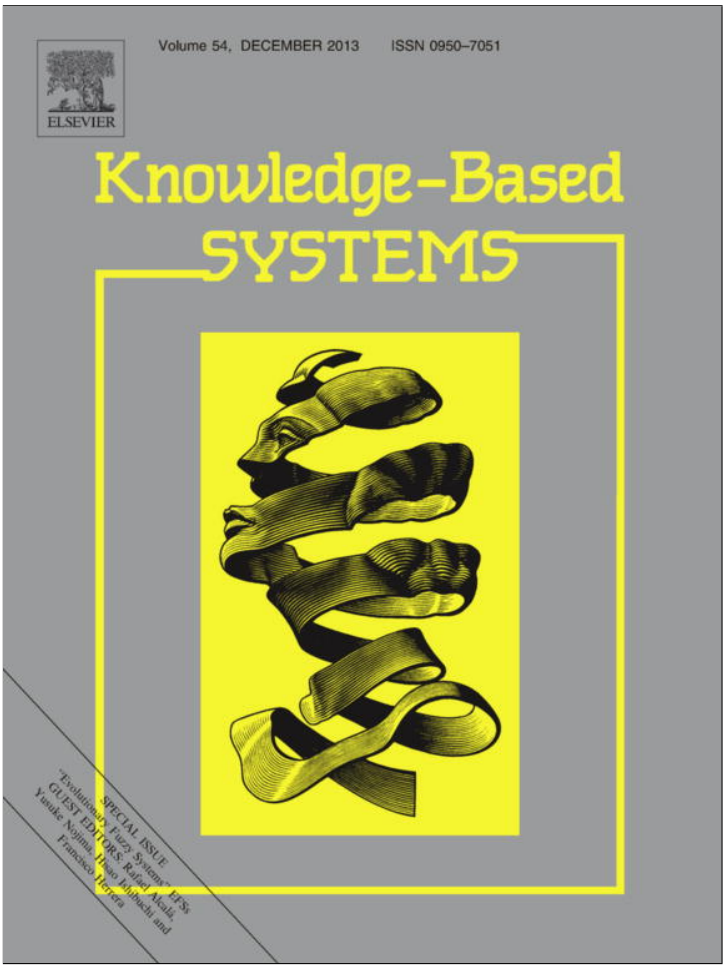


Knowledge-Based Systems 54 (2013) 22–31

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Repeated double cross-validation for choosing a single solution in evolutionary multi-objective fuzzy classifier design

Hisao Ishibuchi, Yusuke Nojima *

Graduate School of Engineering, Osaka Prefecture University, 1-1 Gakuen-cho, Naka-ku, Sakai, Osaka 599-8531, Japan

ARTICLE INFO

Article history:
Available online 5 October 2013

Keywords:
Fuzzy rule-based classifiers
Multi-objective genetic fuzzy systems
Evolutionary multi-objective optimization
Multi-objective genetics-based machine learning
Repeated double cross-validation

ABSTRACT

The main advantage of multi-objective genetic fuzzy systems (MoGFS) is that a number of non-dominated fuzzy rule-based systems are obtained along the tradeoff surface among conflicting objectives. Accuracy maximization, complexity minimization and interpretability maximization have often been used for multi-objective design of fuzzy rule-based classifiers. A number of non-dominated fuzzy rule-based classifiers are obtained by a single run of MoGFS. A human decision maker is supposed to choose a single final classifier from a number of obtained classifiers according to his/her preference. One problem, which has not been discussed in many studies on MoGFS, is how to choose a single final classifier. In this paper, we discuss classifier selection with no intervention of the decision maker. Whereas complexity and interpretability are very important factors in classifier selection, we concentrate on the maximization of generalization ability as the first step towards a more general handling of classifier selection. We propose the use of repeated double cross-validation (rdCV) to choose a single final classifier and to evaluate the generalization ability of the selected classifier. We also discuss how our approach can be applied to parameter specification, formulation selection and algorithm choice.

1. Introduction

Two conflicting objectives are often involved in classifier design: accuracy maximization and complexity minimization. These two objectives were combined into a weighted sum fitness function in genetic fuzzy rule selection in the mid-1990s [1,2]. In the late-1990s [3–5], they were used as separate objectives in a two-objective approach. Since the late-1990s, evolutionary multi-objective optimization (EMO) algorithms [6–8] have been used for multi-objective design of fuzzy systems [9–11]. This research area, which is included in a much broader research field called genetic fuzzy systems (GFS [12–15]), is often referred to as multi-objective genetic fuzzy systems (MoGFS). This is because multi-objective genetic algorithms (e.g., NSGA-II [16] and SPEA [17]) have been mainly used in MoGFS. Recently multi-objective approaches have also been used for various classifier design such as neural networks and decision trees [18–22].

In this paper, we use the following simple two-objective formulation to clearly explain an accuracy-complexity tradeoff relation in a two-dimensional objective space:

$$\text{Minimize } f_1(S) = \mathit{Error}(S), \quad f_2(S) = \mathit{Complexity}(S), \tag{1}$$

where S is a fuzzy rule-based classifier, $\mathit{Error}(S)$ is a percentage error rate of S on training data, and $\mathit{Complexity}(S)$ is the number of fuzzy rules in S.

In MoGFS, we can use multiple complexity measures such as the number of fuzzy rules and the total number of antecedent conditions [23–25]. It is also possible to use multiple accuracy measures such as a true positive rate and a false positive rate [26,27].

In the EMO community, performance measures such as hypervolume and generational distance have been proposed to evaluate the search ability of EMO algorithms [28]. Those measures can be also used to evaluate the search ability of MoGFS. However, high search ability of MoGFS does not always mean high performance of obtained fuzzy rule-based classifiers. This is because the search ability of MoGFS is measured by the accuracy on training data while classifier performance should be measured by the accuracy on unseen test data.

Fig. 1 illustrates the relation between training data accuracy and test data accuracy. If high training data accuracy always leads to high test data accuracy as in Fig. 1(a), good classifiers are obtained from MoGFS with high search ability. However, as in Fig. 1(b), high search ability of MoGFS does not always mean high performance of classifiers due to overfitting to training data.

As shown in Fig. 1, a number of non-dominated fuzzy rule-based classifiers are obtained from a single run of MoGFS. One problem, which has not been discussed in many studies on MoGFS,

* Corresponding author. Tel.: +81 722549198.
E-mail addresses: hisaoi@cs.osakafu-u.ac.jp (H. Ishibuchi), nojima@cs.osakafu-u.ac.jp (Y. Nojima).

0950-7051/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.knosys.2013.09.023

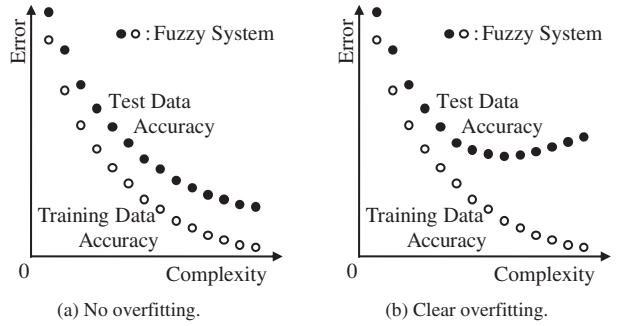


Fig. 1. Two typical cases of the relation between training data accuracy and test data accuracy.

is how to choose a single final classifier. In MoGFS, a human decision maker is supposed to choose a single final classifier based on his/her preference with respect to various factors of fuzzy rule-based classifiers such as accuracy, complexity and interpretability. In this paper, we discuss classifier selection with no intervention of the decision maker. We concentrate on classifier selection for generalization ability maximization. Of course, complexity and interpretability of fuzzy rule-based classifiers are also important factors when a single final classifier is chosen by the decision maker. Thus classifier selection based on those factors is an important research issue. Interactive classifier selection with the decision maker is also an important research issue. Those issues are left as future research topics. In this paper, we concentrate on classifier selection for generalization ability maximization as the first step towards such a more general handling of classifier selection.

A simple classifier selection method with no intervention of the decision maker in MoGFS is to choose the fuzzy rule-based classifier with the highest training data accuracy. When we use the two-objective formulation in (1), this method means the selection of the fuzzy rule-based classifier with the highest complexity (i.e., the selection of the right-most open circle in Fig. 1). This method does not work well in Fig. 1(b) with overfitting while it works well in Fig. 1(a) with no overfitting. One may think that it is the best way to choose the fuzzy rule-based classifier with the highest test data accuracy. The direct use of test data accuracy is unrealistic because "unseen" test data are not available in the classifier selection phase. Test data are available only when the selected classifier is evaluated (i.e., after the classifier selection phase is completed). In this paper, we propose a classifier selection method for generalization ability maximization to choose a single fuzzy rule-based classifier without using test data accuracy.

Our idea is to use repeated double cross-validation (rdCV) [29] for classifier selection and classifier evaluation. Our rdCV approach has a nested structure of two CV loops. The inner CV loop is used to find the best complexity with the highest validation data accuracy, which is used for classifier selection. The outer CV loop is used to evaluate the test data accuracy of the selected classifier in the same manner as in the standard CV procedure.

This paper is a modified and extended version of our conference paper [30]. In [30], we proposed the use of rdCV for classifier selection in MoGFS together with a complicated best complexity estimation method. In this paper, we propose two simple methods for the best complexity estimation, which are used in rdCV-based classifier selection in MoGFS. The performance of the proposed methods are examined through computational experiments on a number of data sets. We also discuss the use of rdCV for parameter specification, formulation selection and algorithm choice.

This paper is organized as follows. In Section 2, we explain the basic idea of classifier selection in MoGFS. In Section 3, we explain the overall framework of rdCV for classifier selection in MoGFS together with two classifier selection methods. The performance of the proposed methods is examined through computational experiments in Section 4. In Section 5, we discuss how our rdCV-based classifier selection methods can be applied to parameter specification, formulation selection and algorithm choice. Finally, we conclude this paper in Section 6.

2. Classifier selection in MoGFS

We assume that we have a MoGFS algorithm and a set of available patterns. By applying the MoGFS algorithm to the given data, a number of non-dominated fuzzy rule-based classifiers are obtained. Our problem is how to choose a single final fuzzy rule-based classifier with high generalization ability.

In this situation, classifier selection is difficult because we do not know the generalization ability of each classifier. So we usually divide the available patterns into training data and validation data. The training data are used in the MoGFS algorithm to design non-dominated classifiers while the validation data are used to evaluate each of the obtained classifiers. A single classifier with the highest validation data accuracy is selected. One difficulty of this approach is that the selected classifier totally depends on the partition of the available patterns into training and validation data. Another difficulty is that all the available patterns are not used in the MoGFS algorithm for classifier design. If we use almost all patterns as training data for the design of non-dominated classifiers, we cannot use many patterns to evaluate the obtained classifiers for classifier selection. However, the use of many patterns for classifier selection leads to the decrease in the size of training data for classifier design by the MoGFS algorithm.

Our idea for classifier selection is to examine the accuracy-complexity tradeoff relation by iterating the train-validation procedure using different data partitions. Results from multiple runs of the train-validation procedure are used to estimate the best complexity with the highest validation data accuracy. The estimated best complexity is used to choose a single final classifier from non-dominated classifiers that are obtained by applying the MoGFS algorithm to all the available patterns. The basic framework of our approach can be summarized as follows:

(i) The available patterns are divided into training data and validation data. The MoGFS algorithm is applied to the training data to design non-dominated fuzzy rule-based classifiers. The accuracy of each classifier on the validation data is calculated together with its complexity. This train-validation procedure is iterated using different data partitions.
(ii) Results in (i) are used to estimate the best complexity with the highest validation data accuracy.
(iii) The MoGFS algorithm is applied to all the available patterns to design non-dominated fuzzy rule-based classifiers. Using the estimated best complexity in (ii), a single final classifier is chosen from the obtained non-dominated classifiers.

The train-validation procedure in (i) is iterated using different data partitions. Thus the estimated best complexity does not depend on a single data partition. Moreover, all the available patterns are used to design non-dominated classifiers in (iii).

From each run of the train-validation procedure in (i), we obtain a number of non-dominated classifiers. Since the train-validation procedure is iterated using different data partitions, we obtain different sets of non-dominated classifiers in (i). The main problem in the above-mentioned procedure for classifier selection is how to estimate the best complexity with the highest validation data accuracy in (ii) using different sets of non-dominated classifiers

obtained in (i). In this paper, we propose two simple methods for the best complexity estimation.

One idea for the best complexity estimation is to first estimate the best complexity with the highest validation data accuracy in each run of the train-validation procedure. The final estimation is calculated as the average value of the estimated best complexity over multiple runs of the train-validation procedure. This method is referred to as the first-estimate-then-average (FETA) method. Another idea is to first calculate the average validation data accuracy for each complexity over multiple runs of the train-validation procedure. The final estimation of the best complexity is obtained as the complexity with the highest average validation data accuracy. This approach is referred to as the first-average-then-estimate (FATE) method. These two methods are explained in detail in the next section.

3. Repeated double CV for classifier selection

3.1. Basic Idea of repeated double cross-validation

When we evaluate the performance of classifiers, we usually use cross-validation (CV) [31–34]. In k-fold cross-validation (kCV), available patterns are divided into k subsets of the same size. One subset is used as test data and the others $(k-1)$ subsets are used as training data. A classifier is designed using the training data. The accuracy of the designed classifier is evaluated using the test data. This train-test procedure is iterated k times so that all the k subsets are used as test data. A single execution of kCV consists of those k iterations. Since the calculated test data accuracy depends on the data partition into k subsets, the execution of kCV is usually iterated several times using different data partitions. We denote t iterations of kCV as $t \times k$CV, which includes tk runs of the train-test procedure. For example, 5×10CV means five iterations of 10CV.

The performance of the MoGFS algorithm can be evaluated in the same manner. In each run in $t \times k$CV, the MoGFS algorithm is applied to the training data to search for non-dominated fuzzy rule-based classifiers. A single classifier is selected from the obtained ones in each run. The selected classifier is evaluated using the corresponding test data in each run. This train-test procedure is iterated $t \times k$ times (i.e., $t \times k$ runs).

Our idea is to use a cross-validation mechanism for classifier selection in each of $t \times k$ runs of the MoGFS algorithm in $t \times k$CV. That is, the training data in each run in $t \times k$CV are used as available patterns in another CV procedure for classifier selection. This idea leads to the same framework as repeated double Cross-Validation (rdCV [29]), which has a nested structure of two CV loops (i.e., inner and outer CV loops) as shown in Fig. 2. The rdCV has already been utilized for performance evaluation of classifiers and regression models [35–37] (for details, see [38,39]). However, at the best of our knowledge, rdCV has not been used in multi-objective machine learning including MoGFS.

The outer CV loop is the same as the standard CV for performance evaluation of classifiers. We use k_{Out}-fold CV (i.e., k_{Out}CV), which is iterated t_{Out} times (i.e., $t_{Out} \times k_{Out}$CV). The MoGFS algorithm is executed $t_{Out} \times k_{Out}$ times in the outer CV loop (e.g., 10 times in 1×10CV in Fig. 2).

The inner CV loop is applied to training data from the outer CV loop. We use k_{In}-fold CV (i.e., k_{In}CV), which is iterated t_{In} times (i.e., $t_{In} \times k_{In}$CV). The outer CV training data are divided into k_{In} subsets (e.g., $k_{In} = 5$ in Fig. 2). One subset is used as validation data while the other $(k_{In}-1)$ subsets are used as inner CV training data. The MoGFS algorithm is applied to the inner CV training data to design non-dominated fuzzy rule-based classifiers. The validation data are used to evaluate the accuracy of each classifier. The train-validation procedure is iterated $t_{In} \times k_{In}$ times for the same outer CV training data to estimate the best complexity with the highest validation data accuracy. Using the estimated best complexity for the current outer CV training data, a single classifier is chosen from the non-dominated ones obtained by the MoGFS algorithm for the current outer CV training data.

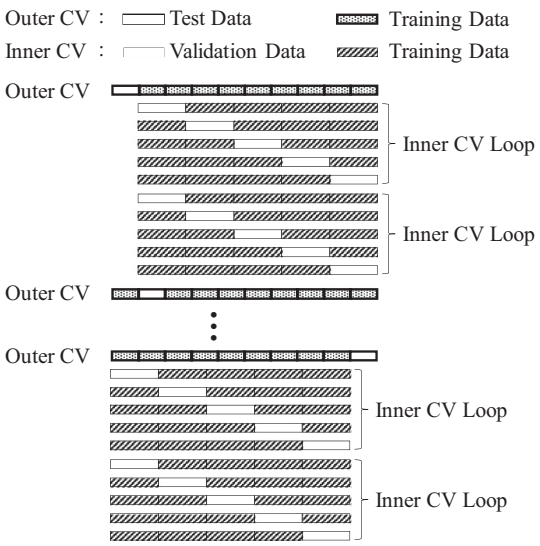


Fig. 2. A rdCV procedure with 1×10CV in the outer loop and 2×5CV in the inner loop.

Since $t_{Out} \times k_{Out}$CV is used in the outer CV loop, the total number of runs of the MoGFS algorithm in the inner CV loop is $t_{Out} \times k_{Out} \times t_{In} \times k_{In}$. Thus the total number of runs of the MoGFS in the outer and inner CV loop is $(t_{Out} \times k_{Out} + t_{Out} \times k_{Out} \times t_{In} \times k_{In}) = t_{Out} \times k_{Out}(1 + t_{In} \times k_{In})$, which is $(1 + t_{In} \times k_{In})$ times larger than the number of runs in the standard CV (i.e., $t_{Out} \times k_{Out}$). For example, when 5×10CV is used in the inner CV loop, the total number of runes of the MoGFS in our rdCV approach is 51 times larger than the case of the standard CV approach. Such a computational overhead is always needed when our rdCV approach is used for any multiobjective machine learning algorithms.

3.2. Implementation of rdCV for MoGFS

In the inner CV loop, a number of non-dominated fuzzy rule-based classifiers are obtained from each run of the MoGFS algorithm on inner CV training data. The error rate on the validation data is calculated for each classifier. A different set of non-dominated fuzzy rule-based classifiers is usually obtained from each of $t_{In} \times k_{In}$ runs in the inner CV loop since different inner CV training data are used in each run. Thus, $t_{In} \times k_{In}$ sets of non-dominated fuzzy rule-based classifiers are obtained together with the validation data error rate of each classifier. We use those results to estimate the best complexity with the highest validation data accuracy for the current outer CV training data. The estimated best complexity is used to choose a single classifier from the obtained non-dominated ones by the MoGFS algorithm for the current outer CV training data.

We propose the following two methods for the best complexity estimation:

First-estimate-then-average method (FETA): In this methods, first the best complexity is estimated in each run of the MoGFS algorithm in the inner CV loop. Then the average value of the estimated best complexity is calculated over $t_{In} \times k_{In}$ runs. The calculated average value is used as the best complexity to choose

a single final classifier for the current outer CV training data in the outer CV loop.

First-average-then-estimate method (FATE): In this method, first the average error rate for each complexity is calculated over $t_{In} \times k_{In}$ runs. Then the best complexity is estimated from the calculated average error rate for each complexity. The estimated best complexity is used to choose a single final classifier for the current outer CV training data in the outer CV loop.

In the following, we explain each of these methods in details. Let us assume that four and three non-dominated fuzzy rule-based classifiers in Fig. 3(a) and (b) are obtained from the first and second runs of MoGFS in the inner CV loop, respectively. In Fig. 3, we assume that multi-objective classifier design is formulated as the two-objective minimization problem of the error rate and the number of fuzzy rules.

In the FETA method, we first estimate the best complexity from each run. In Fig. 3(a), the smallest validation error rate is obtained from the classifier with four fuzzy rules. Thus the best complexity is estimated as four from the first run. From the second run in Fig. 3(b), the best complexity is estimated as three. In this manner, the best complexity is estimated from each of the $t_{In} \times k_{In}$ runs of the MoGFS algorithm in the inner CV loop. The average value of the estimated best complexity over the $t_{In} \times k_{In}$ runs is used as the estimated best complexity for classifier selection for the current outer CV training data in the outer CV loop. It should be noted that the obtained best complexity is not always an integer value in the FETA method. For the current outer CV training data in the outer CV loop, a single classifier with the most similar complexity to the estimated best complexity is chosen.

In the FATE method, we first calculate the average error rate on validation data for each complexity (i.e., each number of fuzzy rules in Fig. 3). It should be noted that a different set of non-dominated fuzzy rule-based classifiers is obtained from each of the $t_{In} \times k_{In}$ runs of the MoGFS algorithm in the inner CV loop as illustrated in Fig. 3. This means that fuzzy rule-based classifiers with a particular complexity (i.e., a particular number of fuzzy rules) are not always obtained in all the $t_{In} \times k_{In}$ runs. Thus the calculated average error rate for each complexity is not always the average over the $t_{In} \times k_{In}$ runs. When we estimate the best complexity, we use only the average error rates calculated over at least α% of the $t_{In} \times k_{In}$ runs (e.g., $\alpha = 80$). If fuzzy rule-based systems with a particular complexity are not obtained from at least α% of the $t_{In} \times k_{In}$ runs, the complexity is not selected as the best complexity.

In our former study [30], we used a substitute method for calculating the average error rate for each complexity over all the $t_{In} \times k_{In}$ runs. In the substitute method, a similar classifier with less complexity was used in the average error rate calculation when a fuzzy rule-based classifier with a particular complexity was not obtained. The substitute method in our former study [30] is illustrated in Fig. 4. In Fig. 4(a), no fuzzy rule-based classifier with

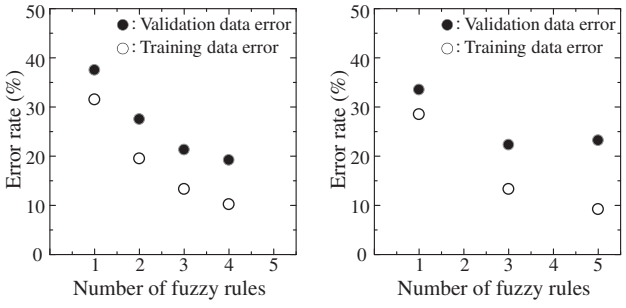


Fig. 3. Two sets of non-dominated fuzzy rule-based classifiers.

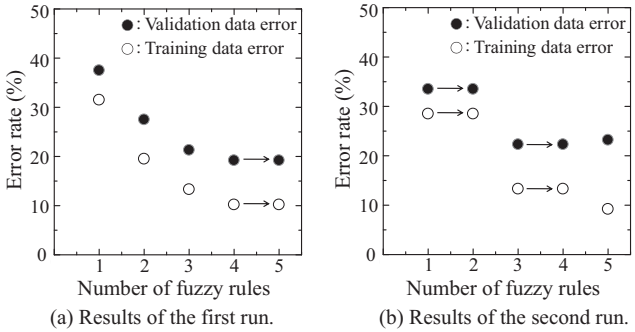


Fig. 4. Illustration of the use of other fuzzy rule-based classifiers with similar complexity as dummy classifiers for the calculation of the average error rates on validation data [30].

five fuzzy rules is not obtained. So the obtained classifier with four fuzzy rules is used as a dummy classifier in the average error rate calculation over classifiers with five fuzzy rules. In Fig. 4(b), the obtained classifiers with one and three fuzzy rules are used as dummy classifiers with two and four fuzzy rules, respectively. Due to the use of dummy classifiers, the average error rate is not always calculated over fuzzy rule-based classifiers with the same complexity. Thus the estimated best complexity does not correctly indicate the complexity of fuzzy rule-based classifiers with the highest validation data accuracy. Thus we do not use such a substitute method in this paper.

After the best complexity is estimated in the inner CV loop, we go back to the current run of the MoGFS in the outer CV loop where a number of non-dominated fuzzy rule-based classifiers are obtained by the MoGFS algorithm using the current outer CV training data. A classifier with the most similar complexity to the estimated best complexity is chosen among them. The similarity is measured by the difference in the number of fuzzy rules. When we have two classifiers with the same similarity to the best complexity, we choose the simpler one with less complexity. For example, when the estimated best complexity is 3.5 fuzzy rules, a classifier with three fuzzy rules is selected rather than that with four fuzzy rules. After a single classifier is selected, the error rate of the selected classifier is calculated using test data. A single run of the MoGFS algorithm in the outer CV loop involves all of these procedures (i.e., $t_{In} \times k_{In}$ runs of the MoGFS algorithm for complexity determination in the inner CV loop, a single run of the MoGFS algorithm to design non-dominated classifiers in the outer CV loop, classifier selection using the estimated best complexity, error rate calculation of the selected classifier on test data).

Here we explain the handling of two special situations. One is the handling of overlapping classifiers in the objective space, and the other is the handling of the tiebreak situation in the best complexity determination.

It is possible that multiple non-dominated classifiers with the same objective vector are simultaneously obtained by a single run of the MoGFS algorithm. While those classifiers have the same training data accuracy, they may have different validation data accuracy in the inner CV loop. This is also the case for test data accuracy in the outer CV loop. Thus we calculate the average error rate over the overlapping classifiers. For example, let us assume that five classifiers with the same complexity and the same training data accuracy are simultaneously obtained. These five classifiers are overlapping with each other in the objective space. We also assume that four of them are exactly the same classifier with a validation data error rate of 10% and the other has a validation data error rate of 20%. In this case, these overlapping classifiers are handled as a single classifier with a validation data error rate of

12% (i.e., $(4 \times 10 + 1 \times 20)/5$). Overlapping classifiers are handled in this manner in the outer and inner CV loops.

It is also possible that multiple complexity values have the same best validation data accuracy in the inner CV loop. In this case, we choose the smallest complexity among them in both the FETA method and the FATE method.

4. Computational experiments

4.1. Setting of computational experiments

In our computational experiments, we use 17 data sets in Table 1. All attribute values are normalized into real numbers in the unit interval [0,1]. That is, each pattern is handled as a point in the n-dimensional pattern space $[0,1]^n$ where n is the number of attributes in each data set.

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

$$\textit{Rule } R_q : \text{If } x_1 \text{ is } A_{q1} \text{ and} \ldots \text{and } x_n \text{ is } A_{qn} \text{ then Class } C_q \text{ with } CF_q, \quad (2)$$

where R_q is the label of the qth fuzzy rule, $\mathbf{x} = (x_1, \ldots, x_n)$ 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 (which is a real number in [0,1]). We use 45 antecedent fuzzy sets from fuzzy partitions with granularities 1–9 including "*don't care*". Some of them are shown in Fig. 5(a)–(d) where an integer is attached to each fuzzy set. The unit interval [0,1] is used as a special antecedent fuzzy set "*don't care*" since all attribute values are real numbers in [0,1].

As an MoGFS algorithm, we use our multi-objective fuzzy genetics-based machine learning (MoFGBML) algorithm [25] which is based on a hybrid FGBML algorithm [41] with a Pittsburgh-style algorithm framework and a Michigan-style search mechanism. Each fuzzy rule is coded by its n antecedent fuzzy sets. Thus each classifier with N fuzzy rules is coded by a string of length nN. The integers attached to fuzzy sets in Fig. 5 are used for classifier coding. Our MoFGBML algorithm is applied to the two-objective problem for minimizing the percentage error rate and the number of fuzzy rules as explained in Section 1.

The other parameters on our MoFGBML algorithm were the same as those in [30] except for the number of initial rules (i.e., 30) and the maximum number of rules (i.e., 60) in a string. The replacement probability with "*don't care*" was also different. We specified it as $(n - 5)/n$ in this paper. These changes are based on our preliminary empirical studies.

Table 1
Data sets used in this paper (Available from UCI Database and Keel Project [40]).

Data	Patterns	Attributes	Classes
Appendicitis	106	7	2
Australian	690	14	2
Bands	365	19	2
Bupa	345	6	2
Cleveland	297	13	5
Dermatology	358	34	6
Glass	214	9	6
Haberman	306	3	2
Heart	270	13	2
Mammographic	830	5	2
Pima	768	8	2
Saheart	462	9	2
Sonar	208	60	2
Vehicle	846	18	4
Wdbc	569	30	2
Wine	178	13	3
Wisconsin	683	9	2

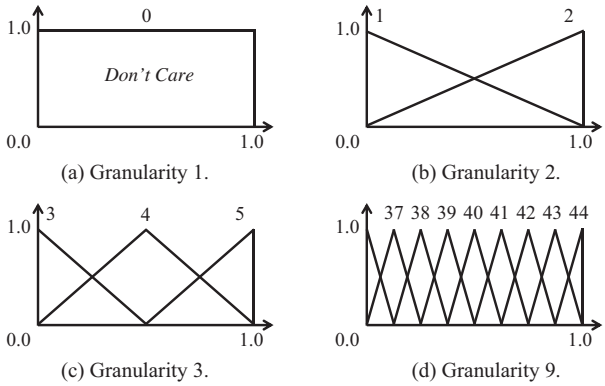


Fig. 5. Examples of fuzzy partitions used in our computational experiments.

Our rdCV approach is implemented as $5 \times 10\text{CV}_{\text{Out}} \times 5 \times 10\text{CV}_{\text{In}}$ where 10CV is iterated five times for the test data accuracy evaluation in the outer CV loop and five times for the best complexity estimation in the inner CV loop. The number of runs of our MoFGBML algorithm is 5×10 in the outer CV loop while it is $50 \times 5 \times 10$ in the inner CV loop (i.e., 2550 runs in total). The class balance is maintained as much as possible in random data partition into ten subsets for each 10CV. In Moreno-Torres et al. [42], they discussed data shift and introduced some data partitioning methods. The comparison with other data partitioning methods would be an interesting future research topic. There is a possibility that we can obtain better classifiers with higher generalization ability. There is also a possibility that we can reduce the number of inner CV iterations using other data partitioning methods.

4.2. Illustration of our rdCV approach

Before reporting experimental results on the 17 data sets, we first explain our rdCV approach (i.e., $5 \times 10\text{CV}_{\text{Out}} \times 5 \times 10\text{CV}_{\text{In}}$) using the heart data set with 270 patterns. In the outer CV loop, the given 270 patterns are divided into ten subsets with 27 patterns for 10CV. In the first run of our MoFGBML algorithm in the outer CV loop, one of the ten subsets is used as test data. The other nine subsets are used as training data (i.e., outer CV training data).

The outer CV training data are further subdivided into ten subsets for 10CV in the inner CV loop. One of the ten subsets of the outer CV training data is used as validation data, and the other nine subsets are used as training data (i.e., inner CV training data). Our MoFGBML algorithm is applied to the inner CV training data to search for non-dominated fuzzy rule-based classifiers. Each of the obtained classifier is evaluated using the validation data. In this manner, 5×10CV is performed for the same outer CV training data in the inner CV loop. Then the best complexity is estimated using the validation error rate of each fuzzy rule-based classifier.

When we use the FETA method, the best complexity is estimated in each run in the 5×10CV in the inner CV loop. In Fig. 6, we show the experimental results of the first two runs. The best complexity is estimated in each run as 4 in Fig. 6(a) and 7 in Fig. 6(b). From the 50 runs, we obtain the average value 7.52 of the estimated best complexity. In Fig. 7, we show the histogram of the 50 values of the estimated best complexity. We can see that the estimated best complexity has a large variety over 50 runs.

When we use the FATE method, first the average validation error rate is calculated for each complexity (i.e., each number of fuzzy rules) over the 50 runs. Then the best complexity with the highest validation data accuracy is estimated. In Fig. 8(a), we show

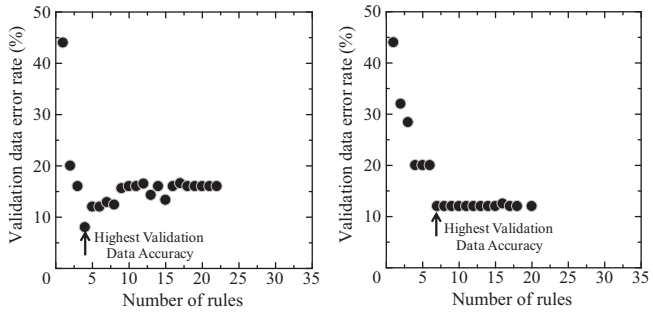


Fig. 6. Results of the first two runs of our MoFGBML algorithm in the inner CV loop.

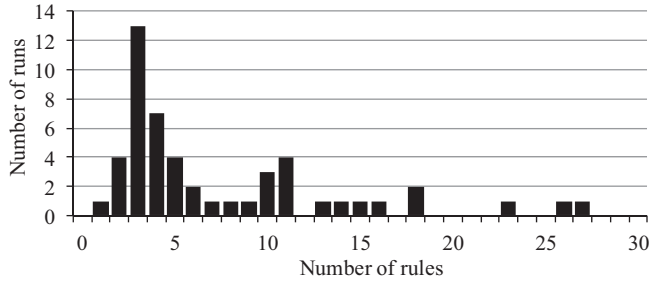


Fig. 7. Histogram of the 50 values of the estimated best complexity.

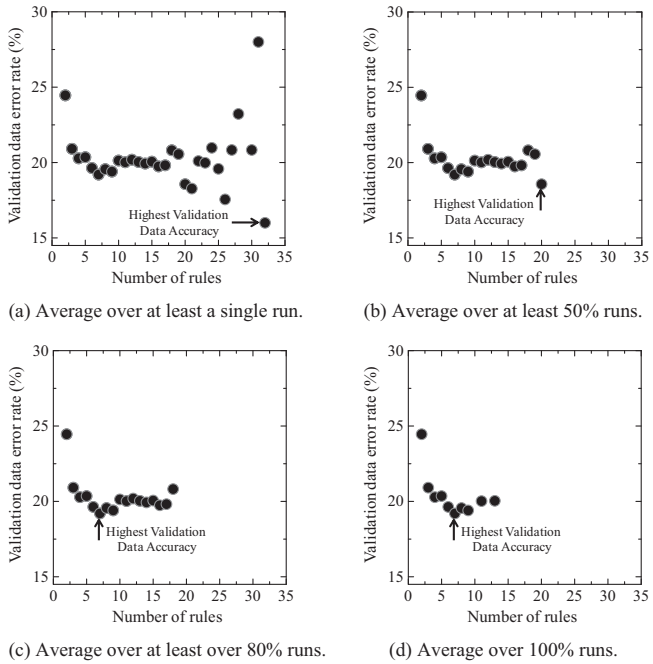


Fig. 8. Average error rates on validation data in the inner CV loop.

the average validation error rate for each complexity. Some average error rates in Fig. 8(a) are calculated over all the 50 runs while others are calculated over only a few runs. In Fig. 8(b)–(d), we show the average error rates calculated over at least α% of the 50 runs for $\alpha = 50$, 80 and 100, respectively. Fig. 8(a) shows all results obtained from at least one of the 50 runs (i.e., $\alpha = 2$). Depending on the value of α, the best complexity is estimated as 32 ($\alpha = 2$ in Fig. 8(a)), 20 ($\alpha = 50$ in Fig. 8(b)), and 7 ($\alpha = 80$ in Fig. 8(c) and $\alpha = 100$ in Fig. 8(d)).

The estimated best complexity is used for choosing a single classifier in the first run of our MoFGBML algorithm in the outer CV loop. In Fig. 9(a), we show the results of the first run in the outer CV loop. One of the obtained 23 fuzzy rule-based classifiers in Fig. 9(a) is selected using the estimated best complexity. For example, when the estimated best complexity is 7.52 (i.e., the results of the FETA method), one classifier with eight fuzzy rules is selected. Then the error rate on the test data of the first run in the outer CV loop is calculated. This completes the first run in the 5×10CV in the outer CV loop. In this manner, 50 runs in the 5×10CV are performed in the outer CV loop. In Fig. 9(b)–(d), we show the average results over the 50 runs in the outer CV loop. As in Fig. 8 (i.e., as in the inner CV loop), a different set of non-dominated classifiers is obtained in each of 50 runs of our MoFGBML algorithm in the outer CV loop. Thus the average results in Fig. 9(b)–(d) are shown in the same manner as in Fig. 8 (i.e., average results over at least a single run, 50% runs, and 100% runs).

4.3. Some other methods for classifier selection

As mentioned in Section 1, a simple method for classifier selection is to choose the classifier with the highest training data accuracy. In Fig. 9(a), the right-most classifier with 27 fuzzy rules is selected. This method is referred to as the highest training data accuracy method. As shown in Fig. 9(b)–(d), usually fuzzy rule-based classifiers do not severely overfit to the training data. Thus the highest training data accuracy method works well in many cases. Of course, the selected classifier also has the highest complexity.

Another method mentioned in Section 1 is to choose the classifier with the highest validation data accuracy. In this method, a part of the training data in the outer CV loop are used as validation data as in the inner CV loop of our rdCV-based two methods. However, the validation data accuracy is directly used for choosing the classifier with the highest validation data accuracy (not for estimating the best complexity). When 10% of the training data are used as validation data, this method is the same as choosing the classifier with the highest validation data accuracy in the first

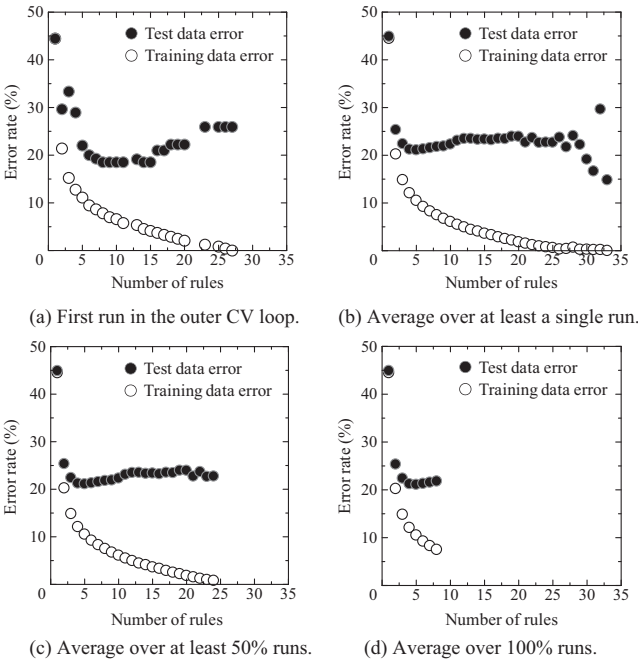


Fig. 9. Results in the outer CV loop.

run in the inner CV loop. That is, the classifier with four rules in Fig. 6(a) is selected. In other words, the first run in the inner CV loop of our rdCV approach is used as the first run in the outer CV loop in this approach. Actually, no inner CV loop is needed in this approach. This approach is referred to as the single train-validation method. As shown in Fig. 7, classifier selection based on 10% validation data often chooses totally different classifiers with a large variety in their complexity. This is because each of the obtained classifiers is evaluated by only 10% of the training data when a single classifier is selected. That is, classifier selection is performed using only a single validation data set, which is 10% of the training data. In our rdCV approach, the best complexity estimation is performed using average results over a large number of different validation data sets in the inner CV loop. As a result, the variety of complexity among the chosen classifiers by our approach is much smaller than that by classifier selection based on a single validation data set.

4.4. Experimental results

We compared five classifier selection methods: our rdCV-based FETA method, our rdCV-based FATE method, the highest training data accuracy method, the single train-validation method, and the median complexity method [43].

In the FATE method, the value of α was specified as $\alpha = 80$. This means that the average validation data error rate was calculated only when fuzzy rule-based classifiers are obtained for a particular complexity from at least 80% runs in the inner CV loop (i.e., 40 out of 50 runs in the 5×10CV in the inner CV loop).

In the single train-validation method, the outer CV training data were randomly divided into 90% training data and 10% validation data. A classifier with the highest validation data accuracy was selected from non-dominated classifiers obtained from the 90% training data. Except for this single train-validation method, all the outer CV training data were used to design non-dominated classifiers from which a final classifier was selected.

In the median complexity method, a classifier with the median complexity among the obtained non-dominated classifiers was chosen (for details, see [43]). Only the two methods of our rdCV approach have the inner CV loop. The same outer CV loop with 5×10CV was used for all the five classifier selection methods. Average test data error rates are summarized for the 17 data sets in Table 2.

In Table 2, the lowest error rate is shown by boldface for each data set. We can see from Table 2, the highest test data accuracy was obtained for ten data sets from one of our two rdCV-based methods. For four data sets, the highest test data accuracy was obtained from the highest training data accuracy method. This observation suggests that our MoFGBML algorithm did not overfit to training data of those data sets.

We can also see from Table 2 that the difference in the average test data error rates between the highest training data accuracy method and the median complexity method is small for most data sets. This observation suggests that the complexity of highest training data accuracy classifiers can be decreased without severely deteriorating their generalization ability (e.g., see Fig. 9(b)).

For most data sets, better results were obtained from our rdCV-based two methods than the single train-validation method. Especially for small data sets such as the appendicitis, glass, and sonar data sets, we can observe large differences between our rdCV-based methods and the single train-validation method. This observation suggests that multiple runs of the train-validation procedure with different data partitions in our rdCV methods can somewhat alleviate the difficulty in the single train-validation classifier selection method for small data sets.

We perform a Wilcoxon signed-ranks test [44] with the average error rates for each method. Table 3 shows R+/R−/p-value for each pairwise comparison. The statistical difference (i.e., p-value < 0.1) is highlighted in bold face. The statistical test supports that our rdCV methods outperform alternative methods with respect to test data accuracy. It also shows our rdCV methods are not statistically different.

In Table 4, we summarize the average number of fuzzy rules in the selected fuzzy rule-based classifiers. This table shows that the size of fuzzy rule-based classifiers selected by our rdCV-based two methods is about 1/3 of the highest training data accuracy classifiers on average.

In Fig. 10, the five methods are compared in the accuracy-complexity space using the average test data error rate and the average number of fuzzy rules over the 17 data sets (i.e., the average results shown in the bottom row of Tables 2 and 4). This figure explains the characteristic features of each method. For example, we can see that the highest training data accuracy method finds accurate classifiers with many fuzzy rules while the median complexity method finds simpler classifiers. Fig. 10 also shows that our rdCV-based methods dominate the other three methods in the accuracy-complexity space.

From Fig. 10, one may think why the median complexity method was worse than the highest training data accuracy method with respect to the average test data error rate. Of course, the highest

Table 2
Average error rates on test data.

	Our rdCV FETA	Our rdCV FATE	Single Validation	Median Complexity	Highest Training
Appendicitis	**16.08**	17.02	19.96	17.16	18.83
Australian	14.87	**14.69**	15.29	15.16	15.48
Bands	32.50	32.63	34.80	**32.45**	32.70
Bupa	34.84	35.12	34.53	34.38	**33.90**
Cleveland	**45.83**	46.44	46.80	46.59	45.94
Dermatology	5.17	**5.12**	6.33	13.81	5.72
Glass	34.28	33.62	37.23	34.54	**33.59**
Haberman	26.95	26.52	**26.39**	27.45	27.90
Heart	**21.72**	22.13	22.08	23.35	23.63
Mammographic	18.26	18.15	**18.10**	18.56	19.53
Pima	**26.13**	26.45	26.23	26.70	26.93
Saheart	32.04	**31.84**	35.21	33.16	33.86
Sonar	25.63	25.45	28.26	25.36	**24.86**
Vehicle	**30.43**	31.03	31.30	30.69	30.69
Wdbc	5.73	**5.26**	6.24	5.76	5.49
Wine	10.60	10.60	10.71	33.35	**10.30**
Wisconsin	**3.84**	3.87	4.36	4.15	4.16
Average	**22.64**	22.70	23.75	24.86	23.15

Table 3
Wilcoxon signed-ranks test. Results are presented as R+/R−/*p*-Value.

	Our rdCV FETA	Our rdCV FATE	Single validation	Median complexity	Highest training
Our rdCV FETA	–	75/61/0.717	**137/16/0.004**	**137/16/0.004**	**114/39/0.076**
Our rdCV FATE	61/75/0.717	–	**132/21/0.008**	**130/23/0.011**	**112/41/0.092**
Single validation	**16/137/0.004**	**21/132/0.008**	–	63/90/0.523	47/106/0.163
Median complexity	**16/137/0.004**	**23/130/0.011**	90/63/0.523	–	62/74/0.756
Highest training	**39/114/0.076**	**41/112/0.092**	106/47/0.163	74/62/0.756	–

Table 4
Average number of fuzzy rules.

Data	Our rdCV FETA	Our rdCV FATE	Single validation	Median complexity	Highest training
Appendicitis	2.46	2.96	1.66	3.30	7.70
Australian	9.00	6.34	10.72	15.52	37.18
Bands	16.56	18.72	14.12	19.48	45.88
Bupa	13.82	12.82	14.42	14.88	39.12
Cleveland	14.24	18.52	8.04	24.86	53.54
Dermatology	6.62	7.30	6.26	4.74	10.56
Glass	14.22	17.90	11.90	13.66	32.74
Haberman	4.34	2.24	3.94	7.32	16.80
Heart	7.26	10.82	6.20	11.52	25.90
Mammographic	4.92	3.84	4.88	8.98	19.74
Pima	15.12	12.38	12.28	21.20	50.34
Saheart	10.62	3.52	54.22	22.74	52.30
Sonar	6.28	8.40	5.52	6.42	13.68
Vehicle	21.40	26.30	18.78	21.10	48.14
Wdbc	4.72	5.76	3.94	4.42	9.88
Wine	3.14	3.34	3.18	2.10	4.68
Wisconsin	4.76	6.50	3.44	6.38	13.86
Average	9.38	9.86	10.79	12.27	28.36

training data accuracy method is more likely to suffer from the overfitting to the training data than the medium complexity method. In this sense, the medium complexity method can be a good heuristic to choose a single classifier with high generalization ability when classifiers with high complexity severely overfits to training data as explained in Fig. 1(b). However, this heuristics does not always work well. For example, in Table 4, the average number of fuzzy rules in the selected classifiers by the medium complexity method is smaller than the number of classes in the dermatology and wine data sets. This means that the complexity of the selected classifiers for these data sets is too small.

4.5. Parameter specifications in our rdCV approach

In our rdCV approach, the best complexity is estimated in the inner CV loop using $t_{\mathrm{In}} \times k_{\mathrm{In}}\mathrm{CV}_{\mathrm{In}}$. Thus the performance of our rdCV approach may strongly depend on the specifications of the two parameters t_{In} and k_{In}. We examined the performance of the FETA method for the following specifications in the inner CV loop: 5×10CV, 4×10CV, 3×10CV, 2×10CV, 1×10CV, 5×5CV, and 5×2CV. Experimental results are shown in Fig. 11(a) and (b).

From Fig. 11(a), we can see that the number of iterations of 10CV (i.e., the specification of t_{In} in the $t_{\mathrm{In}} \times k_{\mathrm{In}}\mathrm{CV}_{\mathrm{In}}$) does not have a large effect on the performance of the FETA method. Actually, the difference in the experimental results between 1×10CV and 5×10CV in the inner CV loop is not statistically significant when we use a Wilcoxon signed-ranks test with $\alpha = 0.10$. The *p*-value is 0.687. However, the number of data partitions (i.e., the specification of k_{In} in the $t_{\mathrm{In}} \times k_{\mathrm{In}}\mathrm{CV}_{\mathrm{In}}$) has a large effect in Fig. 11(b). For example, when we used 5×2CV instead of 5×10CV in the inner CV loop, simpler and less accurate fuzzy rule-based classifiers were selected. This may be because only 50% of the outer CV training data were used as training data in the inner 5×2CV. That is, there is a large difference in the size of training data between the inner and outer CV loops. When 5×10CV was used in the inner CV loop, 90% of the outer CV training data were used as training data in the inner CV loop. In this case, the size of training data is not so different between the inner and outer CV loops. Similar observations are obtained from Fig. 12(b) for our rdCV-based FATE method.

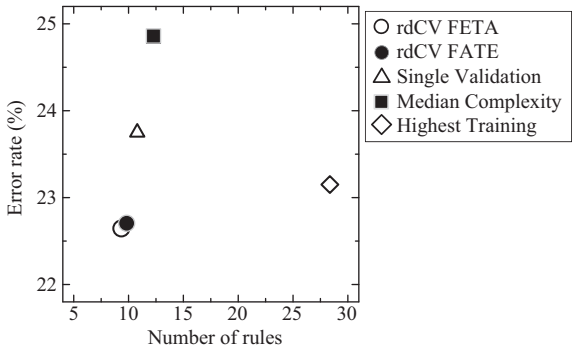
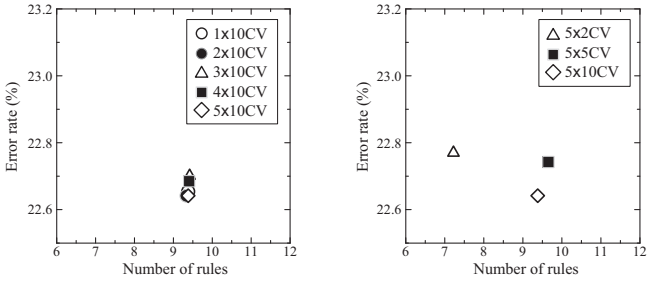
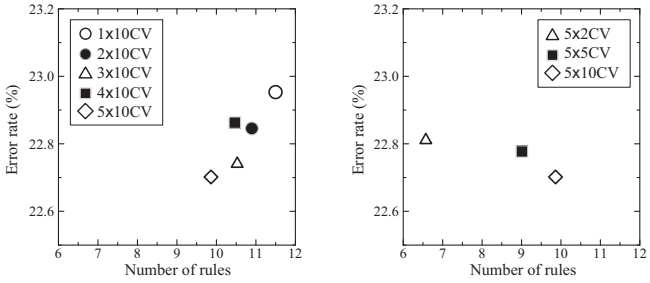


Fig. 10. The average test data error rate and the average number of fuzzy rules over the 17 data sets by the five classifier selection methods.



(a) Effects of the number of iterations. (b) Effects of the number of partitions.

Fig. 11. The average test data error rate and the average number of fuzzy rules over the 17 data sets by our rdCV-based FETA method.



(a) Effects of the number of iterations. (b) Effects of the number of partitions.

Fig. 12. The average test data error rate and the average number of fuzzy rules over the 17 data sets by our rdCV-based FATE method.

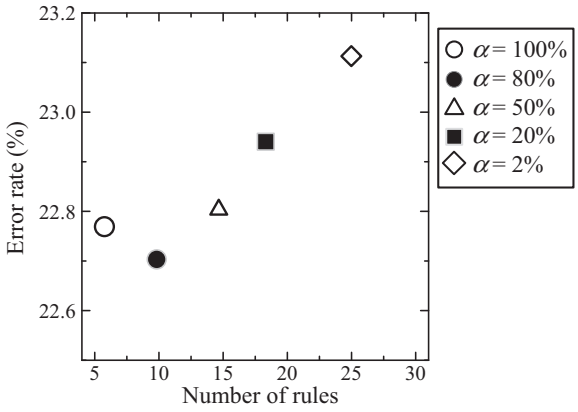


Fig. 13. Results of our rdCV-based FATE method with different specifications of α.

In our rdCV-based FATE method, we have another important parameter α. Only when fuzzy rule-based classifiers with a particular complexity are obtained at least α% runs in the inner CV loop, the average validation data error rate is calculated for the complexity in this method. In Fig. 13, we show experimental results for five specifications of α in the inner 5×10CV: $\alpha = 2\%$ (1 run), 20% (10 runs), 50% (25 runs), 80% (40 runs), 100% (50 runs). We can see from Fig. 13 that the increase in the value of α decreased the average number of fuzzy rules in the selected classifiers. This is because fuzzy rule-based systems with a large number of fuzzy rules were not obtained in many runs in the inner CV loop (i.e., the average validation error rate was not calculated for such a large number of fuzzy rules when the value of α is large).

5. Repeated double CV for other settings in MoGFS

It is not likely that a single algorithm with a fixed parameter specification always works well on various data sets. A different algorithm with a different parameter specification may be needed for a different data set. In this section, we discuss how our rdCV approach can be used for parameter specification, formulation selection and algorithm choice.

Our idea is to examine different parameter specifications, different formulations and different algorithms in the inner CV loop of our rdCV approach. It should be noted that we cannot use test data for these tasks. Only after a single classifier is obtained, test data are used to evaluate the obtained classifier. Various parameter values, formulations and algorithms can be compared in the inner CV loop without using test data. It should be also noted that our rdCV approach is not used for comparing different MoGFS algorithms but used for choosing an appropriate MoGFS algorithm for each data set. A different MoGFS algorithm will be chosen for a different data set. The inner CV loop plays a role of a manager to utilize a team of different MoGFS algorithms.

In general, MoGFS algorithms have a number of parameters to be pre-specified. For example, one important parameter in our MoFGBML is the range of granularities of fuzzy partitions. In the previous section, we used fuzzy partitions with granularities 1–9. This is not necessarily the best specification for all data sets. We can use other specifications.

In addition to granularities 1–9, let us examine granularities 1–3, 1–5 and 1–7. In our rdCV approach, we can examine each specification in the inner CV loop. Then we can choose the best specification together with the best complexity for each run in the outer CV loop. Let us explain how our rdCV-based FETA method works for granularities specification through computational experiments on the heart data set with 270 patterns using the FETA method.

As in the previous section, we use 5×10CV in the outer CV loop and 5×10CV in the inner CV loop. The inner CV loop is used to choose one of the three specifications together with the best complexity. For each run of our MoFGBML algorithm in the outer CV loop, the 5×10CV in the inner CV loop is applied to the outer CV training data in the same manner as in the previous section for each of the three specifications: 1–3, 1–5 and 1–7. In each run in the inner CV loop, the best error rate on validation data and the best complexity are identified as shown in the previous section. The estimated validation data accuracy is obtained by calculating the average value of the best error rate over 50 runs in the inner CV loop for each specification. Using the estimated validation data accuracy for each specification, the best specification can be selected (i.e., the specification with the highest estimated validation data accuracy is selected). In this manner, one of the three specifications is selected for each of 50 runs in the outer CV loop.

Among 50 runs of our MoFGBML in the outer CV loop, granularities 1–3, 1–5, 1–7 and 1–9 are selected in 39, 7, 1 and 3 runs, respectively. When we use granularities 1–3, 1–5, 1–7 and 1–9 in all the 50 runs, the average error rate on test data is calculated as 19.91%, 22.00%, 20.63% and 21.72%, respectively. We can see from these results that our approach selects granularities with lower error rates more often. It should be noted that we cannot use test data error rates for parameter specifications.

Since the estimated validation data accuracy is obtained together with the estimated best complexity for each of different settings, we can choose the best setting with the best validation data accuracy. This idea can be used for various specifications in MoGFS (e.g., the specification of the shape of membership functions, the choice of an EMO algorithm, the specification of a termination condition of MoGFS, the choice of a multiobjective formulation of classifier design).

6. Conclusions

In this paper, we discussed classifier selection from a number of non-dominated fuzzy rule-based classifiers obtained by a MoGFS algorithm. For classifier selection and test data accuracy evaluation, we proposed the use of repeated double Cross-Validation (rdCV). In the inner CV loop of our rdCV approach, the MoGFS algorithm was applied to the training data to design a number of non-dominated fuzzy rule-based classifiers. The validation data accuracy and the complexity of each non-dominated classifier were examined. This train-validation procedure was iterated for different data partitions. We proposed two methods for specifying the best complexity with the highest validation data accuracy using experimental results in the inner CV loop with multiple runs of the train-validation procedure. The main characteristic of our rdCV-based approach is to use validation data accuracy for complexity determination instead of classifier selection. As a result, multiple runs can be performed in the inner CV loop for complexity determination.

We examined the performance of our rdCV-based two methods for classifier selection through computational experiments on 17 data sets. Our rdCV-based two methods were compared with the highest training data accuracy method, the single train-validation method and the median complexity method. For ten data sets out of the 17 data sets, the best test data accuracy was obtained from one of our rdCV-based two methods. One future research issue is the handling of three-objective formulations (e.g., [23,45,46]) in our rdCV approach. Examination of our rdCV approach in more advanced MoGFS algorithms with various tuning mechanisms (e.g., [43]) is also an interesting future research issue. In our rdCV approach, classifier selection was performed for generalization ability maximization with no intervention of the decision

maker. Classifier selection based on the decision maker's preference is an interesting and promising future research issue where we should take into account not only the generalization ability of fuzzy rule-based classifiers but also their complexity and interpretability.

References

[1] H. Ishibuchi, K. Nozaki, N. Yamamoto, H. Tanaka, Construction of fuzzy classification systems with rectangular fuzzy rules using genetic algorithms, Fuzzy Sets Syst. 65 (2/3) (1994) 237–253.

[2] H. Ishibuchi, K. Nozaki, N. Yamamoto, H. Tanaka, Selecting fuzzy if-then rules for classification problems using genetic algorithms, IEEE Trans. Fuzzy Syst. 3 (3) (1995) 260–270.

[3] H. Ishibuchi, T. Murata, I.B. Turksen, Selecting linguistic classification rules by two-objective genetic algorithms, in: Proc. of 1995 IEEE International Conference on Systems, Man and Cybernetics, Vancouver, Canada, October 1995, pp. 1410–1415.

[4] H. Ishibuchi, T. Murata, I.B. Turksen, Single-objective and two-objective genetic algorithms for selecting linguistic rules for pattern classification problems, Fuzzy Sets Syst. 89 (2) (1997) 135–150.

[5] H. Ishibuchi, T. Murata, M. Gen, Performance evaluation of fuzzy rule-based classification systems obtained by multi-objective genetic algorithms, Computers Ind. Eng. 35 (3–4) (1998) 575–578.

[6] K. Deb, Multi-Objective Optimization using Evolutionary Algorithms, John Wiley & Sons, Chichester, 2001.

[7] C.A. Coello Coello, D.A. Van Veldhuizen, G.B. Lamont, Evolutionary Algorithms for Solving Multi-Objective Problems, Kluwer Academic Publishers, 2002.

[8] K.C. Tan, E.F. Khor, T.H. Lee, Multiobjective Evolutionary Algorithms and Applications, Springer, Berlin, 2005.

[9] H. Ishibuchi, Multiobjective genetic fuzzy systems: review and future research directions, in: Proc. of 2007 IEEE International Conference on Fuzzy Systems, London, UK, July 2007, pp. 913–918.

[10] H. Ishibuchi, Y. Nojima, Multiobjective genetic fuzzy systems, in: L. Mumford, L.C. Jain (Eds.), Computational Intelligence: Collaboration, Fusion and Emergence, Springer, Berlin, 2009, pp. 131–173.

[11] M. Fazzolari, R. Alcalá, Y. Nojima, H. Ishibuchi, F. Herrera, A review of the application of multi-objective evolutionary fuzzy systems: current status and further directions, IEEE Trans. Fuzzy Syst. 21 (1) (2013) 45–65.

[12] O. Cordón, F. Gomide, F. Herrera, F. Hoffmann, L. Magdalena, Ten years of genetic fuzzy systems: current framework and new trends, Fuzzy Sets Syst. 141 (1) (2004) 5–31.

[13] F. Herrera, Genetic fuzzy systems: Status, critical considerations and future directions, Int. J. Comput. Intel. Res. 1 (1) (2005) 59–67.

[14] F. Herrera, Genetic fuzzy systems: taxonomy, current research trends and prospects, Evol. Intel. 1 (1) (2008) 27–46.

[15] O. Cordón, A historical review of evolutionary learning methods for Mamdani-type fuzzy rule-based systems: designing interpretable genetic fuzzy systems, Int. J. Approx. Reason. 52 (6) (2011) 894–913.

[16] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput. 6 (2) (2002) 182–197.

[17] E. Zitzler, L. Thiele, Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach, IEEE Trans. Evol. Comput. 3 (4) (1999) 257–271.

[18] S.N. Qasem, S.M. Shamsuddin, A.M. Zain, Multi-objective hybrid evolutionary algorithms for radial basis function neural network design, Knowledge-Based Syst. 27 (2012) 475–497.

[19] J. Pacheco, S. Casado, F. Angel-Bello, A. Álvarez, Bi-objective feature selection for discriminant analysis in two-class classification, Knowledge-Based Syst. 44 (2013) 57–64.

[20] C.K. Goh, E.J. Teoh, K.C. Tan, Hybrid multiobjective evolutionary design for artificial neural networks, IEEE Trans. Neural Netw. 19 (9) (2008) 1531–1548.

[21] H.M. Zhao, A multi-objective genetic programming approach to developing Pareto optimal decision trees, Decision Support Syst. 43 (3) (2007) 809–826.

[22] Y.C. Jin, B. Sendhoff, Pareto-based multiobjective machine learning: an overview and case studies, IEEE Trans. Syst. Man Cybern.: Part C – Appl. Rev. 38 (3) (2008) 397–415.

[23] H. Ishibuchi, T. Nakashima, T. Murata, Three-objective genetics-based machine learning for linguistic rule extraction, Inform. Sci. 136 (1–4) (2001) 109–133.

[24] H. Ishibuchi, T. Yamamoto, Fuzzy rule selection by multi-objective genetic local search algorithms and rule evaluation measures in data mining, Fuzzy Sets Syst. 141 (1) (2004) 59–88.

[25] H. Ishibuchi, Y. Nojima, Analysis of interpretability-accuracy tradeoff of fuzzy systems by multiobjective fuzzy genetics-based machine learning, Int. J. Approx. Reason. 44 (1) (2007) 4–31.

[26] R. Muñoz-Salinas, E. Aguirre, O. Cordón, M. García-Silvente, Automatic tuning of a fuzzy visual system using evolutionary algorithms: single-objective versus multiobjective approaches, IEEE Trans. Fuzzy Syst. 16 (1) (2008) 485–501.

[27] P. Ducange, B. Lazzerini, F. Marcelloni, Multi-objective genetic fuzzy classifiers for imbalanced and cost-sensitive datasets, Soft Comput. 14 (7) (2010) 713–728.

[28] E. Zitzler, L. Thiele, M. Laumanns, C.M. Fonseca, V.G. da Fonseca, Performance assessment of multiobjective optimizers: an analysis and review, IEEE Trans. Evol. Comput. 7 (2) (2003) 117–132.

[29] P. Filmoser, B. Liebmann, K. Varmuza, Repeated double cross validation, J. Chemometr. 23 (3–4) (2009) 160–171.

[30] H. Ishibuchi, Y. Nakashima, Y. Nojima, Double cross-validation for performance evaluation of multi-objective genetic fuzzy systems, in: Proc. of 5th IEEE International Workshop on Genetic and Evolutionary Fuzzy Systems, Paris, France, April 11–15, 2011, pp. 31–38.

[31] S.M. Weiss, C.A. Kulikowski, Computer Systems That Learn, Morgan Kaufmann Publishers, San Mateo, 1991.

[32] B. Efron, R.J. Tibshirani, An Introduction to the Bootstrap, Chapman & Hall, London, 1993.

[33] J.S. Urban Hjorth, Computer Intensive Statistical Methods: Validation, Model Selection, and Bootstrap, Chapman & Hall, London, 1994.

[34] B. Efron, R.J. Tibshirani, Improvements on cross-validation: the. 632+ bootstrap method, J. Am. Stat. Assoc. 92 (438) (1997) 548–560.

[35] B. Liebmann, A. Friedl, K. Varmuza, Determination of glucose and ethanol in bioethanol production by near infrared spectroscopy and chemometrics, Anal. Chim. Acta 642 (1–2) (2009) 171–178.

[36] L. Yetukuri, J. Tikka, J. Hollmén, M. Orešič, Functional prediction of unidentified lipids using supervised classifiers, Metabolomics 6 (1) (2010) 18–26.

[37] B. Liebmann, A. Friedla, K. Varmuzaa, Applicability of near-infrared spectroscopy for process monitoring in bioethanol production, Biochem. Eng. J. 52 (2–3) (2010) 187–193.

[38] K. Varmuza, P. Filzmoser, Introduction to Multivariate Statistical Analysis in Chemometrics, CRC Press, Boca Raton, 2009.

[39] The R Project for Statistical Computing. <http://www.r-project.org>.

[40] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera, KEEL data-mining software tool: data set repository, integration of algorithms and experimental analysis framework, J. Multiple-Valued Logic Soft Comput. 17 (2–3) (2010) 255–287.

[41] H. Ishibuchi, T. Yamamoto, T. Nakashima, Hybridization of fuzzy GBML approaches for pattern classification problems, IEEE Trans. Syst. Man, Cybern. – Part B Cybern. 35 (2) (2005) 359–365.

[42] J.G. Moreno-Torres, J.A. Sáez, F. Herrera, Study on the impact of partition-induced dataset shift on k-fold cross-validation, IEEE Trans. Neural Netw. Learn. Syst. 23 (8) (2012) 1304–1312.

[43] R. Alcalá, P. Ducange, F. Herrera, B. Lazzerini, F. Marcelloni, A multiobjective evolutionary approach to concurrently learn rule and data bases of linguistic fuzzy rule-based systems, IEEE Trans. Fuzzy Syst. 17 (5) (2009) 1106–1112.

[44] D.J. Sheskin, Handbook of Parametric and Nonparametric Statistical Procedures, fifth ed., Chapman and Hall, CRC, 2007.

[45] J.M. Alonso, L. Magdalena, O. Cordon, Embedding HILK in a three-objective evolutionary algorithm with the aim of modeling highly interpretable fuzzy rule-based classifiers, in: Proc. of 4th International Workshop on Genetic and Evolutionary Fuzzy Systems: GEFS 2010, Mieres, Spain, March 2010, pp. 15–20.

[46] R. Cannone, J.M. Alonso, L. Magdalena, Multi-objective design of highly interpretable fuzzy rule-based classifiers with semantic cointension, in: Proc. of 5th IEEE International Workshop on Genetic and Evolutionary Fuzzy Systems: GEFS 2011 under SSCI 2011, Paris, France, April 2011, pp. 1–8.