H. Chan, K. B. Faris, and D. M. Hummels, "On the training of radial basis function classifiers," *Neural Networks,* vol. 5, pp. 595–603, Apr. 1992.
[6] M. P. Windham, "Cluster validity for the fuzzy c-means clustering algorithm," *IEEE Trans. Pattern Anal. Mach. Intell.,* vol. 4, pp. 357–363, July 1982.
[7] R. Krishnapuram and J. M. Keller, "A possibilistic approach to clustering," *IEEE Trans. Fuzzy Syst.,* vol. 1, pp. 98–110, May 1993.
[8] J. C. Bezdek and R. J. Hathaway, "Convergence theory for fuzzy c-means: Counterexamples and repairs," *IEEE Trans. Syst., Man, Cybern.,* vol. 17, pp. 873–877, Sept./Oct. 1987.
[9] T. Kohonen, *Self-Organization and Associative Memory,* 2nd Ed. Berlin, Germany: Springer-Verlag, 1987.
[10] R. Fisher, "The use of multiple measurements in taxonomic problems," *Ann. Eugenics,* vol. 7, pt. II, pp. 179–188, 1936.
[11] M. Takatoo, M. Kanasaki, T. Mishima, T. Shibata, and H. Ota, "Gray scale image processing technology applied to vehicle license number recognition system," in *Proc. Int. Workshop Indust. Applicat. Mach. Vision Mach. Intell.,* Tokyo, Japan, Feb. 1987, pp. 76–79.
[12] H. Takenaga, S. Abe, M. Takatoo, M. Kayama, T. Kitamura, and Y. Okuyama, "Input layer optimization of neural networks by sensitivity analysis and its application to recognition of numerals," *Trans. Inst. Elect. Eng. Jpn,* vol. 111-D, no. 1, pp. 36–44, 1991 (in Japanese); transl. English, Scripta Technica, Inc., *Elect. Eng. Jpn.,* vol. 111, no. 4, pp. 130–138, 1991.
[13] S. M. Weiss and I. Kapouleas, "An empirical comparison of pattern recognition, neural nets, and machine learning classification methods," in *Proc. Int. Joint Conf. Artificial Intell.,* Detroit, MI, Aug. 1989, pp. 781–787.
[14] A. Hashizume, J. Motoike, and R. Yabe, "Fully automated blood cell differential system and its application," in *Proc. Int. Union Pure Appl. Chem. 3rd Int. Congress Automat. New Technol. Clinical Lab.,* Kobe, Japan, Sept. 1988, pp. 297–302.

Shigeo Abe (M'79–SM'83) received the B.S. degree in electronics engineering, the M.S. degree in electrical engineering, and the D.Eng. degree, all from Kyoto University, Kyoto, Japan, in 1970, 1972, and 1984, respectively.

Since 1972, he has been with Hitachi Research Laboratory, Hitachi, Ltd., Ibaraki-ken, Japan, and has been engaged in power-system analysis, development of a vector processor, a Prolog processor, neural network theories, and fuzzy system models. From 1978 to 1979 he was a Visiting Research Associate at the University of Texas at Arlington. He is the author of *Neural Networks and Fuzzy Systems: Theory and Applications* (Norwell, MA: Kluwer, 1996).

Dr. Abe was awarded an outstanding paper prize from the Institute of Electrical Engineers of Japan in 1984 and 1995. He is a member of the International Neural Network Society, the Institute of Electrical Engineers of Japan, the Information Processing Society of Japan, the Institute of Electronics, Information and Communication Engineers of Japan, the Society of Instrument and Control Engineers of Japan, and the National Geographic Society.

Ruck Thawonmas (M'97) received the B.Eng. degree in electrical engineering from Chulalongkorn University, Bangkok, Thailand, in March 1987, the M.Eng. degree in information science from Ibaraki University, Hitachi, Japan, in March 1990, and the D.Eng. degree in information engineering from Tohoku University, Sendai, Japan, in January 1994.

From January 1994 to March 1996, he was a Visiting Researcher under the Hitachi Research Visit Programs (HIVIPS) at the Hitachi Research Laboratory, Hitachi, Ltd., Hitachi, Japan. From April 1996 to August 1997 he was with the Institute of Physical and Chemical Research (RIKEN), Wako, Japan, as a Special Postdoctoral Researcher under the Special Postdoctoral Researchers Program in the Laboratory for Artificial Brain Systems Brain Information Processing Group, Frontier Research Program. Since September 1997 he has been with the University of Aizu, Aizu-Wakamatsu, Japan, as an Assistant Professor in the Department of Computer Hardware, Multimedia Devices Laboratory. He is an author and co-author of more than 20 peer-reviewed international journal papers and conference papers. His research interests are in neural networks and, most recently, their applications to signal processing with incomplete data such as a blind separation problem. His other interests include fuzzy systems and computational intelligence models.

Dr. Thawonmas was a recipient of the Japanese Government (Monbusho) Scholarship from April 1987 to March 1993 (during his graduate studies), the Asahi Glass Company Scholarship, and the Tohoku Kaihatsu Memorial Foundation Research Grant from April 1993 to January 1994.