Efficient fuzzy rule generation:
A new approach using data mining principles and rule weighting

O. Dehzangi
Department of Computer Engineering, Islamic Azad University of Marvdasht, Marvdasht , Iran
dehzangi@cse.shirazu.ac.ir

M. J. Zolghadri
Department of Computer Science and Engineering, Shiraz university, Shiraz, Iran
zjahromi@shirazu.ac.ir

S. Taheri
Department of Computer Engineering, Islamic Azad University of Marvdasht, Marvdasht, Iran
Shahram_taheri@cse.shirazu.ac.ir

S.M. Fakhrahmad
Department of Computer Engineering, Islamic Azad University of Shiraz, Shiraz, Iran
fakhrahmad@cse.shirazu.ac.ir

Abstract

Classification Systems have been widely applied in different fields such as medical diagnosis. A fuzzy rule-based classification system (FRBCS) is one of the most popular approaches used in pattern classification problems. One advantage of a fuzzy rule-based system is its interpretability. However, we're faced with some challenges when generating the rule-base. In high dimensional problems, we can not generate every possible rule with respect to all antecedent combinations. In this paper, by making the use of some data mining concepts, we propose a method for rule generation, which can result in a rule-base containing rules of different lengths. Then, our rule learning algorithm based on R.O.C analysis tunes the rule-base to have better classification ability. Our goal in this article, is to check if generating cooperative rule-bases containing rules of different dimensions, can lead to better generalization ability. To evaluate the performance of the proposed method, a number of UCI-ML data sets were used. The results show that considering cooperation in a rule-base tuned by rule weighting process can improve the classification accuracy. It is also shown that increasing the maximum length of rules in the initial rule-base, improves the classification accuracy.

Keywords: *Pattern classification, fuzzy systems, data mining, rule weighting, Noise Removal*

1. Introduction

Fuzzy rule-based systems have been widely used on control problems [1,2,3]. One key feature of fuzzy rule-based systems is their comprehensibility because each fuzzy rule is linguistically interpretable. Recently, fuzzy rule-based systems have been applied successfully on classification problems [4,5,6]. The interest in using fuzzy rule-based classification systems (FRBCS) arises from the fact that those systems consider both accuracy and comprehensibility of the classification result at the same time [7,8,9,10].

Basic idea for designing a FRBCS is to automatically generate fuzzy rules from numeric data (i.e., a number of pre-labeled training examples). Hence, rule-base construction for a classification problem always has been a challenging part of it. In this paper, a novel approach for generating a set of candidate rules of each class is presented using data mining principles in which the number of generated rules is reduced dramatically. A compact rule-base is then constructed by selecting a specified number of candidate rules from each class (using a selection metric).

In many studies, antecedent fuzzy sets were generated and tuned by numerical input data for rule-base construction to improve the classification accuracy of FRBCSs [11,12]. As shown in [13,14], the modification of the membership functions of antecedent fuzzy sets can be replaced by rule weight specification to some extent. Since, the adjustment of membership functions may degrade the interpretability of a FRBCS. In this paper, a learning algorithm is proposed to adjust the weights of the rules (existing in the rule-base) by the training data. This method attends to improve the generalization of FRBCS by minimizing the classification error rate on the training data.

The rest of this paper is organized as follows. In Section 2, a FRBCS is briefly introduced. In Section 3, the process of rule-base construction and the proposed method of generating rules with different lengths is described. Section 4 is devoted to introduction of the proposed method of rule weight learning. In Section 5, the computational experiments are shown. Section 6 concludes the paper.

2. Fuzzy rule-based classification systems

Various methods have been introduced for fuzzy classification [15,16,17]. Let us assume that we have m training patterns $X_p = (x_{p1},\ldots,x_{pn})$, $p = 1,2,\ldots,m$ from M different classes where X_p is an n-dimensional vector of attributes in which x_{pi} is the i-th attribute value of the p-th training pattern ($i = 1,2,\ldots,n$). For our M-class, n-dimensional classification problem, we use fuzzy if-then rules of the form below [18]:

Rule R_q: If x_1 is A_{q1} and … and x_n is A_{qn} then class C_q with CF_q (1)

, where R_q is the label of the q-th fuzzy if-then rule, $X = (x_1,\ldots,x_n)$ is n-dimensional vector of a pattern, presents an antecedent fuzzy set, C_q is a class label,

Fourth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2007)
0-7695-2874-0/07 $25.00 © 2007 IEEE

CF_q is the weight assigned to the q-th rule. In [19], fuzzy rules of other types are introduced.

To calculate the compatibility grade of each training pattern X_p with the antecedent part of the rule $\mathbf{A}_q$ = $(A_{q1},\ldots,A_{qn})$, we use the product operator as follows:

$$\mu_{A_q}(\boldsymbol{x}_p) = \mu_{A_{q1}}(x_{p1}) \cdot \mu_{A_{q2}}(x_{p2}) \cdot \ldots \cdot \mu_{A_{qn}}(x_{pn}), p = 1,2,\ldots,m \quad (2)$$

, where $\mu_{Aqi}(x_{pi})$ is the compatibility grade of x_{pi} with fuzzy membership function A_{qi}. To determine the consequent class of the q-th rule C_q, we measure the confidence degree of the association rule "$\mathbf{A}_q \Rightarrow$ Class h" from the field of data mining for each class, where $\mathbf{A}_q$ is a multi-dimensional fuzzy set representing the antecedent conditions and h is a class label. Confidence of a fuzzy association rule R_q is defined as follows [20,21]:

$$c(A_q \Rightarrow \text{Class h}) = \sum_{X_p \in \text{Class h}} \mu_{A_q}(\boldsymbol{x}_p) \Big/ \sum_{p=1}^{m} \mu_{A_q}(\boldsymbol{x}_p), h = 1,2,\ldots,M \quad (3)$$

, where $\mu_{Aq}(X_p)$ is the compatibility grade of pattern X_p with the antecedent part of the rule R_q, m is the number of training patterns and C_q is a class label. The class with maximum confidence degree is identified to determine the consequent class C_q:

$$q = \text{argmax}\{c(\text{A}_q \Rightarrow \text{Class h}) \mid h = 1, 2, \ldots, M\} \quad (4)$$

An input pattern is classified regarding to the consequent class of the winner rule. By using rules of the form (1), a weight assigned to each rule is used to find the winner rule. Rule weighting has a profound effect on the classification ability of FRBCSs [22]. In [23], several methods of rule weighting have been introduced. In this paper, we use a learning mechanism to find the weight of each rule. The winner rule R_w is chosen for the input pattern X_t in the following manner:

$$\mu(x_t) \cdot CF_w = \max\{\mu_j(x_t) \cdot CF_j \mid R_j, j = 1, 2, \ldots, N\} \quad (5)$$

Note that the classification of a pattern not covered by any rule in the rule-base is rejected. The classification of a pattern X_t is also rejected if two rules with different consequent classes have the same value of $\mu(X_t).CF$ in equation (4).

3. Rule-base construction

For an M-class problem in an n-dimensional feature space, assume that m labeled patterns X_p=[x_{p1}, x_{p2}, …, x_{pn}], p=*1*, *2*, …, m from M classes are given. A simple approach for generating fuzzy rules is to partition the domain interval of each input attribute using a pre-specified number of fuzzy sets (i.e., grid partitioning). Some examples of this partitioning (using triangular membership functions) are shown in Figure 1.

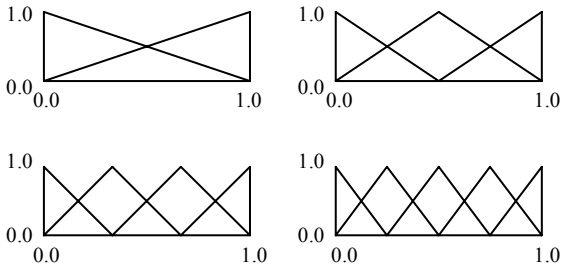

Figure1. Different partitioning of each feature axis.

Given a partitioning of pattern space, one approach is to consider every possible combination of antecedents to generate the fuzzy rules. The problem with grid partitioning is that an appropriate partitioning of each attribute is not usually known. One solution is to simultaneously consider different partitions, as shown in Figure 1. That is, for each attribute, one of the 14 fuzzy sets shown in Figure1 can be used when generating a fuzzy rule. The problem is that for an n-dimensional problem, 14^n antecedent combinations have to be considered. It is impractical to consider such a huge number of antecedent combinations when dealing with high dimensional problems.

One solution for the above problem is presented in [24] by adding the fuzzy set "don't care" to each attribute. The membership function of this fuzzy set is defined as $\mu_{don't\ care}(x) = 1$ for all values of x. The trick is not to consider all antecedent combinations (which is now 15^n) and only short fuzzy rules having a limited number of antecedent conditions are generated as candidate rules. For example, fuzzy rules having only two or less antecedent fuzzy sets (excluding don't care) are investigated.

By increasing the number of antecedents, the rule set grows dramatically. Moreover, within a very large number of rules, usually a small fraction of rules are acceptable. Thus, in many cases, a considerable time is devoted to useless computations. The purpose of the solution presented in this paper, is to avoid the exponential growth of the rule sets in each step. In this approach, we do not generate rules that are hardly probable to be interesting. The method is based over two data mining principles, used for mining frequent item sets:

1) Increasing the length of an item set, the support value will not improve.
2) A set of n items is probable to be frequent (have a good support), if and only if all of its subsets of size n-1 are frequent (the Apriori principle) [25].

In this work, we observe them from a different viewpoint and use them to find fuzzy rules (not item sets) having good supports.

3-1-Generating rules with 1 or 2 antecedents

As mentioned before, a major purpose in this paper is to propose a solution that enables us to generate fu

rules with any number of antecedents. For this purpose, we consider the well-known evaluation measure, *Support* as the primary factor for rule filtering. In equation (6), a simple definition for the fuzzy aspect of the *Support* measure is presented.

$$s(A_j \Rightarrow \text{Class h}) = \frac{1}{m} \sum_{X_p \in \text{Class h}} \mu_{A_j}(x_p) \qquad (6)$$

, where $\mu_j(X_p)$ is the compatibility degree of X_p with the antecedent part of the rule R_j, m is the number of training patterns and h is a class label. After determining a minimum support threshold (denoted by *MinSupp*), a set of 1-dimensional rules (containing one antecedent), is generated. This set is then filtered by selecting only rules having a support value above the *MinSupp*. Combining the rules within this set in the next step, results in the set of 2-dimensional candidate rules. The reason of this issue (that we just combine rules having good supports) refers to the first principle mentioned in Section 2. Another key point in combination of a pair of 1-dimensional rules is the conditions under which the rules can be combined:
1) The rules must not contain similar antecedents on their left-hand sides. 2) The consequent classes of the two rules must be identical. Similarly, the resulting rule set is filtered with respect to the *MinSupp* value.

3-2- Generating rules of higher dimensions

In order to generate rules containing more than two antecedents, a similar procedure is followed. However, in this case, both of the principles (used for mining frequent item sets) must be regarded. Generating 3-dimensional rules is accomplished using the 1 and 2-dimensional candidate rules. Any possible combination of the rules from these two sets, having the same consequent and not containing common antecedents would be a 3-dimensional candidate rule. The second principle is used to avoid the time-consuming evaluation of some useless rules (which can not have high support values). A rule resulting from a combination will be evaluated only if all of its 2-dimensional sub-rules are present in the candidate set of the previous stage (i.e., all the sub-rules have good supports). Otherwise, we do not measure the support of the rule, since it can not be even a candidate rule. As an example, the support of the rule R: If X1 is A1 and X2 is A2 and X3 is A3 → C1 is computed only if all the following sub-rules have good supports:

If X1 is A1 and X2 is A2 → C1, If X1 is A1 and X3 is A3 → C1, If X2 is A2 and X3 is A3 → C1

Similarly, generating an n-dimensional rule is performed by the combination of n-1 and 1-dimensional candidate rules.

An important challenge here is to find a solution for this problem: How should we control the presence of all sub-rules in order to avoid efficiency reduction? Moreover, combining 1 and n-dimensional rules (even with respect to the mentioned conditions) may lead to some repeating combinations. Another challenging problem is how to avoid generating repeating rules, which could be so helpful to the efficiency of the process. To achieve these two goals, we make use of the efficiency of SQL and accomplish the primary phases of the rule generation process using this language.

In a relational database, for all rule sets containing 1,2,3, … antecedents, we create new tables, namely Rules_1D, Rules_2D, Rules_3D, … each having a similar structure, shown in Figure 2.

Field Name	Field Type	Description
antecedent	String	Left-Hand Side of the rule
consequent	String	Right-Hand Side of the rule
supp	Double	Rule's Support
conf	Double	Rule's Confidence
Fslave	Double	The Selection metric
IsAccepted	Boolean	Is the rule accepted?

Figure 2. Structure of the table used for holding rules

A table having the same structure is also created for the final rule-base, which will be a hybrid set containing rules of various dimensions. To facilitate the operation of parsing the rules (performed by SQL), we save each rule in a record having a unified structure.

Following the above process, it will also be possible to generate rules having 4 and more antecedents, for any data set having arbitrary number of features.

Although the set of rules is pruned to some extent, in some cases the number of rules is still large. This problem gets more sensible as we increase the number of antecedents. In order to obtain a more concise data set, we divide the set of candidate rules into M distinct groups, according to their consequents (M is the number of classes). The rules in each group are sorted by an appropriate evaluation factor and the final rule-base is constructed by selecting p rules from each class, i.e., in total, $M.p$ rules are selected. Many evaluation measures have already been proposed [26]. In this work, we use the measure proposed in [27] as the rule selection metric, which evaluates the rule $A_j \Rightarrow class\ C_j$ through the following equation:

$$e(R_j) = \sum_{X_p \in \text{Class } C_j} \mu_{A_q}(\boldsymbol{x}_p) - \sum_{X_p \notin \text{Class } C_j} \mu_{A_q}(\boldsymbol{x}_p) \qquad (7)$$

4. Learning rule weights

For an M-class problem, assume that a rule-base consisting of N fuzzy classification rules of the form (1) exists. This rule-base is constructed by generating candidate rules and then selecting them based on the criterion in (6) as mentioned in the previous section. In this section, we propose an algorithm to learn the weights of the rules using a set of labeled training patterns. The calculated weight is optimal in the sense that it maximizes the classification rate of the FRBCS

on training data. Initially, all rules are assumed to have a weight of one (i.e. CF_k=1, K=1,2,...,N). For easier discussion, consider the following rule as a typical rule in the rule-base for optimization.

R_k: If x_1 is A_{k1} and … and x_n is A_{kn} then class T with CF_k (8)

Where, $T \in [C_1, C_2 \ldots, C_M]$. The optimal weight of this rule (given the weights of all other rules) can be found as follows. At first, all the training patterns which are not covered by this rule (i.e., $\mu_k(X_t)$=0) are removed from the training set. Then, considered rule is removed from the rule-base (CF_k=0). Subsequently, training patterns of class T which are classified correctly are removed from the training set. Similarly, training patterns of other classes which are misclassified with the current rule-base are also removed from the training set. For each training pattern left in the training set, a score is calculated using the following measure:

$$S(x_t) = \frac{\max\{CF_j \cdot \mu_j(x_t) \mid Consequent(R_j) \neq \text{Class P}\}}{\mu_k(x_t)} \quad (9)$$

After assigning the score to each left pattern, the algorithm given in Table 1 can be used to find the best threshold for this rule.

The resulting threshold is considered as the weight of the rule and is used to find the winner rule in (5). In Table 1, the two following concepts are used for finding the best threshold:

TP[1]: number of patterns of class T which are classified correctly.

FP[2]: number of patterns not of class T which are misclassified.

That is, the proposed algorithm in Table 1 finds the best threshold by maximizing *Accuracy*= TP-FP.

The above procedure claculates the optimal weight of the rule R_k assuming that the weights of all other rules are given and fixed. The search for the best combination of rule weights is conducted by optimizing each rule in turn assuming that the order of the rules to be optimized is fixed.

It should be considered that the learning algorithm is sensetive to the order of the rules presented to the algorithm. For example, first few rules presented to the learning algorithm may have a very different probability to be prunned compared to those which are visited later. In simulation results reported in section 6, we fix the ordering using a fuzzy rule evaluation metric.

Table1. Algorithm for finding the best threshold

Inputs: patterns X_t, scores $S(X_t)$
Output: the value of best threshold (*best-th*)
  current = classification accuracy corresponding to the threshold of *th* = 0 (i.e., classifying everything as negative)
  optimum = *current*
  best-th = 0
  rank the patterns in ascending order of their scores
  {assume that X_k and X_{k+1} are two successive patterns in the list}
  for any threshold *th* = (*Score*(X_k)+*Score*(X_{k+1}))/2
    current = accuracy corresponding to the specified threshold (i.e., all patterns X_t having *Score*(X_t) < *th* are classified as positive)
    if *current* > *optimum* **then**
      optimum = *current*
      best-th = *th*
    end if
  end for
  {assume that *last* is the score of last pattern in the list and τ is a small positive number}
  current =accuracy corresponding to *th* = *last* + τ (i.e., classifying everything as positive)
  if *current* > *optimum* **then**
    optimum = *current*
    best-th = *th*
  end if
return *best-th*

5. Computational experiments

In order to evaluate the performance of the proposed method, we used the data sets shown in Table 2 available from UCI ML repository. To construct an initial rule-base for a specific problem, using the method explained in section 3, all the rules of length 1, 2, 3, and 4 (i.e. having 1, 2, 3, and 4 number of antecedent conditions excluding don't care) were generated.

Table 2. Some statistics of the data sets used in our computer simulations.

Data set	Number of attributes	Number of patterns	Number of Classes
Iris	4	150	3
Wine	13	178	3
Thyroid	5	215	3
Sonar	60	208	2
Bupa	6	345	2
Pima	8	768	2
Glass	9	214	6

In the preprocessing step for specifying noisy examples, number of effective neighbors for each pattern was determined as 3 after some trial experiments. To assess the generalization ability of the proposed method, we used 10CV technique which is a case of n-fold cross validation. A rule-base was then constructed by selecting 100 candidate rules from each class using (7) as the selection metric. The propo

[1] True Positive
[2] False Positive

method of rule weighting was then used to optimize each rule from the generated candidate rule-base. The results obtained by generating rules of maximum length 1 through 4 are shown in Table 3 through 6 respectively. As it can be observed, increasing the maximum length of generated candidate rules results in higher classification rates.

Table 3. The results achieved by generating rules of max-length 1

Data set	Accuracy (%)	# of rules
Iris	94.6	3.81
Wine	92.4	6.92
Thyroid	90.0	6.33
Sonar	74.5	12.33
Bupa	56.4	9.51
Pima	74.6	9.4
Glass	54.2	10.42

Table 4. The results achieved by generating rules of max-length 2

Data set	Accuracy (%)	# of rules
Iris	95.3	3.87
Wine	95.6	6.97
Thyroid	94.2	7.03
Sonar	79.8	14.38
Bupa	59.8	9.78
Pima	77.7	9.63
Glass	61.4	10.82

Table 5. The results achieved by generating rules of max-length 3

Data set	Accuracy (%)	# of rules
Iris	96.5	4.34
Wine	97.0	7.25
Thyroid	95.8	6.93
Sonar	80.6	15.76
Bupa	62.4	10.25
Pima	78.9	9.79
Glass	65.3	10.66

Table 6. The results achieved by generating rules of max-length 4

Data set	Accuracy (%)	# of rules
Iris	95.6	5.95
Wine	97.7	8.46
Thyroid	95.5	7.89
Sonar	82.2	18.21
Bupa	65.2	12.95
Pima	75.3	10.29
Glass	68.2	13.58

In Table 7, our proposed method is compared to another successful rule-based method as benchmark results called C4.5 reported by Elomaa and Rousu [28]. How ever, the comparison has been made on the available results of the C4.5 method. As shown in Table 6, except in one case, the proposed classifier in this paper shows higher classification rates.

Table 7. Accuracy Comparison of the proposed classifier with C4.5

Data set	The proposed classifier (%)	C4.5 classifier	
		Worst (%)	Best (%)
Iris	**95.6**	94.0	94.9
Pima	**75.3**	72.8	75.0
Sonar	**82.2**	67.4	76.7
Wine	**97.7**	92.2	94.4
Glass	68.2	68.8	**72.7**

6. Conclusion

In this paper, a novel method of rule-base construction using data mining principles and a rule weighting mechanism was proposed. Using the proposed method for rule generation, it will be possible to generate rules having different lengths, efficiently. It is much more useful when dealing with high dimensional data sets, were the existing methods are not able to generate rules containing more than 2 or 3 antecedent conditions. The proposed scheme achieves two other goals, too. Firstly, the classification rate is improved by adjusting rule weights. Secondly, redundant rules are removed during the learning process which results in a compact rule-base.

The proposed scheme also has a mechanism to cope with noisy training examples. This is accomplished by identifying and subsequently discarding noisy training examples. Using this high quality subset instead of the full training set helps to improve the generalization ability of the learned classifier.

For evaluation of the proposed method, 7 data sets from UCI ML repository were used. The experimental results on these data sets showed that the method can be used to construct a compact rule-base with a high classification accuracy. In comparison with C4.5 as an alternative rule-based method of classification, we showed that the proposed classifier is more accurate and results in higher classification rates.

8. References

[1] M. Arima, E. H. Hara, J. D. Katzberg, ″A fuzzy logic and rough sets controller for HVAC systems″, *Proceedings of the IEEE WESCANEX*, New York, 1995, pp. 133–138.

[2] O. Cordón, F. Herrera, A. Peregrín, ″Applicability of the fuzzy operators in the design of fuzzy logic controllers″, *Fuzzy Sets and Systems*, 1997, pp. 15–41.

[3] P. Y. Glorennec, ″Application of fuzzy control for building energy management. In: Building Simulation″, *International Building Performance Simulation Association* 1. Sophia Antipolis, France, pp. 197–201, 1991.

[4] A. Bárdossy, L. Duckstein, ″Fuzzy rule-based modeling with applications to geophysical″, *biological and engineering systems*, CRC Press,1995.

[5] J. C.Bezdek, S. K.Pal,, "Fuzzy Models for Pattern Recognition, Methods that Search for Structures in Data", *IEEE Press*, Boca Raton, 1992.

[6] Z. Chi, H. Yan, T. Pham, "Fuzzy algorithms with applications to image processing and pattern recognition", *World Scientific*, New York, 1996.

[7] Y. Jin, "Fuzzy Modeling of High-dimensional Systems: Complexity Reduction and Interpretability Improvement", *IEEE Trans. on Fuzzy Systems,* 2000, pp. 212-221.

[8] Jin, Y., Von Seelen, W., and Sendhoff, B.: On Generating FC3 Fuzzy Rule Systems from Data Using Evolution Strategies, *IEEE Trans. on Systems, Man and Cybernetics - Part B: Cybernetics* 29 (1999) 829-845.

[9] H. Roubos, and M. Setnes, "Compact and Transparent Fuzzy Models and Classifiers Through Iterative Complexity Reduction", *IEEE Trans. on Fuzzy Systems,* 2001, pp. 516- 24.

[10] M. Setnes, R. Babuska, and B. Verbruggen, "Rule-based Modeling: Precision and Transparency", *IEEE Trans. on Systems, Man, and Cybernetics - Part C: Applications and Reviews,* 1998, pp. 165-169.

[11] D. Nauck and R. Kruse, "A neuro-fuzzy method to learn fuzzy classification rules from data", *Fuzzy Sets and Systems*, 1997, pp. 277-288.

[12] S. Abe and R. Thawonmas, "A fuzzy classifier with ellipsoidal regions," *IEEE Trans. on Fuzzy Systems*, 1997, pp. 358-368.

[13] D. Nauck and R. Kruse, "How the learning of rule weights affects the interpretability of fuzzy systems," *Proc. of 7th IEEE International Conference on Fuzzy Systems*, 1998, pp. 1235-1240.

[14] H. Ishibuchi and T. Yamamoto, "Rule weight specification in fuzzy rule-based classification systems", *IEEE Trans. on Fuzzy Systems*, 2005, pp. 428-435.

[15] A. Gonzalez, R. Pérez, ″Completeness and consistency conditions for learning fuzzy rules", *Fuzzy Sets and Systems*, 1998, pp. 37–51.

[16] H. Ishibuchi, K. Nozaki and H. Tanaka, ″Distributed representation of fuzzy rules and its application to pattern classification", *Fuzzy Sets and Systems,* 1992, pp. 21–32.

[17] H. Ishibuchi, K. Nozaki, N. Yamamoto and H. Tanaka, ″Construction of fuzzy classification systems with rectangular fuzzy rules using genetic algorithms", *Fuzzy Sets and Systems,* 1994*, pp. 237–253*.

[18] H. Ishibuchi, K. Nozaki, and H. Tanaka, "Distributed Representation of Fuzzy Rules and Its Application to Pattern classification", *Fuzzy Sets and Systems,* 1992, pp. 21-32.

[19] H.Ishibuchi, T. Nakashima, and T. Morisawa, "Voting in Fuzzy Rule-Based Systems for Pattern Classification problems", *Fuzzy Sets and Systems,* 1999, pp. 223-238.

[20] T. P. Hong, C. S. Kuo, and S. C. Chi, "Trade-off between Computation Time and Number of Rules for Fuzzy Mining from Quantitative Data", *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 2001, pp. 587-604.

[21] H. Ishibuchi, T. Yamamoto, and T. Nakashima, "Fuzzy Data Mining: Effect of Fuzzy Discretization", *Proc. of 1st IEEE International Conference on Data Mining,* 2001, pp. 241-248.

[22] H. Ishibuchi, and T. Nakashima, "Effect of Rule Weights in Fuzzy Rule-Based Classification Systems", *IEEE Trans. on Fuzzy Systems,* 2001, pp. 506-515.

[23] H. Ishibuchi, and T. Yamamoto, "Comparison of Heuristic Rule Weight Specification Methods", *Proc. of 11th IEEE International Conference on Fuzzy Systems,* 2002, 908-913.

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

[25] R. Agrawal, R. Srikant, "Fast Algorithms for Mining Association Rules in Large Databases", *Proc. of the 20th nternational Conference on Very Large Data Bases*, 1994, pp. 487-499.

[26] H. Ishibuchi, T. Yamamoto, "Comparison of heuristic criteria for fuzzy rule selection in classification problems", Fuzzy *Optimization and Decision Making,* 2004, pp. 119-139.

[27] A. Gonzalez, R. Perez, "SLAVE: A genetic learning system based on an iterative approach", *IEEE Trans. on Fuzzy Systems*, 1999, pp. 176-191.

[28] T. Elomaa, and, J. Rousu, "General and efficient multisplitting of numerical Attributes", m*achine Learning*, 1999, pp. 201-244.