Ensemble Classifier Design by Parallel Distributed Implementation of Genetic Fuzzy Rule Selection for Large Data Sets

Yusuke Nojima, *Member, IEEE*, Shingo Mihara, and Hisao Ishibuchi, *Senior Member, IEEE*

***Abstract*—Evolutionary algorithms have been actively applied to knowledge discovery, data mining and machine learning under the name of genetics-based machine learning (GBML). The main advantage of using evolutionary algorithms in those application areas is their flexibility: Various knowledge extraction criteria such as accuracy and complexity can be easily utilized as fitness functions. On the other hand, the main disadvantage is their large computation load. It is not easy to apply evolutionary algorithms to large data sets. The scalability improvement to large data sets is one of the main research issues in GBML. In our former studies, we proposed an idea of parallel distributed implementation of GBML and examined its effectiveness for genetic fuzzy rule selection. The point of our idea was to realize a quadratic speed-up by dividing not only a population but also training data. Training data subsets were periodically rotated over sub-populations in order to prevent each sub-population from over-fitting to a specific training data subset. In this paper, we propose the use of parallel distributed implementation for the design of ensemble classifiers. An ensemble classifier is designed by combining base classifiers, each of which is obtained from each sub-population. Through computational experiments on parallel distributed genetic fuzzy rule selection, we examine the generalization ability of designed ensemble classifiers under various settings with respect to the size of training data subsets and their rotation frequency.**

I. Introduction

Fuzzy rule-based classifiers can be a promising knowledge representation framework for classification problems since each fuzzy rule is linguistically interpretable (e.g., If x_1 is *large* and x_2 is *small* then Class 1). Evolutionary algorithms have been frequently used for the design of fuzzy rule-based classifiers under the name of fuzzy genetics-based machine learning (fuzzy GBML), genetic fuzzy systems (GFS) and evolutionary fuzzy systems (EFS) [1, 2]. One hot research issue in the field of fuzzy GBML (and GBML in general) is the scalability improvement of evolutionary algorithms to large data sets. This is because even an evaluation of a single fuzzy rule-based classifier needs a long computation time in the case of large data sets. Since the execution of a fuzzy GBML algorithm involves tens of thousands of evaluations of fuzzy rule-based classifiers, its application to large data sets is very difficult.

This work was supported in part by the Japan Society for Young Scientist (B): KAKENHI (22700239).

Y. Nojima, S. Mihara, and H. Ishibuchi are with the Department of Computer Science and Intelligent Systems, Osaka Prefecture University, Sakai, Osaka 599-8531, JAPAN. (phone: +81-72-254-9198; fax: +81-72-254-9915; e-mail: nojima@cs.osakafu-u.ac.jp, mihara@ci.cs.osakafu-u.ac.jp, hisaoi@cs.osakafu-u.ac.jp).

One approach for decreasing the computation time of evolutionary algorithms is their parallel implementation [4, 5]. When we use an island model on a multi-core computer, the number of islands (i.e., the number of sub-populations) is usually the same as the number of CPU cores. Let N_{CPU} be the number of sub-populations (i.e., the number of CPU cores). The computation time of an evolutionary algorithm can be potentially decreased through its parallel implementation to $1/N_{\mathrm{CPU}}$ in comparison with its non-parallel implementation. In the fields of knowledge discovery, data mining and machine learning, data reduction such as feature and instance selection [6-9] is a well-known and frequently-used approach to the scalability improvement of knowledge extraction methods to large data sets.

In our former studies [10-12], we proposed an idea of parallel distributed implementation of genetic fuzzy rule selection. The point of our idea was to realize a quadratic speed-up by simultaneously utilizing the two scalability improvement approaches mentioned above. Specifically saying, we used parallel implementation of genetic fuzzy rule selection [13] together with data reduction. A population was divided into a number of sub-populations for parallel implementation while training data were divided into multiple training data subsets for data reduction. A single training data subset and a single sub-population were assigned to each CPU core in a multi-core CPU computer. In order to prevent each sub-population from over-fitting to a specific training data subset, training data subsets were periodically rotated over sub-populations (e.g., every 100 generations). It was shown in [10] that a parallel distributed implementation of genetic fuzzy rule selection decreased its computation time to 1/9 (i.e., $1/N_{\mathrm{CPU}}^2$) in comparison with its non-parallel implementation without any clear deterioration in the generalization ability of obtained fuzzy rule-based classifiers. Moreover, we proposed the use of very small training data subsets in our parallel distributed genetic fuzzy rule selection in [12] where the number of training data subsets was much larger than the number of CPU cores. This means that only a small portion of the available training data was used for genetic fuzzy rule selection at each generation. The computation time of genetic fuzzy rule selection was further reduced by this idea.

The use of multiple classifiers as an ensemble is one of the most promising approaches to the design of reliable classifies with high generalization ability [14]. Especially bagging is a well-known and frequently-used method [15]. There are several recent studies on GFS with bagging [16-18]. In

978-1-4244-8126-2/10/$26.00 ©2010 IEEE

bagging, a number of training data subsets are generated by random sampling with replacement. The size of each subset is usually the same as the original training data set. A base classifier for constructing an ensemble is obtained from each training data subset. Bagging is, however, time-consuming especially when we apply it to a large data set since the size of training data subsets is the same as the original training data. To tackle with this problem, small training data subsets are used for generating base classifiers in [19]. On the other hand, disjoint training data subsets are used in [20].

In this paper, we propose the use of parallel distributed implementation of genetic fuzzy rule selection for the design of fuzzy rule-based ensemble classifiers. As in our previous work [12], we use small training data subsets. That is, the number of training data subsets is larger than the number of CPU cores (i.e., the number of sub-populations). A single base classifier is selected from each sub-population. We examine two strategies for choosing the best classifier from each sub-population. In one strategy, the best classifier is locally identified using only the current training data subset assigned to each sub-population. On the other hand, all the available training data are used in the second strategy to choose the best classifier from each population. The selected classifier from each sub-population is combined to design an ensemble classifier.

This paper is organized as follows. First we explain parallel distributed implementation of genetic fuzzy rule selection with small training data subsets in Section II. Next we propose the use of parallel distributed implementation for the design of ensemble classifiers in Section III. Then we examine the performance of fuzzy rule-based ensemble classifies designed by our ensemble method with the above-mentioned two strategies for choosing a base classifier from each sub-population. Finally we conclude this paper in Section IV.

II. Parallel Distributed Implementation of Genetic Fuzzy Rule Selection

A. Fuzzy Rule-based Classifier

Let us assume that we have m training (i.e., labeled) patterns $\mathbf{x}_p = (x_{p1}, \ldots, x_{pn}), p = 1, 2, \ldots, m$ from M classes in the n-dimensional continuous pattern space where x_{pi} is the attribute value of the p-th training pattern for the i-th attribute ($i = 1, 2, ..., n$). For simplicity of explanation, we assume that all the attribute values have already been normalized into real numbers in [0, 1].

For our n-dimensional pattern classification problem, we use fuzzy rules of the following type:

$$\text{Rule } R_q\text{: If } x_1 \text{ is } A_{q1} \text{ and } \ldots \text{ and } x_n \text{ is } A_{qn} \text{ then Class } C_q \text{ with } CF_q, \quad (1)$$

where R_q is a rule label, $\mathbf{x} = (x_1, \ldots, x_n)$ is an n-dimensional pattern vector, A_{qi} is an antecedent fuzzy set ($i = 1, 2, \ldots, n$), C_q is a class label, and CF_q is a rule weight.

We simultaneously use multiple fuzzy partitions with different granularities in fuzzy rule generation. We use four homogeneous fuzzy partitions in Fig. 1 (i.e., 14 triangular fuzzy sets). We also use a domain interval [0, 1] as an antecedent fuzzy set to represent a *don't care* condition. That is, we use the 15 antecedent fuzzy sets in total.

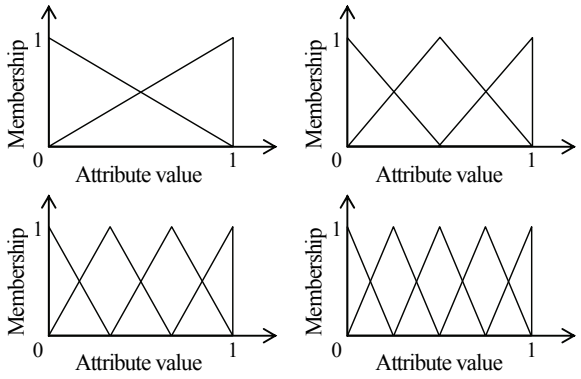

Fig. 1. Four fuzzy partitions used in our computational experiments.

The consequent class C_q and the rule weight CF_q of each fuzzy rule R_q can be specified in a heuristic manner by compatible training patterns with the antecedent part of R_q. For details, see [10-12, 21].

Since we use the 15 antecedent fuzzy sets for each attribute of our n-dimensional pattern classification problem, the total number of possible fuzzy rules is 15^n. For high-dimensional data sets, the total number of possible fuzzy rules becomes quite large. Moreover, it is very difficult to intuitively understand long fuzzy rules with many antecedent conditions. Thus, we generate short fuzzy rules with only a small number of antecedent conditions. In this paper, we examine short fuzzy rules of length $L_{\max}$ or less (e.g., $L_{\max} = 3$) in order to generate understandable fuzzy rules as candidates in fuzzy rule selection. It should be noted that the length of a fuzzy rule is defined by the number of its antecedent conditions excluding *don't care*.

B. Parallel Distributed Genetic Fuzzy Rule Selection

We use one CPU core as a server and the other CPU cores as clients in our parallel distributed implementation of genetic fuzzy rule selection [10, 11]. Our implementation can be written as follows. The server performs all the following steps except for Step 5 while Step 5 is perfomed by the clients in parallel.

Step 1: Generate a number of fuzzy rules (say, N rules) from the whole training data set D.

Step 2: Randomly generate N_{pop} binary strings of length N as an initial population P where N_{pop} is the population size.

Step 3: Randomly divide the current population P of N_{pop} binary strings into N_{CPU} sub-populations $\{P_1, P_2, \ldots, P_{N_{\text{CPU}}}\}$, and randomly divide the training data set D of m patterns into training data subsets $\{D_1, D_2, \ldots, D_d\}$ where d is the number of training data subsets.

Step 4: Assign the N_{CPU} sub-populations to the N_{CPU} client CPUs, and assign the first N_{CPU} training data subsets to the N_{CPU} client CPUs.

Step 5: Locally perform genetic fuzzy rule selection at each client CPU for a prespecified number of generations as follows (Step 5-1 to Step 5-5):

Step 5-1: Evaluate strings in the assigned sub-population using the assigned training data subset.

Step 5-2: Generate an offspring population by tournament selection, uniform crossover, and biased mutation.

Step 5-3: Evaluate strings in the offspring population using the assigned training data subset.

Step 5-4: Update the generation by the (μ+λ) ES generation update strategy.

Step 5-5: If a prespecified condition in terms of the training data subset exchange interval is not satisfied, return to Step 5-2. Otherwise go to Step 6.

Step 6: Rotate the d training data subsets as shown in Fig. 2 over the N_{CPU} client CPUs.

Step 7: If a prespecified termination condition of genetic fuzzy rule selection is not satisfied, return to Step 5. Otherwise go to Step 8.

Step 8: Choose the best string S_{best} from the whole population for the design of a standard (i.e., non-ensemble) fuzzy rule-based classifier.

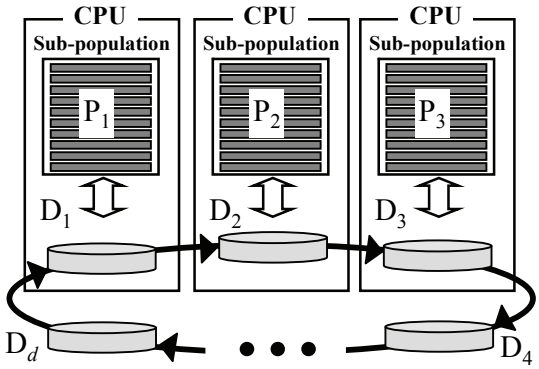

Fig. 2. Training data subdivision and its periodical rotation.

In [10, 11], we used the rule length, minimum support level, and minimum confidence level to extract candidate fuzzy rules from numerical data for rule selection. In this paper, we use the product of support and confidence in order to extract the same number of fuzzy rules for each class in Step 1 (maximum 300 fuzzy rules for each class in our experiments).

In Step 2, each binary string of length N ($N = 300M$ where M is the number of classes) is denoted as $S = s_1 s_2 s_3 \cdots s_N$ where $s_i = 1$ and $s_i = 0$ mean that the i-th candidate rule is included in and excluded from the rule set S, respectively.

In Step 3, we divide the given training patterns into d training data subsets keeping the class balance unchanged whereas we did not take into account the class balance in our former studies [10, 11]. In this paper, we examine a larger number of training data subsets than the number of CPU cores as [12]. As shown in Fig. 2, some training data subsets are not assigned to any sub-populations. However, the periodical rotation of the training data subsets makes it possible for each CPU to use all training patterns in the long run (i.e., over a large number of generations).

In Step 5, we use tournament selection, uniform crossover, biased mutation to generate an offspring population as in standard non-parallel fuzzy genetic rule selection [13, 22]. The biased mutation changes the current bit value from 0 to 1 with a small probability and from 1 to 0 with a large probability to decrease the number of 1's in each string.

We use the following three objectives [22, 23] to find an accurate and compact fuzzy classifier S at each CPU core:

$f_1(S)$: The number of correctly classified training patterns in the assigned training data subset by S,

$f_2(S)$: The number of selected fuzzy rules in S,

$f_3(S)$: The total number of antecedent conditions in S.

These three objectives are combined into the following weighted sum fitness function:

$$fitness(S) = w_1 \cdot f_1(S) - w_2 \cdot f_2(S) - w_3 \cdot f_3(S), \qquad (2)$$

where w_1, w_2 and w_3 are non-negative weights. Eq. (2) is maximized in genetic fuzzy rule selection. As a result, the accuracy is maximized while the complexity is minimized.

The first objective $f_1(S)$ is calculated for each string in the sub-population by classifying all training patterns in the assigned training data subset at each client CPU core. We use a single winner-based (i.e., winner-take-all) classification method. In this method, a single winner rule is identified for each training pattern using the product of the rule weight of each fuzzy rule and the compatibility grade of its antecedent part with the training pattern.

Since we use the single winner-based classification method, some rules may be used for the classification of no training patterns. Whereas the existence of such an unnecessary rule in the rule set S has no effect on the accuracy of S (i.e., the first objective $f_1(S)$ of the weighted sum fitness function), it deteriorates the complexity of S (i.e., the second and third objectives: $f_2(S)$ and $f_3(S)$). Thus we remove all the unnecessary fuzzy rules responsible for the classification of no training patterns before we calculate the second and third objectives.

In Step 8, we finally choose one solution (i.e., a single fuzzy rule-based classifier) with the best fitness value of Eq. (2) from the combined population. This selection is performed using all the available training patterns.

III. Ensemble Classifier Design

In our previous studies [10-12], we chose only a single fuzzy rule-based classifier with the best fitness value over all the sub-populations (i.e., the combined population) as we have just explained in the previous section. In this section, we propose the use of our parallel distributed framework for the design of ensemble classifies. The basic idea is to choose a

single classifier from each sub-population and combine the chosen classifiers as an ensemble classifier.

We examine two selection strategies: One is to choose a classifier with the best fitness value of Eq. (2) from each sub-population using the currently assigned training data subset. Each of the chosen classifiers can be viewed as being adjusted to a specific training data subset. Since the selection of each classifier can be performed locally at each sub-population, we refer to the generated ensemble classifier designed by this selection strategy as **Ensemble**$_{\text{Local}}$. The other strategy is to choose a classifier with the highest training data accuracy (i.e., the best value of $f_1(S)$) for all the available training patterns from each sub-population. Since the entire training data set rather than a single training data subset is used, we refer to the ensemble classifier designed by this selection strategy as **Ensemble**$_{\text{Global}}$.

In **Ensemble**$_{\text{Local}}$, each base classifier is selected using a different training data subset. Thus we can expect that **Ensemble**$_{\text{Local}}$ has a larger diversity in its base classifiers than **Ensemble**$_{\text{Global}}$. On the other hand, the entire training data are used for choosing each base classifier in **Ensemble**$_{\text{Global}}$. Thus we can expect that **Ensemble**$_{\text{Global}}$ has base classifiers with higher individual classification accuracy on average than **Ensemble**$_{\text{Local}}$.

In this paper, we use the simple majority voting scheme (strictly speaking "plurality voting") which is to classify each pattern as a class with the maximum number of votes by the base classifiers. When multiple classes have the same maximum number of votes, one class is randomly chosen among those classes with the maximum number of votes.

IV. Computational Experiments

In our computational experiments, we used three data sets in Table I available from the UCI machine learning repository.

TABLE I
DATA SETS USED IN OUR COMPUTATIONAL EXPERIMENTS

Data set	Number of attributes	Number of patterns	Number of classes
Phoneme	5	5404	2
Satimage	36	6435	6
Pendig	16	10992	10

We examined the classification ability of ensemble classifiers designed by the proposed method with the two selection strategies. Their generalization ability was evaluated by iterating the ten-fold cross validation procedure three times (i.e., 3 x 10 CV). That is, each result was the average over 30 runs. We used a workstation with two Xeon 3.0 GHz dual processors (i.e., four CPU cores, in total). We coded the program with C++ and Shell Script under Linux OS. We used one of the four CPU cores as a server CPU. The other three were used as client CPUs.

First we generated maximum 300 fuzzy rules for each class of each data set by using the product of confidence and support. The maximum rule length of each fuzzy rule was specified as 2 for Satimage data set, and 3 for Phoneme and Pendig data sets. In this paper, we specified the population size as 300. That is, the size of each sub-population for parallel distributed implementation was 100. The total number of evaluations was specified as 300300, which was used as the termination condition of genetic fuzzy rule selection. This termination condition is equivalent to 1001 generations with a population of size 300 in the case of non-parallel implementation of evolutionary algorithms. The weight vector in Eq. (2) was specified as $\mathbf{w} = (100, 1, 1)$.

We examined the effects of the number of training data subsets and their rotation interval as in [12]. It should be noted that the performance of fuzzy rule-based classifies was examined in [12] whereas we report the performance of ensemble classifies in this paper. The following 6×5 combinations of parameter specifications were examined in this paper.

Number of training data subsets: 3, 6, 9, 12, 15, 27,

Rotation interval (generation): 10, 50, 100, 200, none,

where "none" means that training data subsets were not rotated. When the number of training data subsets was large and the rotation interval was long, some training data subsets were not used during the execution of our parallel distributed genetic fuzzy rule selection. In an extreme case of no rotation, only the first three training data subsets were used (see Fig. 2) in the previous section. This setting looks like distributed data mining [24].

In Table II, we summarize experimental results by the standard non-parallel implementation of genetic fuzzy rule selection with a single population and no data subdivision. A single best fuzzy rule-based classifier was chosen in each execution of genetic fuzzy rule selection in Table II.

TABLE II
RESULTS BY THE NON-PARALLEL IMPLEMENTATION

Data set	Training data accuracy	Test data accuracy	Number of fuzzy rules	Computation time [min]
Phoneme	81.01	80.34	14.57	49.1
Satimage	83.45	82.12	34.60	185.0
Pendig	89.52	88.71	51.63	501.3

In Figs. 3-6, we summarize experimental results of parallel distributed genetic fuzzy rule selection on the Phoneme data set. As in Table II, a single best fuzzy rule-based classifier was chosen in each execution. In Fig. 3 and Fig. 4, the increase in the number of training data subsets (i.e., the decrease in the size of training data subsets) had a negative effect on the accuracy of obtained fuzzy rule-based classifiers. Frequent rotation of training data subsets had a positive effect in Fig. 3 and Fig. 4. In Fig. 5, frequent rotation of training data subsets seems to decrease slightly the average number of fuzzy rules in obtained fuzzy rule-based classifiers. The most prominent observation in Figs. 3-6 is the decrease in the

computation time in Fig. 6 by the increase in the number of training data subsets (i.e., by the decrease in the size of training data subsets). The computation time was decreased from 49.1 minutes in Table II in the non-parallel case to less than one minute in Fig. 6. This observation clearly shows a drastic speed-up by our parallel distributed implementation.

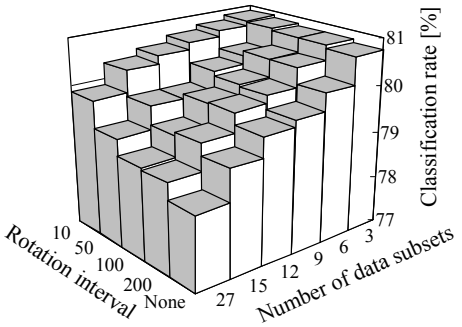

Fig. 3. Training data accuracy on the Phoneme data set.

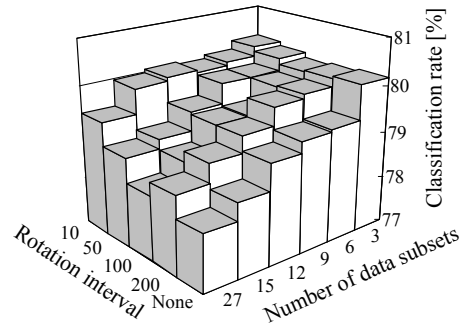

Fig. 4. Test data accuracy on the Phoneme data set.

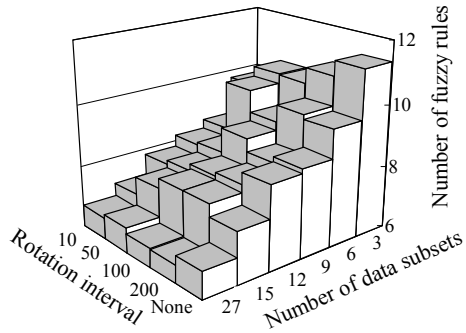

Fig. 5. The number of fuzzy rules on the Phoneme data set.

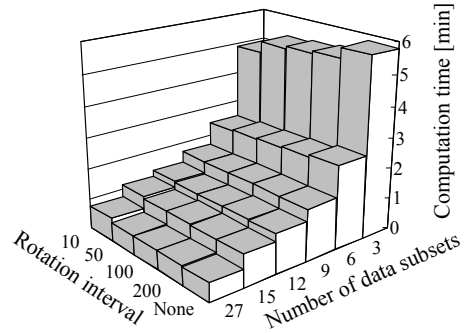

Fig. 6. Computation time on the Phoneme data set.

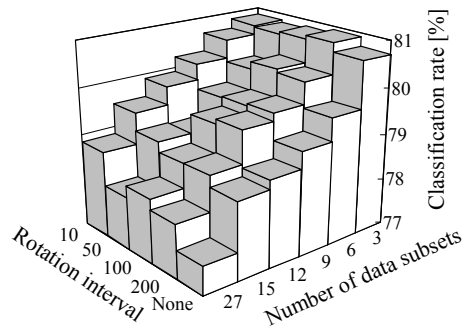

Fig. 7. Training data accuracy of **Ensemble**$_{\text{Local}}$ on the Phoneme data set.

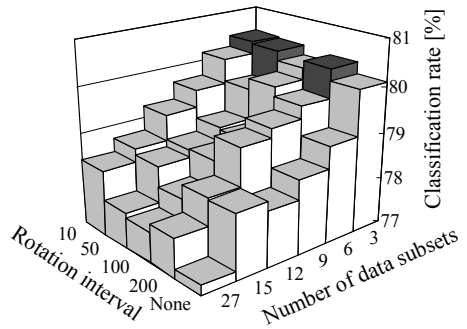

Fig. 8. Test data accuracy of **Ensemble**$_{\text{Local}}$ on the Phoneme data set.

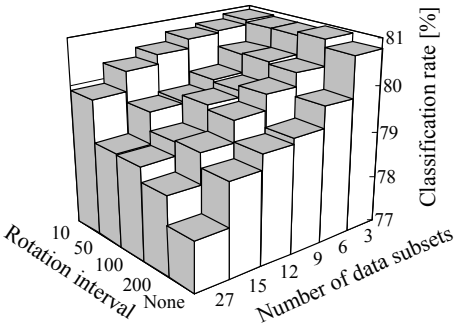

Fig. 9. Training data accuracy of **Ensemble**$_{\text{Global}}$ on the Phoneme data set.

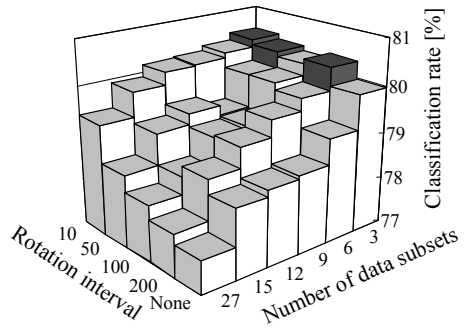

Fig. 10. Test data accuracy of **Ensemble**$_{\text{Global}}$ on the Phoneme data set.

Figs. 7-10 show experimental results by the proposed ensemble method. The dark gray bars in these figures show that higher classification rates were obtained than the case of non-parallel implementation in Table II (i.e., 81.01% training data accuracy and 80.34% test data accuracy). When we specified the number of training data subsets as three, better

test data accuracy was obtained by the proposed ensemble method with some settings of parameter values in Fig. 8 and Fig. 10 than non-parallel implementation in Table II. With respect to the comparison between the two selection strategies of base classifiers, slightly better results were obtained by **Ensemble**$_{\text{Global}}$ than **Ensemble**$_{\text{Local}}$ in Figs. 7-10.

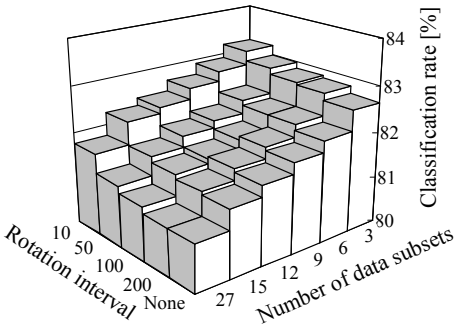

Fig. 11. Training data accuracy on the Satimage data set.

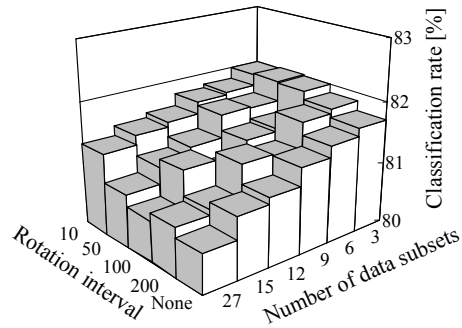

Fig. 12. Test data accuracy on the Satimage data set.

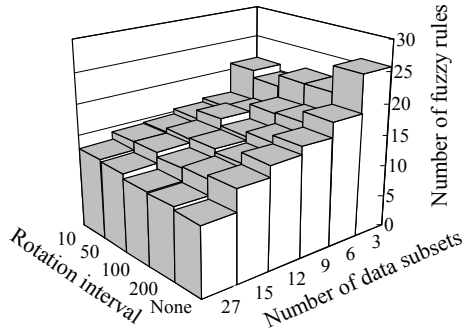

Fig. 13. The number of fuzzy rules on the Satimage data set.

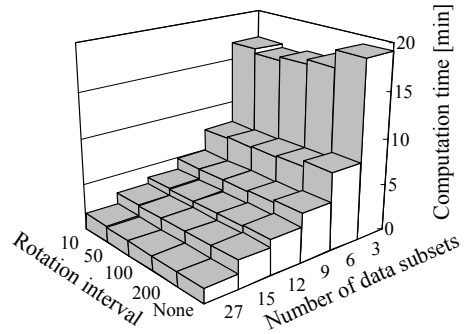

Fig. 14. Computation time on the Satimage data set.

Experimental results on the Satimage data set are shown in Figs. 11-18. The above-mentioned observations from Figs. 3-10 on the Phoneme data set are applicable to experimental results on the Satimage data set in Figs. 11-18.

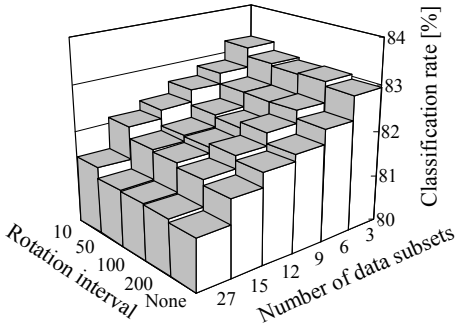

Fig. 15. Training data accuracy of **Ensemble**$_{\text{Local}}$ on the Satimage data set.

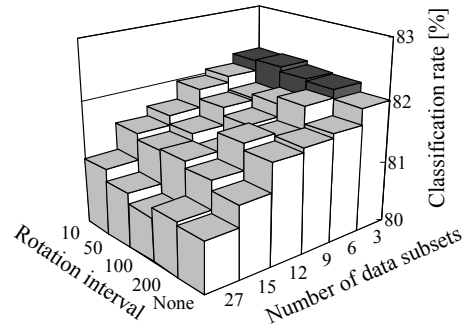

Fig. 16. Test data accuracy of **Ensemble**$_{\text{Local}}$ on the Satimage data set.

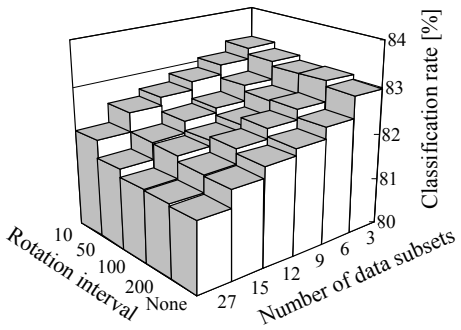

Fig. 17. Training data accuracy of **Ensemble**$_{\text{Global}}$ on the Satimage data set.

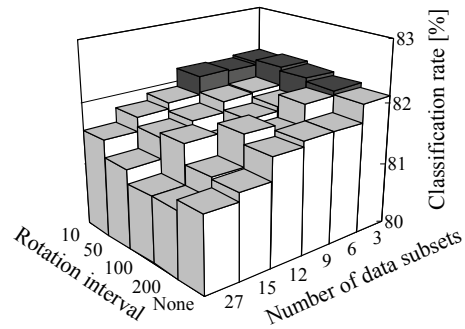

Fig. 18. Test data accuracy of **Ensemble**$_{\text{Global}}$ on the Satimage data set.

Performance improvement by ensemble classifiers in comparison with single fuzzy rule-based classifiers was

demonstrated more clearly in Figs. 19-26 on the Pendig data set. For example, even when the number of training data subsets was 15 (i.e., the size of each training data subset was 1/15 of the entire training data set), higher test data accuracy was obtained by rotating training data subsets every 10 generation in **Ensemble**$_{\text{Global}}$ in Fig. 26.

In this case, the computation time was decreased from 501.3 minutes by the non-parallel implementation in Table II to 8.8 minutes in Fig. 26. That is, higher test data accuracy was obtained by the proposed ensemble approach in less than 1/50 computation times in comparison with the non-parallel genetic fuzzy rule selection for on the Pendig data set.

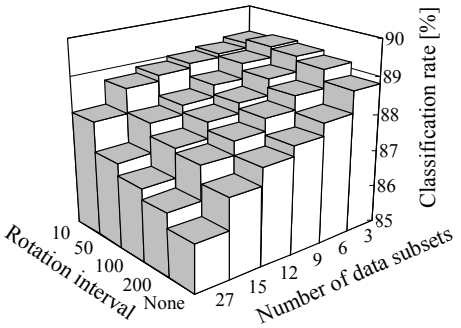

Fig. 19. Training data accuracy on the Pendig data set.

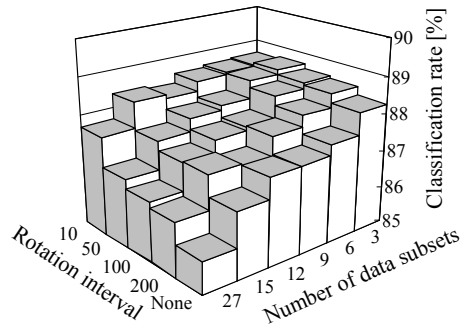

Fig. 20. Test data accuracy on the Pendig data set.

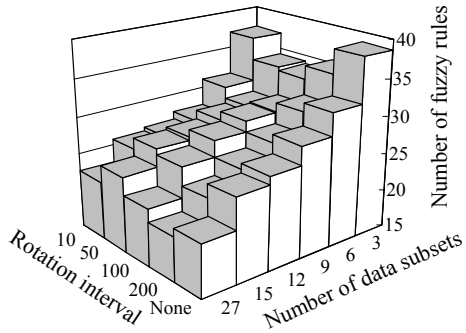

Fig. 21. The number of fuzzy rules on the Pendig data set.

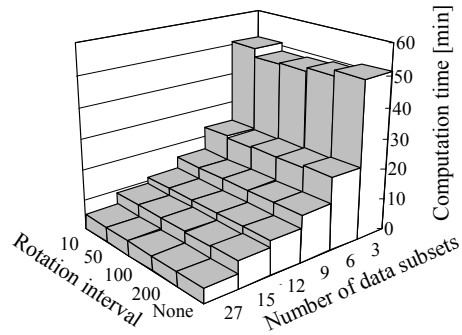

Fig. 22. Computation time on the Pendig data set.

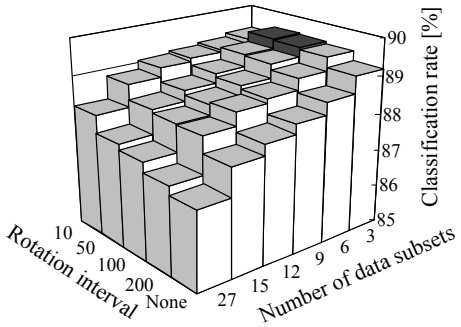

Fig. 23. Training data accuracy of **Ensemble**$_{\text{Local}}$ on the Pendig data set.

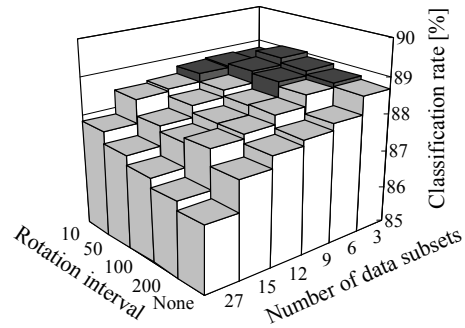

Fig. 24. Test data accuracy of **Ensemble**$_{\text{Local}}$ on the Pendig data set.

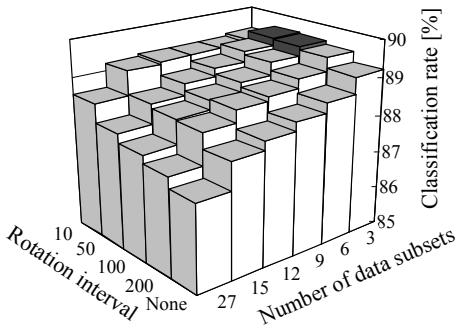

Fig. 25. Training data accuracy of **Ensemble**$_{\text{Global}}$ on the Pendig data set.

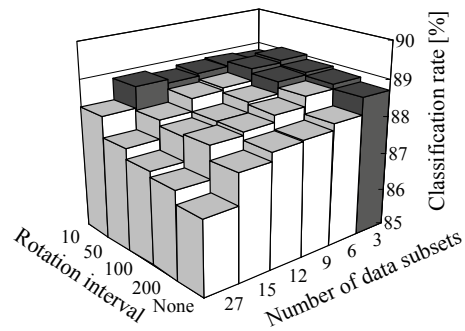

Fig. 26. Test data accuracy of **Ensemble**$_{\text{Global}}$ on the Pendig data set.

By the proposed ensemble method, the total number of rules in an ensemble classifier was about three times of a single ensemble classifier because of three base classifiers. Of course, there exist some overlapping rules among the base classifiers. Since we used different base classifier selection strategies, there is a difference in the total number of rules between $\mathbf{Ensemble}_{\mathrm{Local}}$ and $\mathbf{Ensemble}_{\mathrm{Global}}$. But, that was not large. For example, the total number of rules in $\mathbf{Ensemble}_{\mathrm{Local}}$ and that in $\mathbf{Ensemble}_{\mathrm{Global}}$ were 29.27 and 30.20 for Phoneme data set, when we specified the rotation interval and the number of data subsets as ten and three, respectively.

V. Conclusion

In this paper, we proposed the use of parallel distributed implementation of genetic fuzzy rule selection for the design of ensemble fuzzy rule-based classifiers. In the proposed approach, a single base classifier was chosen from each sub-population at the final generation to construct an ensemble classifier. We examined two strategies for choosing a single base classifier from each sub-population. Our experimental results on three data sets showed that the proposed approach worked well even when the size of training data subsets was small. In some cases, our approach was more than 50 times faster than that of the standard non-parallel implementation of genetic fuzzy rule selection.

There are still a number of remaining issues to be discussed. For example, we need to perform further computational experiments on much more data sets in order to qualitatively evaluate the effectiveness of the proposed ensemble method. Through those experiments, we may be able to examine the relation between the data set size and the effectiveness of the proposed ensemble method. There are some future extensions. For example, reconstruction of a single classifier from the rules in the obtained ensemble classifier may be an interesting approach to make a simpler and more accurate classifier than the ensemble one. Moreover, the use of fuzzy GBML algorithms [23, 25] instead of genetic fuzzy rule selection can remove the necessity of a heuristic prescreening procedure for candidate fuzzy rule prescreening. Since the use of evolutionary multiobjective optimization algorithms can remove the weight vector specification in the proposed method, parallel distributed implementation of multiobjective fuzzy GBML algorithms [26, 27] would be a nice approach.

References

[1] O. Cordon, F. Herrera, F. Hoffman, and L. Magdalena, *Genetic Fuzzy Systems*, World Scientific, 2001.

[2] F. Herrera, "Genetic fuzzy systems: Taxonomy, current research trends and prospects," *Evolutionary Intelligence*, vol. 1, pp. 27-46, 2008.

[3] H. Ishibuchi, "Multiobjective genetic fuzzy systems: Review and future research directions," *Proc. of 2007 IEEE International Conference on Fuzzy Systems*, pp. 913-918, 2007.

[4] E. Alba and M. Tomassini, "Parallelism and evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 5, pp. 443-462, 2002.

[5] E. Cantu-Paz, "A survey of parallel genetic algorithms," *IlliGAL Report* No. 95003, 1997.

[6] H. Liu and H. Motoda, *Instance Selection and Construction for Data Mining*, Kluwer Academic Publishers, 1998.

[7] H. Liu and H. Motoda, *Feature Selection for Knowledge Discovery and Data Mining*, Kluwer Academic Publishers, 1998.

[8] J. R. Cano, F. Herrera, and M. Lozano, "Stratification for scaling up evolutionary prototype selection," *Pattern Recognition Letters*, vol. 26, no. 7, pp. 953-963, 2005.

[9] J. R. Cano, F. Herrera, and M. Lozano, "On the combination of evolutionary algorithms and stratified strategies for training set selection in data mining," *Applied Soft Computing*, vol. 6, no. 3, pp. 323-332, 2006.

[10] Y. Nojima, H. Ishibuchi, and I. Kuwajima, "Parallel distributed genetic fuzzy rule selection," *Soft Computing*, vol. 13, no. 5, pp. 511-519, 2009.

[11] Y. Nojima, I. Kuwajima, and H. Ishibuchi, "Data set subdivision for parallel distributed implementation of genetic fuzzy rule selection," *Proc. of 2007 IEEE International Conference on Fuzzy Systems*, pp. 2006-2011, 2007.

[12] Y. Nojima, H. Ishibuchi, and S. Mihara, "Use of very small training data subsets in parallel distributed genetic fuzzy rule selection," *Proc. of 4th International Workshop on Genetic and Evolutionary Fuzzy Systems*, pp. 27-32, 2010.

[13] H. Ishibuchi, K. Nozaki, N. Yamamoto, and H. Tanaka, "Selecting fuzzy if-then rules for classification problems using genetic algorithms," *IEEE Trans. on Fuzzy Systems*, vol. 3, no. 3, pp. 260-270, 1995.

[14] L. Rokach, "Ensemble-based classifiers," *Artificial Intelligence Review*, vol. 33, pp. 1-39, 2010.

[15] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, pp. 123-140, 1996.

[16] O. Cordon, A. Quirin, and L. Sanchez, "A first study on bagging fuzzy rule-based classification systems," *Proc. of 3rd International Workshop on Genetic and Evolutionary Fuzzy Systems*, pp. 11-16, 2008.

[17] O. Cordon and A. Quirin, "Comparing two genetic overproduce-and-choose Strategies for fuzzy rule-based multiclassification systems generated by bagging and mutual information-based feature selection," *International Journal of Hybrid and Intelligent Systems*, vol. 7, no. 1, pp. 45-64, 2010.

[18] L. Sanchez, O. Cordon, A. Quirin, and K. Trawinski, "Introducing a genetic fuzzy linguistic combination method for bagging fuzzy rule-based multiclassification systems," *Proc. of 4th International Workshop on Genetic and Evolutionary Fuzzy Systems*, pp. 75-80, 2010.

[19] L. Breiman, "Pasting small votes for classification in large database and on-line," *Machine Learning*, vol. 36, pp. 85-103, 1999.

[20] N. V. Chawla, T. E. Moore, L. O. Hall, K. W. Bowyer, W. P. Kegelmeyer, and C. Springer, "Distributed learning with bagging-like performance," *Pattern Recognition Letters*, vol. 24, pp. 445-471, 2003.

[21] H. Ishibuchi, T. Nakashima, and M. Nii, *Classification and Modeling with Linguistic Information Granules: Advanced Approaches to Linguistic Data Mining*, Springer, Berlin, 2004.

[22] H. Ishibuchi and T. Yamamoto, "Comparison of heuristic criteria for fuzzy rule selection in classification problems," *Fuzzy Optimization and Decision Making*, vol.3, no.2, pp.119-139, 2004.

[23] H. Ishibuchi, T. Nakashima, and T. Murata, "Three-objective genetics-based machine learning for linguistic rule extraction," *Information Science*, vol.136, no.1-4, pp.109-133, 2001.

[24] D. Mokeddem and H. Belbachir, "A survey of distributed classification based ensemble data mining method," *Journal of Applied Sciences*, vol. 9, no. 20, pp. 3739-3745, 2009.

[25] H. Ishibuchi, T. Yamamoto, and T. Nakashima, "Hybridization of fuzzy GBML approaches for pattern classification problems," *IEEE Trans. on Systems, Man, and Cybernetics*- Part B: Cybernetics, vol. 35, no. 2, pp.359-365, 2005.

[26] H. Ishibuchi and Y. Nojima, "Analysis of interpretability-accuracy tradeoff of fuzzy systems by multiobjective fuzzy genetics-based machine learning," *International Journal of Approximate Reasoning*, vol. 44, no. 1, pp. 4-31, 2007.

[27] H. Ishibuchi, Y. Nakashima, and Y. Nojima, "Search ability of evolutionary multiobjective optimization algorithms for multi-objective fuzzy genetics-based machine learning," *Proc. of 2009 IEEE International Conference on Fuzzy Systems*, pp. 1724-1729, 2009.