International Journal of Applied Science and Engineering
2006. 4, 1: 41-52

Generating Weighted Fuzzy Rules from Training Data for Dealing with the Iris Data Classification Problem

Yung-Chou Chen[a], Li-Hui Wang[b], and Shyi-Ming Chen[c*]

[a] *Department of Electronic Engineering, National Taiwan University of Science and Technology, Taipei 106, Taiwan, R.O.C.*
[b] *Department of Finance, Chihlee Institute of Technology, Banciao City, Taipei County 220, Taiwan, R.O.C.*
[c] *Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology, Taipei 106, Taiwan, R.O.C.*

Abstract: The most important task in the design of fuzzy classification systems is to find a set of fuzzy rules from training data to deal with a specific classification problem. In this paper, we present a new method to generate weighted fuzzy rules from training data to deal with the Iris data classification problem. First, we convert the training data to fuzzy rules, and then we merge those fuzzy rules in order to reduce the number of fuzzy rules. Then, we calculate the weight of each input variable appearing in the generated fuzzy rules by the relationships of input variables. The proposed weighted fuzzy rules generation method gets a higher average classification accuracy rate than the existing methods.

Keywords: fuzzy classification systems; fuzzy sets; Iris data; membership functions; weighted fuzzy rules.

1. Introduction

One of important applications of fuzzy set theory [22] is in the fuzzy classification systems. There are two approaches to obtain fuzzy rules for fuzzy classification systems. One of them is given directly by experts; the other is produced through an automatic learning process. In recent years, some methods [1-10, 12-18, 20-21] have been presented to generate fuzzy rules from training instances.

In [1], Castro et al. presented a method for learning maximal structure rules for dealing with the Iris data [11] classification problem. In [2], Castro et al. presented an inductive learning algorithm in fuzzy systems. In [3], Chang et al. presented a method to generate fuzzy rules from numerical data based on the exclusion of attribute terms for dealing with the Iris data classification problem. In [4], Chen et al. presented a method to generate fuzzy rules from numerical data for handling the Iris data classification problem. In [5], Chen et al. presented a method to generate fuzzy rules from relational database systems for estimating null values. In [6], Chen et al. presented a method for constructing fuzzy decision trees and generating fuzzy classification rules from training examples. In [9],

* Corresponding author; e-mail: smchen@et.ntust.edu.tw *Accepted for Publication: August 17, 2005*

Chen et al. used clustering techniques to handling the Iris data classification problem. In [12], Hayushi et al. presented a fuzzy neural expert system with automated extraction of fuzzy if-then rules from a trained neural network. In [14], Kasabov presented a method for learning fuzzy rules and approximate reasoning in fuzzy neural networks and hybrid systems. In [15], Lin et al. presented a method for generating weighted fuzzy rules from training data for handling fuzzy classification problems. In [16], Nozaki et al. presented a heuristic method for generating fuzzy rules from numerical sets. In [17], Wang et al. presented a method for generating fuzzy rules by learning from examples. In [18], Wu et al. presented a method for constructing membership functions and fuzzy rules from training examples for handling the Iris data classification problem. In [20], Yuan et al. presented a method for fuzzy rules generation based on the induction of fuzzy decision trees. In [21], Yuan et al. presented a genetic algorithm for generating fuzzy classification rules.

In this paper, we present a new method to generate weighted fuzzy rules from a set of training data to deal with the Iris data [11] classification problem. The proposed method is an extension of the fuzzy rule generation method presented in [1], but it generates less fuzzy rules and can get a higher average classification accuracy rate than the fuzzy rule generation method presented in [1]. First, we convert the training data into fuzzy rules, and then we merge these fuzzy rules in order to reduce the number of fuzzy rules. Furthermore, because each input variable appearing in the generated fuzzy rules may have a different degree of importance, we present a method to calculate the weight of each input variable appearing in antecedent parts of the generated fuzzy rules. The proposed weighted fuzzy rules generation method gets a higher average classification accuracy rate than the ones presented in [1] and [9].

The rest of this paper is organized as follows. In Section 2, we briefly review Castro's method for fuzzy rules generation from [1]. In Section 3, present a new method to generate weighted fuzzy rules from training data. In Section 4, we use an example to illustrate the proposed method. In Section 5, we show the experimental results of the proposed method. The conclusions are discussed in Section 6.

2. A review of Castro's fuzzy rules generation method

In this section, we briefly review the fuzzy rules generation method presented by Castro et al. [1]. Assume that there is a training data set Π containing

$$\Pi = \{\boldsymbol{a_1}, \boldsymbol{a_2}, \cdots, \boldsymbol{a_m}\},$$

where each datum $\boldsymbol{a_i}$ in the data set Π has multiple input values and a single output value represented as follows:

$$\boldsymbol{a_i} = ((x_{i1}, x_{i2}, \cdots, x_{in}), y_i),$$

where x_{ij} denotes the value of the jth input variable of the ith training datum $\boldsymbol{a_i}$; y_i denotes the value of the output variable of the ith training datum, where $1 \le i \le m$ and n is the number of input variables.

The form of a fuzzy rule is shown as follows:

IF X_1 is L_1 AND X_2 is L_2 AND $\cdots$ AND X_n is L_n **THEN** Y is y_j, (1)

where X_1, X_2, $\cdots$, and X_n are input variables; L_1, L_2, $\cdots$, and L_n are the set of labels represented by membership functions; Y is the output variable; y_j is an output value of the output variable Y. The fuzzy rule shown in Eq. (1) can be represented as follows:

$$((L_1, L_2, \cdots, L_n), y_j).$$

Castro's method for fuzzy rules generation can be divided into two main parts. One of them is to deal with training and learning; the other is to deal with classification. We describe them as follows:

2.1. Training and learning

(1) To convert the numerical data into fuzzy rules: By converting all numerical values of the initial training data set into fuzzy rules and put them in the set of initial rules. We divide the domain of each input variable into several equal length intervals and each interval is associated with a membership function. Then, we assign each membership function a label to represent it. If a value x_j in the membership function associated with label L_j has the largest membership value, then the value x_j is converted into L_j.

(2) To establish a set of definitive rules: At the beginning, we have an empty set of definitive rules and then we take a fuzzy rule R from the set of initial rules. We check if there are any fuzzy rules in the set of definitive rules that can subsume [1] in the fuzzy rule R. Let fuzzy rule $A = ((L_1, L_2, \cdots, L_n), y_i)$ and let fuzzy rule $B = ((L'_1, L'_2, \cdots, L'_n), y_j)$. Fuzzy rule A subsumes in fuzzy rule B, if and only if $L_1 \subseteq L'_1$, $L_2 \subseteq L'_2$, $\cdots$, $L_n \subseteq L'_n$, and $y_i = y_j$. If there are some fuzzy rules in the set of definitive rules that can subsume in the fuzzy rule R, then we ignore them and take another fuzzy rule from the set of initial rules. If there are no fuzzy rules in the set of definitive rules, then we perform the amplification process [1] to the fuzzy rule R until we can not amplify this fuzzy rule any more, and then put this amplified fuzzy rule in the set of definitive rules. After this, we take another fuzzy rule from the set of initial rules again and repeat all previous actions until all fuzzy rules in the set of initial rules are taken out.

Example 2.1: Assume that there is a set of labels {N, Z, P} and assume that there is a fuzzy rule R: ((N, Z, P), 1), then the amplification of the set of labels are shown as follows [1]:

((N, Z, P), 1), (({N, Z}, Z, P), 1), (({N, Z, P}, Z, P), 1), (({N, Z, P}, {N, Z}, P), 1), (({N, Z, P}, {N, Z, P}, P), 1), (({N, Z, P,}, {N, Z, P}, {N, P}), 1), (({N, Z, P,}, {N, Z, P}, {N, Z, P}), 1).

Definition 2.1 [1]***:*** If amplification from fuzzy rule R to fuzzy R' is possible, where fuzzy rule $R' = ((L'_1, L'_2, \cdots, L'_n), y_j)$, then there is no fuzzy rule R'', where fuzzy rule $R'' = ((L''_1, L''_2, \cdots, L''_n), y_j)$, that $L''_1 \subseteq L'_1$, $L''_2 \subseteq L'_2$, $\cdots$, $L''_n \subseteq L'_n$ and $y_i \neq y_j$.

Finally, we can get a set of fuzzy rules that can be used to deal with the classification problem from the set of definitive rules.

2.2. Processing classification

(1) Convert the numerical values of the testing data into labels, where the labels represent the membership functions.

(2) Processing the fuzzy rules: by comparing all converted data derived in the last step to the generated fuzzy rules, we can get the following two states:

(i) If it subsumes in one or more fuzzy rules and those fuzzy rules have the same output, then the output is the classification result.

(ii) If it subsumes in two or more fuzzy rules and those fuzzy rules do not have the same output, then we must calculate the degree of convenience [1] of this data for fuzzy rules. The way to calculate the degree of convenience of one rule is to map each input value to the membership functions, calculate the corresponding membership values, and take the minimal value of all the membership values as the degree of convenience, i.e., Degree of Convenience $= \min\{\varphi_i(x_i)\}$.

The classification result is the output of the fuzzy rule having the maximal value of degree of convenience.

3. A new method to generate weighted fuzzy rules from training data

In this section, we present a new algorithm to generate weighted fuzzy rules from training

data to deal with the Iris data [11] classification problem, where the data that we deal with are multiple input and single output data (MISO). Assume that there are m instances (i.e. $\boldsymbol{a_1}$, $\boldsymbol{a_2}$, $\cdots$, $\boldsymbol{a_m}$) of data to be trained. Assume that the ith training data $\boldsymbol{a_i}$ has n input values and one output value, shown as follows:

$$a_i = ((x_{i1}, x_{i2}, ..., x_{in}), y_i),$$

where x_{ij} is the jth input value of input variable X_j of the ith training data $\boldsymbol{a_i}$, and x_{ij} is a real number; y_i is the value of output variable Y of the ith training data, where $i = 1, 2, \ldots, m$ and $j = 1, 2, \cdots, n$.

Let I be a set of initial rules. Based on [1], we can convert the initial training data into fuzzy rules one to one and put them into the set of initial rules. In other words, each initial training datum is converted into a fuzzy rule. By this step, we can build a set of initial rules, which have the same number of fuzzy rules as the number of data in the initial training data. We describe the method of conversion as follows. Assume that we want to convert each input value of every initial training datum into a label, which represents a membership function. First, we get an initial training datum $\boldsymbol{a_i}$ that is not converted yet. Then, we get the jth input value of $\boldsymbol{a_i}$, denoted by x_{ij}, which is not converted. Assume that the domain of input variable X_j was divided into z parts (all parts are equal). If $z = 7$, we can define 7 labels, as shown in Figure 1, where each label represents a membership function.

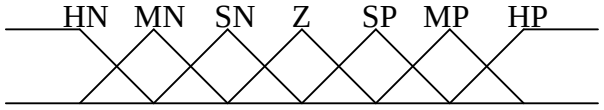

Figure 1. Membership functions of the corresponding labels.

Then, we map x_{ij} into every membership function that its label represents, shown in Figure 1, and calculate the membership values respectively. We compare all of the membership values, take the label which has the maximal membership value, and replace the numerical value x_{ij} by the label. Then, we take the value of another input variable and repeat the action described above until all of the values of input variables are converted into labels. Then, we can get a fuzzy rule which is converted from $\boldsymbol{a_i}$, and we put this rule into the set of initial fuzzy rules. After this step, we take another initial training datum which has not been converted, repeat the step described above until the all of the training data are converted into fuzzy rules and these fuzzy rules are put into the set of initial rules.

Then, we deal with the fuzzy rules converted from the initial training data set. First, we take a fuzzy rule R from the set of initial rules. If the set of definitive rules is empty, then we let the fuzzy rule R be a member of the set of definitive rules; if the set of definitive rules is not empty, then we take one fuzzy rule R' which has the same output as the fuzzy rule R and merge it with the fuzzy rule R. The action of merging is defined as follows.

Definition 3.1: Assume that fuzzy rule $A = ((L_1, L_2, \cdots, L_n), y_i)$ and fuzzy rule $B = ((L'_1, L'_2, \cdots, L'_n), y_j)$. Then, fuzzy rule A and fuzzy rule B are merged into fuzzy rule C, where $C = ((L''_1, L''_2, \cdots, L''_n), y_k)$, if and only if $y_i = y_j$. Then $L''_1 = L_1 \cup L'_1$, $L''_2 = L_2 \cup L'_2$, $\cdots$, and $L''_n = L_n \cup L'_n$, and $y_k = y_i = y_j$, where "$\cup$" is the union operator.

Example 3.1: Assume that fuzzy rule A and fuzzy rule B are merged into fuzzy rule C, where

A = (({SN, Z}, {HP}, {HN, SP, MP, HP}, {HN}), 1),

B = (({HN, MN, SN}, {Z}, {HN, MN}, {HN, MN, SP}), 1),

then

C = (({HN, MN, SN, Z},{HP, Z},{HN, MN, SP, MP, HP}, {HN, MN, SP}), 1).

Definition 3.2: Assume that fuzzy rule $A = ((L_1, L_2, \cdots, L_n), y_i)$ and fuzzy rule $B = ((L'_1, L'_2, \cdots, L'_n), y_j)$. Then, fuzzy rule A and fuzzy rule B are in collision, if and only if $y_i \neq y_j$, $L_1 \cap L'_1 \neq \phi$, $L_2 \cap L'_2 \neq \phi$, $\cdots$, and $L_n \cap L'_n \neq \phi$, where ϕ denotes the empty set.

Definition 3.3: The merge of fuzzy rule A and fuzzy rule B into fuzzy rule C is allowed, if and only if fuzzy rule C does not collide with any of the fuzzy rules which are in the set of initial rules.

If the merge of fuzzy rule R with fuzzy rule R' is allowed, then the fuzzy rule R is merged with fuzzy rule R' into fuzzy rule R'', and we use fuzzy rule R'' to replace R' which is in the set of definitive rules. If the merge of fuzzy rule R with fuzzy rule R' is not allowed, then take another fuzzy rule which was not taken before and have the same output with fuzzy rule R and repeat the process described above. If the fuzzy rule R can't merge with any fuzzy rules in the set of definitive rules, then the fuzzy rule R becomes a new member of the set of definitive rules. Repeat those processes until all of the fuzzy rules in the set of initial rules are processed. Finally, we can obtain a set of fuzzy rules from the set of definitive rules to be used for reasoning.

In the following, we present an algorithm to generate fuzzy rules from a set of training data. The algorithm is now presented as follows:

Step 1: Convert each training data in the initial training data set into a fuzzy rule and put them into the set of initial rules.

Step 2: **If** the set of initial rules is empty or all of the fuzzy rules in the set of initial rules have been taken **then Stop**;
else take a fuzzy rule R form the set of initial rules.

Step 3: **If** the set of definitive rules is empty or all of the fuzzy rules which are in the set of definitive rules that have the same output with fuzzy rule R have attempted to merge with fuzzy rule R **then** fuzzy rule R becomes one member of the set of definitive rules;
else go to Step 4.

Step 4: Take a fuzzy rule R' which has the same output with fuzzy rule R and has not attempted to merge with fuzzy rule R from the set of definitive rules;
If the merge of fuzzy rule R with fuzzy rule R' is allowed **then** merge them into fuzzy rule R'' and replace R' by fuzzy rule R'', and go to Step 2
else go to Step 3.

The set of definitive rules is produced by the above four steps. Now, we use those fuzzy rules that are in the set of definitive rules, for classification. First, we convert the testing datum into labels, and then we test if it is subsumed in the fuzzy rules that are in the set of definitive rules. We can obtain two cases:

(1) If the testing datum subsumes one or more fuzzy rules, and those fuzzy rules have the same output, then this output is the classification result.

(2) If the testing datum can not subsume in any fuzzy rules or subsume in two or more fuzzy rules which do not have the same output, then we must use Definition 3.4 to calculate the degree of weighted convenience. The output of a fuzzy rule that has the maximal value of degree of weighted convenience is the classification result. The definition of the degree of weighted conveniences is presented as follows.

Definition 3.4: Assume that there exists a testing datum $T = (x_1, x_2, \cdots, x_n)$ and a fuzzy rule $R = ((L_1, L_2, \cdots, L_n), y_k)$. The degree of weighted convenience is the summation of the multiplications of the membership value $\varphi_{L_i}(x_i)$ and the individual weight w_i of input variable X_i, i.e.,

Degree of Weighted Convenience of fuzzy rule R = $\sum_{i=1}^{n} \varphi_{L_i}(x_i) * w_i$, (2)

where $\varphi_{L_i}(x_i)$ denotes the degree of membership of x_i in the label L_i. The method to calculate the weight of each input variable is presented as follows. First, we find out the whole domain *WD* of the input variable X_i. Then, we find out the individual domain of the input variable X_i for each type of output. Let *PD* be formed by a set of intervals I_1, I_2, $\cdots$, I_p which are not overlapping with the domain of the input attribute variable X_i for each type of classification output. Then, let

$$v_i = \frac{|PD|}{|WD|}, \quad (3)$$

where $|PD| = |I_1| + |I_2| + \cdots + |I_p|$; $|I_j|$ denotes the length of the interval I_j, for $1 \le j \le p$; |WD| denotes the length of the whole domain WD, and let

$$w_i = \left(\frac{v_i}{\max(v_1, v_2, ..., v_n)} \right)^2, \quad (4)$$

where w_i is the weight of the input attribute variable X_i and $1 \le i \le n$.

4. An example

In this section, we apply the proposed method to deal with the Iris data [11] classification problem. The Iris data contains 150 instances as shown in Table 1. There are three species of the Iris data, i.e., Iris-Setosa, Iris-Versicolor and Iris-Virginica. The Iris data have 4 input attributes (i.e., Sepal Length (SL), Sepal Width (SW), Petal Length (PL) and Petal Width (PW)) and one output attribute. The characteristics of the input attributes of the Iris data are shown in Table 2.

The output attribute of the Iris data consists of three types of output values, as shown in Table 3.

In order to clearly illustrate the proposed fuzzy rules generation algorithm, we only chose 15 instances from the Iris data for illustration, where 5 instances for each species (i.e. Setosa, Versicolor and Verginica) are chosen. The chosen instances for this example are shown in Table 4.

We assume that the number of labels for each input attribute is 7, i.e., HN, MN, SN, Z, SP, MP and HP. In [1], the membership functions of the input attributes SL, PW, PL and SW can be defined as shown in Figure 2, Figure 3, Figure 4 and Figure 5, respectively.

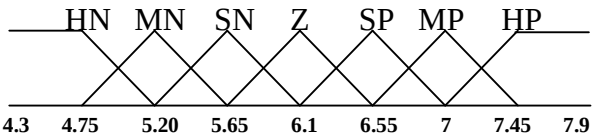

Figure 2. Membership functions of the input attribute SL [1].

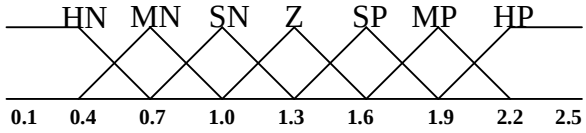

Figure 3. Membership functions of the input attribute PW [1].

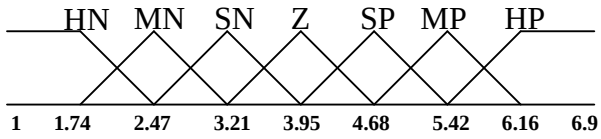

Figure 4. Membership functions of the input attribute PL [1].

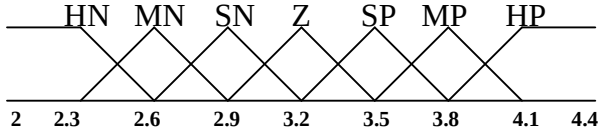

Figure 5. Membership functions of the input attribute SW [1].

Table 1. Iris data [11].

Iris-Setosa					Iris-Versicolor					Iris-Virginica				
SL	SW	PL	PW	Output	SL	SW	PL	PW	Output	SL	SW	PL	PW	Output
5.1	3.5	1.4	0.2	1	7.0	3.2	4.7	1.4	2	6.3	3.3	6.0	2.5	3
4.9	3.0	1.4	0.2	1	6.4	3.2	4.5	1.5	2	5.8	2.7	5.1	1.9	3
4.7	3.2	1.3	0.2	1	6.9	3.1	4.9	1.5	2	7.1	3.0	5.9	2.1	3
4.6	3.1	1.5	0.2	1	5.5	2.3	4.0	1.3	2	6.3	2.9	5.6	1.8	3
5.0	3.6	1.4	0.2	1	6.5	2.8	4.6	1.5	2	6.5	3.0	5.8	2.2	3
5.4	3.9	1.7	0.4	1	5.7	2.8	4.5	1.3	2	7.6	3.0	6.6	2.1	3
4.6	3.4	1.4	0.3	1	6.3	3.3	4.7	1.6	2	4.9	2.5	4.5	1.7	3
5.0	3.4	1.5	0.2	1	4.9	2.4	3.3	1.0	2	7.3	2.9	6.3	1.8	3
4.4	2.9	1.4	0.2	1	6.6	2.9	4.6	1.3	2	6.7	2.5	5.8	1.8	3
4.9	3.1	1.5	0.1	1	5.2	2.7	3.9	1.4	2	7.2	3.6	6.1	2.5	3
5.4	3.7	1.5	0.2	1	5.0	2.0	3.5	1.0	2	6.5	3.2	5.1	2.0	3
4.8	3.4	1.6	0.2	1	5.9	3.0	4.2	1.5	2	6.4	2.7	5.3	1.9	3
4.8	3.0	1.4	0.1	1	6.0	2.2	4.0	1.0	2	6.8	3.0	5.5	2.1	3
4.3	3.0	1.1	0.1	1	6.1	2.9	4.7	1.4	2	5.7	2.5	5.0	2.0	3
5.8	4.0	1.2	0.2	1	5.6	2.9	3.6	1.3	2	5.8	2.8	5.1	2.4	3
5.7	4.4	1.5	0.4	1	6.7	3.1	4.4	1.4	2	6.4	3.2	5.3	2.3	3
5.4	3.9	1.3	0.4	1	5.6	3.0	4.5	1.5	2	6.5	3.0	5.5	1.8	3
5.1	3.5	1.4	0.3	1	5.8	2.7	4.1	1.0	2	7.7	3.8	6.7	2.2	3
5.7	3.8	1.7	0.3	1	6.2	2.2	4.5	1.5	2	7.7	2.6	6.9	2.3	3
5.1	3.8	1.5	0.3	1	5.6	2.5	3.9	1.1	2	6.0	2.2	5.0	1.5	3
5.4	3.4	1.7	0.2	1	5.9	3.2	4.8	1.8	2	6.9	3.2	5.7	2.3	3
5.1	3.7	1.5	0.4	1	6.1	2.8	4.0	1.3	2	5.6	2.8	4.9	2.0	3
4.6	3.6	1.0	0.2	1	6.3	2.5	4.9	1.5	2	7.7	2.8	6.7	2.0	3
5.1	3.3	1.7	0.5	1	6.1	2.8	4.7	1.2	2	6.3	2.7	4.9	1.8	3
4.8	3.4	1.9	0.2	1	6.4	2.9	4.3	1.3	2	6.7	3.3	5.7	2.1	3
5.0	3.0	1.6	0.2	1	6.6	3.0	4.4	1.4	2	7.2	3.2	6.0	1.8	3
5.0	3.4	1.6	0.4	1	6.8	2.8	4.8	1.4	2	6.2	2.8	4.8	1.8	3
5.2	3.5	1.5	0.2	1	6.7	3.0	5.0	1.7	2	6.1	3.0	4.9	1.8	3
5.2	3.4	1.4	0.2	1	6.0	2.9	4.5	1.5	2	6.4	2.8	5.6	2.1	3
4.7	3.2	1.6	0.2	1	5.7	2.6	3.5	1.0	2	7.2	3.0	5.8	1.6	3
4.8	3.1	1.6	0.2	1	5.5	2.4	3.8	1.1	2	7.4	2.8	6.1	1.9	3
5.4	3.4	1.5	0.4	1	5.5	2.4	3.7	1.0	2	7.9	3.8	6.4	2.0	3
5.2	4.1	1.5	0.1	1	5.8	2.7	3.9	1.2	2	6.4	2.8	5.6	2.2	3
5.5	4.2	1.4	0.2	1	6.0	2.7	5.1	1.6	2	6.3	2.8	5.1	1.5	3
4.9	3.1	1.5	0.1	1	5.4	3.0	4.5	1.5	2	6.1	2.6	5.6	1.4	3
5.0	3.2	1.2	0.2	1	6.0	3.4	4.5	1.6	2	7.7	3.0	6.1	2.3	3
5.5	3.5	1.3	0.2	1	6.7	3.1	4.7	1.5	2	6.3	3.4	5.6	2.4	3
4.9	3.1	1.5	0.1	1	6.3	2.3	4.4	1.3	2	6.4	3.1	5.5	1.8	3
4.4	3.0	1.3	0.2	1	5.6	3.0	4.1	1.3	2	6.0	3.0	4.8	1.8	3
5.1	3.4	1.5	0.2	1	5.5	2.5	4.0	1.3	2	6.9	3.1	5.4	2.1	3
5.0	3.5	1.3	0.3	1	5.5	2.6	4.4	1.2	2	6.7	3.1	5.6	2.4	3
4.5	2.3	1.3	0.3	1	6.1	3.0	4.6	1.4	2	6.9	3.1	5.1	2.3	3
4.4	3.2	1.3	0.2	1	5.8	2.6	4.0	1.2	2	5.8	2.7	5.1	1.9	3
5.0	3.5	1.6	0.6	1	5.0	2.3	3.3	1.0	2	6.8	3.2	5.9	2.3	3
5.1	3.8	1.9	0.4	1	5.6	2.7	4.2	1.3	2	6.7	3.3	5.7	2.5	3
4.8	3.0	1.4	0.3	1	5.7	3.0	4.2	1.2	2	6.7	3.0	5.2	2.3	3
5.1	3.8	1.6	0.2	1	5.7	2.9	4.2	1.3	2	6.3	2.5	5.0	1.9	3
4.6	3.2	1.4	0.2	1	6.2	2.9	4.3	1.3	2	6.5	3.0	5.2	2.0	3
5.3	3.7	1.5	0.2	1	5.1	2.5	3.0	1.1	2	6.2	3.4	5.4	2.3	3
5.0	3.3	1.4	0.2	1	5.7	2.8	4.1	1.3	2	5.9	3.0	5.1	1.8	3

Table 2. The characteristics of input attributes of the Iris data.

Input attributes	Minimal value (cm)	Maximal value (cm)
Sepal Length (SL)	4.3	7.9
Sepal Width (SW)	2.0	4.4
Petal Length (PL)	1.0	6.9
Petal Width (PW)	0.1	2.5

Table 3. Output values of the output attributes.

Type	Output attribute
1	Iris-Setosa
2	Iris-Versicolor
3	Iris-Virginica

Table 4. Initial training data.

Iris-Setosa					Iris-Versicolor					Iris-Virginica				
SL	SW	PL	PW	Output	SL	SW	PL	PW	Output	SL	SW	PL	PW	Output
4.6	3.4	1.4	0.3	1	6.6	2.9	4.6	1.3	2	6.1	3	4.9	1.8	3
5.7	3.8	1.7	0.3	1	5	2	3.5	1	2	6.1	2.6	5.6	1.4	3
5.2	3.4	1.4	0.2	1	6.2	2.2	4.5	1.5	2	6.9	3.1	5.4	2.1	3
4.5	2.3	1.3	0.3	1	5.9	3.2	4.8	1.8	2	6.7	3.1	5.6	2.4	3
4.4	3.2	1.3	0.2	1	6	2.9	4.5	1.5	2	6.2	3.4	5.4	2.3	3

After the membership functions of the input attributes have been defined, according to [1], we convert each training datum into a fuzzy rule, respectively. First, we take a training datum ((4.6, 3.4, 1.4, 0.3), 1) from Table 4. The value of the attribute SL is 4.6, we map the value "4.6" into the membership function HN and we can see that the membership value is 1; we map the value "4.6" into the membership MN, SN, Z, SP, MP and HP, respectively, and we can see that the membership values are 0, respectively. We can see that when we map the value "4.6" into the membership function HN, we get the largest membership value, so we convert the value "4.6" into "HN". In the same way, we can convert ((4.6, 3.4, 1.4, 0.3), 1) into (({HN}, {SP}, {HN}, {HN}), 1). Repeating the above steps, we can convert the initial training data shown in Table 4 into the set of initial rules as shown in Table 5.

Table 5. Fuzzy rules in the set of initial rules.

Iris-Setosa	Iris-Versicolor	Iris-Virginica
(({HN}, {SP}, {HN}, {HN}), 1)	(({SP}, {SN}, {SP}, {Z}), 2)	(({Z}, {SN}, {SP}, {MP}), 3)
(({SN}, {MP}, {HN}, {HN}), 1)	(({MN}, {HN}, {SN}, {SN}), 2)	(({Z}, {MN}, {MP}, {Z}), 3)
(({MN}, {SP}, {HN}, {HN}), 1)	(({Z}, {HN}, {SP}, {SP}), 2)	(({MP}, {Z}, {MP}, {HP}), 3)
(({HN}, {HN}, {HN}, {HN}), 1)	(({Z}, {Z}, {SP}, {MP}), 2)	(({SP}, {Z}, {MP}, {HP}), 3)
(({HN}, {Z}, {HN}, {HN}), 1)	(({Z}, {SN}, {SP}, {SP}), 2)	(({Z}, {SP}, {MP}, {HP}), 3)

Then, we take the fuzzy rules sequentially from the set of initial rules shown in Table 5 and perform the merging process. Finally, we can get a set of definitive fuzzy rules from the set of initial rules. There are 5 fuzzy rules in the set of definitive rules, shown as follows:

R_0: (({HN, MN, SN}, {HN, Z, SP, MP}, {HN}, {HN}), 1),

R_1: (({MN, Z, SP}, {HN, SN}, {SN, SP}, {SN, Z, SP}), 2),

R_2: (({Z}, {Z}, {SP}, {MP}), 2),

R_3: (({Z}, {MN, SN, SP}, {SP, MP}, {Z, MP, HP}), 3),

R_4: (({SP, MP}, {Z}, {MP}, {HP}), 3).

Now, we apply those fuzzy rules to deal with the classification. We use an instance (4.7, 3.2, 1.3, 0.2) of the Iris data shown in Table 1 as a testing datum to illustrate the classification process. First, we convert this testing datum into ({HN}, {Z}, {HN}, {HN}). We take this converted datum into all of the generated fuzzy rules, respectively, and then we can see that only "R_0: (({HN, MN, SN}, {HN, Z, SP, MP}, {HN}, {HN}), 1)" can subsume in it (because $\{HN\} \subseteq \{HN, MN, SN\}$, $\{Z\} \subseteq \{HN, Z, SP, MP\}$, $\{HN\} \subseteq \{HN\}$ and $\{HN\} \subseteq \{HN\}$). Thus, the classification result indicates that it is belonging to Type 1, i.e., Iris-Setosa. From Table 1, we can see that it is a correct classification.

We use the other instance (4.4, 2.9, 1.4, 0.2) of the Iris data shown in Table 1 as the testing datum. First, we convert this testing datum into ({HN}, {SN}, {HN}, {HN}). We take this converted testing datum into all of the generated fuzzy rules, respectively, and then we can see that no fuzzy rule can subsume in it. Thus, we must calculate the degree of weighted convenience of each generated fuzzy rule.

First, we calculate the weight w_i of each input attribute, for $1 \le i \le 4$, shown as follows.

According to all of the input values of the input attributes SL, SW, PL, and PW shown in Table 4, based on Eq. (3), we have $v_1 = 0.52$, $v_2 = 0.39$, $v_3 = 1$, and $v_4 = 0.73$, respectively.

Because of $\max(v_1, v_2, v_3, v_4) = v_3 = 1$, by Eq. (4), we obtain:

$$w_1 = (0.52/1)^2 = 0.27,$$
$$w_2 = (0.39/1)^2 = 0.15,$$
$$w_3 = (1/1)^2 = 1,$$
$$w_4 = (0.73/1)^2 = 0.54.$$

Then, we calculate the degree of weighted convenience of fuzzy rule R_0,

R_0: (({HN, MN, SN}, {HN, Z, SP, MP}, {HN}, {HN}), 1).

According to the previous discussions, we have

$$\varphi_{01}(4.4) = 1,$$
$$\varphi_{02}(2.9) = 0,$$
$$\varphi_{03}(1.4) = 1,$$
$$\varphi_{04}(0.2) = 1.$$

Therefore, based on Eq. (2), we can calculate the degree of weighted convenience of the fuzzy rule R_0 shown as follows:

0.27 * 1 + 0.15 * 0 + 1 * 1 + 0.54 * 1 = 1.81.

Similarly, we can see that the degrees of weighted convenience of the fuzzy rules R_1, R_2, R_3 and R_4 are 0.15, 0, 0.15 and 0, respectively.

From the above calculations, we can see that the degree of weighted convenience of the fuzzy rule R_0 has the maximal value. Thus, the output of the fuzzy rule R_0 is the classification result. Because the output of the fuzzy rule R_0 is Type 1: Iris-Setosa, we can see that the classification result of the testing datum (4.4, 2.9, 1.4, 0.2) is Iris-Setosa. From Table 1, we can see that it is a correct classification.

5. Experimental results

We have implemented the proposed method on a Pentium 4 PC by using Borland C++ Build Version 5.0 to deal with the Iris data classification. In [1], Castro et al. used 120 instances of the Iris data as the training data set, and the other 30 instances of Iris data as the testing data set. The average classification accuracy rates of the Castro's method with different numbers of labels are shown in Table 6, where the values "*a*/*b*" in Table 6 are explained as follows: *a* denotes the number of fuzzy rules and *b* denotes the classification accuracy rate. From Table 6, we can see that Castro's method [1] has the highest classification accuracy rate when the number of labels is 11. In this paper, we also use 120 instances of the Iris data as the training set and the other 30 instances of the Iris data as the testing data set. The average classification accuracy rates of the proposed method for different numbers of labels after executing 1000 runs are shown in Table 7. From Table 7, we can see that when the number of labels is 11, we can get the highest average classification accuracy rate. Table 8 shows a comparison of the average classification accuracy rate of the proposed method with the ones presented in [1] and [9], where 120 instances of the Iris data are used as the training set and the other 30 instances of the Iris data are used as the testing data set. From Table 8, we can see that the proposed weighted fuzzy rules generation method gets a higher average classification

accuracy rate than the ones presented in [1] and [9].

6. Conclusions

In this paper, we have presented a new method to generate weighted fuzzy rules from training data to deal with the Iris data classification problem. First, we convert the training data to fuzzy rules, and then we merge those fuzzy rules in order to reduce the number of fuzzy rules. Then, we calculate the weight of each input variable appearing in the generated fuzzy rules by the relationships of input variables. From Table 7, we can see that when the number of labels is 11, we get the highest average classification accuracy rate. From Table 8, we can see that the proposed weighted fuzzy rules generation method gets a higher average classification accuracy rate than the ones presented in [1] and [9]. We also can see that the proposed method generates less average number of fuzzy rules than the method presented in [1]. Furthermore, from Table 8, we also can see that the average number of generated fuzzy rules of the method presented in [9] is zero due to the fact that it is based on clustering techniques.

Table 6. Experimental results of Castro's method [1].

Test	7 Labels	9 Labels	11 Labels
1	15/100	15/96.6	13/96.6
2	13/96.6	15/93.3	13/96.6
3	13/96.6	14/83.3	11/96.6
4	12/96.6	12/96.6	12/96.6
5	13/96.6	15/93.3	13/96.6
6	14/96.6	11/93.3	11/96.6
7	11/93.3	12/86.6	12/86.6
8	11/93.3	13/90	11/90
9	13/96.6	14/93.3	13/96.6
10	15/100	14/100	12/100

(The meaning of "a/b" is described as follows: *a* denotes the number of fuzzy rules and *b* denotes the classification accuracy rate.)

Table 7. Experimental results of the proposed method under different numbers of labels.

Number of labels	7 Labels	9 Labels	11 Labels
Average classification accuracy rate	96.48%	96.06%	96.7%
Average number of generated fuzzy rules	8.269	8.772	8.849

Table 8. A comparison of the average classification accuracy rate for different methods.

Methods	Average classification accuracy rate	Average number of generated fuzzy rules
Castro's method [1] (training data set: 120 instances; testing data set: 30 instances; the number of labels is 11)	96.6%	12.1
Chen-and-Yu's method [9] (training data set: 120 instances; testing data set: 30 instances; no labels are used)	96.65%	0
The proposed method (training data set: 120 instances; testing data set: 30 instances; the number of labels is 11)	96.7%	8.849

Acknowledgements

This work was supported in part by the National Science Council, Republic of China, under grant NSC 91-2213-E-011-037.

References

[1] Castro, J. L., Castro-Schez, J. J., and Zurita, J. M. 1999. Learning maximal structure rules in fuzzy logic for knowledge acquisition in expert systems. *Fuzzy Sets and Systems,* 101, 3: 331-342.

[2] Castro, J. L. and Zurita, J. M. 1997. An inductive learning algorithm in fuzzy systems. *Fuzzy Sets and Systems,* 89, 2: 193-203.

[3] Chang, C. H. and Chen, S. M. 2000. A new method to generate fuzzy rules from numerical data based on the exclusion of attribute terms. *Proceedings of the 2000 International Computer Symposium: Workshop on Artificial Intelligence*, Chiayi, Taiwan, Republic of China: 57-64.

[4] Chen, S. M., Lee, S. H., and Lee, C. H. 1999. A new method for generating fuzzy rules from numerical data for handling fuzzy classification problems. *Applied Artificial Intelligence: An International Journal,* 33, 8: 645-664.

[5] Chen, S. M. and Yeh, M. S. 1998. Generating fuzzy rules from relational database systems for estimating null values. *Cybernetics and Systems: An International Journal*, 29, 6: 363-376.

[6] Chen, S. M. and Lin, S. Y. 2000. A new method for constructing fuzzy decision trees and generating fuzzy classification rules from training examples. *Cybernetics and Systems: An International Journal*, 31, 7: 763-785.

[7] Chen, S. M., Kao, C. H., and Yu, C. H. 2002. Generating fuzzy rules from training data containing noise for handling classification problems. *Cybernetics and Systems: An International Journal*, 33, 7: 723-749.

[8] Chen, S. M. and Chen, Y. C. 2002. Automatically constructing membership functions and generating fuzzy rules using genetic algorithms. *Cybernetics and Systems: An International Journal*, 33, 8:

841-863.

[9] Chen, S. M. and Yu, C. H. 2004. A new method for handling fuzzy classification problems using clustering techniques. *International Journal of Applied Science and Engineering*, 2, 1: 90-104.

[10] Chen, Y. C. and Chen, S. M. 2000. A new method to generate fuzzy rules for fuzzy classification systems. *Proceedings of the 2000 Eighth National Conference on Fuzzy Theory and Its Applications,* Taipei, Taiwan, Republic of China.

[11] Fisher, R. 1936. The use of multiple measurements in taxonomic problems. *Ann. Eugenics*, 7: 179-188.

[12] Hayushi, Y. 1992. Fuzzy neural expert system with automated extraction of fuzzy if-then rules from a trained neural network. In "*Analysis and Management of Uncertainty: Theory and Applications*" (edited by Ayyub, B. M., Gupta, M. M., and Kanal, L. N.), North-Holland, Amsterdam: 171-181.

[13] Ishibuchi, H. and Tanaka, H. 1993. Neural network that learn from fuzzy If-Then rules. *IEEE Transactions on Fuzzy Systems,* 1, 1: 85-97.

[14] Kasabov, N. K. 1996. Learning fuzzy rules and approximate reasoning in fuzzy neural networks and hybrid systems. *Fuzzy Sets and Systems,* 82, 2: 135-149.

[15] Lin, H. L. and Chen, S. M. 2000. Generating weighted fuzzy rules from training data for handling fuzzy classification problems. *Proceedings of the 2000 International Computer Symposium: Workshop on Artificial Intelligence*, Chiayi, Taiwan, Republic of China: 11-18.

[16] Nozaki, K., Ishihuchi, H., and Tanaka, H. 1997. A simple but powerful heuristic method for generating fuzzy rules from numerical sets. *Fuzzy Sets and Systems,* 86, 3: 251-270.

[17] Wang, L. X. and Mendel, J. M. 1992. Generating fuzzy rules by learning from examples. *IEEE Transactions on Systems, Man, and Cybernetics*, 22, 6: 1414-1427.

[18] Wu, T. P. and Chen, S. M. 1999. A new method for constructing membership functions and fuzzy rules from training examples. *IEEE Transactions on systems, Man, and Cybernetics-Part B: Cybernetics*, 29, 1: 25-40.

[19] Yao, J. M., Dash, M., Tan, S. T., and Liu, H. 2000. Entropy-based fuzzy clustering and fuzzy modeling. *Fuzzy Sets and Systems*, 113, 3: 381-388.

[20] Yuan, Y. and Shaw, M. J. 1995. Induction of fuzzy decision trees. *Fuzzy Sets and Systems*, 69, 2: 125-139.

[21] Yuan, Y. and Zhuang, H. 1996. A genetic algorithm for generating fuzzy classification rules. *Fuzzy Sets and Systems,* 84, 1: 1-19.

[22] Zadeh, L. A. 1965. Fuzzy sets. *Information and Control*, 8: 338-353.