# Improving Generalization of Fuzzy IF–THEN Rules by Maximizing Fuzzy Entropy

Xi-Zhao Wang, *Senior Member, IEEE*, and Chun-Ru Dong, *Member, IEEE*

***Abstract*—When fuzzy IF–THEN rules initially extracted from data have not a satisfying performance, we consider that the rules require refinement. Distinct from most existing rule-refinement approaches that are based on the further reduction of training error, this paper proposes a new rule-refinement scheme that is based on the maximization of fuzzy entropy on the training set. The new scheme, which is realized by solving a quadratic programming problem, is expected to have the advantages of improving the generalization capability of initial fuzzy IF–THEN rules and simultaneously overcoming the overfitting of refinement. Experimental results on a number of selected databases demonstrate the expected improvement of generalization capability and the prevention of overfitting by a comparison of both training and testing accuracy before and after the refinement.**

***Index Terms*—Classification, fuzzy entropy, fuzzy IF–THEN rules, maximum entropy principle, parametric fuzzy IF–THEN rules, rule-based reasoning.**

## I. Introduction

KNOWLEDGE acquisition or extraction is regarded as the bottleneck of expert system development in the artificial intelligence field. Knowledge may be expressed in different forms such as mappings, graphics, tables, neural networks, rough sets, especially fuzzy IF–THEN rules. So far, fuzzy IF–THEN rules are the most commonly used representation tools of knowledge with uncertainty and have wide applications to computational intelligence [1].

So far, a large number of approaches to fuzzy IF–THEN rule generation from numerical and nominal data for classification have been proposed in recent decades. It is difficult to give a complete survey on fuzzy rule generation/extraction. The following is a brief and incomplete summary on this topic.

1) Fuzzy rules for classification can be extracted from data through a simple heuristic procedure [2]–[6], where the heuristic information plays a key part in the rule extraction.
2) The induction approach of the fuzzy decision tree, which first generates a decision tree and then converts the tree into a set of fuzzy IF–THEN rules, is important for fuzzy rule extraction [7], [8].
3) Fuzzy rules for classification can be generated by using the neuro-fuzzy technique [9]–[13]. In [14], the authors give a survey on this technique.
4) Fuzzy rules can also be generated and evaluated by using genetic algorithms [15]–[19], [47].
5) Fuzzy rule extraction can be conducted by using the rough set technique in which fuzzy upper/lower approximations and the fuzzy core are used to generate fuzzy attribute reduction [20].
6) Other techniques such as data mining [21], [22], simulated annealing [23], hierarchical partitioning [24], support vector machines [25], B-spline approximation [26], etc., can be used.

Weighted fuzzy rules incorporate several parameters into the fuzzy rules, which aims to enhance the representation power of fuzzy rules. The key issues for weighted rules are how to determine the weight values by using a soft computing method and not an experience-based method. From literature, one can find many investigations on the weighted fuzzy rules. The following is brief survey on weighted fuzzy rules.

The certainty grade is considered to be the rule weight, and the effect of the weight for improving the performance of classification is investigated in detail [27], and furthermore, a method of rule weight specification is developed through heuristic algorithms [28]. A type of fuzzy weighted additive rules are discussed in [29], and the optimization technique is suggested to access the weights. This type of weighted additive rules are developed further in [30], and a quadratic programming is given to obtain the weights. With respect to the weight specification, the authors in [31] propose to train a neural network, while the authors in [32] propose to use of the receiver operating characteristics analysis. Weighted fuzzy rules have had many real applications. For example, the authors in [33] propose a new weighted fuzzy reasoning algorithm and further apply it to the engineering domain. Yeung and Tsang in [34] incorporate the concept of local and global weight into the fuzzy rules that the representation power of fuzzy rules get further enhanced.

Due to many uncertain factors such as the insufficiency of training data and limitations of the methods of rule extraction, fuzzy IF–THEN rules extracted from data by using the previous one or some methods are not likely to have a satisfying performance. In this case, one can consider that the extracted fuzzy IF–THEN rules require a refinement. For example, we can map a set of fuzzy IF–THEN rules into a neural network. The parameters in the IF–THEN rules correspond to connection weights of the neural network, and therefore, the refinement of IF–THEN rules can be completed by training the neural network [11], [35]. This type of refinement aims at minimizing the error function on the training set via training the neural network. It is really helpful to improve the training accuracy, but our new experiments show that this type of refinement is likely to lead to an overfitting. One

Manuscript received February 11, 2007; revised June 21, 2007, September 15, 2007, and November 28, 2007; accepted November 30, 2007. First published April 30, 2008; current version published June 11, 2009. This work was supported in part by the National Nature Science Foundation of China under Grant 60473045 and in part by the plan of first 100 excellent scientists in educational committee of Hebei province.

The authors are with the Department of Mathematics and Computer Science, Hebei University, Baoding City, Hebei Province 071002, China (e-mail: xizhaowang@ieee.org; dongcr@hbu.cn).

Digital Object Identifier 10.1109/TFUZZ.2008.924342

of the reasons for overfitting may be that we have adopted the refinement criterion of training error minimization. The overfitting phenomenon refers to the fact that the generalization capability has been downgraded although the training accuracy is significantly improved. Motivated by changing the refinement criterion to improve the generalization capability, we propose in this paper a new refinement scheme based on the fuzzy entropy maximization, instead of the training error minimization. As a type of criterion of information, the (fuzzy) entropy maximization principle has been widely applied to many fields such as pattern recognition and image processing [36], [37].

Now, we analyze the training process in which fuzzy IF–THEN rules are used for classification problems. Consider a two-class problem, and suppose that two *very similar* objects are selected for training, object A belongs to class 1, and object B belongs to class 2. When we would like to generate a set of fuzzy IF–THEN rules from a training set including objects A and B, most existing algorithms assume that the target output is (1,0) for object A and is (0,1) for object B. Since the two objects are very similar, we consider the assumption of target-output-being-crisp nonreasonable. Due to the similarity between objects A and B, it may be reasonable that we consider that the degree of object A belonging to class 1, which is unnecessarily 1, is bigger than the degree of object A belonging to class 2, which is unnecessarily 0. We may suppose that the target output for object A is, say, (0.6,0.4), instead of (1,0). In this situation, the classification uncertainty exists inherently for the two objects. This paper makes an attempt to recover and utilize this type of inherent classification uncertainty information during the process of refinement of fuzzy IF–THEN rules.

The following is our main idea for the refinement of fuzzy IF–THEN rules. Suppose that we have extracted a set of initial fuzzy IF–THEN rules from a training set by using a given training algorithm in which a Boolean vector is considered the target output, and the set of initial fuzzy IF–THEN rules need to refine. Then, a group of parameters, called global weights, are incorporated into the set of fuzzy IF–THEN rules. The reasoning results of the fuzzy IF–THEN rules will be changing with diverse values of parameters. We attempt to adjust these parameters such that the fuzzy entropy of actual outputs of the fuzzy IF–THEN rules on the training set attains maximum. The adjustment is subject to a number of constraints that indicate that a training object after refinement has a correct output of crisp class if the training object can be classified correctly before refinement. Those constraints imply that the training accuracy will not reduce after refinement. It is worth noting that the fuzzy entropy maximization implies a fuzzification of Boolean vectors, which tries to recover and utilize the inherent classification uncertainty information lost in the training process. The fuzzy entropy maximization application to refining parameters of fuzzy IF–THEN rules, which is realized in this paper by solving a quadratic programming problem, is expected to improve the fuzzy IF–THEN rules generalization capability.

The rest of this paper is organized as follows. Section II gives a globally weighted fuzzy IF–THEN rule-reasoning algorithm. Section III addresses the parameter refinement based on fuzzy entropy maximization, derives the parameter-solving problem as a quadratic program, and gives the procedure of parameter refinement. Section IV experimentally demonstrates the generalization capability improvement after parameter refinement, and the last section concludes this paper.

## II. Globally Weighted Fuzzy IF–THEN Rules Reasoning

According to Zadeh's initial definition of generalized modus Ponens [38], the reasoning model of fuzzy IT-THEN rules is described as

A fuzzy IF–THEN rule: IF 'x is A' THEN 'y is B'
A given fact: 'x is A*'
A conclusion: 'y is B*'.

There are many approaches to achieve a fuzzy classification result by matching an object to a set of fuzzy rules. For example, one can achieve the fuzzy classification by using nonlinear regression represented as a Takagi–Sugeno fuzzy model recently published in [46]. It is easy to check that different reasoning mechanism directly influences the results of fuzzy classification. In this paper, we focus on a type of globally weighted fuzzy production rules (WFPRs) [34] and its corresponding reasoning mechanism. A WFPR is a parametric fuzzy IF–THEN rule with the conjunctive form

$$\wedge_{j=1}^{n}(V_j = A_j) \Rightarrow (U = C)\,, g$$

where $V_j (j = 1, 2, \ldots, n)$ and $U$ are variables; $A_j (j = 1, 2, \ldots, n)$ and $C$ are fuzzy values of these variables (in other words, $A_j (j = 1, 2, \ldots, n)$ are fuzzy sets); the parameter $g$ is a real number in [0,1] denoting the global weight of the rule $R$; and $\wedge$ denotes the conjunction AND. WFPRs degenerate to fuzzy IF–THEN rules in commonsense when the global weight $g$ is ignored.

Consider a set of $m$ WFPRs: $S = \{R_i,\ i = 1, 2, \ldots, m\}$ and a given fact, the reasoning model that slightly modifies the generalized modus Ponens is described as follows:

A set of WFPRs:

$$R_i:\ \wedge_{j=1}^{n}(V_j = A_j^{(i)}) \Rightarrow (U = C)\,,\ g_i, \qquad i = 1, 2, \ldots, m$$

$$\frac{\text{A given fact: } (V_j = B_j)\ j = 1, 2, \ldots, n}{\text{A conclusion: } U = D, \mathrm{CF}(D)}$$

where $g_i$ represents the global weight assigned to the $i$th rule $R_i$ and $\mathrm{CF}(D)$ is the certainty factor of the conclusion. It is worth noting that the $m$ rules have the same consequent $C$. It is possible that the $j$th proposition of the antecedent of the $i$th rule (i.e., $V_j$) is missing. In this case, the membership function of $A_j^{(i)}$ is defined as const 1.

How to draw the conclusion $U = D,\ \mathrm{CF}(D)$: The following is a scheme to draw the conclusion and to evaluate its certainty factor. We call the scheme globally weighted reasoning [39].

### A. Globally Weighted Reasoning Algorithm

*Step A1)* For an observed object $B = (B_1, B_2, \ldots, B_n)$ and each IF–THEN rule $R_i$ within $S$, the membership of the attribute value $B_j$ belonging to $A_j^{(i)}$ is calculated by $a_j^{(i)} = A_j^{(i)}(B_j)$, where $A_j^{(i)}(\cdot)$ denotes its membership function. The overall degree of matching object B to rule $R_i$, $a_i$, is defined as

$$a_i = \frac{1}{m}(\min_{1\leq j\leq n} a_j^{(i)}), \qquad i = 1, 2, \ldots, m \tag{1}$$

where *m* is the number of fuzzy IF–THEN rules.

*Step A2)* The conclusion's certainty factor $\mathrm{CF}(D)$ is given by

$$\mathrm{CF}(D) = \sum\nolimits_{i-1}^{m} (a_i g_i) \tag{2}$$

It is easy to see that certainty factor of conclusion is a linear combination of the global weight parameters $g_1, g_2, \ldots, g_m$.

For classification problems, the reasoning result *D* is exactly equal to *C*, and the value given in (2) denotes the degree of truth of the object belonging to class *C*. If there are *K* classes (corresponding to *K* sets of fuzzy IF–THEN rules) and $C_k$ represents the label of the *k*th class $(1 \leq k \leq K)$, then the computed result in (2) refers to the degree of truth of some class, which is denoted by $x_k (k = 1, 2, \ldots, K)$. The normalized form of the inferred result is defined as $(d_1, d_2, \ldots, d_k)$, where $d_k = x_k / \mathrm{Max}_{1\leq j\leq K} x_j \ (k = 1, 2, \ldots, K)$.

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

We now consider a classification problem with *K* classes. Let $\Omega = \{x_1, x_2, \ldots, x_N\}$ be a training set from which a set of initial fuzzy IF–THEN rules $S = \{R_i, \ i = 1, 2, \ldots, m\}$ have been extracted. (One can find many approaches to fuzzy IF–THEN rule extraction from references, but this paper does not discuss the specific fuzzy IF–THEN rule extraction methods.) Suppose that the set of extracted IF–THEN rules have not satisfying performance and, therefore, require a refinement. There exist a number of refinement methodologies, e.g., the Tsang *et al.* work [35] in which a set of fuzzy IF–THEN rules are mapped into a neural network with the parameters of fuzzy IF–THEN rules corresponding to the connection weights of the neural network, and then, the refinement is conducted by training the neural network. This paper proposes a new refinement viewpoint that basically results from the fuzzy entropy maximization principle.

Considering the global weights of the set of weighted fuzzy IF–THEN rules as a number of parameters to be refined, we propose the reasoning scheme in Fig. 1, where $\{g_i, \ i = 1, 2, \ldots, m\}$ denotes the global weights. The value of $\mathrm{CF}_k$, representing the possibility of the object $x_i$ belonging to the *k*th class $(1 \leq k \leq K)$ is dependent on the global weight $(g_1, g_2, \ldots, g_m)$, that is

$$\mathrm{CF}_k = \mathrm{CF}_k (g_1, g_2, \ldots, g_m).$$

We now give an example to illustrate how $\mathrm{CF}_k (1 \leq k \leq K)$ dependent on $(g_1, g_2, \ldots, g_m)$.

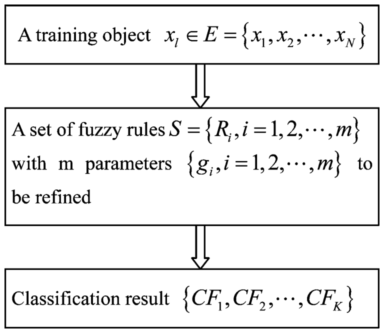
A training object $x_l \in E = \{x_1, x_2, \cdots, x_N\}$
A set of fuzzy rules $S = \{R_i, i = 1, 2, \cdots, m\}$ with m parameters $\{g_i, i = 1, 2, \cdots, m\}$ to be refined
Classification result $\{CF_1, CF_2, \cdots, CF_K\}$

Fig. 1. Reasoning process for an object.

*Example 1:* Consider the following four fuzzy IF–THEN rules:

| | V1 | V2 | V3 | Class | Global weight |
|---|---|---|---|---|---|
| Rule1 | Big | | Small | C1 | g1 |
| Rule2 | Small | | Big | C1 | g2 |
| Rule3 | | Big | Big | C2 | g3 |
| Rule4 | | Small | Small | C2 | g4 |

where $V_1$, $V_2$, and $V_3$ are three variables, and the fuzzy sets (Big) and (Small) are defined as

$$\mathrm{Big}(x) = \begin{cases} 1, & x \geq 1 \\ x, & x \in (0,1) \\ 0, & x \leq 0 \end{cases}$$

$$\mathrm{Small}(x) = \begin{cases} 0, & x \geq 1 \\ 1 - x, & x \in (0,1) \\ 1, & x \leq 0. \end{cases}$$

Suppose that $A = (0.6, 0.7, 0.2)$ is an object to be classified. Matching object $A$ to rules 1 and 2 according to the globally weighted reasoning algorithm [i.e., (1) and (2)], we have

$$\mathrm{CF}_1 = \frac{0.6 g_1 + 0.2 g_2}{2}.$$

Similarity, matching object $A$ to rules 3 and 4, we obtain

$$\mathrm{CF}_2 = \frac{0.2 g_3 + 0.3 g_4}{2}.$$

Therefore, the classification result of object $A$ matching rules 1–4 is

$$\left(\frac{0.6 g_1 + 0.2 g_2}{2}, \frac{0.2 g_3 + 0.3 g_4}{2}\right)$$

which is obviously dependent on $(g_1, g_2, g_3, g_4)$. Before weight refinement, the weight vector $(g_1, g_2, g_3, g_4)$ is regarded as (1,1,1,1), which results in a classification result (0.4, 0.25), and therefore, the crisp decision is that object $A$ belongs to class 1. After weight refinement, the weight $(g_1, g_2, g_3, g_4)$will change. For example, if the refined weight vector $(g_1, g_2, g_3, g_4)$ is (0.2, 0.3, 0.6, 0.8), then the classification result will be (0.09, 0.18), and the crisp decision is that object $A$ belongs to class 2, which shows that the result of crisp classification for an object to be classified is really dependent on the value of weight.

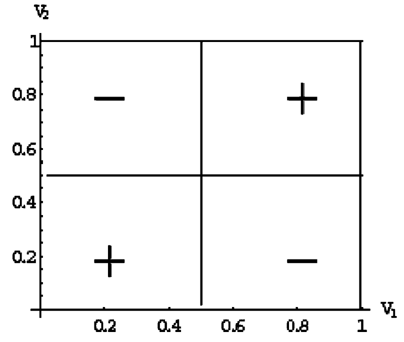

Fig. 2. Classification result for $(g_1, g_2, g_3, g_4) = (0.5, 0.5, 0.5, 0.5)$.

The following example 2 illustrates that the change of global weights will have much impact on the classification boundary.

*Example 2:* Consider the following four globally weighted fuzzy IF–THEN rules

| | V₁ | V₂ | Class | Global weight |
|---|---|---|---|---|
| Rule1 | Big | Big | $C_1$ | $g_1$ |
| Rule2 | Small | Small | $C_1$ | $g_2$ |
| Rule3 | Big | Small | $C_2$ | $g_3$ |
| Rule4 | Small | Big | $C_2$ | $g_4$ |

where the membership functions of fuzzy sets Big and Small are defined as Example 1. We now use these four rules with different global weights and the globally weighted reasoning algorithm proposed in (1) and (2) to classify each point in $[0, 1] \times [0, 1]$ and then to observe the classification boundary. The classification results for four groups of weights are shown in Figs. 2–5, where sign $+$ denotes class $C_1$, and sign – denotes class $C_2$. From Figs. 2–5, one can see that the global weights have a significant impact on the classification boundary of reasoning results.

The current focus is what criterion will be used to determine global weights $(g_1, g_2, \ldots, g_m)$ and how will they be determined? Noting that each training object has a crisp target classification and an actual classification output can be obtained by matching the object to the set of fuzzy IF–THEN rules when $(g_1, g_2, \ldots, g_m)$ are known, one traditional approach to refinement is to adjust the values of $(g_1, g_2, \ldots, g_m)$ such that the difference between the target classification and the actual output is as small as possible. Refinement method based on training-error-reduction is commonly used. This type of refinement can certainly improve (at least does not reduce) the training accuracy. However, this approach to refinement is very likely to lead to an over fitting and is not helpful to improve the inductive capability of this set of fuzzy IF–THEN rules. This paper proposes using the maximum fuzzy entropy as a criterion to refine parameters $g_1, g_2, \ldots, g_m$.

## III. Parameter Refinement Based on Fuzzy Entropy Maximization

As a type of criterion of information, the fuzzy entropy maximization principle has been widely applied to many fields such as pattern recognition and image processing [35]–[37]. This paper will apply this principle to the parameter determination of rule-based fuzzy reasoning.

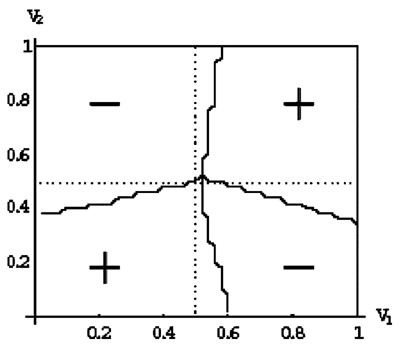

Fig. 3. Classification result for $(g_1, g_2, g_3, g_4) = (0.5, 0.37, 0.24, 0.63)$.

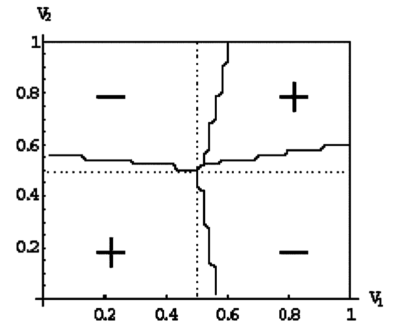

Fig. 4. Classification result for $(g_1, g_2, g_3, g_4) = (0.35, 0.65, 0.5, 0.5)$.

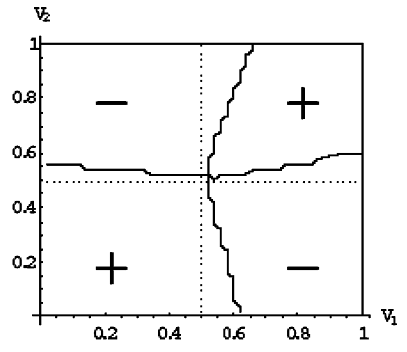

Fig. 5. Classification result for $(g_1, g_2, g_3, g_4) = (0.32, 0.73, 0.45, 0.57)$.

*Definition:* Let A be a fuzzy set defined on a space *X* with the membership function *A*(*x*) and *F*(*X*) be the set of all fuzzy sets defined on *X*. A mapping from *F*(*X*) to [0,1], *H*(*A*) is called fuzzy entropy of *A* if *H*(*A*) satisfies [35], [36] the following:

1) *H*(*A*) attains its minimum iff $\forall x \in X, A(x) = 0$ or $A(x) = 1$.
2) *H*(*A*) attains its maximum iff $\forall x \in X, A(x) = 1/2$.
3) When *A* is more fuzzy than *B*, i.e.,

$$\forall x \in X, 1/2 \geq A(x) \geq B(x) \text{ or}$$
$$1/2 \leq A(x) \leq B(x), \; H(A) \geq H(B).$$

4) $H(A) = H(X - A)$ for all $A \in F(X)$.

There are many functions *H*(·) satisfying the conditions (1)–(4). For example, similar to Shannon's entropy [40], the fuzzy entropy of a finite fuzzy set $A = (\mu_1, \mu_2, \ldots, \mu_T)$ [41] can be defined as

$$H_f\,(A) = -\sum_{j=1}^{T}\left(\mu_j \ln \mu_j + (1-\mu_j)\ln(1-\mu_j)\right) \qquad (3)$$

which represents a type of fuzziness of the fuzzy *A*. It is easy to check that (4), shown below, satisfies the aforementioned four conditions, and therefore, (4) is a quite simple fuzzy entropy function

$$E_f\,(A) = \sum_{j=1}^{T}\left(\mu_j(1-\mu_j)\right). \qquad (4)$$

This paper will use (4) as our fuzzy entropy definition for discussion. When the membership $\mu_j$ of a fuzzy set *A* is equal to 0.5 for all *j*, the fuzzy entropy of the fuzzy set attains the maximum. In this case, the fuzzy set *A* has the maximum fuzziness. The fuzzy entropy maximization implies that, for drawing a fuzzy set as our conclusion, we prefer a fuzzy set with bigger fuzziness to other fuzzy sets. In other words, we consider that an event with much uncertainty (fuzzy entropy) will bring us more information when it occurs.

*Maximum fuzzy entropy principle:* Consider a reasoning process that includes a number of parameters to be determined. With respect to a given fact, the reasoning conclusion will be a parametric fuzzy set, which implies that the reasoning conclusion will be changing with diverse parameters. We prefer the parametric fuzzy set with maximum fuzzy entropy (to other fuzzy sets) as our reasoning conclusion, subject to the given constraints. The fuzzy entropy maximization can be realized by parameter adjustment.

Why does the fuzzy entropy maximization can improve the classification accuracy? The following is an intuitive explanation.

Suppose that we have a classification problem with *n* class and A is an object to be classified.

Without any additional information for classification available, a most reasonable classification result for A should be that the possibility of A belonging to each one of the *n* classes is equal (i.e., 1/*n*), which can be achieved by maximizing the entropy of A.

If some additional information for classification is available (i.e., there exists a training set in which each example's class is known), then to get a reasonable and fair classification for A, we should maximize the entropy of A subject to some constraints, each constraint represents that a training example can be classified correctly. These constraints mean that the available information for classification has been utilized and that the remaining uncertain information for classification is handled by maximum entropy. The reasonable and fair classification for A is expected to lead a precision increase.

Since A is an object remaining to classify, we have to complete the entropy maximization of A via training set's entropy maximization.

Unfortunately, thus far we cannot give a formal mathematical formulation for the earlier explanation. It will be considered as an important issue for further study.

Fig. 6 is an outline of weight-refinement procedure.

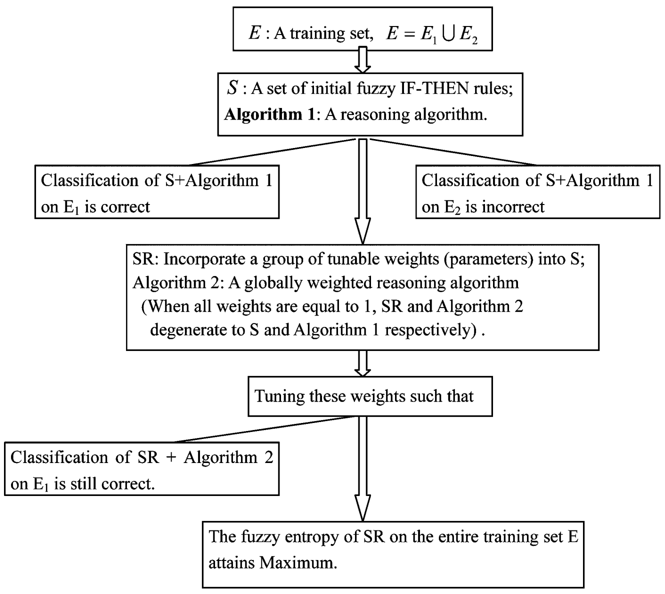
$E$ : A training set, $E = E_1 \cup E_2$
$S$ : A set of initial fuzzy IF-THEN rules;
**Algorithm 1**: A reasoning algorithm.
Classification of S+Algorithm 1 on E$_1$ is correct
Classification of S+Algorithm 1 on E$_2$ is incorrect
SR: Incorporate a group of tunable weights (parameters) into S;
Algorithm 2: A globally weighted reasoning algorithm
(When all weights are equal to 1, SR and Algorithm 2 degenerate to S and Algorithm 1 respectively).
Tuning these weights such that
Classification of SR + Algorithm 2 on E$_1$ is still correct.
The fuzzy entropy of SR on the entire training set E attains Maximum.

Fig. 6. Outline of weight-refinement procedure.

Consider Fig. 1 where the reasoning result for a training object $x_i$ is $\{\mathrm{CF}_1^{(i)}, \mathrm{CF}_2^{(i)}, \ldots, \mathrm{CF}_K^{(i)}\}$. Each $\mathrm{CF}_j^{(i)}$ denotes the possibility with which the object $x_i$ belongs to the *j*th class. Noting that each $\mathrm{CF}_j^{(i)}$ is a function of parameters $(g_1, g_2, \ldots, g_m)$ that remain to be determined, we have

$$\mathrm{CF}_j^{(i)} = \mathrm{CF}_j^{(i)}\,(g_1, g_2, \ldots, g_m)\,. \qquad (5)$$

Then, the fuzzy entropy of the reasoning result with respect to object $x_i$ can be expressed as

$$E_f\,(x_i;\; g_1, g_2, \ldots, g_m) = \sum_{j=1}^{K}(\mathrm{CF}_j^{(i)}(g_1, g_2, \ldots, g_m)(1-\mathrm{CF}_j^{(i)}(g_1, g_2, \ldots, g_m))). \qquad (6)$$

The fuzzy entropy on the training set is defined as

$$E_f\,(g_1, g_2, \ldots, g_m) = \sum_{i=1}^{N} E_f\,(x_i;\; g_1, g_2, \ldots, g_m). \qquad (7)$$

Our parameter-refinement method attempts to maximize the fuzzy entropy (7) subject to a number of constraints that we formulate as follows.

From Fig. 1, we know that each training object $x_i$ matching to a set of fuzzy IF–THEN rules with *m* parameters $\{g_i,\ i = 1, 2, \ldots, m\}$ will lead to a consequent $\{\mathrm{CF}_1^{(i)}, \mathrm{CF}_2^{(i)}, \ldots, \mathrm{CF}_K^{(i)}\}$ dependent on the parameters. We first consider a special case in which all parameters are equal to 1. This case corresponds to the initial (nonrefined) fuzzy IF–THEN rules extracted from the data set. Matching $x_i$ to the set of fuzzy IF–THEN rules with *m* parameters being equal to 1, we obtain a consequent denoted by $\{d_1^{(i)}, d_2^{(i)}, \ldots, d_K^{(i)}\}$, that is

$$d_j^{(i)} = \mathrm{CF}_j^{(i)}\,(1, 1, \ldots, 1) \qquad (7a)$$

for $j = 1, 2, \ldots, K$. Supposing that $x_i$ belongs to the *j*th class, we then define an index set $I$ as

$$I = \left\{ i \,\middle|\, 1 \le i \le N,\ d_{j_i}^{(i)} > \max_{p \ne j_i} d_p^{(i)} \right\} \tag{8}$$

representing the training objects that can be classified correctly by the initial fuzzy IF–THEN rules, and then, we can mathematically formulate our parameter refinement problem as the programming problem with constraints, shown in (9) at the bottom of the page, where the index set $I$ is defined as (8). It is noted that the number of constraints in (9) is equal to the number of objects that can be correctly classified by the initial fuzzy IF–THEN rules extracted from training set.

From the globally weighted reasoning algorithm given in Section II, we know that $\mathrm{CF}_j^{(i)}(g_1, g_2, \ldots, g_m)$ is a linear combination of parameters $(g_1, g_2, \ldots, g_m)$. Therefore, (9) is a quadratic programming problem with linear constraints. We now derive its standard mathematical form.

Suppose *m* initial IF–THEN rules are classified *K* group (where *K* is the number of classes) and the *k*th group has $r_k$ IF–THEN rules with the same consequent. The parameter vector is defined as

$$\begin{aligned} G &= (g_1, g_2, \ldots, g_m)^T \\ &= (\underline{g_1, g_2, \ldots, g_{r_1}}; \underline{g_{r_1+1}, g_{r_1+2}, \ldots, g_{r_1+r_2}}; \\ &\quad \ldots; \underline{g_{r_1+\cdots+r_{K-1}+1}, \ldots, g_m})^T \\ &= \left(G_1^T; G_2^T; \cdots; G_K^T\right)^T. \end{aligned} \tag{10}$$

The superscript *T* denotes the transpose of a matrix or a vector throughout this paper. According to the globally weighted reasoning algorithm given in Section II [i.e., (1)], we match the *i*th training object to the *K* groups of initial IF THEN rules and then obtain a matching result, which is denoted by

$$\begin{aligned} A_i &= (a_{i1}, a_{i2}, \ldots, a_{im})^T \\ &= (\underline{a_{i1}, a_{i2}, \ldots, a_{ir_1}}; \underline{a_{i(r_1+1)}, \ldots, a_{i(r_1+r_2)}}; \\ &\quad \ldots; \underline{a_{i(r_1+\cdots+r_{K-1}+1)}, \ldots, a_{im}})^T \\ &= \left(A_{i1}^T; A_{i2}^T; \ldots; A_{iK}^T\right)^T \end{aligned} \tag{11}$$

from which the index set I is determined by (8). According to the globally weighted-reasoning algorithm given in Section II, the objective function in (9) can be expressed as

$$\begin{aligned} &E_f(g_1, g_2, \ldots, g_m) \\ &= \sum_{i=1}^{N}\sum_{k=1}^{K} \left(G_k^T A_{ik}\left(1 - A_{ik}^T G_k\right)\right) \\ &= \sum_{k=1}^{K} \left( G_k^T \left(\sum_{i=1}^{N} A_{ik}\right) - G_k^T \left(\sum_{i=1}^{N} A_{ik} A_{ik}^T\right) G_k \right) \\ &= -G^T A G + B^T G \end{aligned} \tag{12}$$

where the matrix *A* and the vector *B* are defined as

$$A = \begin{pmatrix} \sum_{i=1}^{N} A_{i1}A_{i1}^T & & & \\ & \sum_{i=1}^{N} A_{i2}A_{i2}^T & & \\ & & \ddots & \\ & & & \sum_{i=1}^{N} A_{iK}A_{iK}^T \end{pmatrix}_{m\times m},$$

$$B = \begin{pmatrix} \sum_{i=1}^{N} A_{i1} \\ \sum_{i=1}^{N} A_{i2} \\ \vdots \\ \sum_{i=1}^{N} A_{iK} \end{pmatrix}_{m\times 1}. \tag{13}$$

Noting that the consequent of matching the *i*th training object to the initial set of fuzzy rules (i.e., weighted fuzzy rules with all global weights being 1) is denoted by $\{d_1^{(i)}, d_2^{(i)}, \ldots, d_K^{(i)}\}$, we suppose $j_i = \arg\max_{1\le j\le K} d_j^{(i)}$. Then, the constraints given in (9) can be expressed as

$$G_{j_i}^T A_{ij_i} > G_k^T A_{ik}, \qquad 1 \le k \ne j_i \le K,\ i \in I. \tag{14}$$

In matrix form, the previous constraints can be rewritten as

$$\begin{pmatrix} (C_1)_{(K-1)\times m} \\ (C_2)_{(K-1)\times m} \\ \vdots \\ (C_L)_{(K-1)\times m} \end{pmatrix}_{L(K-1)\times m} \begin{pmatrix} G_1 \\ G_2 \\ \vdots \\ G_K \end{pmatrix}_{m\times 1} > \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}_{m\times 1} \tag{15}$$

where $L = |I|$, and for each $i \in I$

$$(C_i)_{(K-1)\times m} = \begin{pmatrix} -A_{i1}^T & 0 & \cdots & A_{ij_i}^T & \cdots & 0 \\ 0 & -A_{i2}^T & \cdots & A_{ij_i}^T & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & A_{ij_i}^T & \cdots & -A_{iK}^T \end{pmatrix}. \tag{16}$$

Summarizing the previous derivations, we give the following parameter-refinement procedure.

*Parameter-refinement procedure:*

*Step 1)* extracting a set of fuzzy IF–THEN rules from a given training-set according to a given learning algorithm;

$$\begin{cases} \mathrm{Max}\left( \begin{aligned} &E_f(g_1, g_2, \ldots, g_m) \\ &= \sum_{i=1}^{N}\sum_{j=1}^{K} \left(\mathrm{CF}_j^{(i)}(g_1, g_2, \ldots, g_m)(1 - \mathrm{CF}_j^{(i)}(g_1, g_2, \ldots, g_m))\right) \end{aligned} \right), \\ \text{Subject to } \mathrm{CF}_{j_i}^{(i)} > \max_{p\ne j_i} \mathrm{CF}_p^{(i)},\ i \in I \end{cases} \tag{9}$$

*Step 2)* dividing these rules into $K$ groups (where $K$ is the number of classes) so that rules in one group have the same consequent;

*Step 3)* for each group, matching training objects to the initial IF–THEN rules to obtain the vector $A_i$ (11);

*Step 4)* expressing the parameter vector as (10) and determining the index set *I* by (8);

*Step 5)* determining the integer $j_i$ for each $i \in I$ according to $j_i = \arg\max_{1 \le j \le K} d_j^{(i)}$, which is defined as (7a);

*Step 6)* solving the quadratic programming problem given by the quadratic objective function (12) subject to linear constraints (15) to get these parameters.

Since the maximum fuzzy entropy is attained on candition that all output components are 0.5, one may argue about how to determine the exact class when we use the maximum fuzzy entropy principle to adjust these weights, which leads to such a case that most class outputs are 0.5. Actually, since the number of training objects is much bigger than the number of rules (weights) and many constraints exist, it is almost impossible that the weight adjustment based on maximum entropy brings about such a result that all class outputs are 0.5. Our experiments in Section IV testify to this fact.

## IV. Experimental Demonstration

### A. Generating the Initial Fuzzy IF–THEN Rules

We use fuzzy decision tree induction to generate a set of initial fuzzy IF–THEN rules. What technique is used to generate the initial IF–THEN rules is unimportant since our focus in this paper is the rule refinement after rule generation. The complete algorithm for generating fuzzy decision trees can be found in many references [7], [8]. Here, we outline the main steps of generation procedure:

1) fuzzifying the continuous attributes to get a number of linguistic terms;
2) choosing a heuristic to generate a fuzzy decision tree; and
3) converting this tree into a set of fuzzy IF–THEN rules.

In our experiments, the heuristic is selected as the well-known fuzzy ID3 and the linguistic terms (fuzzy sets) are selected as normal fuzzy numbers with the following form:

$$f(x) = \begin{cases} e^{-(x-c)^2/r^2}, & x > c \\ e^{-(x-c)^2/l^2}, & x \le c \end{cases} \tag{17}$$

where *c* is the center, and *r* and *l* are determined by solving

$$e^{-(x-c_1)^2/r^2}\Big|_{x=(c_1+c_2)/2} = e^{-(c_2-x)^2/l^2}\Big|_{x=(c_1+c_2)/2} = \frac{1}{2} \tag{18}$$

for any two adjacent centers $c_1$ and $c_2$ ($c_1 < c_2$). It results in $r = l = (c_2 - c_1)/2\sqrt{\ln 2}$.

In this way, if we use *n* centers $c_1 < c_2 < \cdots < c_n (n \ge 2)$for a continuous attribute, then *n* normal membership functions $f_1, f_2, \ldots, f_n$ will be expressed as

$$f_1(x) = \exp\left(-\frac{4\ln 2(x-c_1)^2}{(c_2-c_1)^2}\right) \tag{19}$$

$$f_j(x) = \begin{cases} \exp\left(-\frac{4\ln 2(x-c_j)^2}{(c_j-c_{j-1})^2}\right), & x > c_j \\ \exp\left(-\frac{4\ln 2(x-c_j)^2}{(c_{j+1}-c_j)^2}\right), & x \le c_j \end{cases}$$
$$j = 2, 3, \ldots, n-1 \tag{20}$$

$$f_m(x) = \exp\left(-\frac{4\ln 2(x-c_{n-1})^2}{(c_n-c_{n-1})^2}\right). \tag{21}$$

### B. Databases Selected for Demonstration

The nine databases employed for our experiments are obtained mainly from user–computer interface (UCI) machine learning database repository [42]. Their features are briefly described as follows.

1) *Iris plant data:* This database is created by Fisher [43]. It contains 150 objects with four numerical attributes. The predicted attribute has three values; one class is linearly separable from the other two.
2) *Mango leaf data:* This set is used by Pal [44] 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 (objects). It has three classes representing three kinds of mango.
3) *Wine data:* It contains 178 examples with three classes (class A—59; class B—71; class C—48) and 13 numerical attributes [42].
4) *Image segmentation data:* It contains 2310 instances with seven classes and 19 numerical attributes [42].
5) *Thyroid gland data:* This set contains 215 objects of three different kinds of thyroid grand [42]. Each object consists of five numerical attributes.
6) *Pima India diabetes data:* This database contains 768 objects related to the diagnosis of diabetes [42] (500 positive and 268 negative). It has eight numerical attributes.
7) *Glass identification database:* This database has 214 instances related to seven classes of glass [42]. Each instance has nine numerical attributes. In our experiments, we consider the first class as the positive and the other six classes as the negative.
8) *Auto-mpg data:* This database has 398 instances and nine attributes. In our experiments, we only use the five numerical attributes and one integer-valued attribute [42]. The mpg attribute is regarded as the class attribute. Moreover, since the attribute horsepower has six missing values, we only use 392 of 398 instances.
9) *Sonar database:* This database contains 208 patterns, 111 patterns belonging to metal class, and 97 patterns belonging to rock class [42]. Each pattern is a set of 60 numbers ranging from 0.0 to 1.0.

### C. Experimental Procedure

We use the cross-validation procedure. Each database is randomly partitioned into ten disjoint subsets, and the size of each subset is *m*/10, where *m* is the number of examples of the dataset.

TABLE I
AVERAGED TRAINING AND TESTING ACCURACY BEFORE AND AFTER THE REFINEMENT FOR THE SELECTED NINE DATABASES

| Databases | Number of classes | Training / testing before refinement | Training / testing after refinement |
|---|---|---|---|
| Iris plant data | 3 | 0.95 0.92 | 0.96 0.94 |
| Mango leaf data | 3 | 0.83 0.79 | 0.84 0.82 |
| Wine data | 3 | 0.90 0.78 | 0.91 0.79 |
| Segmentation data | 7 | 0.80 0.76 | 0.83 0.79 |
| Thyroid grand data | 3 | 0.83 0.80 | 0.86 0.85 |
| Auto-mpg data | 2 | 0.73 0.70 | 0.75 0.73 |
| Sonar signal data | 2 | 0.85 0.75 | 0.86 0.80 |
| Pima diabetes data | 2 | 0.73 0.71 | 0.76 0.75 |
| Glass identification | 2 | 0.70 0.68 | 0.75 0.74 |

The procedure is then run ten times, each time using different one of these subsets as the validation (testing) set and combining the other nine subsets for the training set. The training and testing accuracies are then averaged. For each time, the experimental procedure is the following.

1) Generate a set of initial IF–THEN rules from the training set according to the fuzzy decision tree induction mentioned in Section IV-A.
2) Calculate the training accuracy (the percentage of training examples that can be classified correctly by the initial IF–THEN rules) and the testing accuracy (the percentage of testing examples classified correctly) by using the globally weighted reasoning algorithm given in Section II, where all global weights are equal to 1.
3) Acquire the parameters (global weights) for all initial IF–THEN rules by using the parameter-refinement procedure proposed in Section III.
4) Calculate the training accuracy and the testing accuracy by using the globally weighted reasoning algorithm given in Section II, where all global weights are from Step 3.

The experimental results (averaged training and testing accuracies for the ten times) are summarized in Table I.

### D. Experimental Results and Analysis

From Table I, we can see that, for each dataset, the testing accuracy after the refinement is bigger than the testing accuracy before the refinement while keeping the training accuracy nondecreased. For example, the testing accuracy of glass identification data takes an average value as 0.68 before the refinement and 0.74 after the refinement, which makes an increase of 6%. Meanwhile, the testing accuracy of thyroid grand data is also increased on average by 5%, etc. The magnitude of increase is different, depending on the individual databases. For example, the increase is little for the database wine. We analyzed the database of wine and found that there exist many points that are very similar but with different classes. Compared with the best results published in [16], we consider decision tree induction nonsuitable for this dataset.

Moreover, one can learn that, before the refining, the testing accuracy of sonar signal data is 0.75, although the training accuracy is 0.85, which implies that the over-fitting phenomenon has happened. After incorporating global weights into the if-then rules and refining them according to the maximum fuzzy entropy principle, the testing accuracy has been greatly improved. This means that our method can avoid the over-fitting phenomenon to a great extent. Finally, we can pay attention to the fact that the training accuracy of all databases is not reduced after the refinement, which is guaranteed by the constraints given in (15).

*Remark:* This technique first requires a fundamental rule-generation algorithm-A. The outputs of the algorithm-A are a set of rules with parameters. In this paper, the algorithm-A is selected as the basic fuzzy decision induction that is a simple approach to fuzzy rule generation without further data preprocessing and genetic optimization. It is possible that the performance of algorithm-A on some specific datasets is worse than the results published such as in [16], [19], and [46]. This technique then applies a parametric refinement scheme to refine outputs of algorithm-A. After the refinement, we expect a significant improvement in learning accuracy. Mainly, this paper focuses on this improvement. However, it is possible that our results after the refinement are still worse than the results published such as in [16], [19], and [46] on some specific datasets. The reason is that our results are strongly dependent of the initially selected algorithm-A. If the published results (generated rules) are considered as the outputs of our algorithm-A, we may get the better results than (at least not worse than) the published ones after the parametric refinement.

The focus of this paper is the improvement of performance before and after refining algorithm-A (not the algorithm-A itself). It is worth noting that the algorithm-A can be anyone that can generate fuzzy IF–THEN rules from data.

### E. Experimental Comparison Between Two Approaches to Weight Refinement Based on the Fuzzy Entropy Maximum and Training Error Reduction

This section provides an experimental comparison between two approaches to weight refinement. One is the entropy-maximum-based approach proposed in this paper, while the other is the training error-reduction-based approach, which is realized by genetic algorithm (GA) [45]. A brief introduction for GAs is placed in the Appendix.

In our experiments, the fitness function $f(g_1, g_2, \ldots, g_m)$ is defined as the training accuracy with respect to the weight parameters. The evaluation of $f(g_1, g_2, \ldots, g_m)$ is described as follows.

Given weights $g_1, g_2, \ldots, g_m \Rightarrow$ A set of weighted fuzzy IF–THEN rules $\Rightarrow$ A training set and the globally weighted reasoning algorithm $\Rightarrow$ Training accuracy $f(g_1, g_2, \ldots, g_m)$.

Following Table I, the experimental results are summarized in Table II, where the population size is $M = 500$, $\Omega = [0, 1]^m$, and the mutation probability is $p_m = 0.01$. It is worth noting that, since $f(1, 1, \ldots, 1)$ denotes the initial training accuracy, we suppose that the point $(1, 1, \ldots, 1)$ is within the population of every generation. It implies that the evolution process will not lead to a decrease of accuracy.

TABLE II
EXPERIMENTAL RESULT OF WEIGHT-REFINEMENT METHOD BASED ON TRAINING ERROR REDUCTION FOR THE SELECTED NINE DATABASES

| Databases | Number of classes | Training / testing before refinement | Training / testing after refinement |
|---|---|---|---|
| Iris plant data | 3 | 0.95 0.92 | 0.97 0.93 |
| Mango leaf data | 3 | 0.83 0.79 | 0.87 0.80 |
| Wine data | 3 | 0.90 0.78 | 0.94 0.79 |
| Segmentation data | 7 | 0.80 0.76 | 0.85 0.71 |
| Thyroid grand data | 3 | 0.83 0.80 | 0.87 0.81 |
| Auto-mpg data | 2 | 0.73 0.70 | 0.76 0.72 |
| Sonar signal data | 2 | 0.85 0.75 | 0.88 0.71 |
| Pima diabetes data | 2 | 0.73 0.71 | 0.78 0.69 |
| Glass identification | 2 | 0.70 0.68 | 0.74 0.70 |

TABLE III
RULE POSTPRUNING APPROACH

| Databases | # of preconditions | | Training/testing accuracy | |
|---|---|---|---|---|
| | Before pruning | After pruning | Before prunning | After prunning |
| Auto-mpg | 49 | 42 | 0.727 0.698 | 0.748 0.721 |
| Pima | 66 | 57 | 0.731 0.712 | 0.745 0.734 |

TABLE IV
OUR PROPOSED APPROACH

| Databases | # of preconditions | Training / testing before refinement | Training / testing after refinement |
|---|---|---|---|
| Auto-mpg | 49 | 0.727 0.698 | 0.753 0.730 |
| Pima | 66 | 0.731 0.712 | 0.764 0.751 |

Comparing the last column of Tables I and II, we find that, with respect to the testing accuracy (i.e., the generalization capability), the approach based on fuzzy entropy maximum is superior to the approach based on the training error reduction. From the last column of Table II, one can see that, for databases sonar signal data and Pima diabetes data, the over-fitting phenomenon has already occurred.

Over fitting is a well-known problem in the neural networks-related literature. Normally, there are two methods to overcome the over-fitting problem in feed-forward neural network training [48]. One is the weight-decay training, and the other is to provide a set of validation data while training.

Our proposed method is a rule-based technique that efficiently prevents the over fitting while refining the rule weights. It is quite different from the neural-network-based techniques. To conduct a brief comparison for the two different techniques, a medical image database of computerized tomography (CT) diagnosis is selected. The database has 16 features and 290 cases with normal and abnormal classes.

The result of comparison shows that each of them can overcome the overfitting. Our rule-based method needs to 1) determine a number of linguistic terms for each attribute; 2) generate a decision tree; 3) convert the tree into a set of fuzzy rules; and 4) solve a quadratic programming problem for parameter refinement while the NN-based method needs to use gradient descent technique (with weight decay constraints) to train. With respect to the computational complexity, the experiment shows that the running time of our approach (132 s) is slightly longer than the NN-based approach (119 s) for this dataset. The computational complexity for solving the quadratic programming problem (12)–(15) needs to reduce for large databases.

Without a theoretical comparison, it is hard to say which technique is significantly better than the other for preventing over-fitting problems.

## *F. Experimental Comparisons for Balancing the Training Accuracy and Generalization*

Since it is widely accepted that the training error-minimization method is not enough to optimize classifiers, an additional popular approach to avoiding the overfitting in decision tree induction, i.e., the rule postpruning [48], is selected in this section for comparison with our proposed approach. The method of rule postpruning is briefly described as follows:

1) fuzzifying the continuous attributes to get a number of linguistic terms;
2) for a given training set, choosing ID3 as a heuristic to grow the tree until the training data is fit as well as possible;
3) converting the learned tree into an equivalent set of fuzzy IF–THEN rules by creating one rule for each path from the root to a leaf;
4) pruning each rule by removing any preconditions that result in increasing its learning accuracy;
5) after pruning, the remaining rules are considered as a new classifier.

The nine selected databases are again used for the comparison. The experimental procedure is described in Section IV-C for our proposed approach and is described earlier for the rule postpruning approach. The experimental results show that, for improving the learning accuracy and preventing over fitting, our proposed approach is slightly superior to the rule postpruning approach, which is slightly superior to the training error-reduction-based approach. However, for the computational complexity, the rule postpruning approach is the best. To some extent, it means that our achievements for accuracy improvement are at the price of increasing computational complexity.

Tables III and IV, where the number of preconditions is the average value of cross validation, show the experimental results for the Pima diabetes data set and the auto-mpg data.

Another experimental comparison is conducted between our proposed approach and the SEE5, which is a popular commercial rule-generation software in which two parameters (the interval center and the leaf-standard) are used to balance the training accuracy and the generalization. SEE5 is the extended version of C4.5 [49] and generates crisp IF–THEN rules. The parameter of leaf-standard in SEE5 has the following features. If the leaf-standard is very high, then the rules generated will be many, and the training accuracy will be high, which possibly leads to an overfitting. If the leaf-standard is low, then the rules

TABLE V
COMPARISON WITH SEE5

| Databases | Training / testing before refinement | Training / testing after refinement |
|---|---|---|
| Iris plant data | 0.97 0.93 | 0.97 0.94 |
| Mango leaf data | 0.90 0.85 | 0.95 0.87 |
| Wine data | 0.98 0.94 | 0.98 0.94 |
| Segmentation data | 0.90 0.89 | 0.95 0.92 |
| Thyroid grand data | 0.98 0.96 | 0.98 0.96 |
| Auto-mpg data | 0.88 0.75 | 0.88 0.77 |
| Sonar signal data | 0.86 0.78 | 0.87 0.79 |
| Pima diabetes data | 0.83 0.76 | 0.83 0.77 |
| Glass identification | 0.84 0.73 | 0.85 0.76 |

generated will be few, and the training accuracy will be low. In SEE5, this parameter has been optimized by some optimization techniques to balance the training accuracy and the rule number (generalization), and users need not to set up the value of this parameter. So far, the rules generated by SEE5 have been considered to be one of the best results for an inductive learning problem.

The antecedents of rules generated by SEE5 consist of a number of propositions such as $(a \le x \le b)$, where $x$ is a numerical attribute. In order to compare it with our proposed refinement approach, we need to fuzzify such propositions. The proposition $(a \le x \le b)$ is fuzzified as a triangular membership function:

$$f(x) = \begin{cases} 1 - \dfrac{1}{2}\dfrac{2x-a-b}{b-a}, & \text{if } \dfrac{a+b}{2} \le x \le \dfrac{3b-a}{2} \\ 1 - \dfrac{1}{2}\dfrac{a+b-2x}{b-a}, & \text{if } \dfrac{3a-b}{2} \le x \le \dfrac{a+b}{2} \\ 0, & \text{otherwise.} \end{cases}$$

The main idea for the previous fuzzification is that the membership in the support area is greater than or equal to 0.5.

The experimental results are shown in Table V. The results imply that SEE5 is preventing overfitting by setting up a relatively low leaf-standard. It is difficult for SEE5 to find a boundary value of leaf-standard so that the over fitting will occur once the boundary is exceeded. Furthermore, Table V shows that the parameter leaf-standard in SEE5, which balances the training accuracy and the generalization, is not the best. By using our proposed approach, the generalization capability can be further improved while keeping the rule number unchanged.

## V. CONCLUSION

Improving the generalization capability of fuzzy if-then rules extracted from training sets is very important for a rule-based classification system. This paper proposes an approach to the refinement of parametric fuzzy if-then rules based on fuzzy entropy maximization, rather than based only on the training error reduction. Its main features can be described as follows.

1) The uncertainty information that may be lost in the classification process can be sufficiently utilized by maximizing the fuzzy entropy.
2) The hard classification (partition) can be softened by maximizing the fuzzy entropy.
3) While improving the testing accuracy and keeping the simplicity of rules, the proposed approach can effectively avoid the overfitting due to the use of maximum entropy.

## APPENDIX

## GENERAL PROCEDURE OF A GA FOR OPTIMIZATION OF FUNCTIONS

In this Appendix, we briefly recall the general procedure of a GA for the optimization of functions. Let $f\,(x_1, x_2, \ldots, x_n)$ be a real function defined on $\Omega \subset R^n$. The function $f$ can be very complicated and noncontinuous, but $f$ is bounded. The GA is usually used to solve the following optimization problem:

$$\mathrm{Max}_{(x_1, x_2, \ldots, x_n) \in \Omega} f\,(x_1, x_2, \ldots, x_n)\,.$$

The function $f$ is called fitness function in the GA. We now briefly recall the GA as follows.

Step 1. *Initializing a population:* This step first requires us to specify a parameter $M$ (the population size) and then randomly select $M$ $n$-dimensional vectors in $\Omega$. In our experiments, $M = 500$. Let $G$ denote the population. Each element of $G$ is called a chromosome.

Step 2. *Coding each chromosome:* For each $(x_1, x_2, \ldots, x_n) \in G$, the binary code is used. It means that each chromosome is a 0–1 string.

Step 3. *Creating new chromosomes by mating current chromosomes:* This step is completed by reproduction, crossover, and mutation (where the crossover rate is 1.00 and the mutation probability is 0.01).

Step 4. *Generating the population of next generation:* The $M$ parents and their $M$ children are placed together to form a set of $2M$ chromosomes. From this set, we choose the first $M$ chromosomes with highest fitness values as the population of next generation.

Step 5. If the evolution attains a specified number of generations, then stop and return the best chromosome, else go to step 3.

## REFERENCES

[1] D. Ruan and E. Kerre, Eds., *Fuzzy IF–THEN Rules in Computational Intelligence*. Norwell, MA: Kluwer, 2000.
[2] H. Ishibuchi, K. Nozaki, and H. Tanaka, "Distributed representation of fuzzy rules and its application to pattern classification," *Fuzzy Sets Syst.*, vol. 52, no. 1, pp. 21–32, Nov. 1992.
[3] S. Abe and M.-S. Lan, "A method for fuzzy rules extraction directly from numerical data and its application to pattern classification," *IEEE Trans. Fuzzy Syst.*, vol. 3, no. 1, pp. 18–28, Feb. 1995.
[4] 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, no. 3, pp. 251–270, Mar. 1997.
[5] X.-Z. Wang, Y.-D. Wang, X.-F. Xu, W.-D. Ling, and D.-S. Yeung, "A new approach to fuzzy rule generation: Fuzzy extension matrix," *Fuzzy Sets Syst.*, vol. 123, no. 3, pp. 291–306, Nov. 2001.

[6] C. Huang and C. Moraga, "Extracting fuzzy if-then rules by using the information matrix technique," *J. Comput. Syst. Sci.*, vol. 70, no. 1, pp. 26–52, Feb. 2005.

[7] Y. Yuan and M. Shaw, "Induction of fuzzy decision trees," *Fuzzy Sets Syst.*, vol. 69, no. 2, pp. 125–139, Jan. 1995.

[8] X.-Z. Wang, E.-C.-C. Tsang, and D.-S. Yeung, "A comparative study on heuristic algorithms for generating fuzzy decision trees," *IEEE Trans. Syst., Man, Cybern., B*, vol. 31, no. 2, pp. 215–226, Apr. 2001.

[9] N. Kasabov, "Learning fuzzy rules and approximate reasoning in fuzzy neural networks and hybrid systems," *Fuzzy Sets Syst.*, vol. 82, no. 2, pp. 135–149, Sep. 1996.

[10] D. Nauck and R. Kruse, "A neuro-fuzzy method to learn fuzzy classification rules from data," *Fuzzy Sets Syst.*, vol. 89, no. 3, pp. 77–288, Aug. 1997.

[11] E.-C.-C. Tsang, X.-Z. Wang, and D.-S. Yeung, "Improving learning accuracy of fuzzy decision trees by hybrid neural networks," *IEEE Trans. Fuzzy Syst.*, vol. 8, no. 5, pp. 601–614, Oct. 2000.

[12] V. Uebele, S. Abe, and M.-S. Lan, "A neural network based fuzzy classifier," *IEEE Trans. Syst., Man, Cybern.*, vol. 25, no. 2, pp. 353–361, Feb. 1995.

[13] S. Mitra and L. I. Kuncheva, "Improving classification performance using fuzzy MLP and two level selective partitioning of the feature space," *Fuzzy Sets Syst.*, vol. 70, no. 1, pp. 1–13, Feb. 1995.

[14] S. Mitra and Y. Hayashi, "Neuro-fuzzy rule generation: Survey in soft computing framework," *IEEE Trans. Neural Netw.*, vol. 11, no. 3, pp. 748–768, May 2000.

[15] P.-P. Angelov and R.-A. Buswell, "Automatic generation of fuzzy rule-based models from data by genetic algorithms," *Inf. Sci.*, vol. 150, no. 1–2, pp. 17–31, Mar. 2003.

[16] J.-A. Roubos, M. Setnes, and J. Abonyi, "Learning fuzzy classification rules from labeled data," *Inf. Sci.*, vol. 150, pp. 77–93, 2003.

[17] A. Gonzalez and R. Perez, "SLAVE: A genetic learning system based on an iterative approach," *IEEE Trans. Fuzzy Syst.*, vol. 7, no. 2, pp. 176–191, Apr. 1999.

[18] H. Ishibuchi, T. Murata, and I. B. Turksen, "Single-objective and two-objective genetic algorithms for selecting linguistic rules for pattern classification problems," *Fuzzy Sets Syst.*, vol. 89, no. 2, pp. 135–149, Jul. 1997.

[19] H. Ishibuchi and T. Yamamoto, "Fuzzy rule selection by multi-objective genetic local search algorithms and rule evaluation measures in data mining," *Fuzzy Sets Syst.*, vol. 141, pp. 59–88, 2004.

[20] T.-P. Hong, T.-T. Wang, S.-L. Wang, and B.-C. Chien, "Learning a coverage set of maximally general fuzzy rules by rough sets," *Exp. Syst. Appl.*, vol. 19, no. 2, pp. 97–103, Aug. 2000.

[21] Y.-C. Hu, R.-S. Chen, and G.-H. Tzeng, "Finding fuzzy classification rules using data mining techniques," *Pattern Recogn. Lett.*, vol. 24, pp. 509–519, 2003.

[22] H. Ishibuchi and T. Yamamoto, "Heuristic extraction of fuzzy classification rules using data ming techniques: An empirical study on benchmark data sets," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, 25–29, Jul. 2004, vol. 1, pp. 161–167.

[23] F. Guély, R. La, and P. Siarry, "Fuzzy rule base learning through simulated annealing," *Fuzzy Sets Syst.*, vol. 105, no. 3, pp. 353–363, Aug. 1999.

[24] M.-A. Kbir, H. Benkirane, K. Maalmi, and R. Benslimane, "Hierarchical fuzzy partition for pattern classification with fuzzy if-then rules," *Pattern Recogn. Lett.*, vol. 21, no. 6–7, pp. 503–509, Jun. 2000.

[25] J.-L. Castro, L.-D. Flores-Hidalgo, C.-J. Mantas, and J.-M. Puche, "Extraction of fuzzy rules from support vector machines," *Fuzzy Sets Syst.*, vol. 158, no. 18, pp. 2057–2077, 2007.

[26] J. Zhang, S. Köper, and A. Knoll, "Extracting compact fuzzy rules based on adaptive data approximation using B-splines," *Inf. Sci.*, vol. 142, no. 1–4, pp. 227–248, May 2002.

[27] H. Ishibuchi and T. Nakashima, "Effect of rule weights in fuzzy rule-based classification systems," *IEEE Trans. Fuzzy Syst.*, vol. 9, no. 4, pp. 506–515, Aug. 2001.

[28] H. Ishibuchi and T. Yamamoto, "Rule weight specification in fuzzy rule-based classification systems," *IEEE Trans. Fuzzy Syst.*, vol. 13, no. 4, pp. 428–435, Aug. 2005.

[29] R. Ramakrishnan and C.-J.-M. Rao, "The fuzzy weighted additive rule," *Fuzzy Sets Syst.*, vol. 46, pp. 177–187, 1992.

[30] K.-S. Park and S.-H. Kim, "A note on the fuzzy weighted additive rule," *Fuzzy Sets Syst.*, vol. 77, pp. 315–320, 1996.

[31] A. Blanco, M. Delgado, and I. Requena, "A learning procedure to identify weighted rules by neural networks," *Fuzzy Set Syst.*, vol. 69, pp. 29–36, 1995.

[32] M.-J. Zolghadri and E.-G. Mansoori, "Weighting fuzzy classification rules using receiver operating characteristics (ROC) analysis," *Inf. Sci.*, vol. 177, pp. 2296–2307, 2007.

[33] P. Lau and C.-W. Chan, "An enhanced weighted fuzzy reasoning algorithm for engineering applications," *Comput. Math. Appl.*, vol. 34, no. 1, pp. 81–87, 1997.

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

[35] E.-C.-C. Tsang, D.-S. Yeung, J.-W.-T. Lee, D.-M. Huang, and X.-Z. Wang, "Refinement of generated fuzzy production rules by using a fuzzy neural network," *IEEE Trans. Syst., Man Cybern., B*, vol. 34, no. 1, pp. 409–418, Feb. 2004.

[36] H.-D. Cheng, Y.-H. Chen, and X.-H. Jiang, "Thresholding using two-dimensional histogram and fuzzy entropy principle," *IEEE Trans. Image Process.*, vol. 9, no. 4, pp. 732–735, Apr. 2000.

[37] W.-B. Tao, J.-W. Tian, and J. Liu, "Image segmentation by three-level thresholding based on maximum fuzzy entropy and genetic algorithm," *Pattern Recogn. Lett.*, vol. 24, no. 16, pp. 3069–3078, Dec. 2003.

[38] D. Dubois and H. Prade, Eds., "Fundamentals of fuzzy sets," in *The Handbooks of Fuzzy Sets,* vol. 7. Boston, MA: Kluwer, 2000.

[39] D.-S. Yeung, X.-Z. Wang, and E.-C.-C. Tsang, "Handling interaction in fuzzy production rule reasoning," *IEEE Trans. Syst., Man Cybern., B*, vol. 34, no. 5, pp. 1979–1987, Oct. 2004.

[40] C.-E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech.*, vol. 27, pp. 379–423 and 623–656, Jun, 1948.

[41] A.-D. Luca and S. Termin, "A definition of a nonprobabilistic entropy in the setting of fuzzy sets theory," *Int. J. Control*, vol. 20, pp. 301–312, 1972.

[42] UCI repository of machine learning databases and domain theories, FTP address: [Online]. Available: ftp://ftp.ics.uci.edu/pub/machine-learning-databases.

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

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

[45] M.-L. Raymer, W.-F. Punch, E.-D. Goodman, L.-A. Kuhn, and A.-K. Jain, "Dimensionality reduction using genetic algorithms," *IEEE Trans. Evol. Comput.*, vol. 4, no. 2, pp. 164–171, Jul. 2000.

[46] P. Angelov, X. Zhou, and F. Klawonn, "Evolving fuzzy rule-based classifiers," in *Proc. 1st 2007 IEEE Int. Conf. Comput. Intell. Appl. Signal Image Process.*, Honolulu, HI, Apr. 1–5, pp. 220–225.

[47] M. Setnes and H. Roubos, "GA-fuzzy modeling and classification: Complexity and performance," *IEEE Trans. Fuzzy Syst.*, vol. 8, no. 5, pp. 509–522, Oct. 2000.

[48] T. M. Mitchell, *Machine Learning*. New York: McGraw-Hill, 1997.

[49] J. R. Quinlan, *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann, 1993.

**Xi-Zhao Wang** (A'04–M'04–SM'04) received the B.Sc. and M.Sc. degrees in mathematics from Hebei University, Baoding, China, in 1983 and 1992, respectively, and the Ph.D. degree in computer science from Harbin Institute of Technology, Harbin, China, in 1998.

From 1983 to 1998, he was a Lecturer, an Associate Professor, and a Full Professor with the Department of Mathematics, Hebei University, where, since 2001, he has been the Dean and Professor of the Faculty of Mathematics and Computer Science. From 1998 to 2001, he was a Research Fellow with the Department of Computing, Hong Kong Polytechnic University, Kowloon, Hong Kong. His current research interests include inductive learning with fuzzy representation, fuzzy measures and integrals, neuro-fuzzy systems and genetic algorithms, feature extraction, multiclassifier fusion, and applications of machine learning. He is the author or coauthor of more than 80 international journal papers, has completed over 20 funded research projects, and has supervised over 50 Doctors or Masters theses.

Prof. Wang is an Associate Editor of the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS, PART B and the *International Journal of Information Sciences*, the Chair of the IEEE Systems, Man, and Cybernetics (SMC) Baoding Chapter, IEEE SMC Technical Committee on Computational Intelligence, and the IEEE SMC Technical Committee on Computational Intelligence, and an Executive Member of Chinese Association of Artificial Intelligence. He is the General Co-Chair of the 2002, 2003, 2004, 2005, 2006, and 2007 International Conference on Machine Learning and Cybernetics, which is co-sponsored by IEEE SMC.

**Chun-Ru Dong** received the Bachelors' degree in computational mathematics and the Masters' degree in computer science from Hebei University, Baoding, China, in June 2002 and June 2005, respectively.

He is currently a Lecturer with the Department of Mathematics and Computer Science, Hebei University. His current research interests include inductive learning, fuzzy reasoning, and neuro-fuzzy systems.