Refinement of Generated Fuzzy Production Rules by Using a Fuzzy Neural Network

Eric C. C. Tsang, Daniel S. Yeung, *Senior Member, IEEE*, John W. T. Lee, *Member, IEEE*, D. M. Huang, and X. Z. Wang

***Abstract*—Fuzzy production rules (FPRs) have been used for years to capture and represent fuzzy, vague, imprecise and uncertain domain knowledge in many fuzzy systems. There have been a lot of researches on how to generate or obtain FPRs. There exist two methods to obtain FPRs. One is by painstakingly, repeatedly and time-consuming interviewing domain experts to extract the domain knowledge. The other is by using some machine learning techniques to generate and extract FPRs from some training samples. These extracted rules, however, are found to be nonoptimal and sometimes redundant. Furthermore, these generated rules suffer from the problem of low accuracy of classifying or recognizing unseen examples. The reasons for having these problems are 1) the FPRs generated are not powerful enough to represent the domain knowledge, 2) the techniques used to generate FPRs are pre-matured, ad-hoc or may not be suitable for the problem, and 3) further refinement of the extracted rules has not been done. In this paper we look into the solutions of the above problems by 1) enhancing the representation power of FPRs by including local and global weights, 2) developing a fuzzy neural network (FNN) with enhanced learning algorithm, and 3) using this FNN to refine the local and global weights of FPRs. By experimenting our method with some existing benchmark examples, the proposed method is found to have high accuracy in classifying unseen samples without increasing the number of the FPRs extracted and the time required to consult with domain experts is greatly reduced.**

I. INTRODUCTION

FUZZY logic and fuzzy sets invented by Zadeh [1]–[3] in 1960s help people develop machines which could capture and represent approximate reasoning capability used by human beings. The outcome is that fuzzy logic and fuzzy sets not only increase the reasoning power of many machines but are also easy to implement in many areas. The most widely used area is in fuzzy controllers [4], [5]. The fuzzy controllers with many fuzzy control rules will capture the reasoning process of human operators. Nowadays many useful commercial products with intelligent (approximate reasoning) and automatic control have been developed. FPRs have been the most popular and easiest way to capture and represent fuzzy, vague, imprecise and uncertain domain knowledge. They could be found in different kinds of fuzzy systems, no matter whether it is a fuzzy controller or a fuzzy expert system. The brain or heart of these fuzzy systems is their knowledge (the FPRs) it captured or stored in the knowledge base. Traditionally, these FPRs are provided and extracted from domain experts. It is very difficult and time-consuming to obtain accurate and reasonable FPRs.

In recent years there have been a lot of researches on how to use powerful computers to help us generate and produce FPRs from a set of sample data. Many machine learning techniques could be found in the literatures [6]–[14]. An automated fuzzy knowledge base generation and tuning method is presented in [6]. In [7] Hong *et al.* provide a method to construct membership functions for FPRs generation, while in [8] a way to process individual fuzzy attributes for fuzzy rule induction is given. A GA method to select fuzzy if-then rules for classification problems could be found in [9]. Kao *et al.* propose a method to generate FPRs from training data with noise for classification problems [10]. In [11] Ravi *et al.* present a method for generation of fuzzy rule base and its optimization by using modified threshold accepting. An interpolation technique is used to learn fuzzy rules [12]. Wang *et al.* present a method to optimize and simplify fuzzy rules [13]. A fuzzy decision tree induction technique is proposed to generate fuzzy rules [14]. Although these machine learning techniques allow us to generate a set of FPRs, these extracted rules, however, are found to be far from optimal and sometimes redundant. Using machine learning techniques to generate and produce approximately optimal and useful FPRs should include two phases. The first phase involves generating a set of rough, crude and raw FPRs by using one or more machine learning techniques from a set of training sample while the second phase involves using other techniques to refine, tune or enhance the initial rough FPRs so that approximately optimal and accurate FPRs are obtained. This paper focuses on the second phase by using an enhanced FNN.

In this paper, a set of rough, crude and raw FPRs are assumed to have been extracted or generated by some machine learning techniques. These FPRs, like traditional FPRs, are obtained without any weights (local and/or global) assigned. In order to enhance the representation power of FPRs, the knowledge representation parameters (KRPs) such as local and global weights are included in these FPRs. These local and global weights had been proposed by Yeung *et al.* [15], [16]. A fuzzy neural network (FNN) is proposed to refine or tune the local and global weights of FPRs. A set of weighted FPRs (WFPRs) obtained will be more optimal and accurate in recognizing and classifying unseen samples. It is because those weights with values more or less equal to zero could be deleted so that smaller number of propositions in the antecedent of WFPRs (so-called simple WFPRs) are generated. Furthermore, the

Manuscript received June 13, 2002; revised December 19, 2002. This work was supported by the Hong Kong Research Grant Council under Grant B-Q519. This paper was recommended by Associate Editor R. Jang.

E. C. C. Tsang, D. S. Yeung, and J. W. T. Lee are with the Department of Computing, The Hong Kong Polytechnic University, Kowloon, Hong Kong (e-mail: csetsang@comp.polyu.edu.hk; csdaniel@comp.polyu.edu.hk; csjlee@comp.polyu.edu.hk).

D. M. Huang is with the College of Science, Hebei Agriculture University, Hebei, China.

X. Z. Wang is with the Machine Learning Center, College of Mathematics and Computer Science, Hebei University, Hebei, China.

Digital Object Identifier 10.1109/TSMCB.2003.817033

extracted WFPRs with local and global weights capturing more domain experts' knowledge will have higher accuracy in solving recognition and classification problems. A FNN offers advantages of allowing us to map these KRPs (local and global weights) of FPRs into the connection weights of a FNN and with a modified backpropagation (BP) learning algorithm, we are able to tune, refine and even acquire these parameters. In [17] eleven categories of FNNs have been identified. The FNN used in this paper is similar to the fuzzy-like neuro model where a neural network is used to represent fuzzy rules. The difference is that our FNN is used to represent WFPRs which could be refined or tuned so that approximately optimal rules and higher testing accuracy could be obtained. In [18], [19] two FNN models are proposed to solve parameters tuning of fuzzy membership functions. The problem settings of ANFIS in [19] is that it is used to represent three types of fuzzy inference systems used in fuzzy controlled systems whose rules are parallel in nature, whereas our proposed method could handle multi-level WFPRs and extends the traditional method to a more general one.

This paper is organized as follows. In Section II a WFPR together with its reasoning method is presented. Section III presents a mapping architecture of how a WFPR could be mapped into a FNN. Section IV provides the convergence of fuzzy learning rule and derives the back-propagation learning algorithm for the FNN. In Section V experimentation with some benchmark examples are used to demonstrate the workability of our proposed method. The final section presents a conclusion and future work. Throughout this paper, the notation $\wedge$ is used to denote *min* operation.

II. A WFPR and Its Reasoning Algorithm

A. A Weighted Fuzzy Production Rule

A crisp production rule takes a form of "IF A THEN B", where A, which is called propositions of the antecedent, is a conjunction or disjunction of several crisp subsets, B which is the consequent of the rule, is also regarded as a crisp subset. The propositions and the consequent in a FPR can be linguistic terms such as "big", "high", which are regarded as fuzzy subsets. Usually, all propositions in the antecedent are assumed to have equal degree of importance and a number of rules leading to the same consequent are also regarded as having the same relative degree of importance.

To enhance the knowledge representation power of fuzzy production rules, a generic form of FPRs is suggested in [15], [16] where a threshold value and a local weight are assigned to each proposition while a global weight is assigned to the entire rule. This paper discusses a type of FPRs in which two important knowledge parameters the local and global weights are emphasized. For instance, a conjunctive Weighted Fuzzy Production Rule takes the form of:

$$\begin{aligned}&\mathbf{R}: \text{If } (V_1 \text{ is } A_1[Lw_1]) \text{ AND } --- \text{ AND } (V_n \text{ is } A_n[Lw_n])\\&\qquad \text{THEN } U \text{ is } B, [Gw]\\&\text{Fact 1}: V_1 \text{ is } {A_1}^*, ---, \text{Fact n}: V_n \text{ is } {A_n}^*\\&\text{Conclusion}: U \text{ is } B^*\end{aligned}$$

where $V_1, ---, V_n$ and U are attributes and $A_1, ---, A_n$ and B are the values of these attributes, which are fuzzy. $Lw_i(1 \leq i \leq n)$ is the local weight of the proposition "V_i is A_i" and each Lw_i is nonnegative. Gw denotes the global weight assigned to the entire rule $\mathbf{R}(Gw \geq 0)$.

The local weight is introduced for the purpose of indicating the relative degree of importance of a proposition contributing to its consequent while the global weight concept is used to represent the relative degree of importance of each rule's contribution to reach a final goal.

B. Max-Min Fuzzy Matching Algorithm

When the given fact for an antecedent in a FPR does not match exactly with the antecedent of the rule, the approximate matching and reasoning should be used to draw the consequent. The following is a frequent-used max-min fuzzy matching algorithm [14].

Let RS be a set of fuzzy classification rules without weight and F be an observed object to be classified.

Step 1) For each rule, calculate the membership degree of the observed object in the antecedent. The membership degree of each proposition in the antecedent is a similarity degree between the observed attribute value of the object and the proposition. The overall membership degree of the antecedent is regarded as the minimum among all the proposition memberships (in a conjunctive FPR). The membership of the consequent (the classification to one class) will be set equal to the membership degree of the antecedent.

Step 2) When two or more rules are applied and classify the observed object into the same class with different membership degrees, take the maximum of these membership degrees as the membership degree of the class (consequent).

Step 3) An object may be classified into several classes with different membership degrees. When classification to only one class is required, select the class with the highest membership degree.

From Steps 1 and 2, it is easy to see the classification result of the observed object is a vector in which each component corresponds to a degree of belonging to a certain class. In the following paragraph, we present an example to show that if weights are not assigned it is very difficult to provide a sound crisp decision.

Example 1: Consider the following three fuzzy rules R1, R2, R3 and one observed case F:

R1: IF (T is Hot) AND (O is Sunny) THEN U is Swimming

R2: IF (T is Hot) AND (O is Cloudy) THEN U is Swimming

R3: IF (T is Mild) AND (W is Not-windy) THEN U is Volleyball

Case: $\mathrm{F} = 1.0/\mathrm{Sunny} + 0.2/\mathrm{Cloudy} + 0.5/\mathrm{Hot} + 0.5/\mathrm{Mild} + 0.0/\mathrm{Windy} + 1.0/\mathrm{Not-windy}$. (T: Temperature = Hot, Cool, Mild}, O: Outlook = Sunny, Cloudy, rain}, W: Wind = {Windy, Not-Windy}).

The matching process is as follows: (0/Swimming, 0/Volleyball) $\xrightarrow{R1}$ (0.5/Swimming, 0/Volleyball) $\xrightarrow{R2}$ (0.5/Swimming, 0/Volleyball) $\xrightarrow{R3}$ (0.5/Swimming, 0.5/ Volleyball). One may notice that it is difficult to make a crisp decision because Swimming and Volleyball have the same membership degree.

C. Weighted Fuzzy Reasoning Method

Now Let us state our weighted fuzzy reasoning method. We can regard the method as a kind of similarity-based reasoning algorithm. The similarity degree between the attribute values of an example and the antecedent of the rule is considered as the membership value that indicates to what degree the example belongs to the corresponding term. For instance, the similarity between attribute value "0.6/Hot + 0.4/Mild + 0.0/Cool" and the antecedent "Temperature = Hot" is 0.6.

Consider a set of fuzzy production rules $S = \{R_i, i = 1, 2, \cdots, m\}$ where R_i takes the form of

$$\mathbf{R}_i : \text{If } \left(V_1 \text{ is } A_1^{(i)} \left[Lw_1^{(i)}\right]\right) \text{ AND} - - - \text{ AND } \left(V_n \text{ is } A_n^{(i)} \left[Lw_n^{(i)}\right]\right) \text{ THEN } U \text{ is } B^{(i)},\ [Gw_i].$$

The observed object has attribute values in the following form:

$$\text{Fact1} : V_1 \text{ is } C_1^{(i)},\ \text{Fact2} : V_2 \text{ is } C_2^{(i)},\ - - -,\ \text{Fact}n : V_n \text{ is } C_n^{(i)}$$

For each rule R_i within S, the similarity between the proposition $A_j^{(i)}$ and the observed attribute-value $C_j^{(i)}$, denoted by $SM_j^{(i)}$, is defined as the membership value that indicates to what degree the example belongs to the corresponding term. The overall similarity $SM^{(i)}$ is defined as

$$SM^{(i)} = Min_{1 \le j \le n} \left(Lw_j^{(i)} \cdot SM_j^{(i)}\right).$$

Let there be K fuzzy sets of conclusions (K fuzzy clusters in the learning from fuzzy examples). The conclusions of the given m rules can be classified into K groups, denoted by $CLASS_1, - - -, CLASS_K$. The inferred result is regarded as a vector $(x_1, x_2, \cdots, x_K)$. The degree x_k is determined by the following equation:

$$x_k = \sum_{B^{(i)} = CLASS_k} Gw_i \cdot SM^{(i)} \quad (k = 1, 2, - - -, K).$$

The normalized form of the inferred result is defined as $(d_1, d_2, \cdots, d_K)$ where d_k is the value which indicates to what degree the observed object belongs to $CLASS_k (k = 1, 2, - - -, K)$

$$d_k = \frac{x_k}{Max_{1 \le j \le k} x_j} \quad (k = 1, 2, - - -, K).$$

When the crisp inferred result is needed, one can take the consequent $CLASS$ with maximum $d_k (1 \le k \le K)$.

The max-min fuzzy matching algorithm presented in the previous subsection could easily be modified or enhanced to accommodate such changes. In the following paragraph, we present example 2, which is the same as example 1 but with weights assigned, to indicate how we could obtain a crisp decision by using assigned weights.

Example 2: If the three rules in example 1 are changed into the following three WFPRs:

R1: IF (T is Hot [1.0]) AND (O is Sunny [0.6]) THEN U is Swimming, [1.0].

R2: IF (T is Hot [0.9]) AND (O is Cloudy [1.0]) THEN U is Swimming, [0.8].

R3: IF (T is Mild [0.4]) AND (W is Not-windy [0.9]) THEN U is Volleyball, [1.0].

Then the normalized inferred vector of the case is (1/Swimming, 0.3/Volleyball). It is easy for us to make a crisp decision.

III. Mapping a WFPR and Its Reasoning Algorithm to a FNN

A set of WFPRs and the proposed weighted fuzzy reasoning algorithm can exactly be mapped into a three-layer FNN. These three layers are called Term layer, Rule layer, and Classification layer. We describe the structure of the mapped FNN as follows.

Term layer: This is the input layer (layer i). Each node in this layer represents a linguistic term of an attribute. Since each linguistic term corresponds to an attribute value, the input of each node is regarded as the similarity degree between the observed attribute value and the corresponding term (proposition) of the antecedent in a WFPR. The similarity degree can also be the membership value that indicates to what degree the observed fact belongs to the linguistic term.

Rule layer: This is the only hidden layer (layer j). Each node in this layer represents a given antecedent part of a rule. According to linguistic terms (propositions) appeared in the antecedent part of a rule, the connections between the term layer and the rule layer are determined.

Classification layer: This is the output layer (layer k). Each node in this layer represents a fuzzy cluster. Since the inferred result of a WFPR has generally the form of vector (discrete fuzzy set defined on the space of cluster labels), the output of the network has more than one value. The meaning of each output value after normalization is the membership value that indicates to what degree the training object belongs to the cluster corresponding to the node.

Connection weights: The local weights (shown as Lw_{ij}) of a set of WFPRs are regarded as the connection weights between the term layer and the rule layer. The global weights (shown as Gw_{jk}) of the set of WFPRs are regarded as the connection weights between the rule layer and the classification layer.

Fig. 1 presents a generic conjunctive WFPR mapped to a FNN which could be used to refine and tune local and global weights.

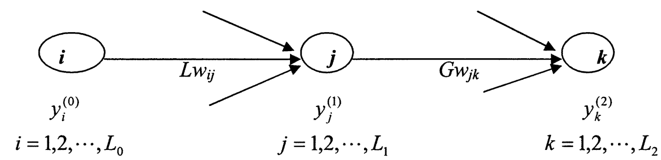

Fig. 1. Generic FNN for a conjunctive WFPR.

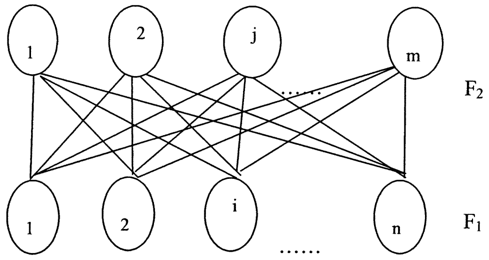

Fig. 2. $\wedge, \cdot$ operator network.

IV. Back-Propagation Algorithm and the Convergence of the Fuzzy Learning Rule

A. The Convergence of the Fuzzy Learning Rule

Let us consider a two-layer feed forward FNN as shown in Fig. 2 in which the neuron unit is a fuzzy neuron with fuzzy operators $(\wedge, \cdot)$.

The training method is presented as follows.

Step 1) Setting the initial connection weights

$$\text{Setting } W_{ij} = 1, \quad i = 1, 2, \cdots, n; \quad j = 1, 2, \cdots, m.$$

Step 2) Calculating the actual output

$$\left(b_i^j\right)' = \overset{n}{\underset{k=1}{\wedge}} \left(W_{kj} \cdot a_i^k\right) \quad j = 1, 2, \cdots, m$$

where $A_i = (a_i^1, a_i^2, \cdots, a_i^n)$ $i = 1, 2, \cdots, n$ is the vector of pattern's inputs.$B_i = (b_i^1, b_i^2, \cdots, b_i^m)$ is the vector of pattern's outputs. $((b_i^1)', (b_i^2)', \cdots, (b_i^m)')$ is the actual response for the input pattern A_i. W_{ij} stands for the connection weight from node i in F_1 to node j in F_2.

Step 3) Adjusting the connection weight

$$\text{Let} \quad \delta_{ij} = \left(b_i^j\right)' - b_i^j$$

$$W_{kj}^{new} = \begin{cases} W_{kj}^{old} - \eta \cdot \delta_{ij} & if\ W_{kj}^{old} \cdot a_i^k < b_i^j \\ W_{kj}^{old} & otherwise \end{cases}$$

where $\eta \in (0, 1]$ denotes the learning rate.

Step 4) Go to Step 3 until $W_{kj}^{new} = W_{kj}^{old}$ hold for all k and j.

Step 5) Repeat Step 2 for the new input and output pattern.

This algorithm is called the fuzzy learning rule. To the fuzzy learning rule, we have the following theorems.

Theorem 1: The fuzzy learning rule is convergent.

Proof: This theorem is an extension of δ-rule found in [24]. From the steps 1 to 5, it is easy to see that this learning rule converges.

Theorem 2: If a solution to the following equation-group exists:

$$\overset{n}{\underset{k=1}{\wedge}} \left(W_{kj} \cdot a_i^k\right) = b_i^j \quad i = 1, 2, \cdots, n; \quad j = 1, 2, \cdots, m$$

then the fuzzy learning rule algorithm can converge to the W° (W° is an $m \times n$ matrix) such that W° satisfy the above equation-group.

Proof: From the theory of fuzzy relation equation [25], one may notice that in each iteration when this neural network learns, it searches for a matrix of weights so that $\wedge_{k=1}^{n}(W_{kj} \cdot a_i^k) = b_i^j$ $i = 1, 2, \cdots, n$; $j = 1, 2, \cdots, m$, i.e., it tries to find a solution for the fuzzy relation equation-group. If W^0 (an $m \times n$ matrix) exists and is the solution of this fuzzy relation equation-group, then it is true that the fuzzy learning rule converges to this W^0. This completes the proof.

Theorem 2 shows that a two-layer FNN with $(\wedge, \cdot)$ operators can produce the fuzzy relation: $A_i = (a_i^1, a_i^2, \cdots, a_i^n) \rightarrow B_i = (b_i^1, b_i^2, \cdots, b_i^m)$ $i = 1, 2, \cdots, n$ by learning.

B. Generic Example of a FNN

To formulate the back-propagation algorithm, let us consider a generic case of our proposed FNN as shown in Fig. 1, where there are L_0 Term nodes, L_1 Rule nodes and L_2 Classification nodes. For a given input vector, e.g. the nth input vector, the feed forward propagation process is described as follows:

$$\text{The initial layer(Term layer)}: \left\{ y_i^{(0)}[n] | i = 1, 2, \cdots, L_0 \right\}$$
$$\text{(the given input vector);}$$
$$\text{The first layer(Rule layer)}: y_j^{(1)}[n] = \wedge_{i=1}^{L_0} \times \left(Lw_{ij} \cdot y_i^{(0)}[n] \right) \quad j = 1, 2, \cdots, L_1 \tag{1}$$

$$\text{The second layer(Class layer)}: y_k^{(2)}[n] = \sum_{j=1}^{L_1} Gw_{jk} \cdot y_j^{(1)}[n], \quad k = 1, 2, \cdots, L_2. \tag{2}$$

Let there be N training sample data. Then, the total error function is usually defined as

$$E = \frac{1}{2} \sum_{n=1}^{N} \sum_{k=1}^{L_2} (d_k[n] - y_k[n])^2 = \sum_{n=1}^{N} \left(\frac{1}{2} \sum_{k=1}^{L_2} (d_k[n] - y_k[n])^2 \right) = \sum_{n=1}^{N} E_n \tag{3}$$

where $d_k[n] = y_k^{(2)}[n] / \max_{1 \le k \le L_2} \{y_k^{(2)}[n]\}$ is a normalization value of the k-th actual output of the n-th training sample $(1 \le k \le L_2)$. It is easy to see from (1), (2), and (3) that the error E is a function with respect to the local weight Lw_{ij} and the global weight $Gw_{jk}(i = 1, \cdots, L_0;\ j = 1, \cdots, L_1;\ k = 1, \cdots, L_2)$. The main objective of learning is to adjust these weights so that the error function reaches minimum or is less than a given small value ε.

C. Enhanced Back-Propagation Algorithm for the FNN

A back-propagation, one of the most popular and powerful learning algorithms, has been proposed for years to learn a multilayer neural network with three or more layers. In our proposed FNN, we establish an enhanced back-propagation algorithm by modifying the smooth derivative introduced in [20], which is briefly described as follows.

The usual derivatives $\partial(y \wedge p)/\partial y = \begin{cases} 1 & if\ y \le p \\ 0 & if\ y > p \end{cases}$ and $\partial(y \vee p)/\partial y = \begin{cases} 1 & if\ y \ge p \\ 0 & if\ y < p \end{cases}$ are regarded as the crisp truth degree of the proposition "y is less than or equal to p" and the crisp truth degree of the proposition "y is greater than or equal to p," respectively. To improve the performance of training, these crisp behaviors will be replaced by fuzzy behaviors which are able to capture the real meaning of $(y \leq p)$ and $(y \geq p)$ in a vague context [20]. Since the relative position of y with respect to p is softened, the relative position could be regarded as the minority degree of p with respect to y, denoted by $\|p \leq y\|$. Noting that when $p \leq y$ then $\|p \leq y\| = 1$, whereas when $p > y$ it is reasonable to consider the minority degree of $\|p \leq y\|$ to be equal to y. The Godel implication is the most suitable one. Consequently, two enhanced derivatives are defined as follows:

$$\frac{\partial(x \vee c)}{\partial x} = \begin{cases} 1 & if\ x \geq c \\ x & if\ x < c \end{cases} \text{ and } \frac{\partial(x \wedge c)}{\partial x} = \begin{cases} 1 & if\ x \leq c \\ c & if\ x > c \end{cases}. \tag{4}$$

Let us now derive the standard back-propagation equations. According to the principle of gradient descent, the back-propagation equations for the FNN as shown in Fig. 1 can be written as

$$Lw_{ij} = Lw_{ij} - \alpha \frac{\partial E_n}{\partial Lw_{ij}} \text{ and } Gw_{jk} = Gw_{jk} - \beta \frac{\partial E_n}{\partial Gw_{jk}} \tag{5}$$

where α and β are the learning rate. Therefore, the problem of derivation is how to evaluate the two partial derivatives appeared in (5).

The detailed derivation are provided in the Appendix I and the derived results are as shown in the equation at bottom of page where the attached $[n]$ has been omitted from each $y_\alpha^\beta (\alpha = i, j, k;\ \beta = 0, 1, 2)$, and all notations have the same meaning as that in (1) and (2).

Theorem 3: The enhanced Back-Propagation Algorithm for FNN converges.

Proof: From Theorem 1 and the traditional gradient descent learning method of a neural network, it is easy to see that our enhanced Back-Propagation algorithm for FNN converges.

V. Applications to Classification Problems

In the following first three experiments, we target to increase the testing accuracy by tuning the local and global weights but with no intention of reducing the number of rules of the extracted rules. It is because the rules we extracted from these three experiments are not large enough for us to test the number of rules that could be reduced. In the fourth experiment we aim to test whether it is possible to reduce the number of propositions in the antecedent of a rule without sacrificing the testing accuracy of the extracted rules.

We choose some well known and widely used machine learning classification problems to verify our enhanced back-propagation learning algorithm found in Section IV for tuning and refining the local and global weights in conjunctive WFPRs.

A. Iris Data

The Iris data set [21] comprises 150 examples with four numerical attributes which are Sepal Length (SL), Sepal Width (SW), Petal Length (PL), and Petal Width (PW). The whole

$$\frac{\partial E_n}{\partial Gw_{jk}} = \begin{cases} (d_k - y_k) \cdot y_j^{(1)} \cdot \frac{\max\left\{y_k^{(2)}\right\} - y_k^{(2)}}{\left(\max\left\{y_k^{(2)}\right\}\right)^2} & if\ y_k^{(2)} \geq \vee_{p \neq k}\, y_p^{(2)} \\ (d_k - y_k) \cdot y_j^{(1)} \cdot \frac{\max\left\{y_k^{(2)}\right\} - \left(y_k^{(2)}\right)^2}{\left(\max\left\{y_k^{(2)}\right\}\right)^2} & if\ y_k^{(2)} < \vee_{p \neq k}\, y_p^{(2)} \end{cases}$$

$$\frac{\partial E_n}{\partial Lw_{ij}} = \frac{\partial}{\partial Lw_{ij}} \left(\frac{1}{2} \sum_{k=1}^{L_2} (d_k - y_k)^2 \right)$$

$$= \begin{cases} \sum_{k=1}^{L_2} \left[(d_k - y_k) \cdot y_i^{(0)} \cdot Gw_{jk} \cdot \frac{\max\left\{y_k^{(2)}\right\} - y_k^{(2)}}{\left(\max\left\{y_k^{(2)}\right\}\right)^2} \right] \\ \qquad if\ y_k^{(2)} \geq \bigvee_{p \neq k} y_p^{(2)}, Lw_{ij} \cdot y_i^{(0)} < \wedge_{p \neq i} \left(Lw_{pj} \cdot y_p^{(0)} \right) \\ \sum_{k=1}^{L_2} \left[(d_k - y_k) \cdot y_i^{(0)} \cdot Gw_{jk} \cdot \frac{\max\left\{y_k^{(2)}\right\} - \left(y_k^{(2)}\right)^2}{\left(\max\left\{y_k^{(2)}\right\}\right)^2} \right] \\ \qquad if\ y_k^{(2)} < \bigvee_{p \neq k} y_p^{(2)}, Lw_{ij} \cdot y_i^{(0)} < \wedge_{p \neq i} \left(Lw_{pj} \cdot y_p^{(0)} \right) \\ \sum_{k=1}^{L_2} \left[(d_k - y_k) \cdot y_i^{(0)} \cdot Gw_{jk} \cdot \left(\bigwedge_{p \neq i} Lw_{pj} \cdot y_p^{(0)} \right) \cdot \frac{\max\left\{y_k^{(2)}\right\} - y_k^{(2)}}{\left(\max\left\{y_k^{(2)}\right\}\right)^2} \right] \\ \qquad if\ y_k^{(2)} \geq \bigvee_{p \neq k} y_p^{(2)}, Lw_{ij} \cdot y_i^{(0)} \geq \bigwedge_{p \neq i} \left(Lw_{pj} \cdot y_p^{(0)} \right) \\ \sum_{k=1}^{L_2} \left[(d_k - y_k) \cdot y_i^{(0)} \cdot Gw_{jk} \cdot \left(\bigwedge_{p \neq i} Lw_{pj} \cdot y_p^{(0)} \right) \cdot \frac{\max\left\{y_k^{(2)}\right\} - \left(y_k^{(2)}\right)^2}{\left(\max\left\{y_k^{(2)}\right\}\right)^2} \right] \\ \qquad if\ y_k^{(2)} < \bigvee_{p \neq k} y_p^{(2)}, Lw_{ij} \cdot y_i^{(0)} \geq \bigwedge_{p \neq i} \left(Lw_{pj} \cdot y_p^{(0)} \right) \end{cases}$$

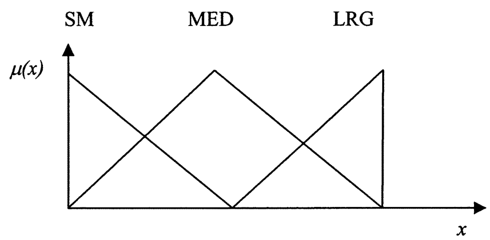

Fig. 3. Three linguistic terms.

data is categorized into three classes: Setosa, Versicolor and Virginica. The main task of our study is to generate a set of WFPRs from the 100 training examples so that the number of fuzzy rules generated is as small as possible and the testing accuracy of these WFPRs with the remaining 50 examples is as high as possible. We set $\alpha = \beta = 0.9$.

Fuzzifying initial Iris data: First of all we need to fuzzify numerical numbers into linguistic terms. Fuzzy clustering based on self-organized learning can be used to generate membership functions [22]. Y. Yuan [14] describes a simple algorithm for generating some type of membership functions.

For the Iris data, the number of linguistic terms for each of the four attributes can be assumed to be three. A simple version of this method is suggested in [23] where three membership functions for each input variable are given. The used abbreviations are as follows: SM—Small; MED—Medium; LRG—Large (Fig. 3).

Extracting initial FPRs and mapping them to a FNN: According to the linguistic terms SM, MED, and LRG, a set of FPRs can be generated by using one of machine learning techniques. We expect the number of generated rules to be as small as possible and the generated fuzzy rules to have as high predictive power as possible. For the Iris data, we quote four fuzzy rules extracted in [23]. These four rules will be regarded as four initial FPRs and are shown in the following rules R1 to R4.

R1: IF (PL is SM $[\mathrm{Lw}_1]$) and (PW is SM $[\mathrm{Lw}_2]$) THEN Setosa $[\mathrm{Gw}_1]$.
R2: IF (PL is MED $[\mathrm{Lw}_3]$) and (PW is MED $[\mathrm{Lw}_4]$) THEN Versicolor $[\mathrm{Gw}_2]$.
R3: IF (PL is LRG $[\mathrm{Lw}_5]$) and (PW is LRG $[\mathrm{Lw}_6]$) and (SL is MED $[\mathrm{Lw}_7]$) THEN Virginica $[\mathrm{Gw}_3]$.
R4: IF (PL is LRG $[\mathrm{Lw}_8]$) and (PW is LRG $[\mathrm{Lw}_9]$) and (SW is MED $[\mathrm{Lw}_{10}]$) THEN Virginica $[\mathrm{Gw}_4]$.

In rules R1 to R4, we assume that they are not assigned with any weights, i.e., local and global weights ($\mathrm{Lw}_\mathrm{i} = \mathrm{Gw}_\mathrm{j} = 1$ for all i and j) are assumed to be equal to 1. These rules are used to test 50 unseen examples of Iris data by means of the frequently used max-min fuzzy matching algorithm presented in Section II. The testing accuracy of the four generated rules is 86.7%. The FNN obtained by mapping the FPRs, R1 to R4, is shown in Fig. 4. In this FNN, we want to refine and tune the local and global weights.

Training the FNN and obtaining WFPRs: We can train the FNN as shown in Fig. 4 by using the back-propagation learning algorithm proposed in the previous section. Iris classification problem is a crisp one and the main objective of generating fuzzy rules is to classify each object correctly. After training the FNN as shown in Fig. 4 with 100 examples, a set of connection weights are obtained. Four WFPRs are shown in the following rules, R1 to R4.

R1: IF (PL is SM [0.70]) and (PW is SM [0.89]) THEN Setosa [0.60].
R2: IF (PL is MED [0.33]) and (PW is MED [0.84]) THEN Versicolor [0.01].
R3: IF (PL is LRG [0.93] and (PW is LRG [0.72] and (SL is MED [6.77]) THEN Virginica [0.43].
R4: IF (PL is LRG [0.93]) and (PW is LRG [0.72]) and (SW is MED [0.68]) THEN Virginica [0.43].

We use the above WFPRs, R1 to R4, to test the 50 unseen examples of Iris data. The result is that there are only two examples that cannot correctly be classified. That is, the testing accuracy of the four WFPRs has been increased to 96% from 86.7%. We have tried to reduce the number of extracted rule by deleting a rule with small or zero global weight. What we found is that although the global weight of R2 is very small after tuning, R2 cannot be deleted as it represents a distinct classification. If we deliberately delete this rule (R2), the classification errors will greatly increase.

B. Rice Data

The Rice data set comprises 105 examples with five attributes. The whole data sets are categorized into two classes. Furthermore, the values of the five attributes and the classified classes are assumed to be continuous. Fuzzy ID3 has been used to extract rules. The five attributes have been fuzzified into three triangular fuzzy sets. They are Large, Middle, and Small. The two classes have also been fuzzified into two triangular fuzzy sets. The 70 examples are used for training while the remaining 35 examples are used for testing. We set $\alpha = \beta = 0.5$. The extracted rules are listed as follows:

R1: IF Stickiness is Small (LW1), THEN Class 1 (Gw1).
R2: IF Stickiness is Middle (Lw2), THEN Class 1 (Gw2).
R3: IF Stickiness is Large (Lw3), THEN Class 2 (Gw3).

The testing accuracy of this data set before and after tuning is listed in Table I.

C. Wine Data

The Wine data set comprises 178 examples with 13 attributes. The whole data sets are categorized into three classes. Fuzzy ID3 has been applied to extract rules. Furthermore, we fuzzified the 13 attributes into triangular fuzzy sets. Each attribute has 3 triangular fuzzy sets. They are Large, Middle, and Small. There are 120 examples used for training while 58 examples are used for testing. We set $\alpha = \beta = 0.5$. The extracted rules are listed as follows.

R1: IF Flavanoids is Large (Lw1), THEN Class 1 (Gw1).
R2: IF Flavanoids is Middle (Lw2), THEN Class 2 (Gw2).
R3: IF Flavanoids is Small (Lw3), THEN Class 3 (Gw3).

The testing accuracy of this dataset before and after tuning is listed in Table I.

The results of the Iris, Rice, and Wine data are summarized in Table I.

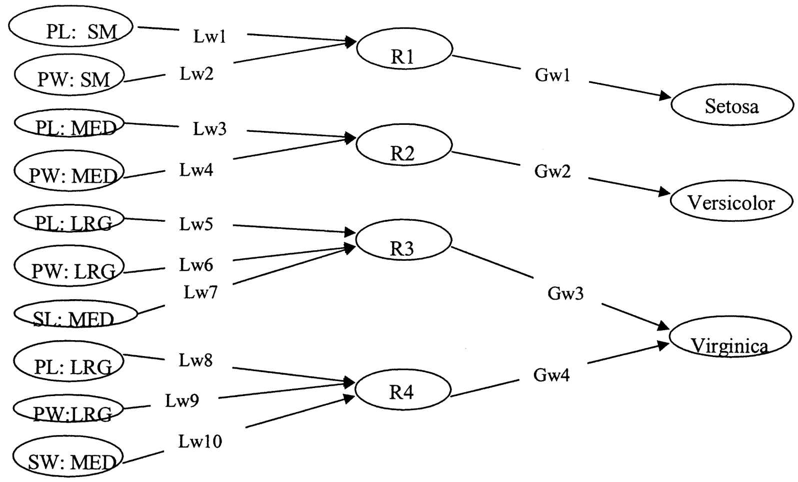

Fig. 4. FNN representing rules R1 to R4.

TABLE I
TESTING RESULTS OF RICE, WINE, IRIS AND ENJOY-SPORT DATA

Data sets	Accuracy of testing *(Lwi=Gwj=1)*	Accuracy of testing after refinement
Rice data	88.6%	93.3%
Wine data	82%	87.6%
Iris data	86.7%	96%
Enjoy-sport data	81.25%	93.75%

D. Enjoy-Sport Data

We use the small training set found in [14] to test whether it is possible to reduce the number of propositions in the antecedent of a rule without sacrificing the testing accuracy of the extracted rules. Four attributes with ten attribute values and three classification results are found in these 16 training samples. Six rules are extracted and listed as follows.

R1: IF Temperature is Hot (Lw1) and Outlook is Sunny (Lw2) THEN Swimming (Gw1).
R2: IF Temperature is Hot (Lw3) and Outlook is Cloudy (Lw4) THEN Swimming (Gw2).
R3: IF Temperature is Hot (Lw5) and Outlook is Rain (Lw6) THEN Weight-Lifting (Gw3).
R4: IF Temperature is Mild (Lw7) and Wind is Windy (Lw8) THEN Weight-Lifting (Gw4).
R5: IF Temperature is Mild (Lw9) and Wind is Not-Windy (Lw10) THEN Volleyball (Gw5).
R6: IF Temperature is Cool (Lw11) THEN Weight-Lifting (Gw6).

When $\mathrm{Lwi} = \mathrm{Gwj} = 1$ the testing accuracy is 81.25%. After tuning the local and global weights, the testing accuracy increases to 93.75%. The corresponding local and global weights are listed as follows:

$$\begin{aligned}&\mathrm{Lw1} = 10.86,\ \mathrm{Lw2} = 6.93,\ \mathrm{Lw3} = 0.67, \mathrm{Lw4} = 0.60,\\&\mathrm{Lw5} = 0.73,\ \mathrm{Lw6} = 0.56, \mathrm{Lw7} = 0.79, \mathrm{Lw8} = 0.53,\\&\mathrm{Lw9} = 0.21, \mathrm{Lw10} = 0.60,\ \mathrm{Lw11} = 1.00, \mathrm{Gw1} = 2.52,\\&\mathrm{Gw2} = 0.01,\ \mathrm{Gw3} = 0.16,\ \mathrm{Gw4} = 0.12, \mathrm{Gw5} = 0.42,\\&\mathrm{Gw6} = 1.0 \text{ and } \alpha = \beta = 0.8.\end{aligned}$$

In this Enjoy-sport data one may notice that R2 has a small global weight (0.01). When R2 is deleted from these extracted rule, the classification errors will increase, (i.e., the testing accuracy decreases from 93.75% to 75%).

From the Iris and Enjoy-sport data examples, one may notice that we could not just delete the entire rule with small or zero global weight from the extracted rule set without considering the number of rules generated and without considering whether the rule to be deleted represents a distinct classification or not. Many factors need to be considered in this issue.

On the other hand, from the Enjoy-sport data, when we delete the condition "If temperature is Mild" $(\mathrm{Lw9} = 0.21)$ from the R5, the testing accuracy is unchanged (93.75%) when the local and global weights are considered. These examples indicate that we could obtain more simple and compact WFPRs.

Since we have assigned the local and global weights to each proposition in the antecedent of a rule and the whole rule respectively to show the degree of importance, there are a lot of applications that are suitable for using our proposed algorithm. For example, in medical diagnosis system, there are many symptoms which are combined together and lead to a disease. For different diseases, the degree of importance of each symptom is different. So it is very useful to assign a weight to each symptom to show and capture the degree of importance.

We know that the learning and testing accuracy and the number of generated rules depend on both the learning algorithm and the selected linguistic terms. When the learning and testing accuracy cannot satisfy user requirements, one may improve the learning and testing accuracy by increasing the number of linguistic terms. However, increasing the number of linguistic terms will result in increasing the number of generated rules, i.e., increase the number of linguistic terms will affect the quality of the generated rules. In our study we propose another way of improving learning and testing accuracy by including local and global weights in FPRs while keeping the number of generated rule constant. This concept has been demonstrated by using the benchmark examples in this section. On the other, when a local weight is found to have small or zero value after refining or tuning by our proposed FNN, the proposition with small or zero local weight can be deleted which resulted in reducing the number of propositions in the antecedent of WFPRs. The advantages of our proposed method are that we could obtain a high learning and testing accuracy while keeping the number of rules as simple and compact as possible.

VI. Conclusion and Future Work

This paper proposes a method to generate and obtain a set of approximately optimal WFPRs by refining and tuning the local and global weights with a FNN. The aim of including local and global weights in FPRs and refinement of these weights is to improve the learning and testing accuracy without increasing the number of rules in the learning problem. When a local weight is found to have small or zero value after refinement, the corresponding proposition in the antecedent of a rule could be deleted. Thus a set of approximately optimal WFPRs could be extracted. We know that the simpler the form of the

$$\begin{aligned}\frac{\partial y_k^{(2)}}{\partial Lw_{ij}} &= \frac{\partial}{\partial Lw_{ij}}\left(\sum_{j=1}^{L_1} Gw_{jk}\cdot\left(\wedge_{p=1}^{L_0} Lw_{pj}\cdot y_p^{(0)}\right)\right)\\ &= \frac{\partial}{\partial Lw_{ij}}\left(Gw_{jk}\cdot\left(\wedge_{p=1}^{L_0} Lw_{pj}\cdot y_p^{(0)}\right)\right)\\ &= Gw_{jk}\cdot\frac{\partial}{\partial Lw_{ij}}\left(\wedge_{p=1}^{L_0} Lw_{pj}\cdot y_p^{(0)}\right)\\ &= Gw_{jk}\cdot\frac{\partial}{\partial Lw_{ij}}\left(\wedge\left(Lw_{ij}\cdot y_i^{(0)},\wedge_{p\neq i}\left(Lw_{pj}\cdot y_p^{(0)}\right)\right)\right)\\ &= Gw_{jk}\cdot\frac{\partial}{\partial\left(Lw_{ij}\cdot y_i^{(0)}\right)}\left(\wedge\left(Lw_{ij}\cdot y_i^{(0)},\wedge_{p\neq i}\left(Lw_{pj}\cdot y_p^{(0)}\right)\right)\right)\cdot\frac{\partial\left(Lw_{ij}\cdot y_i^{(0)}\right)}{\partial Lw_{ij}}\\ &= Gw_{jk}\cdot y_i^{(0)}\cdot\frac{\partial}{\partial\left(Lw_{ij}\cdot y_i^{(0)}\right)}\left(\wedge\left(Lw_{ij}\cdot y_i^{(0)},\wedge_{p\neq i} Lw_{pj}\cdot y_p^{(0)}\right)\right)\\ &= \begin{cases} y_i^{(0)}\cdot Gw_{jk} & if\ Lw_{ij}\cdot y_i^{(0)} < \wedge_{p\neq i}\left(Lw_{pj}\cdot y_p^{(0)}\right)\\ y_i^{(0)}\cdot Gw_{jk}\cdot\left(\wedge_{p\neq i}\left(Lw_{pj}\cdot y_p^{(0)}\right)\right) & otherwise\end{cases}\\ \frac{\partial(d_k-y_k)}{\partial y_k^{(2)}} &= \frac{\partial}{\partial y_k^{(2)}}\left(\frac{y_k^{(2)}}{\max\limits_{1\le k\le L_2}\left\{y_k^{(2)}\right\}}\right)\\ &= \begin{cases}\frac{\max\left\{y_k^{(2)}\right\}-y_k^{(2)}}{\left(\max\left\{y_k^{(2)}\right\}\right)^2} & if\ y_k^{(2)}\ge\vee_{p\neq k} y_p^{(2)}\\ \frac{\max\left\{y_k^{(2)}\right\}-\left(y_k^{(2)}\right)^2}{\left(\max\left\{y_k^{(2)}\right\}\right)^2} & if\ y_k^{(2)}<\vee_{p\neq k} y_p^{(2)}\end{cases}\end{aligned}$$

therefore

$$\frac{\partial E_n}{\partial Lw_{ij}} = \begin{cases} \sum_{k=1}^{L_2}\left[(d_k-y_k)\cdot y_i^{(0)}\cdot Gw_{jk}\cdot\frac{\max\left\{y_k^{(2)}\right\}-y_k^{(2)}}{\left(\max\left\{y_k^{(2)}\right\}\right)^2}\right] \\ \qquad if\ y_k^{(2)}\ge\bigvee_{p\neq k} y_p^{(2)}, Lw_{ij}\cdot y_i^{(0)} < \wedge_{p\neq i}\left(Lw_{pj}\cdot y_p^{(0)}\right)\\ \sum_{k=1}^{L_2}\left[(d_k-y_k)\cdot y_i^{(0)}\cdot Gw_{jk}\cdot\frac{\max\left\{y_k^{(2)}\right\}-\left(y_k^{(2)}\right)^2}{\left(\max\left\{y_k^{(2)}\right\}\right)^2}\right]\\ \qquad if\ y_k^{(2)}<\bigvee_{p\neq k} y_p^{(2)}, Lw_{ij}\cdot y_i^{(0)} < \wedge_{p\neq i}\left(Lw_{pj}\cdot y_p^{(0)}\right)\\ \sum_{k=1}^{L_2}\left[(d_k-y_k)\cdot y_i^{(0)}\cdot Gw_{jk}\cdot\left(\bigwedge_{p\neq i} Lw_{pj}\cdot y_p^{(0)}\right)\cdot\frac{\max\left\{y_k^{(2)}\right\}-y_k^{(2)}}{\left(\max\left\{y_k^{(2)}\right\}\right)^2}\right]\\ \qquad if\ y_k^{(2)}\ge\bigvee_{p\neq k} y_p^{(2)}, Lw_{ij}\cdot y_i^{(0)}\ge\bigwedge_{p\neq i}\left(Lw_{pj}\cdot y_p^{(0)}\right)\\ \sum_{k=1}^{L_2}\left[(d_k-y_k)\cdot y_i^{(0)}\cdot Gw_{jk}\cdot\left(\bigwedge_{p\neq i} Lw_{pj}\cdot y_p^{(0)}\right)\cdot\frac{\max\left\{y_k^{(2)}\right\}-\left(y_k^{(2)}\right)^2}{\left(\max\left\{y_k^{(2)}\right\}\right)^2}\right]\\ \qquad if\ y_k^{(2)}<\bigvee_{p\neq k} y_p^{(2)}, Lw_{ij}\cdot y_i^{(0)}\ge\bigwedge_{p\neq i}\left(Lw_{pj}\cdot y_p^{(0)}\right)\end{cases}.$$

extracted rules, the stronger the generalization capability of the extracted rules. As the computational complexity of finding an optimal set of fuzzy rules is generally NP-hard, the approach to find approximately optimal set of fuzzy rules becomes very important. Hence, the set of extracted fuzzy rules, with high learning and testing accuracy and with small number of the rules, should be considered to be optimal.

This paper also derives a kind of back-propagation algorithm for training the proposed FNN. From the experimental results, one may notice that the representative power of WFPRs is significantly enhanced and better than that of FPRs without weights because the testing accuracy of WFPRs is higher than that of FPRs without weights. Moreover, owing to the learning capability of the proposed FNN, the time required to consult with domain experts to extract a set of WFPRs will greatly be reduced. The proposed back-propagation and the weight refinement algorithm have been applied to some benchmark problems—Iris, Rice, Wine and Enjoy-sport classification problems and the results show that the accuracy of testing the extracted WFPRs after refinement increases. Furthermore, the synergy between WFPRs and a FNN offers a new insight into the construction of better fuzzy intelligent systems in the future.

Our future research work on rule refinement will be on determining the trade-off and strike a balance between the number of rules extracted and the testing accuracy of the extracted rules by using large databases. We will look into the problems of how we could achieve an optimal number of rules by deleting those rules with small or zero global weights. We will also develop an algorithm that will allow us to tune, refine and find optimal rules from a set of rough, crude and raw rules. The robustness and statistical property of this algorithm will also be studied.

APPENDIX
EVALUATION OF PARTIAL DERIVATIVES

We know $E=\sum_{n=1}^{N} E_n$ where

$$E_n=\frac{1}{2}\sum_{k=1}^{L_2}\left(d_k[n]-y_k[n]\right)^2. \tag{6}$$

The meaning of notations $y_i^{(0)}[n]$, $y_j^{(1)}[n]$, $d_k[n]$ and $y_k[n]$ has been explained in Section III. The forth-propagation of input vector is shown in (1) and (2) given in Section III. For simplicity, we omit the attached [n] in (1), (2), (3) and (6) for fixed n. By using the enhanced derivative (4) given in Section III, we evaluate the two partial derivatives in the (5) as follows:

$$\begin{aligned}\frac{\partial E_n}{\partial Gw_{jk}} &= \frac{\partial}{\partial Gw_{jk}}\left(\frac{1}{2}\sum_{q=1}^{L_2}\left(d_k[n]-y_q[n]\right)^2\right)\\ &=(d_k-y_k)\frac{\partial(d_k-y_k)}{\partial Gw_{jk}}\\ &=(d_k-y_k)\cdot y_j^{(1)}\cdot\frac{\partial(d_k-y_k)}{\partial y_k^{(2)}}\end{aligned}$$

$$=\begin{cases}(d_k-y_k)\cdot y_j^{(1)} \\ \cdot\frac{\max\left\{y_k^{(2)}\right\}-y_k^{(2)}}{\left(\max\left\{y_k^{(2)}\right\}\right)^2} & if\ y_k^{(2)}\geq\vee_{p\neq k}\, y_p^{(2)}\\ (d_k-y_k)\cdot y_j^{(1)} \\ \cdot\frac{\max\left\{y_k^{(2)}\right\}-\left(y_k^{(2)}\right)^2}{\left(\max\left\{y_k^{(2)}\right\}\right)^2} & if\ y_k^{(2)}<\vee_{p\neq k}\, y_p^{(2)}.\end{cases}$$

Similarly, we can evaluate another derivative $\partial E_n/\partial Lw_{ij}$ as follows:

$$\begin{aligned}\frac{\partial E_n}{\partial Lw_{ij}} &= \frac{\partial}{\partial Lw_{ij}}\left(\frac{1}{2}\sum_{k=1}^{L_2}(d_k-y_k)^2\right)\\ &=\frac{1}{2}\sum_{q=1}^{L_2}\left(\frac{\partial}{\partial Lw_{ij}}(d_k-y_k)^2\right)\\ &=\sum_{k=1}^{L_2}\cdot(d_k-y_k)\cdot\frac{\partial(d_k-y_k)}{\partial\, y_k^{(2)}}\cdot\frac{\partial y_k^{(2)}}{\partial Lw_{ij}}\end{aligned}$$

noting that (see equation at the bottom of the prevous page).

REFERENCES

[1] L. A. Zadeh, "Fuzzy sets," *Inform. Contr.*, vol. 8, pp. 338–353, 1965.
[2] ——, "Outline of a new approach to the analysis of complex systems and decision processes," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-3, pp. 28–44, 1973.
[3] ——, "The concept of a linguistic variable and its application to approximate reasoning-I,II,III," *Information Sciences*, vol. 8/9, pp. 199–249, 1975.
[4] C. C. Lee, "Fuzzy logic in control systems: fuzzy logic controller—part I," *IEEE Trans. Syst., Man, Cybern.*, vol. 20, no. 2, pp. 404–418, March 1990.
[5] C. T. Lin and C. S. G. Lee, "Neural network based fuzzy logic control and decision system," *IEEE Trans. Comput.*, vol. 40, pp. 1320–1336, 1991.
[6] D. G. Burkhardt and P. P. Bonissone, "Automated fuzzy knowledge base generation and tuning," in *Proc. 1992 IEEE Int. Conf. Fuzzy Systems*, San Diego, CA, 1992, pp. 179–188.
[7] T. P. Hong and J. B. Chen, "Finding relevant attributes and membership functions," *Fuzzy Sets Syst.*, vol. 103, no. 3, pp. 289–404, 1999.
[8] ——, "Processing individual fuzzy attributes for fuzzy rule induction," *Fuzzy Sets Syst.*, vol. 112, no. 1, pp. 127–140, 2000.
[9] H. Ishibuchi, K. Nozaki, N. Yamaoto, and H. Tanaka, "Selecting fuzzy if-then rules for classification problems using genetic algorithms," *IEEE Trans. Fuzzy Syst.*, vol. 3, no. 3, pp. 260–270, 1995.
[10] C. H. Kao and S. M. Chen, "A new method to generate fuzzy rules from training data containing noise for handling classification problems," in *Proc. 5th Conf. Artificial Intelligence and Applications*, Taipei, Taiwan, R.O.C., 2000, pp. 323–331.
[11] V. Ravi, P. J. Reddy, and H.-J. Zimmermann, "Fuzzy rule base generation for classification and its minimization via modified threshold accepting," *Fuzzy Sets Syst.*, vol. 120, no. 2, pp. 271–279, 2001.
[12] T. Sudkamp and R. J. Hammell II, "Interpolating, complete, and learning fuzzy rules," *IEEE Trans. Syst., Man, Cybern.*, vol. 24, no. 2, pp. 332–342, 1994.
[13] X. Wang and J. Hong, "Learning optimization in simplifying fuzzy rules," *Fuzzy Sets Syst.*, vol. 106, no. 3, pp. 349–356, 1999.
[14] Y. Yuan and M. J. Shaw, "Induction of fuzzy decision trees," *Fuzzy Sets Syst.*, vol. 69, pp. 125–139, 1995.
[15] D. S. Yeung and E. C. C. Tsang, "Weighted fuzzy production rules," *Fuzzy Sets Syst.*, vol. 88, pp. 299–313, 1997.
[16] ——, "A multilevel weighted fuzzy reasoning algorithm for expert systems," *IEEE Trans. Syst., Man, Cybern.*, vol. 28, no. 2, pp. 149–158, 1998.
[17] I. Hiraga, T. Furuhashi, and L. C. Jain, "Acquisition of fuzzy knowledge by NN and GA—a survey of the fusion and union methods proposed in Japan," in *Proc. IEEE 2nd Int. Conf. on Knowledge-Based Intelligence Electronic Systems (KES'98)*, 1998, pp. 69–78.
[18] I. Hiraga, Y. Uchikawa, and S. Nakayama, "An acquisition of operator's rules for collision avoidance using fuzzy neural networks," *IEEE Trans. Fuzzy Syst.*, vol. 3, pp. 280–287, Aug. 1995.

[19] J. R. Jang, "ANFIS: Adpative-Neural-Based Fuzzy Inference System," *IEEE Trans. Syst., Man, Cybern.*, vol. 23, pp. 665–685, May/June 1993.

[20] A. Blanco, M. Delgado, and I. Requena, "Identification of fuzzy relational equations by fuzzy neural networks," *Fuzzy Sets Syst.*, vol. 71, pp. 215–226, 1995.

[21] R. Fisher, "The use of multiple measurements in taxonomic problems," *Ann. Eugenics*, vol. 7, pp. 179–188, 1936.

[22] T. Kohonen, *Self-Organization and Associate Memory*. Berlin, Germany: Springer, 1988.

[23] N. K. Kasabov, "Learning fuzzy rules and approximate reasoning in fuzzy networks and hybrid systems," *Fuzzy Sets Syst.*, vol. 82, pp. 135–149, 1996.

[24] M. L. Minsky and S. A. Papert, *Perceptrons*. Cambridge, MA: MIT Press, 1988, pp. 161–170.

[25] Z. Y. He, *Fuzzy Mathematics and Its Applications*. Tianjing, China: Tianjing Science Technology, 1985, pp. 188–215.

Eric C. C. Tsang received the B.Sc. degree in computer studies from the City University of Hong Kong in 1990 and the Ph.D. degree in computing from the Hong Kong Polytechnic University in 1996.

He is an Assistant Professor of the Department of Computing of the Hong Kong Polytechnic University. His main research interests are in the area of fuzzy expert systems, fuzzy Petri nets, fuzzy neural networks, machine learning, and genetic algorithm.

Daniel S. Yeung (M'89–SM'99) received the Ph.D. degree in applied mathematics from Case Western Reserve University, Cleveland, OH, in 1974.

In the past, he has worked as an Assistant Professor of Mathematics and Computer Science at Rochester Institute of Technology, Rochester, NY, as a Research Scientist in the General Electric Corporate Research Center, and a System Integration Engineer at TRW, Inc. Currently, he is a Chair Professor of Computing, The Hong Kong Polytechnic University, Hong Kong. His current research interests include neural-network sensitivity analysis, expert neural-network hybrid systems, off-line handwritten Chinese character recognition, and fuzzy expert systems.

Dr. Yeung was the President of the IEEE Computer Chapter of Hong Kong for 1991 and 1992. He now chairs the technical committees on Cybernetics of the IEEE Systems, Man, and Cybernetics Society. He is also an Associate Editor for the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS and the IEEE TRANSACTIONS ON NEURAL NETWORKS.

John W. T. Lee (M'88) received the B.Sc. degree from the University of Hong Kong, Hong Kong, the M.B.A. degree from the Australian Graduate School of Management, New South Wales University, New South Wales, Australia, and the Ph.D. degree from the University of Sunderland, Sunderland, U.K., in 1972, 1983, and 2000 respectively.

He spent 12 years in the computer industry, working for a number of companies including NCR, Hong Kong and Esso, Australia, before joining The Hong Kong Polytechnic University in 1986. He is currently an Associate Professor in the Department of Computing, The Hong Kong Polytechnic University. His major research interests include fuzzy sets and systems, knowledge discovery and data mining, and applications of artificial intelligence.

Dr. Lee currently chairs the Technical Committee on Expert and Knowledge Based Systems in the IEEE Systems, Man, and Cybernetics Society.

D. M. Huang received the B.Sc. and M.Sc. degrees in mathematics from Hebei Teachers College and Hebei University, P.R.C, in 1984 and 1999, respectively.

She is currently an Associate Professor in the Mathematics Department of Science College, Hebei Agricultural University, P.R.C. Her research interests include fuzzy control theory, Bi-decision trees, fuzzy decision trees, and fuzzy neural networks.

X. Z. Wang received the B.Sc. and M.Sc. degrees in mathematics from the Hebei University, China, in 1983 and 1990, the Ph.D. degree in computer science and engineering from the Harbin Institute of Technology, China, in 1998.

He is currently a Full Professor and Dean of the Faculty of Mathematics and Computer Science, Hebei University. His research interests include fuzzy set theory, fuzzy measure and integral, fuzzy statistics, neural networks and fuzzy control, fuzzy expert systems, neuro-fuzzy systems, machine learning, and data mining.