Improving Learning Accuracy of Fuzzy Decision Trees by Hybrid Neural Networks

E. C. C. Tsang, X. Z. Wang, and D. S. Yeung

***Abstract*—Although the induction of fuzzy decision tree (FDT) has been a very popular learning methodology due to its advantage of *comprehensibility*, it is often criticized to result in poor learning accuracy. Thus, one fundamental problem is how to improve the learning accuracy while the comprehensibility is kept. This paper focuses on this problem and proposes using a hybrid neural network (HNN) to refine the FDT. This HNN, designed according to the generated FDT and trained by an algorithm derived in this paper, results in a FDT with parameters, called weighted FDT. The weighted FDT is equivalent to a set of fuzzy production rules with local weights (LWs) and global weights (GWs) introduced in our previous work. Moreover, the weighted FDT, in which the reasoning mechanism incorporates the trained LWs and GWs, significantly improves the FDTs learning accuracy while keeping the FDTs comprehensibility. The improvements are verified on several selected databases. Furthermore, a brief comparison of our method with two benchmark learning algorithms, namely, fuzzy ID3 and traditional backpropagation, is made. The synergy between FDT induction and HNN training offers new insight into the construction of hybrid intelligent systems with higher learning accuracy.**

***Index Terms*—Approximate reasoning, fuzzy decision trees, hybrid neural networks, knowledge acquisition, learning, local and global weights.**

I. Introduction

In the contemporary research on knowledge engineering, knowledge acquisition (learning), and knowledge interpretation (reasoning) are the main tasks. The two tasks are regarded as two tightly coupled phases in one system, as shown in Fig. 1 [14].

Many learning approaches to knowledge acquisition have been developed. One popular approach is called decision tree induction, which (together with a learning algorithm ID3 and a simple reasoning mechanism) was initially developed by Quinlan [21]. Due to many advantages of this approach, the decision tree induction has been investigated intensively during the recent decade.

Most knowledge associated with human's thinking and perception has imprecision and uncertainty. In addition to the experience of domain experts, learning from examples with fuzzy representation is considered as an essential way of acquiring such knowledge. In the recent decade, for the purpose of acquiring imprecise and uncertain knowledge, the decision-tree induction has been improved such that it is suitable for the fuzzy case. That is fuzzy decision tree (FDT) induction. Investigations to FDT induction could be found in [1], [3], [8], [10]–[13], [17], [25], [28], [29], [32], [33], and [40], which are summarized as follows.

a) Revising algorithms for generating FDTs: Mainly, this kind of revision is reflected within the process of FDT generation and influences the shape and size of FDT. For example, Cios and Sztandera [8] proposed using fuzzy entropy in continuous ID3 algorithm and achieved a remarkable decrease in convergence time. Yuan and Shaw [40] introduced a new heuristic algorithm for generating FDT, which is based on the minimal nonspecificity and dose not use the entropy. Ichihashi *et al.* [11] proposed a method of inducing FDTs in which the interview with domain specialists is considered to be necessary for knowledge acquisition in expert systems. Hayashi [10] proposed an algorithm for generating FDT that incorporates the adjustment mechanism of AND/OR operators such that the FDT has more flexible representation. Wang *et al.* [33] investigated the optimization of FDT and gave a branch-merging algorithm for FDT generation.

b) Improving reasoning mechanism of FDTs: This kind of improvement can be independent of FDT generation, usually including pruning/grafting and fuzzification of nodes of FDT. For example, Maher and Clair [17] gave an UR-ID3 algorithm that incorporates uncertain reasoning technique and improves the robustness of FDT. Sison and Chong [28] suggested using the pruning of FDT to eliminate the irrelevant attributes thus to simplify the rule base. Jeng *et al.* [13] gave a fuzzy inductive learning method for automatic knowledge acquisition by means of fuzzifying the crisp decision tree and reported a remarkable improvement of predictive accuracy. Janikow [12] gave a detailed investigation for FDT, with the purpose of combining symbolic decision trees with approximate reasoning offered by fuzzy representation.

c) Applications to different domains: A lot of applications of FDT can be found in the existing literatures. For example, one can find the applications of FDT to prediction of heater outlet temperature [29], to diagnosis systems [32], to motion planning [25], to ham quality control [1], and to power system security assessment [3].

In FDT induction, comprehensibility has been universally accepted as one main advantage. The comprehensibility refers to that the concepts formed by FDT are understood more easily than that formed by other techniques such as neural networks. Unfortunately, the FDT induction is often reported to have poor learning accuracy (e.g., [24], [33], [40]). Here, the learning accuracy contains two aspects. One is the training accuracy (the correct rate of testing the training set), while the other is the testing accuracy (the correct rate of predicting classes for novel

Manuscript received January 21, 2000; revised May 1, 2000. This work was supported by Research Fellowship Grant G-YY12 from Hong Kong Polytechnic University.

The authors are with the Department of Computing, Hong Kong Polytechnic University, Kowloon, Hong Kong (e-mail: csetsang@comp.polyu.edu.hk; csxzwang@comp.polyu.edu.hk; csdaniel@comp.polyu.edu.hk).

Publisher Item Identifier S 1063-6706(00)08460-5.

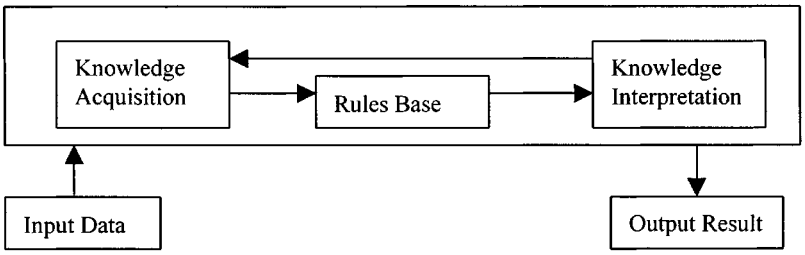

Fig. 1. The knowledge acquisition and knowledge interpretation in a knowledge-based system.

examples). A better learning accuracy is always expected in knowledge acquisition. In the crisp case of learning from examples without noisy data, the training accuracy can attain 100%, but in fuzzy case, it usually fails to achieve that. When the training time is acceptable, one always expects a higher learning accuracy. From the investigations of FDT mentioned above, one may find that the improvement of reasoning mechanism of FDT given by fuzzifying nodes [13] or by fuzzifying branches [17] of the crisp decision tree can raise the learning accuracy. This is an important kind of approach to increase learning accuracy. However, this increase of learning accuracy is accompanied with the decrease of the comprehensibility of FDT. Another approach to improve the learning accuracy is to construct an oblique decision tree in which a linear combination of the original attributes is considered to be a new attribute [23] (in which only the crisp case is considered). Due to the unclear meaning of new attributes, to some extent this approach also lowers the comprehensibility of decision trees. Thus, one fundamental problem is how to improve the learning accuracy while the comprehensibility is kept.

Focusing on this problem, the present paper proposes using a hybrid neural network (HNN) to refine the FDT. It is well known that neural networks together with their learning algorithms usually have a significant merit, namely, their high prediction accuracy due to the highly nonlinear decision boundaries they form. By means of incorporating this merit into the FDT induction, we expect that the learning accuracy of FDTs can be significantly improved while keeping the comprehensibility.

The remains of this paper are organized as follows. Section II gives a brief review for the FDT induction. Section III discusses the weighted FDT induction in which the weighted FDT is equivalent to a set of fuzzy production rules with local weights (LWs) and global weights (GWs) introduced in our previous work and the weighted FDTs reasoning mechanism improves the learning accuracy while keeping the comprehensibility. Section IV investigates the training algorithm for the proposed HNN and Section V verifies the advantages of the weighted FDT on several databases and makes a brief comparison with two benchmark learning algorithms. They are the initial fuzzy ID3 algorithm and the traditional back-propagation algorithm. Section VI offers our conclusions and the last section states the future work.

II. Fuzzy Decision Tree Induction

In this paper, the FDT induction is considered to be an algorithm for generating FDT together with a reasoning mechanism.

A. Fuzzy Representation of Training Example

Fuzzy representation of training examples is introduced in order to handle the increasing uncertainty of learning process. The crisp representation can be regarded as a special case of fuzzy representation. Fuzzy representation of examples may also be expressed in the form of a set of $\langle \text{ATTRIBUTE} = \text{Attribute-value} \rangle$, but the attribute-value is considered a fuzzy set. According to the type of fuzzy sets, the fuzzy representation of Attribute-value can be categorized into six cases, namely, nominal valued, real valued, interval valued, fuzzy number valued, fuzzy vector valued, and mixed-valued attributes.

In this paper, we only deal with the learning from fuzzy vector valued examples, in short, learning from fuzzy examples. Learning from fuzzy examples is a type of supervised inductive learning and the classification of each training example is considered to be known. Traditionally, the classification is crisp (a positive class and a negative class). In our study on learning from fuzzy examples, the classification of training examples is supposed to have fuzzy representations; that is, the classification of each example is no longer a definite class, but a fuzzy vector defined on the cluster space. For instance, when there are only two clusters P and N, the fuzzy representation of classification in this case may be $(0.4/P, 0.6/N)$.

In the following sections, we will make use of a small training set to illustrate our learning process. The small training set (adopted from [40]) is shown in Table I where both attribute-values and classification of each example are represented as the form of fuzzy vector. In this problem of learning from fuzzy examples, there are four attributes namely Outlook, Temperature, Humidity and Wind. Their universes of discourse are

$$U(\text{Outlook}) = \{\text{Sunny}, \text{Cloudy}, \text{Rain}\}$$
$$U(\text{Temperature}) = \{\text{Hot}, \text{Mild}, \text{Cool}\}$$
$$U(\text{Humidity}) = \{\text{Humid}, \text{Normal}\}$$

and

$$U(\text{Wind}) = \{\text{Windy}, \text{Not\_windy}\}$$

respectively. Each attribute value is a fuzzy vector defined on the universe of discourse of the attribute. For instance, the value of the first attribute of the first example in Table I is of the form $(0.9/\text{Sunny} + 0.1/\text{Cloudy} + 0.0/\text{Rain})$, in short, $(0.9, 0.1, 0.0)$. The universe of discourse of the classification is the sport to

TABLE I
A SMALL TRAINING SET WITH FUZZY REPRESENTATION

No.	Outlook			Temperature			Humidity		Wind		Sports Plan		
	Sunny	Cloudy	Rain	Hot	Mild	Cool	Humid	Normal	Windy	Not_	V	S	W
1	0.9	0.1	0.0	1.0	0.0	0.0	0.8	0.2	0.4	0.6	0.0	0.8	0.2
2	0.8	0.2	0.0	0.6	0.4	0.0	0.0	1.0	0.0	1.0	1.0	0.7	0.0
3	0.0	0.7	0.3	0.8	0.2	0.0	0.1	0.9	0.2	0.8	0.3	0.6	0.1
4	0.2	0.7	0.1	0.3	0.7	0.0	0.2	0.8	0.3	0.7	0.9	0.1	0.0
5	0.0	0.1	0.9	0.7	0.3	0.0	0.5	0.5	0.5	0.5	0.0	0.0	1.0
6	0.0	0.7	0.3	0.0	0.3	0.7	0.7	0.3	0.4	0.6	0.2	0.0	0.8
7	0.0	0.3	0.7	0.0	0.0	1.0	0.0	1.0	0.1	0.9	0.0	0.0	1.0
8	0.0	1.0	0.0	0.0	0.2	0.8	0.2	0.8	0.0	1.0	0.7	0.0	0.3
9	1.0	0.0	0.0	1.0	0.0	0.0	0.6	0.4	0.7	0.3	0.2	0.8	0.0
10	0.9	0.1	0.0	0.0	0.3	0.7	0.0	1.0	0.9	0.1	0.0	0.3	0.7
11	0.7	0.3	0.0	1.0	0.0	0.0	1.0	0.0	0.2	0.8	0.4	0.7	0.0
12	0.2	0.6	0.2	0.0	1.0	0.0	0.3	0.7	0.3	0.7	0.7	0.2	0.1
13	0.9	0.1	0.0	0.2	0.8	0.0	0.1	0.9	1.0	0.0	0.0	0.0	1.0
14	0.0	0.9	0.1	0.0	0.9	0.1	0.1	0.9	0.7	0.3	0.0	0.0	1.0
15	0.0	0.0	1.0	0.0	0.0	1.0	1.0	0.0	0.8	0.2	0.0	0.0	1.0
16	1.0	0.0	0.0	0.5	0.5	0.0	0.0	1.0	0.0	1.0	0.8	0.6	0.0

play on the weekend such as {volleyball, swimming, weightlifting}, in short, $\{V, S, W\}$. The classification of each example is a fuzzy vector defined on this universe of discourse. The fuzzy representation of the classification of the first example in Table I, for instance, is of the form $(0.0/V + 0.8/S + 0.2/W)$, in short, $(0.0, 0.8, 0.2)$.

Elements of the universe of discourse of each attribute or classification are regarded as nominal symbols in the learning from crisp examples, but they are regarded as fuzzy vectors defined on the example-label space in the learning from fuzzy examples. These fuzzy vectors refer usually to linguistic terms. For instance, the linguistic term "Sunny" (an element of the universe of discourse of the attribute Outlook) is regarded as the fuzzy vector defined on $\{1, 2, \ldots, 16\}$:

$$\begin{aligned}\text{Sunny} &= 0.9/1 + 0.8/2 + 0.0/3 + \cdots + 1.0/16 \\ &= (0.9, 0.8, 0.0, \ldots, 1.0)\end{aligned}$$

which corresponds to the first column of Table I.

B. Fuzzy Decision Tree and Heuristic Algorithm

A FDT is a generalization of the crisp decision tree (CDT). The generalization is mainly reflected in the following several aspects.

1) A FDT is a fuzzy partition of X while a CDT is a crisp partition of X, where X is the universe of discourse of all training examples.
2) Each node of the FDT is a fuzzy set defined on X while each node of the CDT is a crisp set of X.
3) The intersection of nodes located on the same layer is nonempty in FDT but is empty in CDT.
4) In the fuzzy case, if N is a nonleaf node and $\{N_i\}$ is the set of all son-nodes of N, then $\cup_i N_i \subset N$. In the crisp case, the equality $\cup_i N_i = N$ holds well.
5) Each attribute-value is regarded as a fuzzy set in fuzzy case but as a crisp set in crisp case.
6) Each path from the root to a leaf can be converted to a fuzzy rule with some degree of truth in fuzzy case, but a crisp production rule in crisp case.
7) An example remaining to be classified matches only one path in the CDT, but may match several paths in the FDT.

Since the generation of optimal fuzzy decision tree has been proved to be NP-hard [33], the investigation to heuristic algorithm seems to be very important. A good heuristic algorithm for generating FDTs should strike a balance among the learning accuracy, training hours, the simplicity of generated fuzzy rules, and the capability of tolerating noise. For a given heuristic, the general learning algorithm for generating fuzzy decision trees can be described as follows.

- Consider the whole training set, i.e., $(1, 1, \ldots, 1)$ which is regarded as the first candidate node. *While* there exist candidate nodes
- *do* select one using a given search strategy; if the selected one is not a leaf, then generate its son nodes by selecting the expanded attribute using the given heuristic. These son nodes are regarded as new candidate nodes.

Before training, the α-cut is usually used for the purpose of reducing the fuzziness of training examples. The α cut of a fuzzy set A is defined as

$$A_\alpha(x) = \begin{cases} A(x) & A(x) > \alpha \\ 0 & A(x) \le \alpha \end{cases}.$$

Increasing the value of α can reduce the fuzziness of initial data but a too large α may result in an empty set [40]. Usually α is in the range of $(0, 0.5]$.

In addition to the selection of expanded attributes, the determination of the leaf node is another important issue for decision tree generation. Generally speaking, two key points of an algorithm for generating fuzzy decision trees are: 1) a heuristic for selecting expanded attributes and 2) a standard for judging leaf nodes.

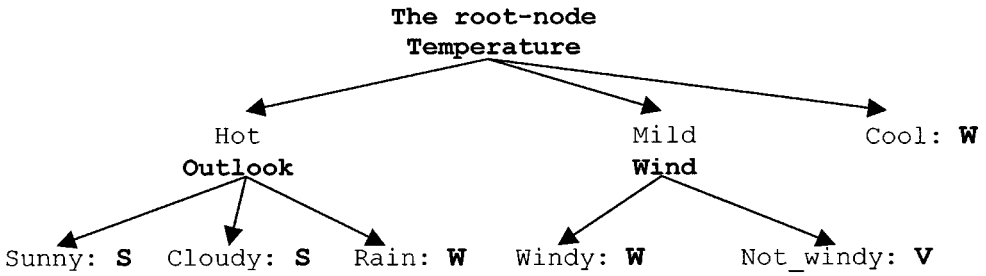

Fig. 2. Fuzzy decision tree for Table I.

C. Fuzzy ID3 Heuristic Algorithm

One popular and powerful heuristic algorithm for generating crisp decision trees is called ID3. The earlier version of ID3, which is based on minimum information entropy to select expanded attributes, was proposed by Quinlan [21]. Subsequently, the fuzzy version of ID3 based on minimum fuzzy entropy was suggested by several authors [8], [29], [32], [34], [35]. Suppose that training examples have a fuzzy representation like Table I, we give (as follows) a generic version of fuzzy ID3, which is founded on the assumption that each attribute-value (linguistic term) is a fuzzy set defined on the universe of discourse on all training examples.

Suppose that the clusters of the learning problem are C_l $(l = 1, 2, \ldots, L)$.

Definition 1: Let N be an arbitrary node of a given fuzzy decision tree. The relative frequency of the node N with respect to the cluster C_l $(1 \leq l \leq L)$ is defined as $f_l(N) = M(N \cap C_l)/M(N)$ where $M(A)$ denotes the sigma count (the sum of all membership degrees) of a fuzzy set A.

In some articles ([22], [40]), $f_l(N)$ is regarded as the subsethood of N in C_l and is interpreted as the degree of truth for the fuzzy rule *IF* N *Then* C_l.

Definition 2: Let N be an arbitrary node of a given fuzzy decision tree. The fuzzy entropy of the node N with respect to the clusters C_l $(l = 1, 2, \ldots, L)$ is defined as $\mathrm{FE}(N) = \sum_{l=1}^{L} f_l(N) \cdot \ln(f_l(N))$.

Definition 3: Consider a nonleaf node S and n attributes $A^{(1)}, \ldots, A^{(n)}$ to be selected. For each k $(1 \leq k \leq n)$, the attribute $A^{(k)}$ takes m_k values of the fuzzy subsets, $A_1^{(k)}, \ldots, A_{m_k}^{(k)}$. Hence, for the attribute $A^{(k)}, m_k$ son nodes of $S, S \cap A_1^{(k)}, \ldots, S \cap A_{m_k}^{(k)}$ will result. Then, the information gain of the attribute $A^{(k)}$ at the node S is defined as

$$\mathrm{Gain}\left(A^{(k)}, S\right) = \mathrm{FE}(S) - \sum_{i=1}^{m_k} \left(\frac{M\left(S \cap A_i^{(k)}\right)}{\sum_{j=1}^{m_k} M\left(S \cap A_j^{(k)}\right)} \right) \times \mathrm{FE}\left(S \cap A_i^{(k)}\right).$$

The fuzzy ID3 heuristic algorithm can now be described as follows. Consider the whole training set, i.e., $(1, 1, \ldots, 1)$ as the first candidate node (the root). Given a leaf standard of frequency β, *while* there exist candidate nodes *do*.

Step 1) Randomly choose one candidate node S with n attributes $A^{(1)}, \ldots, A^{(n)}$ to be selected.

Step 2) If the frequency of some cluster exceeds β at the node S, then regard the node S as a leaf and go to Step 6).

Step 3) Compute $\mathrm{Gain}(A^k, S)$ $(k = 1, 2, \ldots, n)$.

Step 4) Select k_0 such that $\mathrm{Gain}_{k_0}(S) = \mathrm{Max}_{1 \leq k \leq n} \mathrm{Gain}(A^{(k)}, S)$.

Step 5) If $\mathrm{Gain}_{k_0}(S) \leq 0$ then regard the node S as a leaf. If $\mathrm{Gain}_{k_0}(S) > 0$ then select the k_0th attribute as the expanded attribute, generates the son nodes of S and regard these son nodes as new candidate nodes.

Step 6) Label node S, which is no longer a candidate node.

The key points of fuzzy ID3 heuristic are that: 1) the nonpositive gain is regarded as leaf standard and 2) the positive maximum gain is the expanded attribute standard.

D. The Performance of Fuzzy ID3 on a Small Training Set

Consider the small training set with fuzzy representation indicated in Table I. After selecting $\alpha = 0.4$ to cut the training set, one can directly compute $M(\mathtt{V}) = 4.5, M(\mathtt{S}) = 4.2, M(\mathtt{W}) = 6.5$ and $\mathrm{sum} = 15.2$. It results in $\mathrm{FN}(\mathrm{root}) = -(4.5/15.2)\,\mathrm{Log}(4.5/15.2) - (4.2/15.2)\,\mathrm{Log}(4.2/15.2) - (6.5/15.2)\,Log(6.5/15.2) = 1.07$where root denotes the root node. For the first attribute `Outlook`, one can similarly obtain $\mathrm{FN}(\mathrm{root} \cap \mathrm{sunny}) = 1.03, M(\mathrm{root} \cap \mathrm{sunny}) = 6.2$; $\mathrm{FN}(\mathrm{root} \cap \mathrm{cloudy}) = 0.99, M(\mathrm{root} \cap \mathrm{cloudy}) = 4.6$; and $\mathrm{FN}(\mathrm{root} \cap \mathrm{rain}) = 0.00, M(\mathrm{root} \cap \mathrm{rain}) = 2.6$. The averaged fuzzy entropy is $(6.2/13.4)^{\star}1.03 + (4.6/13.4)^{\star}0.99 + (2.6/13.4)^{\star}0.00 = 0.81$. Hence, the information gain of the first attribute $\mathrm{Gain}(\mathtt{Outlook}, \mathrm{root}) = 1.07 - 0.81 = 0.26$.

Similarly, one can compute the information gains of other three attributes: $\mathrm{Gain}(\mathtt{Temperature}, \mathrm{root}) = 0.28$; $\mathrm{Gain}(\mathtt{Humidity}, \mathrm{root}) = 0.05$; $\mathrm{Gain}(\mathtt{Wind}, \mathrm{root}) = 0.22$. Therefore, the expanded attribute at the root should be `Temperature`.

The childnodes of the root can be treated similarly and it finally generates a FDT which is shown in Fig. 2, where the leaf standard of frequency is set to be 0.8.

E. Reasoning Mechanism of FDT

In addition to the heuristic algorithm for generating FDT, another aspect associated with FDT induction is its reasoning mechanism. After generating the FDT, we need a mechanism to predict the classification of novel examples or to test the classification of training examples.

For a generated FDT, each connection from root to leaf is usually called a path. It is clear each path corresponds to a leaf. The connection between two adjacent nodes (father node and

son node) in one path is called a segment of the path. All segments in one path are considered to have equal importance to the cluster labeled at the leaf.

Suppose that the generated FDT contains m leaf nodes, which correspond to m paths denoted by $\text{Tree} = \{\text{Path}_i, i = 1, 2, \ldots, m\}$. Each Path_i consists of several segments denoted by $\text{Path}_i = \{\text{Seg}_1^{(i)}, \text{Seg}_2^{(i)}, \ldots, \text{Seg}_{n_i}^{(i)}$. Each segment of Path_i corresponds to an attribute-value that is regarded as a fuzzy set defined on X (the universe of discourse of all training examples). Let e be an example remaining to be classified. For each i $(1 \leq i \leq m)$, its attribute-values corresponding to Path_i are supposed to be $\{C_1^{(i)}, C_2^{(i)}, \ldots, C_{n_i}^{(i)}\}$. A mechanism commonly used for determining the cluster of the example e is described as follows.

1) For each i $(1 \leq i \leq m)$ and j $(1 \leq j \leq n_i)$, compute $\text{SM}_j^{(i)}$, which denotes the similarity degree between $\text{Seg}_j^{(i)}$ and $C_j^{(i)}$ when $C_j^{(i)}$ is a fuzzy set and denotes the membership degree of $C_j^{(i)}$ belonging to $\text{Seg}_j^{(i)}$ when $C_j^{(i)}$ is a real number.
2) Compute the overall similarity $\text{SM}^{(i)}$ by $\text{SM}^{(i)} = \text{Min}_{1 \leq j \leq n_i}(\text{SM}_j^{(i)})$.
3) Compute x_k $(k = 1, 2, \ldots, K)$ by $x_k = \text{Max}_i\{\text{SM}^{(i)} \,|\, B^{(i)} = \text{CLASS}_k\}$ where K is the number of clusters and $B^{(i)}$ is the cluster labeled at the corresponding leaf.
4) The inferred result is regarded as a fuzzy vector $(x_1, x_2, \ldots, x_k)$ where x_k is the value which indicates to what degree the example e belongs to CLASS_k $(k = 1, 2, \ldots, K)$. When the crisp inferred result is needed, one can take the consequent CLASS with maximum x_k $(1 \leq k \leq K)$.

It is worth noting that the operations min and max used in 2) and 3) can be extended, respectively, to T-norm and S-norm. In addition, if there is more than one maximum x_k in 4), we need another way of defuzzification to give a crisp classification for the example e.

Usually, the reasoning by FDT can be converted into that by a set of fuzzy production rules (FPRs). The purpose of establishing the reasoning mechanism of FDT *in this way* is to compare with the weighted FDT introduced in the following section.

III. Weighted FDT and Weighted FPR

A. Weighted FDT

A weighted FDT refers to a FDT in which several parameters are attached to each leaf node. These parameters attached to the leaf node include the following several aspects.

1) *The degree of truth of the classification corresponding to the leaf node. This parameter is usually called certainty factor (CF).* The CF is an important parameter of the leaf node, which has been given in many methods of generating FDT. A leaf node of FDT can be usually converted to a FPR and then the CF of the leaf is considered to be CF of the FPR. It can be computed in several ways, one popular method is by using the degree of subset hood [22]. That is, if the leaf node corresponds to an FPR taking the form "IF A THEN B" where B is the conclusion fuzzy set and A may be the intersection of several propositional fuzzy sets, then the CF is computed by $M(A \cap B)/M(A)$ where $M(\cdot)$ denotes the cardinality of a fuzzy set. This method of computing CF can be modified by incorporating the concept of LW or by replacing the minimum and summation with S-norm and T-norm, respectively.
2) *The degree of importance of each segment in one path contributing to the classification of the leaf node. These parameters are usually called LWs.* In a nonweighted FDT, all segments in one path from the root to a leaf are considered to have equal importance. In a weighted FDT discussed here, an LW is assigned to a segment in one path to indicate the relative degree of importance of the segment contributing to its leaf node. Since a leaf node of FDT can be usually converted to an FPR and a segment in one path corresponds to a proposition in a FPR, the concept of LW of FDT also specifies that diverse propositions in one FPR should have different importance contributing to its consequent. This LW plays an important role in many real world problems. For example, in medical diagnosis systems it is common to observe that a particular symptom combined with other symptoms may lead to possible diseases. It is necessary to assign an LW to each symptom in order to show the relative degree (weight) of each symptom leading to the consequent (a disease). Many researchers have used this LW concept when employing FPRs to capture medical diagnostic knowledge ([7], [31]).
3) *The degree of importance of the leaf node contributing to the conclusion of classification. This parameter is usually called GW.* In a weighted FDT, since there is generally more than one leaf node having the same classification, a GW is assigned to the path corresponding to the leaf node in this paper to indicate the different importance of different paths (leaf nodes) contributing to the same consequent. We have specified in [37] and [38] that the GW is a concept distinct from the LW.

The most significant merit of decision tree induction should be its comprehensibility which is mainly reflected in the fact that each leaf node of a decision tree can be converted into a production rule. The conversion is exact in crisp case without noise but is usually inexact in the fuzzy case with several parameters. That is, in order to obtain a clear decision, only the main information remains and the other information is ignored. For example, consider a FDTs leaf node, which has a parameter vector $(a_1, a_2, a_3) = (0.3, 0.4, 0.3)$ where a_i, denotes the possibility with that the leaf node is labeled "ith class". When it needs to be converted into a FPR, the information of the first class and the third class will be ignored. If the ignored information is used in the reasoning mechanism of FDT, then the reasoning mechanism of extracted FPRs will be distinct from that of the FDT. This indicates that the FDT is not equivalent to the extracted FPRs. If the criterion of comprehensibility is whether the FDT (including its reasoning mechanism) can be equivalently converted into a set of FPRs, then different types of parameter will affect the comprehensibility of FDT to a certain extent. For example, to some extent, the comprehensibility of the FDT is lowered due to the aforesaid parameter vector

(a_1, a_2, a_3). Although this kind of parameter lowers the comprehensibility of FDT, it can raise the prediction accuracy of FDT to a great extent ([13], [17]).

The increasing complexity of today's knowledge-based system often requires many parameters for knowledge representation. Indeed, some parameters introduced in the FDT can enhance its knowledge representation power and can improve its learning accuracy but, simultaneously, they lower the comprehensibility of the FDT. One problem is what parameters should be introduced for FDT so as not to lower its comprehensibility.

In this paper, we introduce three kinds of parameters, namely, CF, LW, and GW. These parameters have the clear meaning and the weighted FDT with these parameters can be equivalently converted into a set of weighted FPRs introduced in our previous work (see Section III-B or [37]–[39]), therefore, they keep the comprehensibility of FDT. Furthermore, it is found that refining these parameters can improve the learning accuracy of FDT considerably.

We have suggested a heuristic algorithm for a weighted FDT, where the three kinds of parameters can be roughly given by the heuristic. Here we need not to specify this heuristic algorithm with complicated equations since these parameters will be obtained by refining in Section IV. Moreover, the FDT generated in Section II-C can also be considered to have these parameters which are equal to one.

B. Weighted FPR

According to [36], propositional statements are the fundamental building blocks of a rule-based system. It is usually represented in the form of

The (attribute) of (an object) is (attribute-value)

e.g., the outlook of last weekend is sunny.

In our study, the object of the propositional statement is omitted, the attribute is regarded as a variable, and the attribute value is a fuzzy vector defined on a universe of discourse.

Unlike the crisp case, a fuzzy production rule can have linguistic terms like "hot" or "high" in the antecedent and the consequent part. In [37], a generic form of fuzzy production rules has been suggested where threshold value, certainty factor, and LW are assigned to each proposition while GW and certainty factor are assigned to the entire rule. This paper discusses a type of fuzzy production rules in which the LWs and the GWs are emphasized and the effect of other parameters is not considered. For instance, a conjunctive weighted fuzzy production rule in [37] takes the form of

$$\begin{aligned}\mathbf{R}&: \text{If } V_1 \text{ is } A_1 \mathbf{AND}\ V_2 \text{ is } A_2 \ldots \mathbf{AND}\ V_n \text{ is } A_n\\ &\quad \mathbf{THEN}\ U \text{ is } B, \mathrm{LW}_1, \mathrm{LW}_2, \ldots, \mathrm{LW}_n, \mathrm{GW}(R)\\ &\quad \text{Fact 1: } V_1 \text{ is } A_1^*, \quad \text{Fact 2: } V_2 \text{ is } A_2^*, \ldots,\\ &\quad \text{Fact } n\text{: } V_n \text{ is } A_n^*\\ &\quad \text{Conclusion: } U \text{ is } B^* \end{aligned} \tag{1}$$

where $V_1, V_2, \ldots, V_n$ and U are attributes and $A_1, A_2, \ldots, A_n$ and B are the fuzzy values of these attributes. LW_i $(1 \le i \le n)$ is the LW of the proposition "V_i is A_i" and each LW_i is nonnegative. $\mathrm{GW}(R)$ denotes the GW assigned to the rule R $(\mathrm{GW}(R) \ge 0)$.

This type of FPRs can be extracted from the weighted FDT proposed in Section III-A. The weighted FDT can be equivalently converted into a set of weighted FPRs described above. The conversion is straightforward. Paths (leaf nodes), segments of path and three kinds of parameters (CF, LW, and GW) of leaf node in a weighted FDT correspond to FPRs, propositions in the antecedent and three kinds of parameters of FPRs. The three kinds of parameters of these FPRs have the meaning similar to that of the FDT. For example, the LW is to indicate the relative degree of importance of a proposition contributing to its consequent and the GWs indicate the degrees of importance of each rule contributing to the conclusion.

In the following discussions on the three kinds of parameters, LWs and GWs will be refined by a HNN in Section IV, but the CF is not considered adjustable. It depends on LWs and its computational equation is as follows:

$$\mathrm{CF}(R) = \frac{\sum_j \left(\bigwedge_i \left(\mathrm{LW}_i \cdot \mu_{A_i}(j)\right) \bigwedge \mu_B(j)\right)}{\sum_j \mu_B(j)}$$

in which μ denotes the membership function of a fuzzy set and other notations have the same meaning as that in (1).

C. Reasoning Mechanism of Weighted FPR

Generally speaking, the reasoning mechanism of FDT cannot be exactly converted into that of corresponding set of FPRs. However, the reasoning mechanism of weighted FPRs converted from our proposed weighted FDT can always be equivalent to that of the weighted FDT. In the following, we specify this reasoning mechanism of weighted FPRs.

When observations do not exactly match with the antecedent part of the rule, approximate matching and reasoning should be used to deduce a consequent. Fuzzy matching and fuzzy reasoning play a key role in the approximate reasoning process. This type of matching and reasoning is very human like since in many situations human beings have to make decisions based on incomplete and fuzzy information. Incorporating this capability into the knowledge-based system is necessary. As an example, let us consider two fuzzy production rules converted from Fig. 2.

Rule 1) **IF** `Temperature` = Hot **AND** `Outlook` = Cloudy **THEN** `Swimming`

Rule 2) **IF** `Temperature` = Hot **AND** `Outlook` = Rain THEN `W_lifting`

where, for simplicity, we assume that both the LWs and the GWs of the two rules equal to one. The observed fact, for instance, is supposed to have the form

$$\begin{aligned}&\texttt{Temperature} = 0.6/\text{Hot} + 0.4/\text{Mild} + 0.0/\text{Cool}\\ &\texttt{Outlook} = 0.0/\text{Sunny} + 0.5/\text{Cloudy} + 0.5/\text{Rain}.\end{aligned}$$

What conclusion can be drawn? The reasonable conclusion which tallies with person's thinking and perception should have the fuzzy form of (`a/swimming,b/W_lifting`) where `a` and `b` are two real number belonging to $[0, 1]$. If a crisp decision

needs to be made, one can determine the crisp decision according to the maximum of a and b. The problem is that using the fuzzy matching and fuzzy reasoning based on the observation, can a reasonable conclusion be drawn? Using the method proposed in the following section, one can obtain a consequent of (0.5/swimming,0.4/W_lifting).

There are mainly two types of fuzzy reasoning method. One is based on Zadeh's CRI method, while the other is based on similarity measure. In essence, what we propose here is a similarity-based method.

In [38] a similarity-based method was proposed using the degree of subsethood defined by Kosko [16]. Subsequently, it was extended to fuzzy production rules with LWs, GWs, and other parameters [39]. In this type of method, the similarity between the observed fact and the antecedent must be computed according to the selected similarity measure. In this paper, because of our fuzzy representation method (the attribute value is a fuzzy set defined on a linguistic term space), the similarity measure between the attribute value and the antecedent of the rule is regarded as the membership value, which indicates to what degree the example belongs to the corresponding term. For instance, the similarity between attribute value "$0.6/\text{Hot} + 0.4/\text{Mild} + 0.0/\text{Cool}$" and the antecedent "Temperature = Hot" is 0.6.

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

$$\mathbf{R}_i\text{: If } V_1 \text{ is } A_1^{(i)} \textbf{AND } V_2 \text{ is } A_2^{(i)} \ldots \textbf{AND } V_{n_i} \text{ is } A_{n_i}^{(i)} \textbf{ THEN } U \text{ is } B^{(i)}, \mathrm{LW}_1^{(i)}, \mathrm{LW}_2^{(i)}, \ldots, \mathrm{LW}_{n_i}^{(i)}, \mathrm{GW}(R_i).$$

The observed object has attribute values in the following forms:

$$\text{Fact 1: } V_1 \text{ is } C_1^{(i)}, \quad \text{Fact 2: } V_2 \text{ is } C_2^{(i)}, \ldots, \quad \text{Fact } n_i\text{: } V_{n_i} \text{ is } C_{n_i}^{(i)}.$$

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

$$\mathrm{SM}^{(i)} = \min_{1 \le j \le n_i} \left(\mathrm{LW}_j \cdot \mathrm{SM}_j^{(i)} \right).$$

Let there be K classes denoted by $\mathrm{CLASS}_1, \ldots, \mathrm{CLASS}_K$. The inferred result is regarded as a fuzzy vector $(x_1, x_2, \ldots, x_k)$ where x_k is the value which indicates to what degree the observed object belongs to CLASS_k $(k = 1, 2, \ldots, K)$. The degree x_k is determined by the following:

$$x_k = \max_i \left\{ \mathrm{GW}(R_i) \cdot \mathrm{SM}^{(i)} \,\middle|\, B^{(i)} = \mathrm{CLASS}_k \right\} \quad (k = 1, 2, \ldots, K).$$

The normalized form of the inferred result is defined as $(d_1, d_2, \ldots, d_K)$ where

$$d_k = \frac{x_k}{\mathrm{Max}_{1 \le j \le K} x_j} \quad (k = 1, 2, \ldots, K).$$

When the crisp inferred result is needed, one can take the consequent CLASS with maximum d_k $(1 \le k \le K)$. One problem is that the algorithm cannot give a crisp decision if there exists more than one maximum d_k $(1 \le k \le K)$. In that situation, we need another defuzzification method to determine the crisp decision.

It is worth noting that the fuzzy matching and reasoning method proposed here are equivalent to the methods mentioned in [40] when all the LWs and the GWs are equal to one. Hence, our proposed method of fuzzy matching and reasoning indeed generalizes the traditional one.

D. Performance on the Small Training Set

We illustrate the above fuzzy matching and the fuzzy reasoning process. Let us consider the second example in Table I and the set of fuzzy rules converted from Fig. 2. All of LWs and the GWs of these six fuzzy rules are set to one

$$\begin{aligned}
\text{Rule 1) } \mathrm{SM}^{(1)} &= \min(1 \cdot 0.6, 1 \cdot 0.8) \\
&= 0.6. \text{ Consequent: Swimming.} \\
\text{Rule 2) } \mathrm{SM}^{(2)} &= \min(1 \cdot 0.6, 1 \cdot 0.2) \\
&= 0.2. \text{ Consequent: Swimming.} \\
\text{Rule 3) } \mathrm{SM}^{(3)} &= \min(1 \cdot 0.6, 1 \cdot 0.0) \\
&= 0.0. \text{ Consequent: W_lifting.} \\
\text{Rule 4) } \mathrm{SM}^{(4)} &= \min(1 \cdot 0.4, 1 \cdot 0.0) \\
&= 0.0. \text{ Consequent: W_lifting.} \\
\text{Rule 5) } \mathrm{SM}^{(5)} &= 1 \cdot 0.0 \\
&= 0.0. \text{ Consequent: W_lifting.} \\
\text{Rule 6) } \mathrm{SM}^{(6)} &= \min(1 \cdot 0.4, 1 \cdot 1.0) \\
&= 0.4. \text{ Consequent: Volleyball.} \\
\max\left(1 \cdot \mathrm{SM}^{(1)}, 1 \cdot \mathrm{SM}^{(2)}\right) &= 0.6, \\
\max\left(1 \cdot \mathrm{SM}^{(3)}, 1 \cdot \mathrm{SM}^{(4)}, 1 \cdot \mathrm{SM}^{(5)}\right) &= 0.0, \\
\max\left(1 \cdot \mathrm{SM}^{(6)}\right) &= 0.4.
\end{aligned}$$

Hence, the inferred result is

$$(0.4/\text{Volleyball}, 0.6/\text{Swimming}, 0.0/\text{W\_lifting})$$

and its normalized form is $(0.67/\text{V}, 1.00/\text{S}, 0.00/\text{W})$. If a crisp decision is to be made, one can take the second class Swimming.

The matching result of the 16 examples in Table I with respect to the six fuzzy rules learned in Section II-D (where all the LWs and the GWs are set to one) are placed in the middle columns of Table II, labeled "*classification test before modifying weights*." From Table II, one can see that the learning accuracy is not high (examples 2, 8, and 16 cannot be classified correctly). Because the matching result depends on both the LWs and the GWs, we expect to improve the learning accuracy via adjusting the weights using a hybrid neural network instead of consulting with domain experts. From the last three columns of Table II one can see the matching result after learning the weights, where examples 2 and 16 have already been classified correctly. The problem of refining the weights by a HNN remains to be investigated in Section IV.

TABLE II
TEST RESULTS OF LEARNED RULES

No.	Known classification			Classification test before modifying weights			Classification test after modifying weights		
	V	S	W	V	S	W	V	S	W
1	0.00	0.80	0.20	0.00	1.00	0.00	0.00	1.00	0.00
2	1.00	0.70	0.00	0.67	1.00	0.00	1.00	0.82	0.00
3	0.30	0.60	0.10	0.29	1.00	0.43	0.34	1.00	0.51
4	0.90	0.10	0.00	1.00	0.43	0.43	1.00	0.32	0.43
5	0.00	0.00	1.00	0.43	0.14	1.00	0.53	0.18	1.00
6	0.20	0.00	0.80	0.43	0.00	1.00	0.43	0.00	1.00
7	0.00	0.00	1.00	0.00	0.00	1.00	0.00	0.00	1.00
8	0.70	0.00	0.30	0.25	0.00	1.00	0.25	0.00	1.00
9	0.20	0.80	0.00	0.00	1.00	0.00	0.00	1.00	0.00
10	0.00	0.30	0.70	0.14	0.00	1.00	0.14	0.00	1.00
11	0.40	0.70	0.00	0.00	1.00	0.00	0.00	1.00	0.00
12	0.70	0.20	0.10	1.00	0.00	0.43	1.00	0.00	0.43
13	0.00	0.00	1.00	0.00	0.25	1.00	0.00	0.18	1.00
14	0.00	0.00	1.00	0.43	0.00	1.00	0.43	0.00	1.00
15	0.00	0.00	1.00	0.00	0.00	1.00	0.00	0.00	1.00
16	0.80	0.60	0.00	1.00	1.00	0.00	1.00	0.59	0.00

IV. REFINING THE WEIGHTS BY A HYBRID NEURAL NETWORK

From Section III, we know that the proposed weighted FDT including its reasoning mechanism keeps the comprehensibility of decision tree. The remaining problem is how to adjust these weights such that the learning accuracy can be improved. Usually, these weights are given by consultation with domain experts repeatedly. Of course, this kind of consultation is very time consuming. Instead of the consultation, one promising approach to obtain these weights is "*learning them by a connectionist structure*." It will be concerned with the design of a neural network in which the connection weights of the network correspond to the LWs and GWs of the set of FPRs and the output of the network is the classification consequent; the formulation of learning algorithm for training these weights; and the complexity analysis of the algorithm.

A. Mapping a Set of Weighted Fuzzy Production Rules into a Hybrid Neural Network

According to the matching and reasoning mechanism established in Section III, a set of learned fuzzy rules can be mapped into a hybrid neural network which has three layers: term layer, rule layer and classification layer. The key structure of the mapped neural network is described as follows.

1) Term Layer: This is the input layer. Each node 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 fuzzy rule. The similarity degree can also be the membership value which indicates to what degree the observed fact belongs to the linguistic term.

2) Rule Layer: This is the only hidden layer. Each node representing an extracted fuzzy rule corresponds to a leaf of the weighted FDT. According to linguistic terms (propositions) appearing in the antecedent part of a rule, the connections between the term layer and the rule layer are determined.

3) Classification Layer: This is the output layer. Each node represents a cluster. Since the inferred result of the weighted fuzzy rule has generally the form of fuzzy vector (the 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 is the membership value which indicates to what degree the training object belongs to the cluster corresponding to the node.

4) Connection Weights: The LWs of the extracted set of fuzzy rules are regarded as the connection weights between the term layer and the rule layer. The GWs of the set of fuzzy rules are regarded as the connection weights between the rule layer and the classification layer. Noting that the fuzzy rules generated by a fuzzy decision tree algorithm (e.g., Fuzzy ID3) have LWs and GWs being equal to one, all the connection weights of the network are initially set to one. It indicates that the refinement of weights starts from a set of FPRs generated by an initial FDT algorithm.

5) Activation Function: Instead of using the sigmiod function as in traditional neural networks, two activation functions, V_1 and V_2, for the rule layer and the classification layer, respectively, are defined as follows:

$$V_1(\mathrm{LW}_j, x_j) = \mathrm{Min}_j(\mathrm{LW}_j \cdot x_j)$$
$$V_2(\mathrm{GW}_j, y_j) = \mathrm{Max}_j(\mathrm{GW}_j \cdot y_j).$$

These two activation functions are consistent with the reasoning mechanism established in the Section III-C. Noting the use of the operators min and max, the network belongs to a type of hybrid connection neural network.

Fig. 3 shows a hybrid neural network mapped from the six fuzzy rules extracted from the FDT shown in Fig. 2.

B. Training the Neural Network

To formulate the backpropagation algorithm, let us consider a generic case of this kind of hybrid neural network, shown in Fig. 4 where there are L_0 term nodes, L_1 rule nodes and L_2

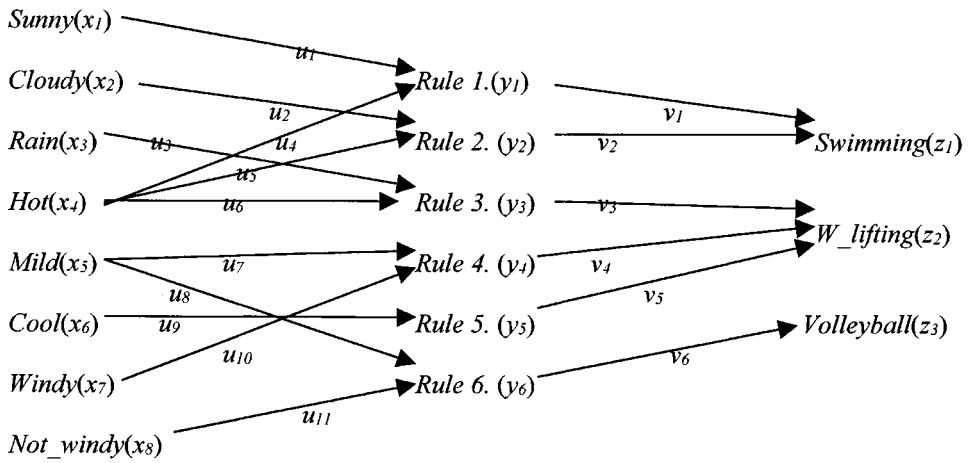

Fig. 3. A neural network with hybrid connections.

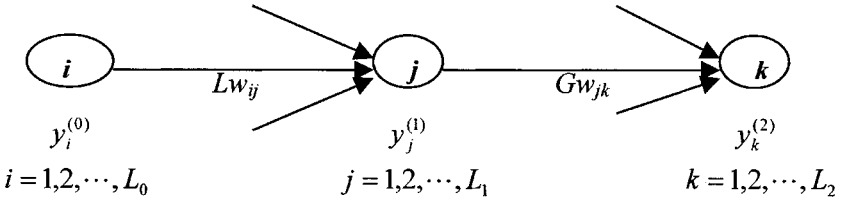

Fig. 4. Generic form of the HNN.

classification nodes. For a given input vector, e.g., the nth input vector, the forth propagation process of the input vector is described as follows:

initial layer (term layer):

$$\left\{ y_i^{(0)}[n] \,\middle|\, i = 1, 2, \ldots, L_0 \right\} \quad \text{(the given input vector)};$$

first layer (rule layer):

$$y_j^{(1)}[n] = \bigwedge_{i=1}^{L_0} \left(\mathrm{Lw}_{ij} \cdot y_i^{(0)}[n] \right) \quad j = 1, 2, \ldots, L_1 \tag{2}$$

second layer (class layer):

$$y_k^{(2)}[n] = \bigvee_{j=1}^{L_1} \left(\mathrm{Gw}_{jk} \cdot y_j^{(1)}[n] \right) \quad k = 1, 2, \ldots, L_2 \tag{3}$$

Let there be N training models. Then, the total error function is defined as

$$\begin{aligned} E &= \frac{1}{2} \sum_{n=1}^{N} \sum_{k=1}^{L_2} \left(y_k^{(2)}[n] - y_k[n] \right)^2 \\ &= \sum_{n=1}^{N} \left(\frac{1}{2} \sum_{k=1}^{L_2} \left(y_k^{(2)}[n] - y_k[n] \right)^2 \right) \\ &= \sum_{n=1}^{N} E_n \end{aligned} \tag{4}$$

in which $y_k[n]$ is the kth actual output of the nth training model ($1 \le k \le L_2$). It is easy to see from (2), (3), and (4) that the error E is a function with respect to LW Lw_{ij} and GW Gw_{jk} ($i = 1, \ldots, L_0; j = 1, \ldots, L_1; k = 1, \ldots, L_2$).

Let us now derive the standard backpropagation equations. According to the principle of gradient descent, the backpropagation equations for the hybrid neural network shown in Fig. 4 can be written as

$$\begin{aligned} \mathrm{Lw}_{ij} &:= \mathrm{Lw}_{ij} - \alpha_{ij} \frac{\partial E_n}{\partial \mathrm{Lw}_{ij}} \quad \text{and} \\ \mathrm{Gw}_{jk} &:= \mathrm{Gw}_{jk} - \beta_{jk} \frac{\partial E}{\partial \mathrm{Gw}_{jk}} \end{aligned} \tag{5}$$

in which α_{ij} (β_{jk}) is the learning rate. The two partial derivatives appearing in (5) are shown as follows:

$$\frac{\partial E_n}{\partial \mathrm{Gw}_{jk}} = \begin{cases} O_k \cdot y_j^{(1)}, & \text{if } T_1 \ge T_2 \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

$$\frac{\partial E_n}{\partial \mathrm{Lw}_{ij}} = \begin{cases} \displaystyle\sum_{k=1}^{L_2} O_k \cdot \mathrm{Gw}_{jk} \cdot y_i^{(0)}, & \text{if } T_1 \ge T_2, \quad T_3 \le T_4 \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

in which, for simplicity, the attached $[n]$ has been omitted from each $y_\alpha^{(\beta)}$ ($\alpha = i, j, k; \beta = 0, 1, 2$)

$$O_k = y_k^{(2)} - y_k, \quad T_1 = \mathrm{Gw}_{jk} \cdot y_j^{(1)}$$
$$T_2 = \bigvee_{q \ne j} \left(\mathrm{Gw}_{qj} \cdot y_q^{(1)} \right), \quad T_3 = \mathrm{Lw}_{ij} \cdot y_i^{(0)}$$

and

$$T_4 = \bigwedge_{p \ne i} \left(\mathrm{Lw}_{pj} \cdot y_p^{(0)} \right).$$

The training procedure is briefly described as follows.

Step 1) Initialization—all the connection weights are initially set to one.
Step 2) Model forth propagation. Computer the output of each node by (2) and (3).
Step 3) Error backpropagation. Compute the adjustment of each weight by (5)–(7).
Step 4) If a given stop criterion is satisfied then stop else repeat Steps 2) and 3).

We now use the training procedure to train the hybrid neural network shown in Fig. 3. The results of training are $u[1-11]$: $0.54, 1.00, 1.00, 0.77, 0.74, 0.81, 0.81, 1.00, 1.00, 1.00, 1.00$ and $v[1-6]$: $0.77, 1.00, 1.00, 1.00, 1.00, 1.00$. According to the reasoning mechanism established in Section III-C, one can use this set of fuzzy rules with trained weights to test the 16 training examples given in Table I. The test result is placed into Table II (the columns labeled "*classification test after modifying weights*"). It should be noted that this set of weighted fuzzy rules is now able to classify objects 2 and 16 correctly.

C. Discussions and Remarks

As a kind of knowledge parameters, the weights in fuzzy production rules are usually acquired according to the following simple procedure.

Step 1) Knowledge engineers together with domain experts specify a set of weighted fuzzy production rules (where the values of weights remain to be determined) and select a set of historical records for testing.
Step 2) Domain experts give the initial values of these weights.
Step 3) Knowledge engineers test this set of weighted fuzzy production rules according to a selected evaluation index. If the evaluation index is acceptable, then go to Step 5), else go to Step 4).
Step 4) Knowledge engineers adjust the values of weights by consulting domain experts, go to Step 3).
Step 5) Stop. These values of weights are finally determined for using in the set of weighted fuzzy production rules.

This paper indicates that the task of consultation with domain specialists [specified in the above Steps 3) and 4)] may be replaced with training hybrid neural networks. It implies that the time spent in the consultation between knowledge engineers and domain experts can be reduced to some extent.

For training a neural network, the initial values of connection weights are usually given randomly. But in our proposed training procedure, they are initially set to one. The reason is twofold. One is the intuitive background that weights being equal to one correspond to a set of FPRs, which are generated by traditional learning algorithms such as Fuzzy ID3. The other is that the initial error with weights being one is usually less than that with weights selected randomly (due to the learning accuracy the algorithm for generating these FPRs has already had). To some extent, this one initialization improves the training performance. For example, the training of the rice taste problem in Section V are 535 epochs when initial weights are one, but are 2268 epochs when initial weights are selected randomly. Table V shows the epochs of training HNNs with one initilization and with random initialization of weight for the five databases.

Compared with the traditional backpropagation algorithm, our proposed method makes use of the max and min functions instead of using the sigmoid function. That implicitly results in the reduction of computational effort. Compare (6) and (7) with the formulation of traditional BP algorithm, one can see that the computational complexity of our proposed algorithm is less than that of the traditional one. Like the traditional backpropagation algorithm, the algorithm proposed here is effective and efficient. The effectiveness is demonstrated in the experiments of next section. Moreover, the performance of the proposed algorithm can be further improved by replacing the crisp derivatives with the smooth derivatives [2]

$$\frac{\partial(\mathrm{Max}(x,c))}{\partial x} = \begin{cases} 1, & \text{if } x \geq c \\ x, & \text{if } x < c \end{cases}$$

and

$$\frac{\partial(\mathrm{Min}(x,c))}{\partial x} = \begin{cases} 1, & \text{if } x \leq c \\ c, & \text{if } x > c \end{cases}$$

but we will report the improvement result separately.

Usually, the convergence of training algorithms depends on their learning rate, the form of input vector, as well as the selection of initial weights. We do not theoretically investigate the convergence of the proposed training algorithm for the hybrid neural network, but will explore it by numerical experiments in the next section. The experimental results in the next section show a better performance for the convergence of the proposed training algorithm. Table IV shows a brief comparison between the BP algorithm and our proposed one with respect to the epochs when HNNs converge. Although there has been some research related to HNNs such as the approximation to continuous functions [4], application to fuzzy controller and expert systems [5], the capability as universal approximators [6], min–max neural networks [26], [27], and so on, much more theoretical study on convergence is really needed.

Moreover, the training of the hybrid neural network cannot generate new rules. This is because the training is a kind of nondestructive learning during which the network structure is kept intact and no new connections appear between adjacent layers. Generally, when the set of fuzzy rules with weights adjusted by a hybrid neural network cannot yet attain a satisfactory accuracy, new rule generation is regarded as necessary. Hence, the research on destructive training with respect to hybrid neural networks is very important and significant.

There has been some work on extracting weighted fuzzy production rules from neural networks (e.g., [14], [30]). The article [14] proposed an algorithm for fuzzy weighted rule extraction from adaptive fuzzy neural networks. The fuzzy neural network consists of five layers and the backpropagation training algorithm is used. The concept of weight in [14] is distinct from that in this paper. Moreover, we proposed a novel approach to tune knowledge representation parameters in a fuzzy production rule

TABLE III
SUMMARY OF THE EMPLOYED DATABASES

Database	Domain	Source	Classes	Attributes [b]	Examples
Rice Taste	Food	[19]	2[a]	5	105
Iris	Biological	[9]	3	4	150
Mango Leaves	Biological	[20]	3	18	166
Thyroid Gland	Medical	[18]	3	5	215
Pima India Diabetes	Medical	[18]	2	8	768

[a] The continuous output is separated into two categories by positive values and negative values.
[b] All attributes are numerical.

TABLE IV
LEARNING ACCURACY (x, y) OF DIFFERENT METHODS WHERE x DENOTES THE TRAINING ACCURACY AND y DENOTES THE TESTING ACCURACY

Database	Fuzzy ID3	WFDTI	BP
Rice Taste	(0.86, 0.84)	(0.97, 0.90)	(0.96, 0.91)
Iris	(0.96, 0.96)	(0.97, 0.97)	(0.97, 0.97)
Mango Leaf	(0.86, 0.80)	(0.96, 0.89)	(0.99, 0.90)
Thyroid Gland	(0.82, 0.78)	(0.93, 0.85)	(0.95, 0.82)
Pima India Diabetes[a]	(0.75, 0.72)	(0.82, 0.78)	(0.82, 0.76)

[a] The iteration does not converge and a threshold is used.

TABLE V
TRAINING EPOCHS OF WEIGHTED FDT AND BP ALGORITHM

Database	WFDTI[a]	WFDTI[b]	BP
Rice Taste	535	2268	2896
Iris	69	186	328
Mango Leaf	689	1677	1893
Thyroid Gland	812	2285	2166
Pima India Diabetes	2819	5671	8000[c]

[a] with initial weight being 1. [b] with initial weight selected randomly. [c] given maximum epochs.

using a fuzzy neural network in [30]. The approach includes the initialization, the feed forward computation and the backward weight adjustment. Both in [14] and in [30], the conventional addition and multiplication are chosen as the inner operation of neural networks.

V. NUMERICAL EXPERIMENTS

In Sections II–IV, we have briefly presented the learning/reasoning process of the proposed method on a small training set. To further understand the performance of this method, we apply it as well as two benchmark learning algorithms (the initial fuzzy ID3 and the traditional back-propagation) to five databases.

A. Databases

The five databases employed for experiments are obtained from various sources. Their features are briefly described below and summarized in Table III.

1) Rice taste data—this database was used by Nozaki [19] to verify a simple and powerful algorithm for fuzzy rule generation. It contains 105 cases with five numerical attributes. The classification attribute is continuous. According to positive values and negative values of the classification attribute, cases are categorized to two classes in our experiments.
2) Iris data—this was the original data Fisher used to illustrate the discriminant analysis [9]. It contains 150 cases of three different kinds of flowers. Each case consists of four numerical attributes.
3) Mango leaf data—this set was used by Pal [20] to investigate the automatic feature extraction based on fuzzy techniques. It provides the information on different kinds of mango leaf with 18 numerical attributes for 166 patterns (cases). It has three classes representing three kinds of mango.
4) Thyroid gland data [18]—this set contains 215 cases of three different kinds of thyroid grand. Each case consists of five numerical attributes.
5) Pima India diabetes data [18]—this database contains 768 cases related to the diagnosis of diabetes (268 positive and 500 negative). It has eight numerical attributes.

B. Experimental Procedures

We call the method proposed in this paper weighted fuzzy decision tree induction—in short, WFDTI. Three methods were compared: fuzzy ID3; traditional BP-algorithm; and

the WFDTI. We select fuzzy ID3 and BP algorithm as the benchmark for evaluating the WFDTI.

Noting that all attributes of the selected six databases are numerical, we need to fuzzify these numerical attributes into linguistic terms. We make use the following simple algorithm for generating triangular type of membership functions ([15], [40]).

Let X be the considered data set. We intend to cluster X into k linguistic terms $T_j, j = 1, 2, \ldots, k$. For simplicity, we assume the type of membership to be triangular. Each linguistic term T_j will have the triangular membership functions as follows (see equation at the bottom of the page). Each pair of adjacent membership functions crosses at the membership value 0.5. The only parameters needed to be determined are the k centers $\{a_1, a_2, \ldots, a_k\}$. An effective method to determine these centers is the Kohonen feature maps algorithm [15]. At the initial time, k centers are set to be distributed evenly on the range of X. Let

$$A = \{a_1, a_2, \ldots, a_k\}; \quad d(X, A) = \sum_{x \in X} \operatorname*{Min}_i |x - a_i|.$$

The centers will be adjusted iteratively. Each iteration consists of the following three steps:

1) randomly take a value x from X, denoted by $x[n]$;
2) search for an integer m such that $|x[n] - a_m[n]| = \operatorname{Min}_j |x[n] - a_j[n]|$;
3) put $a_m[n+1] = a_m[n] + \alpha(x[n] - a_m[n])$ and keep other centers unchanged, where n is the iteration time and α is the learning rate.

The iteration ends when $d(X, A)$ converges.

In our experiments, the number of linguistic terms for each attribute is taken to be three, the parameter α specified in Section II-B for reducing the fuzziness in training process is set to 0.35 and the leaf criterion is taken to be 0.75. The learning accuracy is used to compare the performance of these methods. For each considered database, 50% of the data is uniformly and randomly chosen as the training set and the remaining 50% of cases is held for testing. This procedure is repeated six times. The learning accuracy, namely, the training accuracy and testing accuracy, is the average of the six. Table IV shows the learning accuracy of each method when we applied three methods to different databases. The experimental results of BP algorithm are obtained by using the MATLAB toolbox of neural networks.

C. Remarks

For numerical attributes, the learning accuracy of fuzzy decision tree is usually poor when the number of linguistic terms is very small. To improve the learning accuracy, one can use one of the following approaches:

1) increasing the number of linguistic terms for attributes and tuning the membership functions of these terms; that will result in the increase of the number of extracted fuzzy rules;
2) modifying the reasoning mechanism ([13], [17]); that will lower the comprehensibility of the tree since the classification distribution is used in the modified reasoning mechanism;
3) using new attributes which are given by linear combinations of original attributes; that is called oblique decision tree [23]; it also lowers the comprehensibility of the tree due to the unclear meaning of new attributes;
4) refining knowledge parameters related to the tree is our proposed approach in this paper; compared with the original heuristic algorithm for tree generation, this method has the weakness of increasing training complexity.

In this paper, we argue that the approach 4) is the most promising one. The reason is that it cannot only improve the learning accuracy, but also keep the comprehensibility of the FDT and the simplicity of the extracted fuzzy rules. From Table IV, one can see that our proposed method improves the learning performance of fuzzy ID3. There is no significant difference between our proposed algorithm and BP algorithm, which has been universally considered to have better accuracy, but the concepts formed by our proposed weighted FDT are understood more easily than that formed by traditional neural networks. Noting that there is only one hidden layer and its number of nodes is equal to the number of rules in our method, it can be seen that the complexity of our proposed method is less than that of BP algorithm. Except for the problem of Pima India diabetes, the HNN shows a better performance of convergence where all initial weights are set to one (Table V) and where the number of maximum epochs of training the HNN for each database is set to 8000.

One can see from Tables IV and V that the performance for the Pima India problem is poor. The improvements of learning accuracy obtained by refining the HNN are not significant. We

$$T_1(x) = \begin{cases} 1 & x \le a_1 \\ (a_2 - x)/(a_2 - a_1) & a_1 < x < a_2 \\ 0 & x \ge a_2 \end{cases}$$

$$T_k(x) = \begin{cases} 1 & x \ge a_k \\ (x - a_{k-1})/(a_k - a_{k-1}) & a_{k-1} < x < a_k \\ 0 & x \le a_{k-1} \end{cases}$$

$$T_j(x) = \begin{cases} 0 & x \ge a_{j+1} \\ (a_{j+1} - x)/(a_{j+1} - a_j) & a_j < x < a_{j+1} \\ (x - a_{j-1})/(a_j - a_{j-1}) & a_{j-1} < x < a_j \\ 0 & x \le a_{j-1} \end{cases} \quad 1 < j < k.$$

perform a further analysis on this problem and find that small number of linguistic terms for the attributes is not suitable for this problem. In other words, to achieve a better performance, the scale of the weighted FDT should be very large.

VI. Conclusion

The inclusion of some knowledge parameters such as LW and GW is necessary for enhancing the representation power of FDTs. This paper proposes refining these knowledge parameters in a hybrid neural network to improve the learning performance of FDTs, instead of increasing the numbers of linguistic terms or the complexity (the nodes) of FDTs to improve that. The main advantages of the proposed method are as follows:

1) the learning performance can be improved by refining these parameters without much computational effort.
2) since each knowledge parameter used in FDTs has the clear meaning, the comprehensibility of FDTs can be kept.
3) to determine these knowledge parameters, the training of HNN can replace the task of consultation with domain specialists to a great extent.

The synergy between fuzzy decision tree induction and hybrid neural network offers new insight into the construction of hybrid intelligent systems with better learning accuracy.

VII. Future Work

In this paper, we extend the FDT induction to the weighted FDT induction. The main idea is extracting weighted rules from a decision tree and then refining them by using a hybrid neural network. The weights in a weighted fuzzy production rule are considered knowledge parameters and are mapped into the connection weights of the hybrid neural network to be refined. A training algorithm like traditional BP algorithm is derived. The weighed FDT with trained (refined) weights improves the learning accuracy of the original FDT and keeps the comprehensibility of the FDT.

The knowledge parameters are initially given by domain experts in terms of their domain knowledge. According to the performance of these parameters in a system, knowledge engineers usually need to revise the values of these parameters by consulting domain expert such that a better performance can be achieved. This paper shows, for acquiring these knowledge parameters, the training of hybrid neural networks may replace the task of consulting domain specialists. It implies that the time to consult with domain experts will be reduced if this technique is used in conjunction with consultation with domain experts. However, since the knowledge parameters are closely related to the domain knowledge, the reduction of time of consultation is difficult to formulate. In the next phase, we will implement the parameter refinement in a real-world problem. For this problem, the time to consult with domain experts for determining knowledge parameters will be actually measured, the time for training the hybrid neural network will be given by the training procedure, and a comparison between training HNNs and consulting domain experts will be made.

Moreover, although this paper has derived a training algorithm which is similar to the traditional BP algorithm and contains some parameters like learning rates, it is difficult to specify them for achieving a better convergence performance. The present exploration is experimental. The resulting hybrid neural network still faces the local minimum problem. As an important but difficult problem, the theoretical analysis of convergence for the HNN will be investigated later. In addition, the derivatives for max and min in the formulation are supposed to be crisp. It is likely that the performance of the proposed algorithm can be further improved by replacing the crisp derivatives with smooth derivatives. We will complete this work in the future.

References

[1] G. Adorni *et al.*, "Ham quality control by means of fuzzy decision trees: A case study," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, Anchorage, AK, May 1998, pp. 1583–1588.

[2] 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.

[3] X. Boyen and L. Wehenkel, "Automatic induction of fuzzy decision trees and its application to power system security assessment," *Fuzzy Sets Syst.*, vol. 102, pp. 3–19, 1999.

[4] J. J. Buckley and Y. Hayashi, "Can fuzzy neural nets approximate continuous fuzzy functions?," *Fuzzy Sets Syst.*, vol. 61, pp. 43–51, 1993.

[5] ——, "Hybrid neural nets can be fuzzy controllers and fuzzy expert systems," *Fuzzy Sets Syst.*, vol. 60, pp. 135–142, 1993.

[6] ——, "Hybrid fuzzy neural nets are universal approximators," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, Orlando, FL, June 1994, pp. 238–249.

[7] S. M. Chen, "A new approach to handling fuzzy decision making problems," in *IEEE Trans. Syst., Man, Cybern.*, vol. 18, Nov./Dec. 1988, pp. 1012–1016.

[8] K. J. Cios and L. M. Sztandera, "Continuous ID3 algorithm with fuzzy entropy measures," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, San Diego, CA, Mar. 1992, pp. 469–476.

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

[10] I. Hayashi, T. Maeda, A. Bastian, and L. C. Jain, "Generation of decision trees by fuzzy ID3 with adjusting mechanism of AND/OR operators," in *IEEE Int. Conf. Fuzzy Syst.*, Anchorage, AK, May 1998, pp. 681–685.

[11] H. Ichihashi, T. Shirai, K. Nagasaka, and T. Myoshi, "Neuro-fuzzy ID3," *Fuzzy Sets Syst.*, vol. 81, pp. 157–167, 1996.

[12] C. Z. Janikow, "Fuzzy decision trees: Issues and methods," *IEEE Trans. Syst., Man, Cybern.*, vol. 28, pp. 1–14, Feb. 1998.

[13] B. Jeng, M. Jeng, and T.-P. Liang, "FILM: A fuzzy inductive learning method for automated knowledge acquisition," *Decision Support Syst.*, vol. 21, pp. 61–73, 1997.

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

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

[16] B. Kosko, "Fuzziness versus probability," in *Fuzzy and Neural Systems: A Dynamic System to Machine Intelligence*. Englewood Cliffs, NJ: Prentice-Hall, 1992, pp. 263–298.

[17] P. E. Maher and D. St. Clair, "Uncertain reasoning in an ID3 machine learning framework," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, San Francisco, CA, Mar. 28–Apr. 1, 1993, pp. 7–12.

[18] C. Merz and P. Murphy, UCI Repository of Machine Learning Databases, 1996.

[19] K. Nozaki, H. Ishibuchi, and H. Tanaka, "A simple but powerful heuristic method for generating fuzzy rules from numerical data," *Fuzzy Sets Syst.*, vol. 86, pp. 251–270, 1997.

[20] S. K. Pal, "Fuzzy set theoretic measures for automatic feature evaluation," *Inform. Sci.*, vol. 64, pp. 165–179, 1992.

[21] J. R. Quinlan, "Induction of decision trees," *Machine Learning*, vol. 1, pp. 81–106, 1986.

[22] D. Ruan and E. E. Kerre, "Fuzzy implication and generalized fuzzy method of cases," *Fuzzy Sets Syst.*, vol. 54, pp. 23–37, 1993.

[23] R. Setiano and H. Liu, "A connectionist approach to generating oblique decision trees," *IEEE Trans. Syst., Man, Cybern.*, vol. 29, pp. 440–444, June 1999.

[24] J. W. Shavlik *et al.*, "Symbolic and neural learning algorithms: An experimental comparison," *Mach. Learning*, vol. 6, no. 2, pp. 111–134, 1991.

[25] T. Shibata, T. Abe, K. Tanie, and M. Nose, "Motion planing of a redundant manipulator-criteria of skilled operators by fuzzy-ID3 and GMDH and optimization by GA," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, Yokohama, Japan, Mar. 1995, pp. 99–102.

[26] P. K. Simpson, "Fuzzy min-max neural networks—Part 1: Classification," *IEEE Trans. Neural Networks*, vol. 3, pp. 776–786, Sept. 1992.

[27] ——, "Fuzzy min-max neural networks—Part 2: Clustering," *IEEE Trans. Fuzzy Syst.*, vol. 1, pp. 32–45, Feb. 1993.

[28] L. G. Sison and E. K. P. Chong, "Fuzzy modeling by induction and pruning of decision trees," in *Proc. IEEE Int. Symp. Intell. Contr.*, Columbus, OH, Aug. 1994, pp. 166–171.

[29] T. Tani and M. Sakoda, "Fuzzy modeling by ID3 algorithm and its application to prediction of outlet temperature," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, San Diego, CA, Mar. 1992, pp. 923–930.

[30] E. C. C. Tsang and D. S. Yeung, "Refining local weights and certainty factors using a neural network," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, San Diego, CA, Oct. 1998, pp. 1512–1517.

[31] H. C. Tseng and D. W. Teo, "Medical expert system with elastic fuzzy logic," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, Orlando, FL, June 1994, pp. 2067–2071.

[32] M. Umano, H. Okamoto, I. Hatano, H. Tamura, F. Kawachi, S. Umedzu, J. Kinoshita, and , "Fuzzy decision trees by fuzzy ID3 algorithm and its application to diagnosis systems," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, June 1994, pp. 2113–2118.

[33] X. Z. Wang, B. Chen, G. L. Qian, and F. Ye, "On the optimization of fuzzy decision trees," *Fuzzy Sets Syst.*, vol. 112, pp. 117–125, 2000.

[34] X. Z. Wang and J. R. Hong, "On the handling of fuzziness for continuous-valued attributes in decision tree generation," *Fuzzy Sets Syst.*, vol. 99, pp. 283–290, 1998.

[35] R. Weber, "Fuzzy-ID3: A class of methods for automatic knowledge acquisition," in *Proc. Int. Conf. Fuzzy Logic Neural Networks*, Iizuka, Japan, July 1992, pp. 265–268.

[36] R. Yager, "Approximate reasoning as basis for rule-based expert system," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC–14, pp. 636–643, 1984.

[37] D. S. Yeung and E. C. C. Tsang, "Weighted fuzzy production rules," *Fuzzy Sets Syst.*, vol. 88, pp. 299–313, 1997.

[38] ——, "Improved fuzzy knowledge representation and rule evaluation using fuzzy Petri nets and degree of subsethood," *Int. J. Intell. Syst.*, vol. 9, pp. 1083–1100, 1994.

[39] ——, "A weighted fuzzy production rule evaluation method," in *Proc. FUZZY-IEEE/IFSA'95*, 1995, pp. 461–468.

[40] Y. Yuan and M. J. Shaw, "Induction of fuzzy decision trees," *Fuzzy Sets Syst.*, vol. 69, pp. 125–139, 1995.