ELSEVIER

Fuzzy Sets and Systems 127 (2002) 37–48

FUZZY sets and systems

www.elsevier.com/locate/fss

A neural fuzzy network for word information processing

Chin-Teng Lin *, Fun-Bin Duh, Der-Jenq Liu

Department of Electrical and Control Engineering, National Chiao-Tung University, Hsinchu, Taiwan, ROC

Abstract

A neural fuzzy system learning with fuzzy training data is proposed in this study. The system is able to process and learn numerical information as well as word information. At first, we propose a basic structure of five-layered neural network for the connectionist realization of a fuzzy inference system. The connectionist structure can house fuzzy logic rules and membership functions for fuzzy inference. The inputs, outputs, and weights of the proposed network can be fuzzy numbers of any shape. Also they can be hybrid of fuzzy numbers and numerical numbers through the use of fuzzy singletons. Based on interval arithmetics, a *fuzzy supervised learning* algorithm is developed for the proposed system. It extends the normal supervised learning techniques to the learning problems where only word teaching signals are available. The fuzzy supervised learning scheme can train the proposed system with desired fuzzy input–output pairs. An experimental system is constructed to illustrate the performance and applicability of the proposed scheme.

1. Introduction

Some observations obtained from a system are precise, while some cannot be measured at all. Namely, two kinds of information are available. One is numerical information from measuring instruments and the other is word information from human experts. However, some of data obtained in this manner are hybrid; that is, their components are not homogeneous but a blend of precise and fuzzy information.

Neural networks adopt numerical computations with fault-tolerance, massively parallel computing, and trainable properties. However, numerical quantities evidently suffer from a lack of representation power. Therefore, it is useful for neural networks to be capable of symbolic processing. Most learning methods in neural networks are designed for real vectors. There are many applications that the information cannot be represented meaningfully or measured directly as real vectors. That is, we have to deal with fuzzy information in the learning process of neural networks. Fuzzy set is a good representation form for linguistic data. Therefore, combining neural networks with fuzzy set could combine the advantages of symbolic and numerical processing. In this study, we propose a new model of neural fuzzy system that can process the hybrid of numerical and fuzzy information. The main task is to develop a fuzzy supervised learning algorithm for the proposed neural fuzzy system.

Most of the supervised learning methods of neural networks, for example the perception [1], the backpropagation (BP) algorithm [2,3], process only numerical data. Some approaches have been proposed to process linguistic information with fuzzy inputs,

* Corresponding author. Tel.: +886-3-5712121, Ext. 54315; fax: +886-3-5739497.

E-mail address: ctlin@fnn.cn.nctu.edu.tw (C.-T. Lin).

0165-0114/02/$ - see front matter
PII: S0165-0114(01)00151-8

fuzzy outputs, or fuzzy weights [4–7]. The common points of these approaches are summarized as follows: (1) The α-level sets of fuzzy numbers represent linguistic inputs, linguistic outputs, fuzzy weights, or fuzzy biases. (2) The operations in neural network are performed by using interval arithmetic operations for α-level sets. (3) Fuzzy numbers are propagated through neural networks. (4) Fuzzy weights are usually triangular or trapezoidal fuzzy numbers. Because the real number arithmetic operations in the traditional neural networks are extended to interval arithmetic operations for α-level sets in the above fuzzified networks, the computations become complex (e.g. multiplication of interval) and time-consuming. Moreover, since the fuzzy numbers are propagated through the whole neural networks, the time of computations and the required memory capacities are $2h$ times of those in the traditional neural networks, where h represents the number of α-level sets. In this study, we attack this problem by allowing numerical signals to flow in the proposed network internally and reach the same purpose of processing fuzzy numbers.

The objective of this study is to explore the approach to supervised learning of neural fuzzy systems which receive only word teaching signals. At first, we propose a basic structure of five-layered feedforward network for the network realization of a fuzzy inference system. This connectionist structure can house fuzzy logic rules and membership functions, and perform fuzzy inference. We use α-level sets of fuzzy numbers to represent word information. The inputs, outputs, and weights of the proposed network can be fuzzy numbers of any shape. Since numerical values can be represented by fuzzy singletons, the proposed system can in fact process and learn hybrid of fuzzy numbers and numerical numbers. Based on interval arithmetics, a *fuzzy supervised learning* scheme is developed for the proposed system. It generalizes the normal supervised learning techniques to the learning problems where only word teaching signals are available. The fuzzy supervised learning scheme can train the proposed network with desired fuzzy input–output pairs (or, equivalently, desired fuzzy if-then rules) represented by fuzzy numbers instead of numerical values.

This study is organized as follows: Section 2 describes the fundamental properties and operations of fuzzy numbers and their α-level sets. These operations and properties will be used in later derivation. In Section 3, the basic structure of our neural fuzzy system is proposed. A fuzzy supervised learning algorithm for the proposed system is presented in Section 4. The learning algorithm contains structure and parameter learning phases. In Section 5, simulations are performed to illustrate the performance of the proposed techniques. Finally, conclusions are summarized in the last section.

2. Representation of word information

In our model, we use α-level sets of fuzzy numbers (i.e., convex and normal fuzzy sets), as shown in Fig. 1, to represent word information due to their several good properties such as closeness and good representation form. In using α-level sets, we consider a fuzzy number to be an extension of the concept of the interval of confidence [9]. Instead of considering the interval of confidence at one unique level, it is considered at several levels and more generally at all levels from 0 to 1. Namely, the level of presumption α, $\alpha \in [0,1]$ gives an interval of confidence $A_\alpha = [a_1^{(\alpha)}, a_2^{(\alpha)}]$, which is a monotonical decreasing function of α; that is,

$$(\alpha' > \alpha) \rightarrow (A_{\alpha'} \subset A_\alpha), \tag{1}$$

or

$$(\alpha' > \alpha) \rightarrow ([a_1^{(\alpha')}, a_2^{(\alpha')}] \subset [a_1^{(\alpha)}, a_2^{(\alpha)}]), \tag{2}$$

for every $\alpha, \alpha' \in [0,1]$,

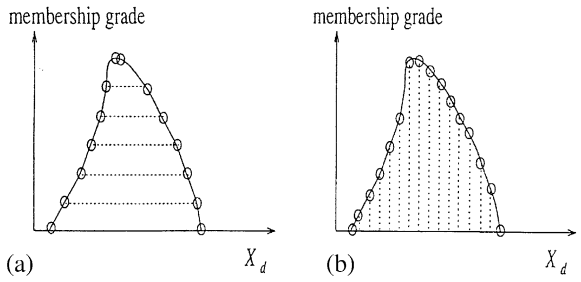

Fig. 1. Representations of fuzzy number. (a) α-level sets of fuzzy number. (b) discretized (pointwise) membership function.

2.1. Basic definitions of fuzzy numbers

Some notations and basic definitions are given in this subsection. We use the uppercase letter to represent a fuzzy set and the lowercase letter to represent a real number.

Let x be an element in a universe of discourse X. A fuzzy set, P, is defined by a membership function, $\mu_P(x)$, as

$$\mu_P : x \to [0, 1]. \tag{3}$$

When X is continuum rather than a countable or finite set, the fuzzy set P is represented as

$$P = \int_X \mu_P(x)/x, \tag{4}$$

where $x \in X$. When X is a countable or finite set,

$$P = \sum_i \mu_P(x_i)/x_i, \tag{5}$$

where $x_i \in X$. We call the form in the above equation as a *discretized* or *pointwise* membership function.

A fuzzy set, P, is *normal* when its membership function, $\mu_P(x)$, satisfies

$$\max_x \mu_P(x) = 1. \tag{6}$$

A fuzzy set is *convex* if and only if each of its α-level sets is a convex set. Equivalently, we may say that a fuzzy set P is convex if and only if

$$\mu_P(\lambda x_1 + (1 - \lambda)x_2) \geqslant \min[\mu_P(x_1), \mu_P(x_2)], \tag{7}$$

where $0 \leqslant \lambda \leqslant 1,\ x_1 \in X,\ x_2 \in X$.

The α-*level set* of a fuzzy set P, P_α, is defined by

$$P_\alpha = \{x | \mu_P(x) \geqslant \alpha\}, \tag{8}$$

where $0 \leqslant \alpha \leqslant 1,\ x \in X$.

A fuzzy set P is convex if and only if every P_α is convex; that is, P_α is a closed interval of $\Re$. It can be represented by

$$P_\alpha = [P_1^{(\alpha)}, P_2^{(\alpha)}], \tag{9}$$

where $\alpha \in [0, 1]$. A convex and normalized fuzzy set whose membership function is piecewise continuous is called a fuzzy number. Thus, a fuzzy number can be considered as containing the real numbers within some interval to varying degrees. Namely, a fuzzy number P may be decomposed into its α-level set, P_α, according to the resolution identity theorem [11] as follows:

$$P = \bigcup_\alpha \alpha P_\alpha = \bigcup_\alpha \alpha[p_1^{(\alpha)}, p_2^{(\alpha)}]$$

$$= \int_x \sup_\alpha \alpha \mu_{P_\alpha}(x)/x. \tag{10}$$

2.2. Basic operations of fuzzy numbers

In this subsection, we introduce some basic operations of α-level sets of fuzzy numbers. These operations will be used in the derivation of our model in the following section. More detailed operations of fuzzy numbers can be found in [9].

Addition. Let A and B be two fuzzy numbers and A_α and B_α their α-level sets, $A = \bigcup_\alpha \alpha A_\alpha = \bigcup_\alpha \alpha[a_1^{(\alpha)}, a_2^{(\alpha)}]$, $B = \bigcup_\alpha \alpha B_\alpha = \bigcup_\alpha \alpha[b_1^{(\alpha)}, b_2^{(\alpha)}]$. Then we can write

$$A_\alpha(+)B_\alpha = [a_1^{(\alpha)}, a_2^{(\alpha)}](+)[b_1^{(\alpha)}, b_2^{(\alpha)}]$$

$$= [a_1^{(\alpha)} + b_1^{(\alpha)}, a_2^{(\alpha)} + b_2^{(\alpha)}], \tag{11}$$

where $\alpha \in [0, 1]$.

Subtraction. The definition of addition can be extended to the definition of subtraction as follows.

$$A_\alpha(-)B_\alpha = [a_1^{(\alpha)}, a_2^{(\alpha)}](-)[b_1^{(\alpha)}, b_2^{(\alpha)}]$$

$$= [a_1^{(\alpha)} - b_2^{(\alpha)}, a_2^{(\alpha)} - b_1^{(\alpha)}], \tag{12}$$

where $\alpha \in [0, 1]$.

Multiplication by an Ordinary Number. Let A be a fuzzy number in $\Re$ and k an ordinary number $k \in \Re$. We have

$$k \cdot A_\alpha = \begin{cases} [ka_1^{(\alpha)}, ka_2^{(\alpha)}], & \text{if } k \geqslant 0, \\ [ka_2^{(\alpha)}, ka_1^{(\alpha)}], & \text{if } k < 0. \end{cases} \tag{13}$$

Multiplication. Here we consider multiplication of fuzzy numbers in $\Re^+$. Consider two fuzzy numbers A and B in $\Re^+$. For the level α of presumption, we have

$$A_\alpha(\cdot)B_\alpha = [a_1^{(\alpha)}, a_2^{(\alpha)}](\cdot)[b_1^{(\alpha)}, b_2^{(\alpha)}]$$

$$= [a_1^{(\alpha)} \cdot b_1^{(\alpha)}, a_2^{(\alpha)} \cdot b_2^{(\alpha)}]. \tag{14}$$

The reader is referred to [9] for the general case that A and B are fuzzy numbers in $\Re$.

The operations on fuzzy numbers can be performed on the basis of the extension principle. Let $G(A, B)$ be an operation on fuzzy numbers A and B by giving a function $g(x, y)$. The membership function of $G(A, B)$, $\mu_{G(A,B)}(z)$, is obtained by extension principle as follows:

$$\mu_{G(A,B)}(z) = \begin{cases} \sup_{(x,y)\in g^{-1}(z)}(\mu_A(x) \wedge \mu_B(y)) & \text{if } g^{-1}(z) \neq \emptyset, \\ 0 & \text{if } g^{-1}(z) = \emptyset, \end{cases} \tag{15}$$

where $x \in A,\ y \in B,\ z \in Z$.

If $G(\cdot)$ is addition, subtraction, or multiplication as above, the result can be easily obtained by using the α-level sets of A and B as above operations. It can be proved easily that these operations based on α-level sets and the extension principle are equivalent [9]. When $G(\cdot)$ is one of the operations given above, if A and B are fuzzy numbers in $\Re$, then $G(A, B)$ is also a fuzzy number.

Difference. We can compute the difference between fuzzy numbers A and B by

$$diff(A,B) = \frac{1}{2}\sum_{\alpha}[(a_1^{(\alpha)} - b_1^{(\alpha)})^2 + (a_2^{(\alpha)} - b_2^{(\alpha)})^2]. \tag{16}$$

Fuzzification. Fuzzification is a mapping from an observed input space to fuzzy sets. Namely, Fuzzification is an operation that obtains the membership grade of x, $\mu_P(x)$, to fuzzy number P. The values $\mu_P(x)$ can be easily obtained by using the following procedure. In this procedure, the notation h represents the number of quantized membership grade.

Procedure: *Fuzzification search*
Inputs. The order set $S = \{p_1^{(0)}, p_1^{(1/h)}, p_1^{(2/h)}, \ldots, p_1^{(1)}, p_2^{(1)}, \ldots, p_2^{(0)}\}, x$.
Output. $\hat{\alpha}$.

Step 1. Use *binary search* or *sequential search* to find the correct position in S; that is, to find $p^{(a)}$ and $p^{(b)}$ such that $p^{(a)} \leqslant x \leqslant p^{(b)}$, $p^{(a)},\ p^{(b)} \in S$.

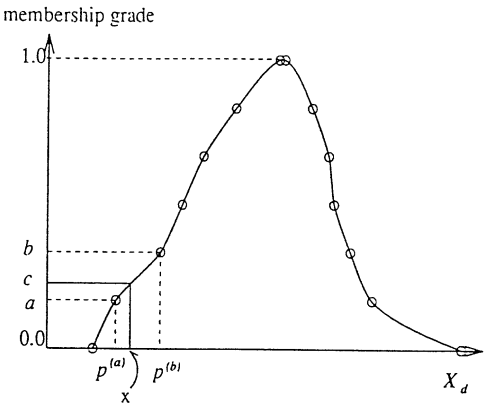

Fig. 2. Illustration of the fuzzification search procedure (the value of c is equal to $\hat{\alpha}$).

Step 2. $\hat{\alpha} = a + (x - p^{(a)})(b - a)/(p^{(b)} - p^{(a)})$.

Step 3. Output $\hat{\alpha}$, and stop.

After this recursive calculation, the value $\hat{\alpha}$ is equal to the value of $\mu_p(x)$ as shown in Fig. 2. If we use the binary search method, the processing time with this procedure is proportional to $\log_2 h$. If we use the sequential search method, the processing time is $O(h)$. Therefore, performing this procedure is very easy and not time consuming.

Defuzzification. In many practical applications such as control and classification, numerical (crisp) data are required. That is, it is essential to transform a fuzzy number to a numerical value. The process mapping a fuzzy number into a nonfuzzy number is called "defuzzification". Various defuzzification strategies have been suggested in [12,13]. In this subsection, we describe two methods (MOM, COA) that transform a fuzzy number in the form of α-level sets into a crisp value.

- *Mean of maximum method* (*MOM*)

The mean of maximum method (MOM) generates a crisp value by averaging the support values whose membership values reach the maximum. For a discrete universe of discourse, this is calculated based on membership function by

$$z_0 = \frac{\sum_{j=1}^{l} z_j}{l}, \tag{17}$$

where l is the number of quantized z values which reach their maximum membership value.

For a fuzzy number Z in the form of α-level sets, the defuzzification method can be expressed according to Eq. (17) as

$$defuzzifier(Z) = z_0 = \frac{(z_1^{(1)} + z_2^{(1)})}{2}, \tag{18}$$

where *defuzzifier* represents a defuzzification operation.

- *Center of area method* (COA)

Assuming that a fuzzy number with a pointwise membership μ_Z has been produced, the center of area method calculates the center of gravity of the distribution for the nonfuzzy value. Assuming a discrete universe of discourse, we have

$$z_0 = \frac{\sum_{j=1}^{n} x_j \mu_Z(x_j)}{\sum_{j=1}^{n} \mu_Z(x_j)}. \tag{19}$$

For a fuzzy number Z with the representation form of α-level sets, it can be expressed according to Eq. (19) as

$$defuzzifier(Z) = z_0 = \frac{\sum_{\alpha} \alpha(z_1^{(\alpha)} + z_2^{(\alpha)})}{2\sum_{\alpha} \alpha}. \tag{20}$$

3. Basic structure of the neural fuzzy system

In this section, we construct an architecture of neural fuzzy system that can process fuzzy and crisp information. Fig. 3 shows the proposed network structure which has a total of five layers. This five-layered connectionist structure performs fuzzy inference effectively. We shall describe the signal propagation in the proposed network layer by layer following the arrow directions shown in Fig. 3. This is done by defining the transfer function of a node in each layer. Signal may flow in the reserve direction in the learning process as we shall discuss in the following sections. In the following description, we shall consider the case of single output node for clarity. It can be easily extended to the case of multiple output nodes. A typical neural network consists of nodes, each of which has some finite fan-in of connections represented by weight values from other nodes and fan-out of connections to other nodes (see Fig. 4). The notations u and U represent the input crisp and fuzzy numbers of the node, respectively. The notations o and O represent, respectively, the output crisp and fuzzy numbers. The superscript in the following formulas indicates the layer number.

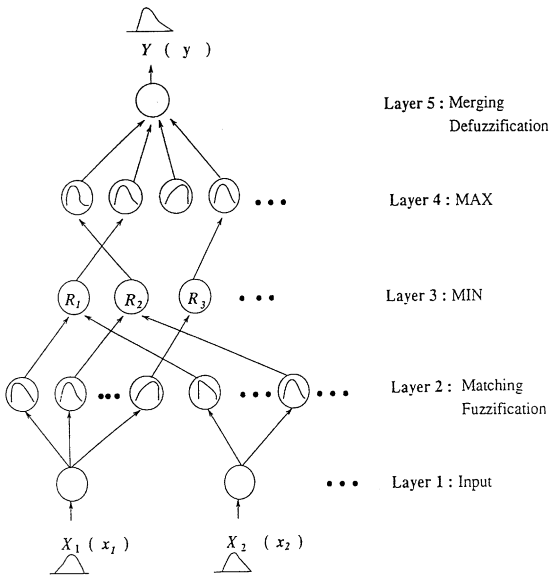

Fig. 3. The five-layered architecture of the proposed neural fuzzy system.

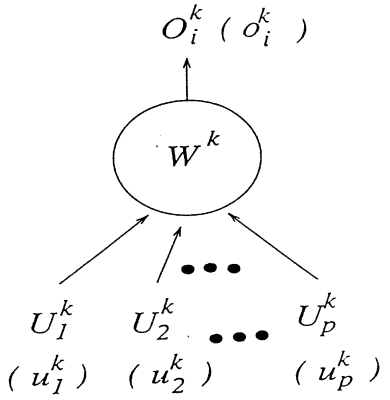

Fig. 4. Basic structure of a node in the proposed neural fuzzy system.

Layer 1 (**Input**). If the input is a fuzzy number, each node in this layer only transmits input fuzzy number X_i to the next layer directly. No computation is done in this layer. That is,

$$O_i^1 = \bigcup_{\alpha} \alpha[o_{i1}^{1(\alpha)}, o_{i2}^{1(\alpha)}] = X_i = \bigcup_{\alpha} \alpha[x_{i1}^{(\alpha)}, x_{i2}^{(\alpha)}]. \tag{21}$$

If the input is a crisp number x_i, it can be viewed as a fuzzy *singleton*, i.e.,

$$O_i^1 = \bigcup_{\alpha} \alpha[o_{i1}^{1(\alpha)}, o_{i2}^{1(\alpha)}] = \bigcup_{\alpha} \alpha[x_i, x_i]. \tag{22}$$

Note that there is no weight to be adjusted in this layer.

Layer 2 (**Matching**/**Fuzzification**). Each node in this layer has exactly one input from some input linguistic node, and feeds its output to rule node(s). For each layer-2 node, the input is a fuzzy number and the output is a numerical number. The weight in this layer is a fuzzy number WX_{ij}. The index i, j means the jth term of the ith input linguistic variable X_i. The transfer function of each layer-2 node is,

$$f_{ij}^2 = diff(WX_{ij}, U_i) = \frac{1}{2}\sum_{\alpha}[(wx_{ij1}^{(\alpha)} - u_{i1}^{2(\alpha)})^2 + (wx_{ij2}^{(\alpha)} - u_{i2}^{2(\alpha)})^2], \tag{23}$$

$$o_{ij}^2 = a(f_{ij}^2) = \mathrm{e}^{-(f_{ij}^2)^2/2\sigma^2}, \tag{24}$$

where σ is the variance of the activation function $a(\cdot)$. It is a constant given in advance. The activation function $a(\cdot)$ is a nonnegative, monotonically decreasing function of $f_{ij}^2 \in [0, \infty]$, and $a(0)$ is equal to 1. For example, $a(\cdot)$ can also be given alternatively as

$$o_{ij}^2 = a(f_{ij}^2) = r^{f_{ij}^2}, \tag{25}$$

where $0 < r < 1$, or

$$o_{ij}^2 = a(f_{ij}^2) = \frac{2}{1 + \mathrm{e}^{-\lambda f_{ij}^2}}, \tag{26}$$

where λ is a nonnegative constant.

Layer 3 (**MIN**). The input and output of the node in this layer are both numerical. The links in this layer perform precondition matching of fuzzy logic rules. Hence, the rule nodes should perform fuzzy AND operation. The most commonly used fuzzy AND operations are *intersection* and *algebraic product* [13]. If intersection is used, we have

$$o_i^3 = \min(u_1^3, u_2^3, \ldots, u_k^3). \tag{27}$$

On the other hand, if algebraic product is used, we have

$$o_i^3 = u_1^3 u_2^3 \ldots u_k^3. \tag{28}$$

Similar to layer one, there is no weight to be adjusted in this layer.

Layer 4 (**MAX**). The nodes in this layer should perform fuzzy OR operation to integrate the fired rules which have the same consequent. The most commonly used fuzzy OR operations are *union* and *bounded sum* [13]. If the union operation is used in this model, we have

$$o_i^4 = \max(u_1^4, u_2^4, \ldots, u_k^4). \tag{29}$$

If the bounded sum is used, we have

$$o_i^4 = \min(1, u_1^4 + u_2^4 + \cdots + u_k^4). \tag{30}$$

The output and input of each layer-4 node are both numerical values.

Layer 5 (**Merging**/**Defuzzification**). In this layer, each node has a fuzzy weight WY_i. There are two kinds of operations in this layer. When we need a fuzzy output Y, the following formula is executed to perform a "merging" action:

$$O^5 = \bigcup_{\alpha} \alpha[o_1^{5(\alpha)}, o_2^{5(\alpha)}] = Y = \frac{\sum_i u_i^5 WY_i}{\sum_i u_i^5}. \tag{31}$$

Namely,

$$Y = \bigcup_{\alpha} \alpha[y_1^{(\alpha)}, y_2^{(\alpha)}], \quad WY_i = \bigcup_{\alpha} \alpha[wy_{i1}^{(\alpha)}, wy_{i2}^{(\alpha)}], \tag{32}$$

where

$$y_1^{(\alpha)} = \frac{\sum_i u_i^5 wy_{i1}^{(\alpha)}}{\sum_i u_i^5}, \tag{33}$$

$$y_2^{(\alpha)} = \frac{\sum_i u_i^5 wy_{i2}^{(\alpha)}}{\sum_i u_i^5}. \tag{34}$$

From the above description we observe that only layer-1 inputs and layer-5 outputs of the proposed network are fuzzy numbers (in the form of α-level sets). Real numbers are propagated internally from layer two to layer four in the network. This makes the operations in our proposed network less time-consuming as compared to the neural networks that can also process fuzzy input/output data but require fuzzy signals flowing in it.

4. Supervised learning of the basic neural fuzzy system

In this section, we shall derive a supervised learning algorithm for the structure of the proposed neural fuzzy system. This algorithm is applicable to the situations that pairs of input–output training data are available.

In general, the fuzzy rules for training are

$$R_p\text{: IF } x_1 \text{ is } X_{p1} \quad \text{and} \ldots \text{and} \quad x_p \text{ is } X_{pn},$$
$$\text{THEN } y \text{ is } Y_p,$$

where $p = 1, 2, \ldots, m$, and m is the total number of training rules. These fuzzy if-then rules can be viewed as the fuzzy input–output pairs, $(X_{p1}, X_{p2}, \ldots, X_{pn}; Y_p)$, where $p = 1, 2, \ldots, m$. If the input or output are crisp data, the corresponding fuzzy elements in the training pairs become numerical elements.

Before the learning of the neural fuzzy system, an initial network structure is first constructed. Then during the learning process, some nodes and links in the initial network are deleted or combined to form the final structure of the network. At first, the number of input (output) nodes is set equal to the number of input (output) linguistic variables. The number of nodes in the second layer is decided by the number of fuzzy partitions of each input linguistic variable x_i, $|T(x_i)|$, which must be assigned by the user. The fuzzy weights WX_{ij} in layer two are initialized randomly as fuzzy numbers. One better way is to distribute the initial fuzzy weights evenly on the interested domain of the corresponding input linguistic variable. As for layer three of the initial network, there are $\prod_i |T(x_i)|$ rule nodes with the inputs of each rule node coming from one possible combination of the terms of input linguistic variables under the constraint that only one term in a term set can be a rule node's input. This gives the preconditions of initial fuzzy rules.

Finally, let us consider the structure of layer four in the initial network. This is equivalent to determining the consequents of initial fuzzy rules. To do the initialization, the number of fuzzy partitions of the output linguistic variable y, $|T(y)|$, must be given in advance. The initial fuzzy weights WY_i in layer four are distributed evenly on the output space. We let the layer-5 nodes to perform *up-down* transmission. In the up-down transmission, the desired outputs are pumped into the network from its output side and the operations in layer five are the same as those in layer two. Signals from both external sides of the network can thus reach the output points of term nodes at layer two and layer four. Since the outputs of term nodes at layer two can be transmitted to rule nodes through the initial architecture of layer-3 links, we can obtain the output (firing strength) of each rule node. Based on the firing strengths of rule nodes (o_i^3) and the outputs of term nodes at layer four (o_j^4), we can decide the correct consequent links of each rule node by unsupervised (self-organized) learning method [21]. The links at layer four are fully connected initially. We denote the weight on the link from the ith rule node to the jth output term node as w_{ij}. The Hebbian learning law is used to update these weights for each training data set. The learning rule is described as

$$\triangle w_{ij} = c o_j^4 o_i^3, \tag{35}$$

where c is a positive constant. After the learning, only one link with the biggest weight in the fan-out of a layer-3 (rule) node is remained as its consequent.

With the above initialization process, the network is ready for learning. We shall next propose a two-phase supervised learning algorithm for our five-layered neural fuzzy system.

4.1. Parameter learning phase

A gradient-descent-based backpropagation algorithm [21,22] is employed to adjust fuzzy weights in layer two and layer four of the proposed network. If the FCLO is used, the error function to be minimized is

$$e = \mathit{diff}(Y, D)$$
$$= \frac{1}{2} \sum_{\alpha} [(y_1^{(\alpha)} - d_1^{(\alpha)})^2 + (y_2^{(\alpha)} - d_2^{(\alpha)})^2], \tag{36}$$

where $Y = \bigcup_\alpha \alpha[y_1^{(\alpha)}, y_2^{(\alpha)}]$ is the current fuzzy output and $D = \bigcup_\alpha \alpha[d_1^{(\alpha)}, d_2^{(\alpha)}]$ is the desired fuzzy output. If the FCNO is used, the error function to be minimized is

$$e = \frac{1}{2}(d - y)^2, \tag{37}$$

where y is the current output and d is the desired output. We assume that $W = \bigcup_\alpha \alpha[w_1^{(\alpha)}, w_2^{(\alpha)}]$ is the

adjustable fuzzy parameter in layer two and layer four. Then to update fuzzy weights means to update the parameters $w_1^{(\alpha)}$ and $w_2^{(\alpha)}$. We shall next derive the update rules for these parameters layer by layer based on the general learning rule

$$w(t+1) = w(t) + \eta\left(-\frac{\partial e}{\partial w}\right), \tag{38}$$

where w represents $w_1^{(\alpha)}$ or $w_2^{(\alpha)}$, and η is the learning rate.

Layer 5. The update rules of $wy_{i1}^{(\alpha)}$ and $wy_{i2}^{(\alpha)}$ are derived from Eqs. (33) and (34) as follows:

$$\frac{\partial e}{\partial wy_{i1}^{(\alpha)}} = \frac{\partial e}{\partial y_1^{(\alpha)}}\frac{\partial y_1^{(\alpha)}}{\partial wy_{i1}^{(\alpha)}} = (y_1^{(\alpha)} - d_1^{(\alpha)})\frac{u_i^5}{\sum_i u_i^5}, \tag{39}$$

$$\frac{\partial e}{\partial wy_{i2}^{(\alpha)}} = \frac{\partial e}{\partial y_2^{(\alpha)}}\frac{\partial y_2^{(\alpha)}}{\partial wy_{i2}^{(\alpha)}} = (y_2^{(\alpha)} - d_2^{(\alpha)})\frac{u_i^5}{\sum_i u_i^5}. \tag{40}$$

The error signals to be propagated to the preceding layer are

$$\delta_1^{5(\alpha)} = \frac{\partial e}{\partial o_1^{5(\alpha)}} = \frac{\partial e_1^{(\alpha)}}{\partial y_1^{(\alpha)}} = (y_1^{(\alpha)} - d_1^{(\alpha)}), \tag{41}$$

$$\delta_2^{5(\alpha)} = \frac{\partial e}{\partial o_2^{5(\alpha)}} = \frac{\partial e_2^{(\alpha)}}{\partial y_2^{(\alpha)}} = (y_2^{(\alpha)} - d_2^{(\alpha)}), \tag{42}$$

where

$$e_1^{(\alpha)} = \frac{1}{2}(y_1^{(\alpha)} - d_1^{(\alpha)})^2, \tag{43}$$

$$e_2^{(\alpha)} = \frac{1}{2}(y_2^{(\alpha)} - d_2^{(\alpha)})^2. \tag{44}$$

Layer 4. In this layer, there is no weights to be adjusted. Only the error signals need to be computed and propagated. The error signal δ_i^4 is derived form Eq. (29) as follows:

$$\delta_i^4 = \frac{\partial e}{\partial o_i^4} = \frac{\partial \sum_\alpha (e_1^{(\alpha)} + e_2^{(\alpha)})}{\partial o_i^4} = \sum_\alpha (\delta_{i1}^{4(\alpha)} + \delta_{i2}^{4(\alpha)}), \tag{45}$$

where

$$\delta_{i1}^{4(\alpha)} = \frac{\partial e_1^{(\alpha)}}{\partial o_i^4} = \frac{\partial e_1^{(\alpha)}}{\partial o_1^{5(\alpha)}}\frac{\partial o_1^{5(\alpha)}}{\partial o_i^4} = \delta_1^{5(\alpha)}\frac{\partial y_1^{(\alpha)}}{\partial o_i^4} = \delta_1^{5(\alpha)}\frac{wy_{i1}^{(\alpha)}\sum_i u_i^5 - \sum_i u_i^5 wy_{i1}^{(\alpha)}}{(\sum_i u_i^5)^2}, \tag{46}$$

$$\delta_{i2}^{4(\alpha)} = \frac{\partial e_2^{(\alpha)}}{\partial o_i^4} = \frac{\partial e_2^{(\alpha)}}{\partial o_2^{5(\alpha)}}\frac{\partial o_2^{5(\alpha)}}{\partial o_i^4} = \delta_2^{5(\alpha)}\frac{\partial y_1^{(\alpha)}}{\partial o_i^4} = \delta_1^{5(\alpha)}\frac{wy_{i2}^{(\alpha)}\sum_i u_i^5 - \sum_i u_i^5 wy_{i2}^{(\alpha)}}{(\sum_i u_i^5)^2}. \tag{47}$$

Layer 3. As in layer four, only the error signals need to be computed. According to Eq. (27), this error signal δ_i^3 can be derived as

$$\delta_i^3 = \frac{\partial e}{\partial o_i^3} = \frac{\partial e}{\partial o_i^4}\frac{\partial o_i^4}{\partial o_i^3} = \begin{cases} \delta_i^4 & \text{if } o_i^4 = \max(u_1^4, \ldots, u_k^4), \\ 0 & \text{otherwise.} \end{cases} \tag{48}$$

Layer 2. In this layer, there are fuzzy weights WX to be adjusted. The update rules can be derived from Eqs. (23) and (24) as follows:

$$\frac{\partial e}{\partial wx_{ij1}^{(\alpha)}} = \frac{\partial e}{\partial o_i^3}\frac{\partial o_i^3}{\partial o_{ij}^2}\frac{\partial o_{ij}^2}{\partial wx_{ij1}^{(\alpha)}} = \delta_i^3\frac{\partial o_i^3}{\partial o_{ij}^2}\frac{\partial o_{ij}^2}{\partial wx_{ij1}^{(\alpha)}}, \tag{49}$$

$$\frac{\partial e}{\partial wx_{ij2}^{(\alpha)}} = \frac{\partial e}{\partial o_i^3}\frac{\partial o_i^3}{\partial o_{ij}^2}\frac{\partial o_{ij}^2}{\partial wx_{ij2}^{(\alpha)}} = \delta_i^3\frac{\partial o_i^3}{\partial o_{ij}^2}\frac{\partial o_{ij}^2}{\partial wx_{ij2}^{(\alpha)}}, \tag{50}$$

where

$$\frac{\partial o_i^3}{\partial o_{ij}^2} = \begin{cases} 1 & \text{if } o_{ij}^3 = \min(u_1^3, \ldots, u_k^3) \\ 0 & \text{otherwise,} \end{cases} \tag{51}$$

and

$$\frac{\partial o_{ij}^2}{\partial wx_{ij1}^{(\alpha)}} = o_{ij}^2 \ln e\left(-\frac{2f_{ij}^2}{2\sigma}\right)\frac{\partial f_{ij}^2}{\partial wx_{ij1}^{(\alpha)}} = -o_{ij}^2\frac{f_{ij}^2}{\sigma^2}(wx_{ij1}^{(\alpha)} - u_{i1}^{2(\alpha)}), \tag{52}$$

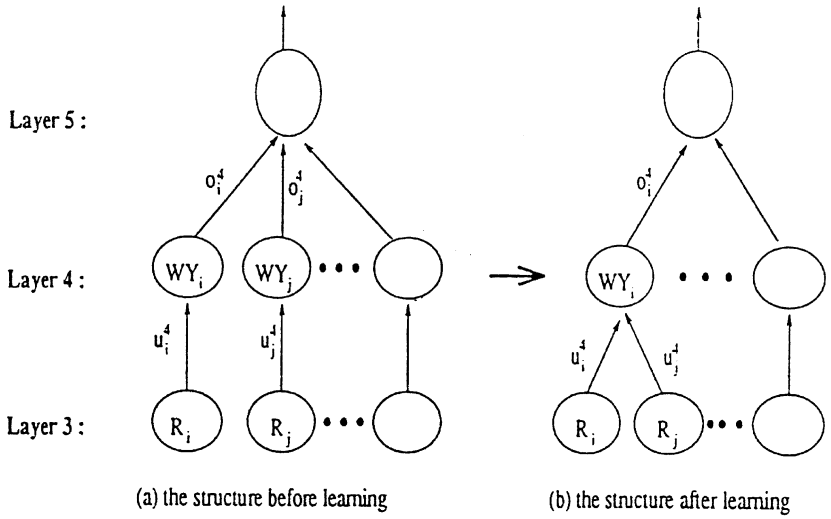

Fig. 5. Illustration of consequent combination.

$$\frac{\partial o_{ij}^2}{\partial wx_{ij2}^{(\alpha)}} = -o_{ij}^2 \frac{f_{ij}^2}{\sigma^2}(wx_{ij2}^{(\alpha)} - u_{i2}^{2(\alpha)}). \tag{53}$$

4.2. Structure learning phase

In this subsection, we propose a structure learning algorithm for the proposed neural fuzzy system to reduce its node and link number. This structure learning algorithm is divided into two parts: One is to merge the fuzzy terms of input and output linguistic variables (*term-node combination*). The other is to do *rule combination* to reduce the number of rules. We shall discuss these two parts separately in the following.

A. *Term-node combination scheme*

Term-node combination is to combine similar terms in the term sets of input and output linguistic variables. We shall present this technique for the term set of output linguistic variables. The whole learning procedure of initialization is described as follows:

Step 1: Perform parameter learning until the output error is smaller than a given value; i.e., $e \leqslant error\_limit$, where *error_limit* is a small positive constant.

Step 2: If $diff(WY_i, WY_j) \leqslant similar\_limit$ and *similar_limit* is a given positive constant, remove term node j with fuzzy weight WY_j and its fan-out links, and connect rule node j in layer 3 to term node i in layer four (see Fig. 5).

Step 3: Perform the parameter learning again to optimally adjust the network weights.

B. *Rule combination scheme*

After the fuzzy parameters and the consequents of the rule nodes are determined, the rule combination scheme is performed to reduce the number of rules. The conditions for applying rule combination has been explored in [17] and are given as follows.

(1) These rule nodes have exactly the same consequents.
(2) Some preconditions are common to all the rule nodes, that is, the rule nodes are associated with the same term nodes.
(3) The union of other preconditions of these rule nodes composes the whole terms set of some input linguistic variables.

If some rule nodes satisfy these three conditions, then these rules can be combined into a single rule. An illustration in shown in Fig. 6.

5. Illustrative examples

In this section, we shall use two examples to illustrate the performance of the proposed neural fuzzy system.

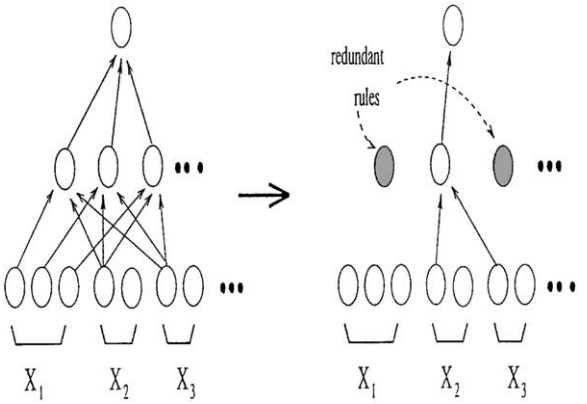

Fig. 6. Illustration of rule combination.

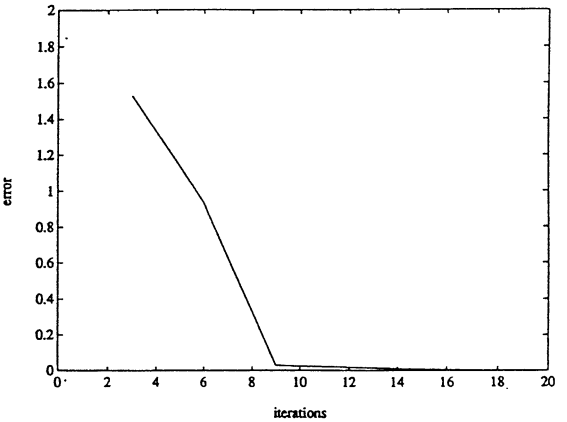

Fig. 8. The learning curve in Example 1.

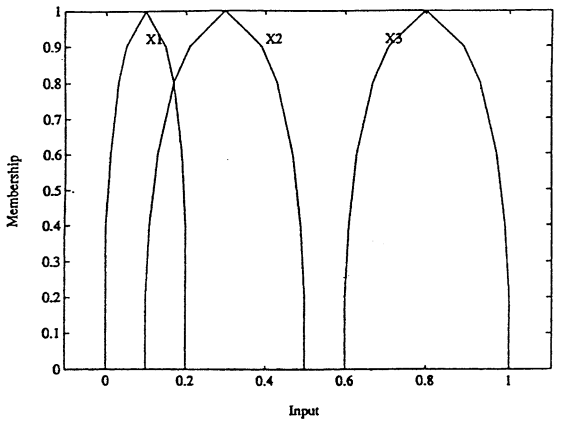

Fig. 7. The membership functions of the input linguistic value "very small" (X1), "small" (X2), "large" (X3) in Example 1.

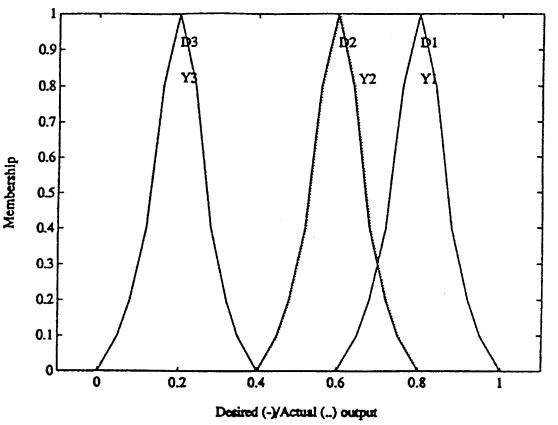

Fig. 9. The actual fuzzy outputs, Y1, Y2, Y3 of the learned neural fuzzy system and the corresponding desired fuzzy outputs, D1, D2, D3 in Example 1.

Example 1. Fuzzy input and fuzzy output.

Consider the following three fuzzy if-then rules for training:

R_1: IF x is very small (X1), THEN y is very large (D1),

R_2: IF x is small (X2), THEN y is large (D2),

R_3: IF x is large (X3), THEN y is small (D3),

where the fuzzy numbers "small", "large", "very small" are given in Fig. 7. Fig. 8 shows the learning curve. The error tolerance is 0.0001 and the number of α-cuts is 6. After supervised learning, the fuzzy outputs of the learned network and the corresponding desired outputs are shown in Fig. 9. The figure shows that they match closely. The two learned (representative) fuzzy rules after learning (condensing) are:

IF x is WX1, THEN y is WY1, and

IF x is WX2, THEN y is WY2,

where the fuzzy weights after learning are shown in Fig. 10. For illustration, Figs. 11 and 12 show the change of fuzzy weights in the learning process. Hence the original three fuzzy rules have been condensed to two rules, and these two sets of rules represent equivalent knowledge.

Example 2. Fuzzy input and fuzzy output.

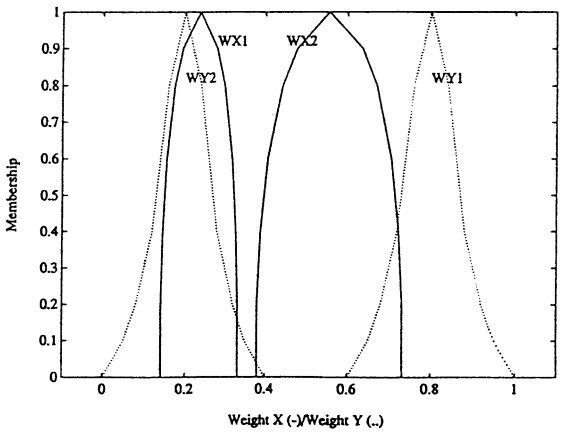

Fig. 10. The learned fuzzy weights of the network in Example 1.

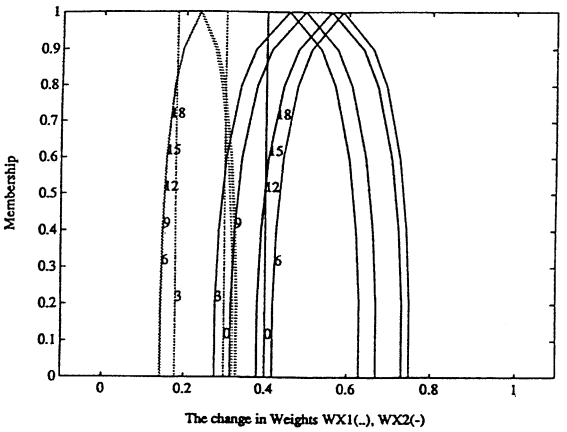

Fig. 11. Time evolving graph of fuzzy weights $WX1$, $WX2$ during the learning process in Example 1.

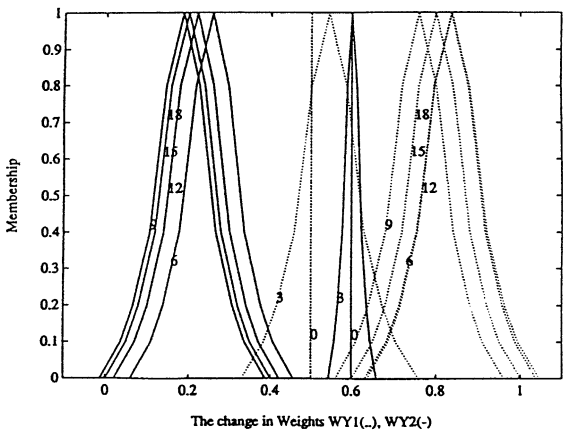

Fig. 12. Time evolving graph of fuzzy weights $WY1$, $WY2$ during the learning process in Example 1.

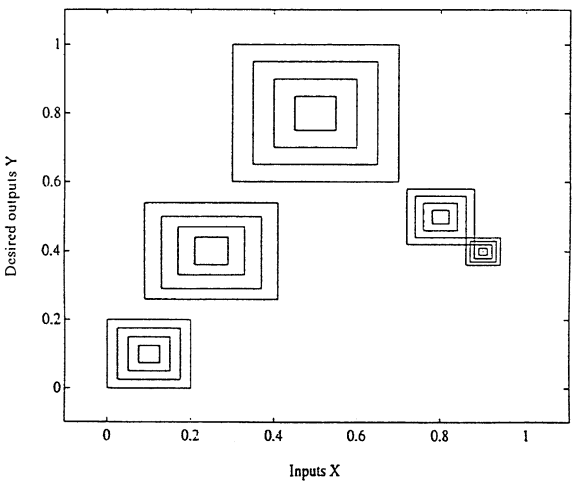

Fig. 13. The training data in Example 2.

There are five training data in Fig. 13. Fig. 15 shows the fuzzy weights after training. In order to examine the generalization ability of the trained neural network, we presented the three fuzzy inputs for testing in Fig. 14. In these figures, five level sets corresponding to $h = 0, 0.25, 0.5, 0.75, 1$ are depicted.

6. Conclusions

In this study, we proposed the learning techniques for neural fuzzy systems to process both numerical and word information. The developed systems have some characteristics and advantages: (1) The inputs and outputs can be fuzzy numbers or numerical numbers. (2) The network weights are fuzzy weights. (3) Owing to the representation forms of the α-level sets, the fuzzy weights, fuzzy inputs, and fuzzy outputs can be fuzzy numbers of any shape. (4) Except the input and output layers, numerical numbers are propagated through the whole network; thus the operations in the proposed neural fuzzy systems are not time-consuming and the required memory capacity is small. The developed systems have fuzzy supervised learning capability. With fuzzy supervised learning, these systems can be used for fuzzy expert systems, fuzzy system modeling, and rule base concentration. When learning with numerical values (real vectors), the pro-

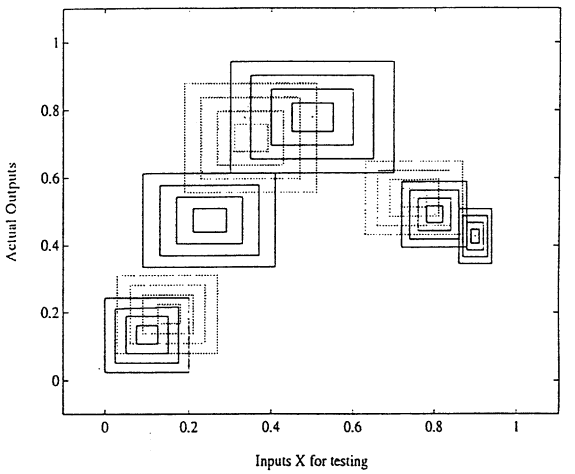

Fig. 14. The actual outputs and testing results in Example 2.

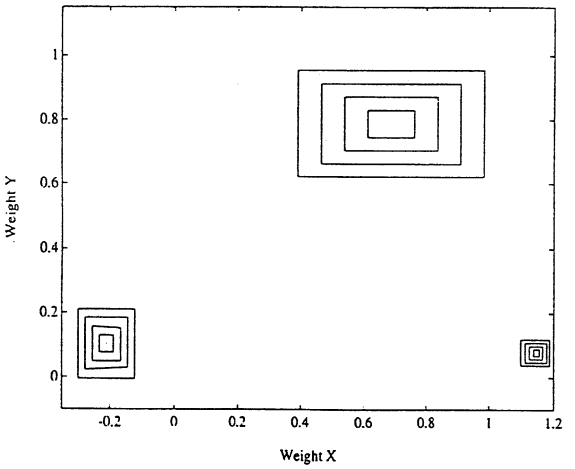

Fig. 15. The fuzzy weights in Example 2.

posed systems can be used for adaptive fuzzy control. Computer simulations and experimental studies satisfactorily verified the performance of the proposed neural fuzzy learning schemes.

References

[1] S.K. Pal, S. Mitra, Multilayer perceptron, fuzzy sets and classification, IEEE Trans. Neural Networks 3 (5) (1992) 683–696.

[2] J.M. Keller, H. Tahani, Backpropagation neural networks for fuzzy logic, Inform. Sci. 62 (1992) 205–221.

[3] S. Horikawa, T. Furuhashi, Y. Uchikawa, On fuzzy modeling using fuzzy neural networks with the backpropagation algorithm, IEEE Trans. Neural Networks 3 (5) (1992) 801–806.

[4] H. Ishibuchi, R. Fujioka, H. Tanaka, Neural networks that learn from fuzzy if-then rules, IEEE Trans. Fuzzy Systems 1 (2) (1993) 85–97.

[5] H. Ishibuchi, H. Tanaka, Fuzzy regression analysis using neural networks, Fuzzy Sets and Systems 50 (1992) 257–265.

[6] H. Ishibuchi, H. Tanaka, H. Okada, Fuzzy neural networks with fuzzy weights and fuzzy biases, Proc. Int'l Joint Conf. on Neural Networks, San Francisco, 1993, pp. 1650–1655.

[7] Y. Hayashi, J.J. Buckley, E. Czogula, Fuzzy neural network, Internat. J. Intelligent Syst. 8 (1993) 527–537.

[8] Y. Hayashi, J.J. Buckley, E. Czogula, Systems engineering application of fuzzy neural networks. Proc. Int'l Joint Conf. on Neural Networks, Baltimore, 1992, pp. 413–418.

[9] A. Kaufmann, M.M. Gupta, Introduction to Fuzzy Arithmetic, Van Nostrand Reinhold, New York, 1985.

[10] K. Uehara, M. Fujise, Fuzzy inference based on families of α-level sets, IEEE Trans. Fuzzy Systems 1 (2) (1993) 111–124.

[11] L.A. Zadeh, The concept of a linguistic truth variable and its application to approximate reasoning—I, II. Inform. Sci. 8 (1975) 199–249, 301–357.

[12] M. Braae, D.A. Rutherford, Fuzzy relations in a control setting, Kyberbetes 7 (3) (1978) 185–188.

[13] C.C. Lee, Fuzzy logic in control systems: Fuzzy logic controller—Part I & II, IEEE Trans. Syst. Man Cybern. SMC-20 (2) (1990) 404–435.

[14] T. Yamakawa, The current mode fuzzy logic integrated circuits fabricated by the standard CMOS process, IEEE Trans. Comput. 35 (2) (1992) 122–130.

[15] K. Uehara, Computational efficiency of fuzzy inference based on level sets, Proc. Spring Nat. Conv. Rec., IEICE, JAPAN, 1989, pp. D-400.

[16] K. Uehara, Fast operation of fuzzy inference based on level sets. Proc. 38th Ann. Conv. Rec. IPS Japan Rec., 1989, pp. 3G-3.

[17] C.T. Lin, C.S.G. Lee, Neural-network-based fuzzy logic control and decision system, IEEE Trans. Comput. 40 (12) (1991) 1320–1336.

[18] C.T. Lin, C.S.G. Lee, Reinforcement structure/parameter learning for neural-network-based fuzzy logic control systems, IEEE Trans. Fuzzy Systems 2 (1) (1995) 46–63.

[19] J.S. Jang, Self-learning fuzzy controllers based on temporal back propagation, IEEE Trans. Neural Networks 3 (5) (1992) 714–723.

[20] T. Tsukamoto, An approach to fuzzy reasoning method, in: M.M. Gupta, R.K. Regade, R.R. Yager (Eds.), Advances in Fuzzy Set Theory and Applications, North-Holland, Amsterdam, 1979.

[21] J.M. Zurada, Introduction to Artificial Neural Systems, West publisher, New York, 1992.

[22] G.E. Hinton, Connectionist learning procedure, Artificial Intelligence 40 (1) (1989) 143–150.

[23] R. Keller, Expert System Technology—Development and Application, Prentice-Hall, NJ, 1987.