Available online at www.sciencedirect.com
ScienceDirect
Neurocomputing 70 (2007) 2581–2587
NEUROCOMPUTING
www.elsevier.com/locate/neucom

Letters

Training T-S norm neural networks to refine weights for fuzzy if–then rules

Xi-Zhao Wang*, Chun-Ru Dong, Tie-Gang Fan

College of Mathematics and Computer Science, Hebei University, Baoding 071002, China

Received 13 January 2006; received in revised form 7 January 2007; accepted 8 January 2007
Communicated by J.Q. Gan
Available online 28 February 2007

Abstract

This correspondence proposes an approach to learning weights of weighted fuzzy if–then rules. According to a given T-S norm-based reasoning mechanism, this approach first maps a set of weighted fuzzy if–then rules into a feed-forward T-S norm network in which connection weights are just the weights of weighted fuzzy if–then rules, and then trains the T-S norm neural network by a derived gradient descent algorithm. Numerical experiments show that the proposed approach is feasible and quite effective. The main contribution of this correspondence is that the mapping relationship between a set of weighted fuzzy if–then rules and a T-S norm neural network is discovered so that the difficult problem of weight acquisition in weighted fuzzy if–then rules can be converted into the training of a T-S norm neural network. A comparison between our T-S norm neural network system and a similar model (NEFCLASS) is made.

Keywords: T-S norm neural network; Improved BP algorithm; Weighted fuzzy if–then rules

1. Introduction

Fuzzy if–then rules are the essential way to uncertain knowledge representation. Fuzzy if–then rules can be extracted from data via many existing inductive learning techniques such as fuzzy decision tree [12,16], genetic algorithm [17], artificial neural network [9,8] and rough set techniques [10], and they also can be presented by knowledge engineers. Usually initially extracted fuzzy if–then rules not have a satisfying testing accuracy, and, therefore, they need to be further refined. The main merit of fuzzy if–then rules is their concise representation form and comprehensibility, while the main defect is the poor reasoning accuracy. To overcome this defect and keep the merits simultaneously, the concepts of local weight [4,15,1] and global weight [14] have been incorporated into the fuzzy if–then rules, called weighted fuzzy if–then rules. However, the determination of these weights is quite difficult since the weights are usually related to domain knowledge.

On the other hand, the artificial neural networks adopting numerical computations with fault-tolerance, massively parallel computing and trainable property are shown to be able to ease the knowledge acquisition problems [18]. The combinations of fuzzy logic and neural networks capture the advantages of these two fields. There are already lots of papers which have been published concerning the integration of fuzzy systems and fuzzy neural networks (e.g. [9,8,4,1,6,3]), while most of these researches have been proposed for the extracting of fuzzy rules, the revising of the membership functions of the linguistic values of input and output linguistic variables or the reduction and combination of the initial fuzzy rules. For example, NEFCLASS [8] presents a neuro-fuzzy system for the classification of data, which is based on a generic model of a fuzzy perceptron and uses a supervised learning algorithm based on fuzzy error back-propagation.

This correspondence proposes an approach to learning weights of weighted fuzzy if–then rules by discovering the mapping relationship between a set of weighted fuzzy if–then rules and a T-S norm neural network. According to

*Corresponding author.
E-mail addresses: wangxz@hbu.cn, xizhaowang@ieee.org (X.Z. Wang).

0925-2312/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2007.01.005

a given T-S norm-based reasoning mechanism, this approach first maps a set of weighted fuzzy if–then rules into a feed-forward T-S norm network, which has three layers (linguistic term layer, rule layer and classification layer) with the connection weights exactly being the weights included in the weighted fuzzy if–then rules, and each node in rule layer is T-norm neuron while each classification layer node is S-norm neuron node, then trains the T-S norm neural network by deriving an improved gradient descent algorithm. Furthermore, the major difference between NEFCLASS [8] and our proposed one is that our approach is to tune the weights in fuzzy rules while NEFCLASS is to adjust membership of fuzzy sets in rule antecedents. We have made a detailed comparison between NEFCLASS model and T-S norm neural network in Section 6. Numerical experiments show that the proposed approach is feasible and quite effective.

Definition 1. A conjunctive weighted fuzzy if–then rule has the following form:

$$R: \textbf{IF } (V_1 \text{ is } A_1[Lw_1]) \quad \text{AND} \cdots \text{AND } (V_n \text{ is } A_n[Lw_n]) \quad \textbf{THEN } U \text{ is } B,\ [Gw], \tag{1}$$

where $V_1, V_2, \ldots, V_n$ and U are attributes (features), $A_1, A_2, \ldots, A_n$ and B are attribute values represented by fuzzy sets, Lw_i $(1 \leqslant i \leqslant n)$ denotes the local weight of proposition (V_i is A_i), Gw denotes the global weight of the rule R, both local weights and global weights are supposed to belong to [0,1]. The local weights are to indicate the relative degree of importance of propositions in antecedents contributing to its consequent, and the global weights are to show the relative degree of importance of a rule within a rule group.

Let $\mathbf{F}: (V_1 \text{ is } A_1^*), \ldots, (V_n \text{ is } A_n^*)$ be a given fact, then we have the following reasoning form:

$$\begin{aligned} R: \ & \textbf{IF } (V_1 \text{ is } A_1\ [Lw_1]) \text{ AND} \cdots \text{AND } (V_n \text{ is } A_n\ [Lw_n]) \\ & \textbf{THEN } U \text{ is } B,\ [Gw], \\ & \text{Given fact } \mathbf{F}:\ (V_1 \text{ is } A_1^*), \ldots, (V_n \text{ is } A_n^*), \\ & \text{Conclusion}:\ U \text{ is } B^*, \end{aligned} \tag{2}$$

where B^* is the reasoning conclusion which refers to the classification in this correspondence. Let $G = \{R_1, R_2, \ldots, R_m\}$ be a set of weighted fuzzy if–then rules. Each one has the following form:

$$R_i: \textbf{IF } (V_1 \text{ is } A_1^{(i)}\ [Lw_1^{(i)}]) \text{ AND} \cdots \text{AND } (V_n \text{ is } A_n^{(i)}\ [Lw_n^{(i)}]) \quad \textbf{THEN } U \text{ is } B^{(i)},\ [Gw_i],$$

where n is the number of attributes. (We make the appointment that if V_j does not appear in the above rule then we have to delete the term $(V_j = A_j^{(i)}[Lw_j^{(i)}])$ from the rule.) In this paper, a given fact $\mathbf{F}$ is represented in the form $(A_1^*, A_2^*, \ldots, A_n^*)$, where A_j^* $(j = 1, 2, \ldots, n)$ not only can be fuzzy sets such as 'high', 'media', 'small', etc., but also can be continuous real numbers. Suppose there are totally K classes denoted by $C_1, C_2, \ldots, C_K$, and the result of matching $\mathbf{F}$ against the rule set G is $\{p_1, p_2, \ldots, p_K\}$ where p_i denotes the membership degree with which $\mathbf{F}$ belongs to class C_i. To introduce an fuzzy reasoning algorithm based on T-S norm, we first recall the definition of T-S norm:

Definition 2. A T norm refers to a mapping $T: [0,1]^2 \to [0,1]$ with the properties:

(T1) $T(a, 1) = a$;
(T2) $a \leqslant b \Rightarrow T(a, c) \leqslant T(b, c)$;
(T3) $T(a, b) = T(b, a)$ and
(T4) $T(a, T(b, c)) = T(T(a, b), c)$.

An S norm is a mapping $S: [0,1]^2 \to [0,1]$ with the properties:

(S1) $S(a, 0) = a$;
(S2) $a \leqslant b \Rightarrow S(a, c) \leqslant S(b, c)$;
(S3) $S(a, b) = S(b, a)$ and
(S4) $S(a, S(b, c)) = S(S(a, b), c)$.

For T-S norm representation, according to (T4) and (S4), we make the following appointments:

$$T_{p=1}^m(a_p) * T(a_1, T_{p=2}^m(a_p)), \quad S_{p=1}^m(a_p) * S(a_1, S_{p=2}^m(a_p)). \tag{3}$$

There have been many forms to represent the T norm and S norm. Here we list commonly used three. More details about the T-S norm can be found in Ref. [7]:

(1) $T_{\min}[a, b] = \min\{a, b\}$, $S_{\max}[a, b] = \max\{a, b\}$;
(2) $T_{\text{Dombi}}[a, b] = \frac{1}{1 + \sqrt[p]{((1-a)/a)^p + ((1-b)/b)^p}}$,
$S_{\text{Dombi}}[a, b] = 1 - \frac{1}{1 + \sqrt[p]{(a/(1-a))^p + (b/(1-b))^p}}$, $p > 0$;
(3) $T_{\text{algebraic}} = ab$, $S_{\text{algebraic}} = a + b - 2ab$.

It is worth noting that [7]

$(T_{\text{Dombi}}, S_{\text{Dombi}}) \to (T_{\min}, S_{\max})$ when $p \to 0$ and

$(T_{\text{Dombi}}, S_{\text{Dombi}}) \to (T_{\text{algebraic}}, S_{\text{algebraic}})$ when $p \to 1$.

One can see that the cases (1) and (3) can be regarded as the special case of (2). Due to the use of parameter p in (2), there is much flexibility for the reasoning based on T-S norm. It means that the reasoning based on T-S norm can be regarded as a general form of approximate reasoning for classification.

2. T-S norm-based reasoning algorithm A

Input $\mathbf{F}$—an observed fact, G—a set of weighted fuzzy if–then rules. **Output**: classification vector $\{p_1, p_2, \ldots, p_K\}$. Using the symbols in Definition 1, for a given T norm and S norm (T, S), the reasoning algorithm is described as follows:

For each R_i , do

Step 1: Computing the similarity between $A_j^{(i)}$ and A_j^*, denoted by $SM_j^{(i)}$, in two cases. (1) A is a number and B is a fuzzy set. In this case, the similarity degree is considered as the membership of A-belonging-to-B. (2) A and B are two

fuzzy sets. In this case, the similarity is

$$SM_j^{(i)} = (1 + DM)^{-1}, \tag{4}$$

where DM is a distance function, e.g., Euclidean distance.

Step 2: Computing the overall similarity $SM^{(i)}$ by

$$SM^{(i)} = \underset{1\leqslant j\leqslant n}{T}(Lw_j^{(i)} \cdot SM_j^{(i)}). \tag{5}$$

Step 3: Computing the classification vector $\{p_1, p_2, \ldots, p_K\}$ by

$$p_k = \underset{1\leqslant i\leqslant m}{S}\{Gw_i * SM^{(i)} | B^{(i)} = C_k\} \quad (k = 1, 2, \ldots, K). \tag{6}$$

When a crisp class is required, the class corresponding to the maximum p_k will be selected.

3. Mapping a set of fuzzy if–then rules to a T-S norm neural network

A set of fuzzy if–then rules can be mapped into a T-S norm network according to a given T-S norm-based reasoning mechanism (specified in algorithm A). Fig. 2 shows such a network, which has three layers (linguistic term layer, rule layer and classification layer) with the connection weights exactly being the weights included in the weighted fuzzy if–then rules. And the activation function of the nodes in rule layer is T norm while that of the nodes in classification layer is the S norm. Let L_0, L_1, L_2 denote the number of nodes in the linguistic term layer, rule layer and classification layer, respectively. Then we have

$$\text{Linguistic term layer} \quad \{y_i^{(0)} | i = 1, 2, \ldots, L_0\}, \tag{7}$$

$$\text{Rule layer} \quad y_j^{(1)} = T_{i=1}^{L_0}(Lw_{ij} y_i^{(0)}), \quad j = 1, 2, \ldots, L_1, \tag{8}$$

Classification layer

$$y_k^{(2)} = S_{j=1}^{L_1}(Gw_{jk} y_j^{(1)}), \quad k = 1, 2, \ldots, L_2, \tag{9}$$

where $y_l^{(w)}$ denotes the output of the lth node of the wth layer corresponding to the given sample, the inputs of the network are a record of the training set, i.e., an n-dimensional vector where n is the feature number. It is obvious that $L_0 = n$. We compute the values of $y_i^{(0)}$ via the formulas in step 1 of algorithm A. Then the error function can be expressed as

$$E = \frac{1}{2}\sum_{n=1}^{N}\sum_{k=1}^{L_2}(y_k - y_k^{(2)})^2 = \sum_{n=1}^{N}\left(\frac{1}{2}\sum_{k=1}^{L_2}(y_k - y_k^{(2)})^2\right) = \sum_{n=1}^{N} E_n, \tag{10}$$

where $\{y_1, y_2, \ldots, y_{L_2}\}$ is the expected output of the sample, N denotes the number of samples.

4. Training the T-S norm neural network by an improved gradient descent algorithm

We want to complete the training by minimizing the above error function via the improved gradient descent technique. In this paper, if $T_{\min}$ and $S_{\max}$ were selected as the activation functions, the partial derivatives of $T_{\min}$ and $S_{\max}$ are defined as [2]

$$\frac{\partial S_{\max}(a,b)}{\partial a} = \begin{cases} 1 & \text{if } a \geqslant b, \\ a & \text{if } a < b, \end{cases} \quad \frac{\partial T_{\min}(a,b)}{\partial a} = \begin{cases} 1 & \text{if } a \leqslant b, \\ b & \text{if } a > b. \end{cases} \tag{11}$$

The formulas for adjusting the weights are the following:

$$\left.\begin{aligned} Lw_{ij} &\leftarrow 1 && \text{if } Lw_{ij} - \alpha\frac{\partial E_n}{\partial Lw_{ij}} \geqslant 1 \\ Lw_{ij} &\leftarrow Lw_{ij} - \alpha\frac{\partial E_n}{\partial Lw_{ij}} && \text{if } Lw_{ij} - \alpha\frac{\partial E_n}{\partial Lw_{ij}} > 0 \\ Lw_{ij} &\leftarrow 0 && \text{if } Lw_{ij} - \alpha\frac{\partial E_n}{\partial Lw_{ij}} \leqslant 0 \end{aligned}\right\}, \tag{12}$$

$$\left.\begin{aligned} Gw_{jk} &\leftarrow 1 && \text{if } Gw_{jk} - \beta\frac{\partial E_n}{\partial Gw_{jk}} \geqslant 1 \\ Gw_{jk} &\leftarrow Gw_{jk} - \beta\frac{\partial E_n}{\partial Gw_{jk}} && \text{if } Gw_{jk} - \beta\frac{\partial E_n}{\partial Gw_{jk}} > 0 \\ Gw_{jk} &\leftarrow 0 && \text{if } Gw_{jk} - \beta\frac{\partial E_n}{\partial Gw_{jk}} \leqslant 0 \end{aligned}\right\}, \tag{13}$$

where α and β are the learning rates for local weights and global weights, respectively, and the adjustment magnitudes in Eqs. (12), (13) are derived as follows:

$$\frac{\partial E_n}{\partial Gw_{jk}} = (y_k^{(2)} - y_k) y_j^{(1)} \frac{\partial S}{\partial a}(Gw_{jk} y_j^{(1)}, S_{\substack{p=1\\p\neq j}}^{L_1}(Gw_{jk} \cdot y_p^{(1)})), \tag{14}$$

$$\frac{\partial E_n}{\partial Lw_{ij}} = \frac{\partial}{\partial Lw_{ij}}\left(\frac{1}{2}\sum_{k=1}^{L_2}(y_k - y_k^{(2)})^2\right) = \sum_{k=1}^{L_2}(y_i^{(0)} Gw_{jk}(y_k^{(2)} - y_k) A_{jk} B_{ij}), \tag{15}$$

where

$$A_{jk} = \frac{\partial}{\partial a} S((Gw_{jk} y_j^{(1)}), S_{\substack{p=1\\p\neq j}}^{L_1}(Gw_{pk}, y_p^{(1)})), \tag{16}$$

$$B_{ij} = \frac{\partial}{\partial a} T((Lw_{ij} \cdot y_i^{(0)}), T_{\substack{q=1\\q\neq i}}^{L_0}(Lw_{qj}, y_q^{(0)})). \tag{17}$$

In our experiment, the T_{Dombi} and S_{Dombi} have been selected as the activation functions:

$$\frac{\partial}{\partial a} T(a,b) \triangleq \frac{\partial}{\partial a} T_{\text{Dombi}}(a,b) = \frac{(((1-a)/a)^p + ((1-b)/b)^p)^{(1-p)/p}(1-a)^{p-1}}{(1 + (((1-a)/a)^p + ((1-b)/b)^p)^{1/p})^2 a^{p+1}}, \tag{18}$$

$$\frac{\partial}{\partial a}S(a,b) \triangleq \frac{\partial}{\partial a}S_{\text{Dombi}}(a,b) = \frac{((a/(1-a))^p + (b/(1-b))^p)^{(1-p)/p}a^{p-1}}{(1+((a/(1-a))^p + (b/(1-b))^p)^{1/p})^2(1-a)^{p+1}}. \quad (19)$$

The detailed process of the improved gradient descent algorithm is as follows:

Step 1: Create a T-S norm neural network according to a set of initially fuzzy if–then rules, and uniformly distributed random numbers are drawn from [0,1] to initialize all network connection weights.

Step 2: Until the termination condition is met, Do
For each object in training set, Do
Propagate the input forward through the network:

(1) Input the object to the network and compute the output $y_u^{(i)}$ $(i = 0, 1, 2)$ of every unit u in the network.
Propagate the errors backward through the network:
(2) For each network output unit k, calculate $\partial E_n/\partial Gw_{jk}$ according to (14), (19).
(3) For each hidden unit j, calculate $\partial E_n/\partial Lw_{ij}$ according to (15)–(18).
(4) Update the value of Lw_{ij} ,Gw_{jk} according to (12), (13).

During the iteration, we select such a restriction that the weight is regarded as 1 (or 0) if the weight is bigger than 1 (or less than 0) to ensure the weight values being in the interval [0,1]. We have conducted a lot of experiments with respect to the restriction. The experimental results show, to some extent, that the restriction affects the convergence. We leave it as a further investigated issue.

5. Experimental demonstration

Many experiments have been conducted for this demonstration. Here we report five cases. These databases are selected from UCI machine learning repository [11] to demonstrate our proposed approach to weight refinement (Table 1). For each database, 50% examples are randomly selected for training and the remaining 50% for testing.

The experimental results are shown in Table 2 from which one can clearly see that there is a significant improvement for the learning rate, particularly for the testing accuracy. In our experiments, the parameter p is set to be 2. Experiments show that the reasoning result is not sensitive to the parameter p. An illustration can be found in Fig. 1.

Table 1
Survey of the selected databases

Database	No. of examples	No. of attributes	No. of classes
Bupa	345	6	2
Pima	768	8	2
Haberman	306	3	2
Glass	814	9	7
Sonar	208	60	2

Table 2
Training and testing accuracy before and after refinement for the selected five databases

Database name	Original results		Final results	
	Training accuracy	Testing accuracy	Training accuracy	Testing accuracy
Bupa	0.65	0.61	0.73	0.73
Pima	0.74	0.72	0.80	0.81
Haberman	0.75	0.75	0.83	0.82
Glass	0.69	0.64	0.79	0.76
Sonar	0.83	0.76	0.89	0.88

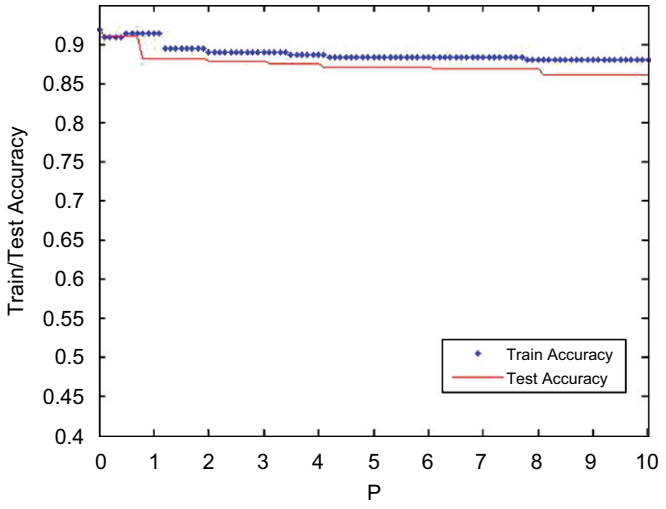

Fig. 1. The train/test accuracy of Sonar with the change of p.

One may argue that the learning accuracy improvement of fuzzy if–then rules can be achieved via many existing techniques such as adjusting the learning parameters in the fuzzy decision tree induction, why this method is preferred? The main reason is that the current approach does not increase the number of initial rules but only adjusts the weight parameters, which has less chance to lead to overfitting. We select an optimized decision tree approach [13], which is to first generate the same initialized fuzzy rules (as used in this paper) and then optimize the tree structure for the fuzzy rule improvement, to be compared with the proposed method. The comparison is listed in Table 3.

Furthermore, in order to show the feasibility of the T-S norm network (Fig. 2), we quote 4 fuzzy if–then rules extracted in [5] corresponding to the IRIS data. These rules will be regarded as 4 initial weighted fuzzy if–then rules (with local weights 1 and global weights 1) and are shown as follows:

R_1: IF ⟨PL is $SM[Lw_1]$⟩ and ⟨PW is $SM[Lw_2]$⟩ THEN Setosa $[Gw_1]$,

Table 3
Comparison between optimized decision tree (ODT) approach and T-S norm network

		Bupa	Pima	Haberman	Glass	Sonar
ODT approach	Train accuracy	0.75	0.79	0.80	0.81	0.85
	Test accuracy	0.73	0.76	0.78	0.75	0.82
T-S norm network	Train accuracy	0.73	0.80	0.83	0.79	0.89
	Test accuracy	0.73	0.81	0.82	0.76	0.88

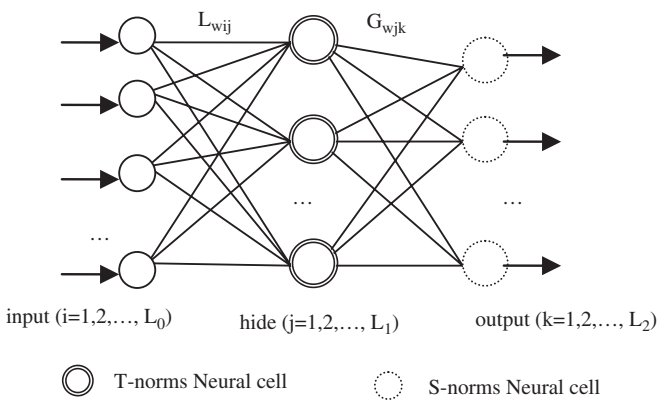

Fig. 2. A T-S norm neural network.

R_2: IF ⟨PL is MED[Lw_3]⟩ and ⟨PW is MED[Lw_4]⟩ THEN Versicolour [Gw_2],
R_3: IF ⟨PL is LRG[Lw_5]⟩ and ⟨PW is LRG[Lw_6]⟩ and ⟨SL is MED[Lw_7]⟩ THEN Virginica [Gw_3],
R_4: IF ⟨PL is LRG[Lw_8]⟩ and ⟨PW is LRG[Lw_9]⟩ and ⟨SW is MED[Lw_{10}]⟩THEN Virginica [Gw_4].

Using the initial rules to test 150 examples of Iris data according to T-S norm-based fuzzy reasoning, the classification accuracy is 82%. Then refine the 4 fuzzy rules via the proposed method, i.e., adjust the values of local weights and global weights. The refined fuzzy if–then rules have been followed.

R_1^*: IF ⟨PL is SM[0.81]⟩ and ⟨PW is SM[0.99]⟩ THEN Setosa [0.68],
R_2^*: IF ⟨PL is MED[1.0]⟩ and ⟨PW is MED[0.92]⟩ THEN Versicolour [0.32],
R_3^*: IF ⟨PL is LRG[1.0]⟩ and ⟨PW is LRG[0.69]⟩ and ⟨SL is MED[0.68]⟩ THEN Virginica [0.82],
R_4^*: IF ⟨PL is LRG[0.38]⟩ and ⟨PW is LRG[0.95]⟩ and ⟨SW is MED[0.76]⟩ THEN Virginica [1.0].

Let R_1^*–R_4^* to test the same 150 examples of Iris data, the error ratio is 11.4%, that is to say, the classification accuracy has been improved by 6.6%.

6. A comparison between NEFCLASS model and our T-S norm neural network system

A NEFCLASS model was proposed ([8]) to learn fuzzy rules via adjusting fuzzy member functions. The following is a minute comparison between NEFCLASS model and our system (Table 4).

Similarities: Both NEFCLASS model and our proposed T-S norm neural network are a three-layer fuzzy perceptron network with the similar specifications:

$$U_1 = \{x_1, \ldots, x_n\} \quad \text{input layer},$$

$$U_2 = \{R_1, \ldots, R_k\} \quad \text{rule layer},$$

$$U_3 = \{c_1, \ldots, c_m\} \quad \text{output layer}.$$

Both the two methods define for each rule unit $u \in U_2$ a propagation function NET_u to calculate the net input $net_u = T_{u' \in U_1}\{W(u', u)(o_{u'})\}$ where T is a t norm, $W(u', u)$ is the connection weight from input unit u' to rule unit u. In addition, both methods complete the learning procedure by adjusting some parameters of the fuzzy production rules.

Difference: (1) *Net architecture*: The connection weights of NEFCLASS are fuzzy sets while the connection weights of our proposed network are real numbers.

(2) O*bjectives*: The NEFCLASS model aims to learn fuzzy rules like:

$$\text{if } x_1 \text{ is } \mu_1 \text{ and } x_2 \text{ is } \mu_2 \text{ and} \ldots \text{and } x_n \text{ is } \mu_n$$
$$\text{then the pattern } (x_1, x_2, \ldots, x_n) \text{ belongs to class } i,$$

where $\mu_1, \ldots, \mu_n$ are fuzzy sets, each μ_j is tunable. The learning process is to tune those μ_j $(j = 1, \ldots, n)$ (i.e., their three parameters). While our T-S norm network system is to learn the weights of weighted fuzzy production rules like Eq. (1), in which local weights and global weights are attached and the fuzzy sets A_j $(j = 1, \ldots, n)$ are nonadjustable. The learning process is to adjust the local

Table 4
Comparison of the network architecture between NEFCLASS model and T-S norm neural network

Network architecture	NEFCLASS	T-S norm neural network
Connection weights from input units to rule units	Each connection between units $x_i \in U_1$ and $R_r \in U_2$ is a fuzzy set $\mu_j^{(i)}$, labelled with a linguistic term $A_j^{(i)}$. These may be terms like small, medium, large, etc.	Lw_{ij} is the connection weight between the ith input unit and jth rule unit, which is exactly the ith local weight of the jth weighted fuzzy production rule, and Lw_{ij} is a scalar
Connection weights from rule units to output units	$W(R, c) \in \{0, 1\}$ for all $R \in U_2$, $c \in U_3$	Gw_{jk} is the connection weight between the jth rule unit and kth output unit, which equals the global weight of the jth weighted fuzzy production rule and Gw_{jk} is a scalar

weights and global weights rather than those fuzzy sets A_j $(j = 1, \ldots, n)$.

(3) *Network input*: The input of NEFCLASS is the continuous-valued number while the input of our system is the similarity degrees of attribute values matching fuzzy rules.

(4) *Learning algorithm of the network*: In our T-S norm neural network system, the propagation functions, i.e., the T norm and S norm, are nondifferentiable, so we define a type of derivative of T norm and S norm, respectively, and then determine a gradient descent-based fuzzy error back-propagation algorithm to train the network. During the training, both the weights from input units to rule units (local weights) and the weights from rule units to output units (global weights) are adjusted. While in NEFCLASS model, a fuzzy error back-propagation algorithm, which is not based on the gradient descent technology, is used to train the NEFCLASS model, and only the weights (fuzzy sets) between the input units and rule units are changed. It is realized by adjusting the incorporated three parameters in the triangular membership functions. The connection weights from rule units to output units remain 1 throughout the learning process.

(5) *Experimental evaluation on IRIS data set*: The NEFCLASS model had been used to obtain rules and fuzzy sets from IRIS data set [8], and the network, with 4 input, 3 hidden and 3 output units, was able to learn the training set completely and it classified 5 of the 75 test patterns incorrectly, i.e., the classification rate on the test set was 93%. To compare the two methods, we also use our T-S norm neural network system to refine the extracted fuzzy rules from IRIS data set. Firstly, 5 initial fuzzy rules, with all weights equal 1.0, are extracted via the fuzzy decision tree methodology [16], and the training rate is 91% while the testing rate is 87%. Secondly, we use the proposed T-S norm network system to learn the local weights and global weights. After the refinement, these weighted fuzzy production rules classify only 2 training and 6 testing patterns incorrectly, i.e., the testing accuracy is 92%.

7. Conclusion and future work

This correspondence proposes an approach to learning weights of weighted fuzzy if–then rules by training a T-S norm neural network. The approach combines both the merit of simple representation and comprehensibility of fuzzy if–then rules and the merit of enhancement of neural network learning capability. The mapping relationship between a set of weighted fuzzy if–then rules and a T-S norm neural network is discovered in this correspondence, and, therefore, the difficult problem of weight acquisition in weighted fuzzy if–then rules can be converted into the training of T-S norm neural networks. An improved gradient descent algorithm is derived to train the neural network. Experiments show that, under the condition that the number of rules is not increasing, the learning/testing accuracy of a set of fuzzy if–then rules can be improved significantly after the weight refinement.

The further research work will focus on the following three points. (1) Improve the training algorithm with respect to the restriction that weights belong to [0,1], including the time complexity and the over-fitting phenomenon. (2) Study whether the reasoning is strongly dependent on the change of T-S norms (since there are many different T-S norms) and further study the T-S norm sensitivity to its parameters.

Acknowledgement

This research work is supported by National Natural Science Foundation of China (60473045).

References

[1] A. Blanco, M. Delgado, I. Requena, A learning procedure to identify weighted rules by neural networks, Fuzzy Sets Syst. 69 (1995) 29–36.
[2] A. Blanco, M. Delgado, I. Requena, Identification of fuzzy relational equations by fuzzy neural networks, Fuzzy Sets Syst. 71 (1995) 215–226.
[3] S.I. Horikawa, T. Furuhashi, Y. Uchikawa, On fuzzy modeling using fuzzy neural networks with the back-propagation algorithm, IEEE Trans. Neural Networks 3 (5) (1992) 801–806.
[4] N. Kasabov, Learning fuzzy production rules for approximate reasoning with connectionist production systems, in: S. Gielen, B. Kappen (Eds.), Proceedings of the International Conference on Artificial Neural Networks, Springer, New York, 1993, pp. 337–345.
[5] N.K. Kasabov, Learning fuzzy rules and approximate reasoning in fuzzy networks and hybrid systems, Fuzzy Sets Syst. 82 (1996) 135–149.
[6] C.T. Lin, C.S.G. Lee, Neural-network-based fuzzy logic control and decision system, IEEE Trans. Comput. C 40 (12) (1991) 1320–1336.
[7] M. Mizumoto, Pictorial representations of fuzzy connectives, part I: cases of t-norms, t-conorms and averaging operators, Fuzzy Sets Syst. 31 (2) (1989) 217–242.
[8] D. Nauck, R. Kruse, A neuro-fuzzy approach for the classification of data, Paper of Symposium on Applied Computing, 1995.
[9] D. Nauck, R. Kruse, Designing neuro-fuzzy systems through back-propagation, in: W. Pedrycz (Ed.), Fuzzy Modelling, Paradigms and Practice, Kluwer, Academic Publishers, Boston, 1996, pp. 203–228.
[10] Z. Pawlak, Rough Sets: Theoretical Aspects of Reasoning About Data, Kluwer Academic Publishers, Dordrecht, 1991.
[11] Repository of machine learning databases and domain theories. FTP address: ⟨ftp://ftp.ics.uci.edu/pub/machine-learning-databases⟩.
[12] M. Umano, H. Okamoto, I. Hatoho, H. Tamura, F. Kawachi, S. Umedzu, J. Kinoshita, Fuzzy decision trees by fuzzy ID3 algorithm and its application to diagnosis systems, in: Proceedings of the Third IEEE Conference on Fuzzy Systems, vol. 3, Orlando, June, 1994, pp. 2113–2118.
[13] X. Wang, B. Chen, G. Qian, F. Ye, On the optimization of fuzzy decision trees, Fuzzy Sets Syst. 112 (2000) 117–125.
[14] D.S. Yeung, E.C.C. Tsang, Weighted fuzzy production rules, Fuzzy Sets Syst. 88 (1997) 299–313.
[15] W. Yu, Z. Bien, Design of fuzzy logic controller with inconsistent rule base, J. Intell. Fuzzy Syst. 2 (1994) 147–159.
[16] Y. Yuan, M.J. Shaw, Induction of fuzzy production rules, Fuzzy Set Syst. 88 (1997) 299–313.
[17] Y. Yuan, H. Zhuang, A genetic algorithm for generating fuzzy classification rules, Fuzzy Sets Syst. 84 (1) (1996).
[18] J.M. Zurada, Introduction to Artificial Neural Systems, Info. Access and Distribution (P) Ltd., Singapore, 1992.

Xi-Zhao Wang is presently the Dean and Professor of the Faculty of Mathematics and Computer Science, Hebei University, China. He received his Ph.D. degree in Computer Science from Harbin Institute of Technology, Harbin, China, in 1998. From 1998 to 2001, he served as a Research Fellow in the Department of Computing, Hong Kong Polytechnic University, Hong Kong. His main research interests include learning from examples with fuzzy representation, fuzzy measures and integrals, neuro-fuzzy systems and genetic algorithms, feature extraction, multi-classifier fusion and applications of machine learning. He has 150+ publications including 2 books, 5 book chapters, and 52 journal papers in IEEE Transactions on PAMI/SMC/FS, Fuzzy Sets and Systems, Pattern Recognition, etc. He has been the PI/co-PI for 16 research projects supported partially by the National Natural Science Foundation of China and the Research Grant Committee of Hong Kong Government.

He is an IEEE Senior Member, the Chair of Chair of IEEE SMC Technical Committee on Computational Intelligence, an Associate Editor of IEEE Transactions on SMC, Part B, an Associate Editor of Pattern Recognition and Artificial Intelligence, a Member of editorial board of Information Sciences. Prof. Wang was the recipient of the IEEE-SMCS Outstanding Contribution Award in 2004 and the recipient of IEEE-SMCS Best Associate Editor Award in 2006.

Chun-Ru Dong is an Assistant Lecturer of the Department of Mathematics and Computer Science, Hebei University, China. He received the B.Sc. in Mathematics and M.E. in Computer Science from Hebei University in 2002 and 2005, respectively. In 2005, he worked as a Research Assistant at the Department of Computing, Hong Kong Polytechnic University, Kowloon. His main research interests include inductive learning, fuzzy reasoning and neuro-fuzzy systems.

Tie-Gang Fan received his Science Master Degree in Applications of Mathematics from Hebei University in 2005. He is presently a Lecturer of the Department of Mathematics and Computer Science, Hebei University, China. His main research interests include neural network and fuzzy reasoning.