International Journal of Computational Intelligence Systems, Vol. 5, No. 2 (April, 2012), 231-253

A Study on the Use of Multiobjective Genetic Algorithms for Classifier Selection in FURIA-based Fuzzy Multiclassifiers

Krzysztof Trawiński , Oscar Cordón , Arnaud Quirin

European Centre for Soft Computing. Mieres, Spain
Edificio Científico-Tecnológico
Calle Gonzalo Gutiérrez Quirós S/N
33600 Mieres, Asturias
E-mail: {krzysztof.trawinski, oscar.cordon, arnaud.quirin}@softcomputing.es

Received 22 November 2010
Accepted 1 April 2011

Abstract

In a preceding contribution, we conducted a study considering a fuzzy multiclassifier system (MCS) design framework based on Fuzzy Unordered Rule Induction Algorithm (FURIA). It served as the fuzzy rule classification learning algorithm to derive the component classifiers considering bagging and feature selection. In this work, we integrate this approach under the overproduce-and-choose strategy. A state-of-the-art evolutionary multiobjective algorithm, namely NSGA-II, is used to provide a component classifier selection and improve FURIA-based fuzzy MCS. We propose five different fitness functions based on three different optimization criteria, accuracy, complexity, and diversity. Twenty UCI high dimensional datasets were considered in order to conduct the experiments. A combination between accuracy and diversity criteria provided very promising results, becoming competitive with classical MCS learning methods.

Keywords: Fuzzy rule-based multiclassification systems, bagging, FURIA, genetic selection of individual classifiers, diversity measures, evolutionary multiobjective optimization, NSGA-II

1. Introduction

Multiclassification systems (MCSs) (also called multiclassifiers or classifier ensembles) have been shown as very promising tools to improve the performance of single classifiers when dealing with complex, high dimensional classification problems in the last few years [1]. This research topic has become especially active in the classical machine learning area, considering decision trees or neural networks to generate the component classifiers, but also some work has been done recently using different kinds of fuzzy classifiers [2, 3, 4, 5, 6, 7, 8].

In our previous studies [9, 10, 11, 12], we proposed a MCS methodology based on classical MCS design techniques such as bagging and feature selection with a fuzzy rule-based system (FRBCS) as a base classifier. As a consequence, fuzzy rule-based multiclassification systems (FRBMCSs) were incorporated into an overproduce-and-choose strategy (OCS) [13]. This MCS desing algorithm is based on the generation of a large number of component classifiers, and a subsequent selection of the subset of them best cooperating. As the main tool we used a multicriteria genetic algorithm (GA) for static component classifier selection guided by several fitness functions based on training error and likelihood, as well as bicriteria fitness functions based on

training error and likelihood or diversity measures. The resulting FRBMCS design approach thus belong to the genetic fuzzy systems (GFSs) family, one of the most successful approaches to hybridize fuzzy systems with learning and adaptation methods in the last fifteen years [14, 15, 16].

In [17] we extended our previous developments by proposing a fuzzy MCS framework based on Fuzzy Unordered Rule Induction Algorithm (FURIA) [18, 19] as the fuzzy rule classification learning algorithm to derive the component classifiers considering bagging and feature selection. We conducted comprehensive experiments with 20 datasets taken from the UCI machine learning repository and provided a deep study of the results obtained. Several FURIA-based fuzzy MCS composition designs were tested including bagging, feature selection, and the combination of bagging and feature selection. We considered three different types of feature selection algorithms: random subspace [20], mutual information-based feature selection (MIFS) [21], and the random-greedy feature selection based on MIFS and the GRASP approach [22]. Finally, our approach was compared against two state-of-the-art MCS algorithms (random forests and bagging decion trees) and also with an application of the fuzzy MCS generation approach with other, less powerful fuzzy classifier derivation method [23, 9]. From the obtained results, we drew the conclusion that FURIA-based fuzzy MCSs were a very powerful tool for dealing with high dimensional classification problems.

Even so, we think that the performance of the latter FURIA-based MCS framework can be improved with an OCS approach based on an evolutionary multiobjective (EMO) algorithm [24] considering diversity measures. In the current work we integrate FURIA-based fuzzy MCSs within the OCS strategy. Since there are many optimization criteria considered for MCS design such as accuracy, complexity, and diversity measures [1, 25, 26, 27], the use of a EMO algorithm came naturally to our mind.

In this paper, we study the behavior of FURIA-based fuzzy MCSs with large size ensembles. To do so, we consider the state-of-the-art NSGA-II algorithm [28] to perform classifier selection. Introducing diversity and complexity measures combined with error measures is an interesting approach, which has led to promising results in the area [12, 25, 26, 27, 29, 30]. Hence, we have embedded three measures of this kind in the objective space of the fitness function combining them with an accuracy index, which resulted in five different bicriteria fitness functions.

We think that such GFS may lead to high quality fuzzy MCSs with a good accuracy-complexity tradeoff. To check this assumption, we present experiments on twenty high dimensional datasets from the UCI machine learning repository.

This paper is set up as follows. In the next section, a state of the art about MCSs, fuzzy MCSs, and MCS selection is presented. Sec. 3 recalls FURIA and our approach for designing FURIA-based fuzzy MCSs, while Sec. 4 describes the proposed NSGA-II for component classifier selection focusing on the different two-objective fitness functions to be considered. The experiments developed and their analysis are shown in Sec. 5. Finally, Sec. 6 collects some concluding remarks and future research lines.

2. Background and related work

This section explores the current literature related to the generation of a FRBMCS. The techniques used to generate MCSs and fuzzy MCSs are described in Sec. 2.1 and 2.2, respectively. Some ways to reduce the size of the ensembles are described in Sec. 2.3. The use of GAs for this purpose is then explored in Sec. 2.4.

2.1. Related work on MCSs

A MCS is the result of the combination of the outputs of a group of individually trained classifiers in order to get a system that is usually more accurate than any of its single components [1]. These kinds of methods have gained a large acceptance in the machine learning community during the last two decades due to their high performance. Decision trees are the most common classifier structure considered and much work has been done in the topic [31, 32], although they can be used with any other

type of classifiers (the use of neural networks is also very extended, see for example [33]).

There are different ways to design a classifier ensemble. On the one hand, there is a classical group of approaches considering *data resampling* to obtain different training sets to derive each individual classifier. In *bagging* [34], they are independently learnt from resampled training sets ("bags"), which are randomly selected with replacement from the original training data set. *Boosting* methods [35] sequentially generate the individual classifiers (weak learners) by selecting the training set for each of them based on the performance of the previous classifier(s) in the series. Opposed to bagging, the resampling process gives a higher selection probability to the incorrectly predicted examples by the previous classifiers.

On the other hand, a second group can be found comprised by a more diverse set of approaches which induct the individual classifier diversity using some ways different from resampling [36]. Feature selection plays a key role in many of them where each classifier is derived by considering a different subset of the original features [27, 37]. *Random subspace* [20], where each feature subset is randomly generated, is one of the most representative methods of this kind.

Finally, there are some advanced proposals that can be considered as *combinations of the two groups*, such as *random forests* [38].

The interested reader is referred to [32, 33] for two reviews for the case of decision tree (both) and neural network ensembles (the latter), including exhaustive experimental studies.

2.2. Previous Work on Fuzzy MCSs

Focusing on fuzzy MCSs, the use of boosting for the design of fuzzy classifier ensembles has been considered in some works [2, 3, 39, 40]. However, only a few contributions for bagging fuzzy classifiers have been proposed considering fuzzy neural networks (together with feature selection) [41], neuro-fuzzy systems [5], and fuzzy decision trees [8, 7] as component classifier structures.

Especially worth mentioning is the contribution [8]. This approach hybridizes Breimann's idea of random forests [38] with fuzzy decision trees [42]. Such resulting fuzzy random forest combines characteristics of MCSs with randomness and fuzzy logic in order to obtain a high quality system joining robustness, diversity, and flexibility to not only deal with traditional classification problems but also with imperfect and noisy datasets. The results show that this approach obtains good performance in terms of accuracy for all the latter problem kinds.

Some advanced GFS-based contributions should also be remarked. On the one hand, an FRBCS ensemble design technique is proposed in [43] considering some niching GA-based feature selection methods to generate the diverse component classifiers, and another GA for classifier fusion by learning the combination weights. On the other hand, another interval and fuzzy rule-based ensemble design method using a single- and multiobjective genetic selection process is introduced in [44, 6]. In this case, the coding scheme allows an initial set of either interval or fuzzy rules, considering the use of different features in their antecedents, to be distributed among different component classifiers trying to make them as diverse as possible by means of two accuracy and one entropy measures. Besides, the same authors presented a previous proposal in [45], where an EMO algorithm generated a Pareto set of FRBCSs with different accuracy-complexity tradeoffs to be combined into an ensemble.

2.3. Determination of the Optimal Set of Component Classifiers in the MCS

Typically, an ensemble of classifiers is post-processed in such a way only a subset of them are kept for the final decision. It is a well known fact that the size of this MCS is an important issue for its tradeoff between accuracy and complexity [32, 33] and that most of the error reduction occurs with the first few additional classifiers [34, 33]. Furthermore, the selection process also participates in the elimination of the duplicates or the poor-performing classifiers.

While in the first studies on MCSs a very small number (around ten) of component classifiers was considered as appropriate to sufficiently reduce the test set prediction error, later research on boosting

(that also holds for bagging) suggested that error can be significantly reduced by largely exceeding this number [46]. This has caused the use of very large ensemble sizes (for example comprised by 1,000 individual classifiers) in the last few years [32].

Hence, the determination of the optimal size of the ensemble is an important issue for obtaining both the best possible accuracy in the test data set without overfitting it, and a good accuracy-complexity trade-off. In pure bagging and boosting approaches, the optimal ensembles are directly composed of all the individual classifiers generated until a specific stopping point, which is determined according to different means (validation data set errors, likelihood, ...). For example, in [32] it is proposed an heuristic method to determine the optimal number guided by the *out-of-bag* error.

However, there is the chance that the optimal ensemble is not comprised by all the component classifiers first generated but on a subset of them carrying a larger degree of disagreement/diversity. This is why different classifier selection methods [47] have been proposed. GAs have been commonly used for this task as we will show in the following subsection.

2.4. Related work on genetic selection of MCSs

In general, the selection of a subset of classifiers is done using the OCS strategy [13, 30], in which a large set of classifiers is produced and then selected to extract the best performing subset. GAs are a popular technique within this strategy. In the literature, performance, complexity and diversity measures are usually considered as search criteria. Complexity measures are employed to simplify the system, whereas diversity measures are used to avoid overfitting. The reader is referred to [10] for a review on these genetic MCS selection approaches.

Among the different genetic OCS methods, we can remark these most related to the current proposal. On the one hand, we find EMO-based approaches such as that in [48], a hierarchical multiobjective GA algorithm, performing feature selection at the first level and classifier selection at the second level, is presented which outperforms classical methods for two handwritten recognition problems. The multiobjective GA allows both performance and diversity to be considered for MCS selection. Another EMO proposal for classifier selecion is presented in [29]. In that contribution, a comparison of a single-objective GA and the NSGA-II EMO algorithm for 14 different objective functions of the mentioned three families of criteria (12 diversity measures, the training error, and the number of classifiers as a complexity measure). The authors applied their study on only one dataset, a digit handwritten recognition problem with 10 classes and 118,735 instances. They concluded saying that the training error is the best criterion for a single GA and a combination of training error and one diversity measure is the best criterion for an EMO algorithm, which supports the developments in the current contribution (see Sec. 6). On the other hand, in [26] a genetic classifier selection method was considered based on a single performance index, either the diversity, including 16 different measures, or the ensemble error. The best results were obtained with the accuracy measure and a specific kind of diversity measures correlated with the error.

3. Bagging FURIA-based fuzzy MCSs

In this section we will detail how the FURIA fuzzy MCSs are designed [17]. A normalized dataset is split into two parts, a training set and a test set. The training set is submitted to an instance selection and a feature selection procedures in order to provide individual training sets (the so-called *bags*) to train FURIA classifiers. After the training, we get an initial FURIA-based fuzzy MCS, which is validated using the training and the test errors, as well as a measure of complexity based on the total number of component classifiers obtained from FURIA. The whole procedure is graphically presented in Fig. 1. FURIA is reviewed in Sec. 3.1, while the instance selection procedure is described in Sec. 3.2.

3.1. FURIA

Fuzzy Unordered Rules Induction Algorithm (FURIA) [18, 19] is an extension of the state-of-the-art rule learning algorithm called RIPPER [49], having its advantages such like simple and comprehensible

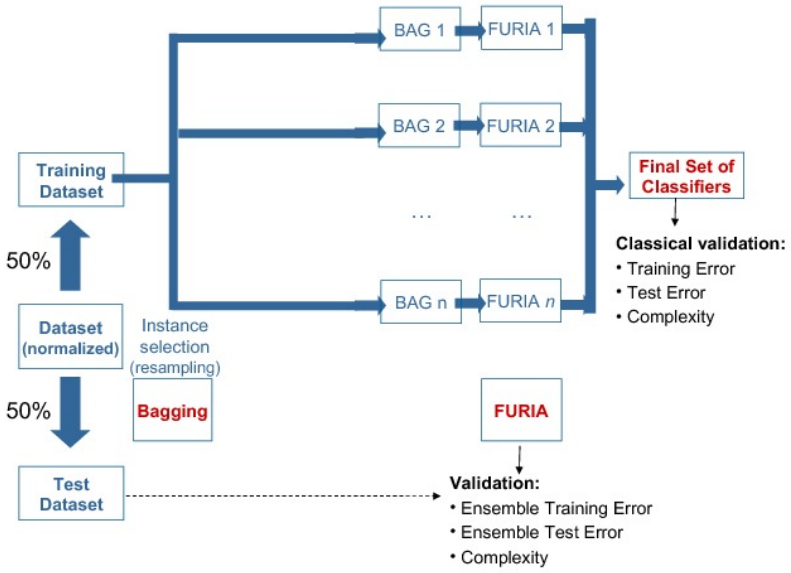

Figure 1: Our framework: after the instance and the feature selection procedures, the component classifiers are derived by the FURIA learning method. Finally, the output is obtained using a voting-based combination method.

fuzzy rule base, and introducing new features. FURIA provides three different extensions of RIPPER:

- It takes an advantage of fuzzy rules instead of crisp ones. Fuzzy rules of FURIA are composed of a class C_j and a certainty degree CD_j in the consequent. The final form of a rule is the following:

  Rule R_j : If x_1 is A_{j1} and ... and x_n is A_{jn} then Class C_j with CD_j; $j = 1,2,...,N$.

  The certainty degree of a given example x is defined as follows:

$$CD_j = \frac{2\frac{D_T^{C_j}}{D_T} + \sum_{x \in D_T^{C_j}} \mu_r^{C_j}(x)}{2 + \sum_{x \in D_T} \mu_r^{C_j}(x)} \qquad (1)$$

  where D_T and $D_T^{C_j}$ stands for the training set and a subset of the training set belonging to the class C_j respectively. In this approach, each fuzzy rule makes a vote for its consequent class. The vote strength of the rule is calculated as the product of the firing degree $\mu_r^{C_j}(x)$ and the certainty degree CD_j. Hence, the fuzzy reasoning method used is the so-called voting-based method [50, 51].
- It uses unordered rule sets instead of rule lists. This change ommits a bias caused by the default class rule, which is applied whenever there is an uncovered example detected.
- It proposes a novel rule stretching method in order to manage uncovered examples. The unordered rule set introduces one crucial drawback, there might appear a case when a given example is not covered. Then, to deal with such situation, one rule is generalized by removing its antecedents. The information measure is proposed to verify which rule to "stretch".

The interested reader is referred to [18] for a full description of FURIA.

3.2. FURIA-based fuzzy MCS design approaches

In our previous contribution [17] we conducted comprehensive experiments considering FURIA-based fuzzy MCSs. Since in [52] it was shown that a combination between bagging and feature selection usually leads to good MCS designs, we decided to follow that idea and we integrated FURIA into a framework of that kind. We aimed to combine the diversity induced by the MCS algorithms and the robustness of the FURIA method in order to desing good performance fuzzy MCS for high dimensional problems. By doing so, we wanted to obtain FURIA-based fuzzy MCSs with a good accuracy-complexity tradeoff. We also tried a combination of FURIA with bagging and feature selection separately.

Three different feature selection methods, random subspace [20], mutual information-based feature selection (MIFS) [21], and the random-greedy feature selection based on MIFS and the GRASP approach [22], were considered. For each feature selection algorithm three different feature subsets of different sizes, which were based on the initial number of features in the classification problem, were tested. Finally, our experiments showed that out of the three following MCS methodologies, that is bagging, feature selection, and bagging with feature selection, the former (see Fig. 1) obtained the best performance when combined with FURIA-based FRBCSs.

Thus, in this contribution we are applying directly a bagging approach in order to generate the initial FURIA-based fuzzy MCSs, which will be later selected by the EMO algorithm. Considering this approach, the bags are generated with the same size as the original training set, as commonly done.

Finally, no weights are considered to combine the outputs of the component classifiers to take the final MCS decision, but a pure voting combination method is applied: the ensemble class prediction will directly be the most voted class in the component classifiers output set.

4. EMO-based MCS selection method

The second stage of our methodology is to consider the OCS strategy. Our aim is to obtain a good accuracy-complexity tradeoff in the FURIA-based fuzzy MCSs when dealing with high dimensional problems. That is, we aim to obtain fuzzy MCS with a low number of base classifiers, which keep a good accuracy. Thus, we have selected the state-of-the-art NSGA-II EMO algorithm in order to generate good quality Pareto set approximations. Five different biobjective fitness functions combining the three existing kinds of optimization criteria (accuracy, complexity, and diversity) are proposed in order to study the best setting. Fig. 2 shows the final structure of the FURIA-based fuzzy MCS design methodology including the OCS stage. The two subsections below presents briefly the algorithm operation mode and its main components.

4.1. Components of NSGA-II

NSGA-II [28] is based on a Pareto dominance depth approach, where the population is divided into several fronts and the depth of each front shows to which front an individual belongs to. A pseudo-dominance rank being assigned to each individual, which is equal to the front number, is a metric used for the selection of an individual.

We have used a standard binary coding in such a way that a binary digit/value/gene is assigned to each classifier. Then, when the variable takes value 1, it means that the current component classifier belongs to the final ensemble, while when the variable is equal to 0, that classifier is discarded. This approach provides a low operation cost, which leads to the high speed of the algorithm.

We have used a generational approach and elitist replacement strategy. The initial population is composed of randomly generated individuals, keeping one of them with the original fuzzy MCS composed of all the existing classifiers. In each generation, to introduce a high amount of diversity, a binary tournament is performed. That means that two individuals are randomly picked from the current population and the best one is selected. The two winners are crossed over to obtain a single offspring that directly substitutes the loser. We have considered the classical two-point crossover and the simple bit-flip mutation. Both operators crossover and mutation are applied with different pre-specified probabilities.

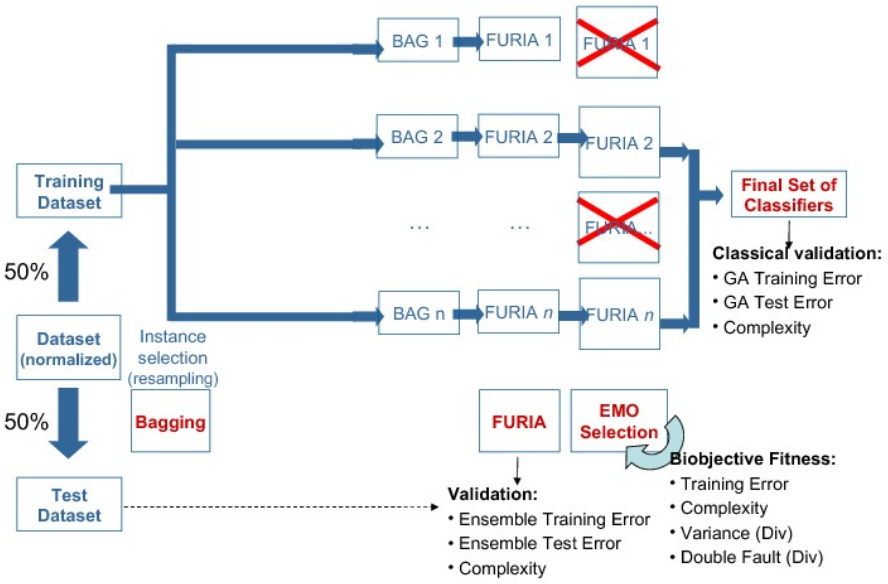

Figure 2: Our framework: after the instance and the feature selection procedures, the component classifiers are derived by the FURIA learning method. Finally, the output is obtained using a voting-based method.

4.2. The four used evaluation criteria for two-objective NSGA-II

In this subsection we describe all the considered optimization criteria. We will utilize measures of three different kinds combined into five different two-objective fitness functions:

- **Accuracy**. We use a standard accuracy measure, the training error (TE). TE is computed as follows. Let $h_1(x),\ldots,h_l(x)$ be the outputs of the component classifiers of the selected ensemble E for an input value $x = (x_1,\ldots,x_n)$. For a given sample $\{(x^k,C^k)\}_{k\in\{1\ldots m\}}$, the TE of that MCS is:

$$TE = \frac{1}{m}\cdot\#\{k \mid C^k \neq \arg\max_{j\in\{1\ldots|E|\}} h_j(x^k)\} \quad (2)$$

  with $|E|$ being the number of classifiers in the selected ensemble.
- **Complexity**. The complexity of the ensemble is directly accounted by the number of classifiers in the ensemble:

$$Complx = |E| \quad (3)$$

- **Diversity**. It seems that obtaining a high diversity between classifiers is the goal to be reached, when aiming to achieve performance improvement of MCSs. In the last few years, a group of researchers devoted their attention to the diversity measures [25, 26, 27]. Two measures can be highlighted from the large amount of proposals in this group: the difficulty (θ) and the double fault (δ):
  1. The difficulty measure θ is computed as follows. Let $X = \{i/|E|\}_{i\in\{0,\ldots,|E|\}}$ and $X_k \in X$ be the proportions of classifiers classifying correctly the instance x^k. Then, θ is:

$$\theta = Var(\{X_1,\ldots,X_k,\ldots,X_m\}) \quad (4)$$

  2. The pairwise measure δ for two classifiers h_i and h_j shows the following expression:

$$\delta_{i,j} = \frac{N_{ij}^{00}}{m} \quad (5)$$

with N_{ij}^{00} being the number of examples missclassified by both h_i and h_j. The global value of the measure for the whole selected ensemble is computed as:

$$\delta_{avg} = \frac{2}{L(L-1)} \sum_{i=1}^{L-1} \sum_{j=i+1}^{L} \delta_{i,j} \qquad (6)$$

with $L = |E|$ being the number of component classifiers in the ensemble.

From these four criteria we have formed five different two-objective fitness functions. A first option is a standard combination of TE with Complx (from now on called fitness function 2a).

In addition, the use of biobjective functions based on diversity measures is justified by previous findings in the specialized literature. Diversity measures have been deeply studied in [25, 26, 27, 29, 30]. The relationship between diversity measures and accuracy is not clear. In [25], it was shown how the ensemble accuracy and diversity is not as strongly correlated as it could be expected. The authors concluded that accuracy estimation can not be substituted by diversity during the MCS design process. These results were confirmed in [26] in our same framework, classifier selection. In the experimental study developed, the authors drew the conclusion that the use of a single-objective function based on a diversity measure does not outperform the direct use of an error rate. Hence, the combined action of both kinds of measures can lead to a better fuzzy MCS performance in our case. In particular, we combined accuracy measures with the said two diversity measures in [12], obtaining promising results. Hence, in this contribution we will use the combination of TE with θ and δ (fitness functions 2b and 2c respectively) in the current contribution, as in our opinion this may lead to an accuracy improvement, when keeping a low number of classifiers.

Finally, we would like to put more stress on the complexity aspect as proposed in [29, 30], so we join diversity measures with complexity into the two remaining two-objective fitness functions (2d and 2e). By doing so we would like to obtain simple ensembles still having high quality in terms of performance.

Table 1 summarizes the composition of the five biobjective fitness functions proposed.

Table 1. The five fitness function proposed.

abbreviation	1st obj.	2nd obj.
2a	TE	Complx
2b	TE	θ
2c	TE	δ
2d	θ	Complx
2e	δ	Complx

5. Experiments and analysis of results

This section reports all the experiments performed. Firstly, we introduce the experimental setup (Sec. 5.1). Then, in Sec. 5.2 the performance of NSGA-II with the considered five two-objective fitness functions when tackling the classifier selection tasks for FURIA-based fuzzy MCSs is analyzed. Two multiobjective metrics, one unary and one binary [24, 53], are considered to do so. We also show graphs of the obtained Pareto front approximations. Furthermore, we study some representative individual solutions selected from the obtained Pareto sets in Sec. 5.3. Finally, we compare the best single values obtained against the result from the previous stage, that is FURIA-based fuzzy MCSs with bagging not considering classifier selection, as well as against two state-of-the-art algorithms, random forests [38] and bagging C4.5 MCSs [54], in Sec. 5.4.

5.1. Experimental setup

To evaluate the performance of the generated FURIA-based fuzzy MCSs, we have selected 20 datasets with different characteristics concerning the number of examples, features, and classes from the UCI machine learning repository (see Table 2). In order to compare the accuracy of the considered classifiers, we used Dietterichs 5×2-fold cross-validation (5×2-cv), which is considered to be superior to paired *k*-fold cross validation in classification problems [55].

Table 2. Data sets considered

Data set	#examples	#attr.	#classes
abalone	4178	7	28
breast	700	9	2
glass	214	9	7
heart	270	13	2
ionosphere	352	34	2
magic	19020	10	2
optdigits	5620	64	10
pblocks	5474	10	5
pendigits	10992	16	10
phoneme	5404	5	2
pima	768	8	2
sat	6436	36	6
segment	2310	19	7
sonar	208	60	2
spambase	4602	57	2
texture	5500	40	11
waveform	5000	40	3
wine	178	13	3
vehicle	846	18	4
yeast	1484	8	10

The FURIA-based fuzzy MCSs generated are initially comprised by 50 classifiers. NSGA-II for the component classifier selection works with a population of 50 individuals and runs during 1000 generations. The crossover probability considered is 0.6 and the mutation probability is 0.1. A different run is developed with each of the five fitness function variants for each initial fuzzy MCS, thus resulting in 10 different runs per dataset as a consequence of the 5×2-cv cross validation procedure. All the experiments have been run in a cluster at the University of Granada, Spain, on Intel quadri-core Pentium 2.4 GHz nodes with 2 GBytes of memory, under the Linux operating system.

To compare the Pareto front approximations of the global learning objectives (i.e. MCS test accuracy and complexity) we consider the two of usual kinds of multiobjective metrics [24, 53]. The first group measures the quality of a single nondominated solution set returned by a multiobjective algorithm, while the second group compares the performance of two different multiobjective algorithms. We have selected one metric for each group, hypervolume ratio (HVR) [24] and C-measure [53], respectively.

An useful unary metric to compare Pareto sets is the S metric, proposed by Zitzler [53], and called hypervolume. It measures the volume enclosed by the Pareto front Y'. When there are only two objectives, $S(Y')$ measures the area covered by the Pareto front by adding the areas covered by each individual nondominated point. In the case of a minimization problem, as ours, there is a need to define a reference point. Nevertheless, the relative value of the S metric usually depends upon an arbitrary choice of this point, getting unexpected metric values if it is not correctly fixed [56]. Besides, when the dimension of the objectives is large, it is interesting to normalise them. Alternatively, the hypervolume ratio (HVR) [24] can be considered to avoid these drawbacks. HVR is a very powerful metric, as it both measures diversity and closeness. The HVR can be simply calculated as follows:

$$HVR = \frac{H_1}{H_2} \tag{7}$$

where H_1 and H_2 are the volume (S metric value) of the Pareto front approximation and the true Pareto front, respectively. When HVR equals 1, then the Pareto front approximation and the true one are equal. Thus, HVR values lower than 1 indicate a generated Pareto front that is not as good as the true Pareto front. In our case, as the truen Pareto fronts are not known, we will consider an approximation (a pseudo-optimal Pareto front) obtained by fusing all the (approximate) Pareto fronts generated for each problem instance by any algorithm variant in any run.

As binary metric we will use the coverage, proposed by Zitzler et al. in [53], which compares a pair of non-dominated sets by computing the fraction of each set that is covered by the other:

$$C(X',X'') = \frac{|\{\forall a'' \in X''; \exists a' \in X' : a' \prec a''\}|}{|X''|} \tag{8}$$

where $a' \prec a''$ indicates that the solution a' dominates the solution a'' in a minimization problem and $Y', \overline{Y} \subseteq Y$ are the sets of objective vectors that correspond to X' and $\overline{X}$ (non-dominated decision vectors), respectively.

Hence, the value $C(X',X'') = 1$ means that all the solutions in X'' are dominated by or equal to solutions in X'. The opposite, $C(X',X'') = 0$, represents the situation where none of the solutions in X'' are covered by the set X'. Note that both $C(X',X'')$ and

$C(X'',X')$ have to be considered, since $C(X',X'')$ is not necessarily equal to $1-C(X'',X')$.

Let us call $\overline{P_i^j}$ the non-dominated solution set returned by NSGA-II usign the variant of fitness function i in the j-th run for a specific problem instance; $P_i = \overline{P_i^1} \bigcup \overline{P_i^2} \bigcup \ldots \bigcup \overline{P_i^{10}}$, the union of the solution sets returned by the ten runs obtained from 5x2cv of algorithm i, and finally $\overline{P_i}$ the set of all non-dominated solutions in the P_i set*(aggregated Pareto fronts). As a complement to the analysis of the results obtained in the two different multiobjective metrics, we will provide graphical representations of some of those aggregated Pareto fronts. When graphically represented, these graphics offers a visual information, not measurable, but sometimes more useful than numeric values.

5.2. *Analysis and comparison of the obtained Pareto front approximations*

This section is devoted to analyze the performance of the proposed NSGA-II-based approach to classifier selection in FURIA-based fuzzy MCS. First, the quality of the obtained Pareto front approximations considering the five fitness functions defined in Sec. 4.2 will be studied in Sec. 5.2.1. Then, a global comparison among the five methods is made by analyzing their performance on the satisfaction of the two final learning problem goals, the test accuracy and the complexity of the obtained fuzzy MCSs.

On the one hand, regarding the former analysis we draw a table with the minimum, maximum, mean, and standard deviation values obtained for each objective in the Pareto set derived for each of the five fitness functions considered. On the other hand, to compare the Pareto front approximations of the global objectives we consider a table with the same structure as above as well as statistics related to the selected multiobjective metrics, HVR and C. We also plot some of the aggregated Pareto front approximations in order to have a taste of their trends, as drawing all of them would not be feasible.

5.2.1. *Analysis of the original Pareto front approximations*

As the five two-objective fitness functions considered handle three different types of measures: accuracy, complexity, and diversity, its direct comparison is practically impossible. Instead, in order to give a flavor of the Pareto fronts obtained, we present their characteristic values. We first gather them for all of the folds out of 5x2cv and average them. We show the statistics for each dataset in a different row in Tables 3 to 5. For each fitness function we show the cardinality of the Pareto set (called *Car.*) and for each objective we present the minimum (called *Min.*), maximum (called *Max.*), mean (called *Avg.*), and standard deviation (called *Dev.*) of the averaged 5x2cv values. Let us recall the objectives of the all fitness functions. Function 2a is composed of training error and complexity, while 2b and 2c combine training error and the diversity measures, variance and double fault, respectively. Finally, 2d and 2e assemble complexity with variance and double fault, respectively. Furthermore, we show a visual representation of the aggregated Pareto front approximations for one selected dataset. Figures 3 to 5 represent a visualization of the fronts obtained for the abalone dataset by the five fitness functions.

A first very important conclusion is that, while the first three fitness function variants, 2a, 2b, and 2c, work properly as they allow the multiobjective genetic classifier selection method to derive a significant number of solutions in the Pareto set approximations (cardinal), the other two, 2d and 2e, show a deceptive behavior. On the one hand, function 2d provided a single solution in 10 out of 20 cases and 1.1 solutions in another 5 out of 20 cases. On the other hand, although function 2e allows us to obtain many different solutions in the solution space (Pareto set), all of them correspond to exactly the same objective values (Pareto front). Thus, the latter two fitness functions are not adequate for generating diverse Pareto front approximations.

*Notice that the pseudo-optimal Pareto front is the fusion of the $\overline{P_i}$ sets generated by every variant of the fitness function in all runs developed.

Table 3: Statistics of the Pareto front approximations with the original objectives.

	2a									2b								
		Obj. 1 - TE				Obj. 2 - Complx					Obj. 1 - TE				Obj. 2 - θ			
	Car.	Min.	Max.	Avg.	Dev.	Min.	Max.	Avg.	Dev.	Car.	Min.	Max.	Avg.	Dev.	Min.	Max.	Avg.	Dev.
abalone	14.5	0.505	0.605	0.532	0.029	2.000	17.700	8.821	4.757	58.6	0.509	0.599	0.546	0.025	0.113	0.128	0.119	0.004
breast	7.0	0.000	0.007	0.002	0.003	2.000	3.000	2.701	0.458	6.3	0.000	0.009	0.005	0.004	0.009	0.038	0.025	0.011
glass	8.9	0.004	0.113	0.034	0.036	2.000	6.800	4.395	1.631	11.1	0.007	0.113	0.060	0.036	0.100	0.410	0.258	0.087
heart	8.8	0.001	0.050	0.018	0.020	2.000	4.500	3.372	0.972	9.5	0.001	0.057	0.029	0.019	0.053	0.230	0.143	0.055
ionosphere	6.8	0.003	0.008	0.005	0.002	2.000	2.800	2.504	0.374	5.2	0.003	0.027	0.015	0.010	0.026	0.083	0.054	0.024
magic	4.3	0.097	0.113	0.102	0.008	2.000	6.600	4.250	2.024	11.1	0.098	0.107	0.102	0.003	0.096	0.189	0.152	0.026
optdigits	142.0	0.000	0.000	0.000	0.000	2.000	2.000	2.000	0.000	196.7	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
pblocks	8.2	0.006	0.016	0.009	0.003	2.000	11.200	6.018	3.295	15.8	0.007	0.015	0.010	0.003	0.015	0.055	0.035	0.011
pendigits	17.2	0.000	0.010	0.002	0.002	2.000	9.300	5.839	2.169	22.7	0.000	0.011	0.004	0.003	0.011	0.082	0.040	0.017
phoneme	7.8	0.058	0.086	0.065	0.010	2.000	9.800	5.990	2.864	14.9	0.059	0.083	0.068	0.008	0.076	0.226	0.163	0.039
pima	10.6	0.022	0.084	0.044	0.021	2.000	10.200	5.568	2.875	19.3	0.024	0.100	0.057	0.023	0.090	0.382	0.239	0.074
sat	13.2	0.010	0.049	0.019	0.011	2.000	16.600	8.013	4.646	38.0	0.011	0.049	0.023	0.010	0.047	0.235	0.148	0.044
segment	14.2	0.000	0.012	0.003	0.004	2.000	6.600	4.782	1.534	12.8	0.000	0.015	0.007	0.005	0.014	0.070	0.041	0.017
sonar	7.9	0.000	0.037	0.011	0.016	2.000	3.300	2.853	0.565	7.5	0.000	0.062	0.029	0.023	0.058	0.220	0.148	0.058
spambase	7.8	0.015	0.033	0.020	0.007	2.000	9.400	5.141	2.703	17.3	0.015	0.029	0.020	0.004	0.029	0.122	0.078	0.024
texture	18.3	0.000	0.020	0.003	0.005	2.000	8.000	5.641	1.867	22.3	0.000	0.021	0.006	0.006	0.020	0.127	0.071	0.025
vehicle	17.0	0.002	0.099	0.023	0.026	2.000	13.400	7.669	3.536	24.9	0.003	0.104	0.043	0.029	0.093	0.421	0.277	0.083
waveform	22.8	0.001	0.059	0.011	0.014	2.000	21.000	10.311	5.517	53.3	0.002	0.067	0.021	0.016	0.062	0.411	0.232	0.081
wine	5.3	0.000	0.001	0.000	0.001	2.000	2.100	2.069	0.048	4.6	0.000	0.006	0.004	0.003	0.006	0.020	0.011	0.008
yeast	9.9	0.156	0.250	0.182	0.029	2.000	11.100	5.950	2.956	28.1	0.158	0.254	0.199	0.029	0.189	0.402	0.324	0.051

Table 4: Statistics of the Pareto front approximations with the original objectives.

	2c									2d								
		Obj. 1 - TE				Obj. 2 - δ					Obj. 1 - θ				Obj. 2 - Complx			
	Car.	Min.	Max.	Avg.	Dev.	Min.	Max.	Avg.	Dev.	Car.	Min.	Max.	Avg.	Dev.	Min.	Max.	Avg.	Dev.
abalone	29.3	0.506	0.622	0.538	0.030	0.428	0.460	0.448	0.008	8.3	0.113	0.151	0.122	0.013	2.000	9.400	5.664	2.561
breast	6.0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.3	0.018	0.018	0.018	0.000	2.000	2.000	2.000	0.000
glass	8.3	0.005	0.162	0.054	0.058	0.009	0.038	0.025	0.010	1.1	0.158	0.158	0.158	0.000	2.000	2.000	2.000	0.000
heart	6.8	0.000	0.064	0.019	0.024	0.000	0.009	0.005	0.003	1.1	0.093	0.093	0.093	0.000	2.000	2.000	2.000	0.000
ionosphere	57.9	0.004	0.004	0.004	0.000	0.000	0.000	0.000	0.000	1.2	0.045	0.045	0.045	0.000	2.000	2.000	2.000	0.000
magic	8.1	0.097	0.115	0.102	0.007	0.067	0.072	0.070	0.002	1.0	0.124	0.124	0.124	0.000	2.000	2.000	2.000	0.000
optdigits	196.7	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	145.7	0.000	0.000	0.000	0.000	2.000	2.000	2.000	0.000
pblocks	10.8	0.006	0.020	0.010	0.005	0.004	0.007	0.006	0.001	1.0	0.022	0.022	0.022	0.000	2.000	2.000	2.000	0.000
pendigits	83.0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.0	0.020	0.020	0.020	0.000	2.000	2.000	2.000	0.000
phoneme	10.0	0.058	0.097	0.068	0.013	0.035	0.044	0.040	0.003	1.0	0.114	0.114	0.114	0.000	2.000	2.000	2.000	0.000
pima	11.7	0.021	0.103	0.046	0.026	0.018	0.037	0.029	0.006	1.0	0.145	0.145	0.145	0.000	2.000	2.000	2.000	0.000
sat	2.8	0.010	0.010	0.010	0.000	0.000	1.031	0.323	0.530	1.0	0.079	0.079	0.079	0.000	2.000	2.000	2.000	0.000
segment	69.9	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.1	0.024	0.024	0.024	0.000	2.000	2.000	2.000	0.000
sonar	4.5	0.000	0.011	0.003	0.003	0.000	0.001	0.001	0.000	1.1	0.101	0.101	0.101	0.000	2.000	2.000	2.000	0.000
spambase	2.4	0.015	0.015	0.015	0.000	0.000	0.550	0.110	0.246	1.1	0.046	0.046	0.046	0.000	2.000	2.000	2.000	0.000
texture	99.8	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.0	0.037	0.037	0.037	0.000	2.000	2.000	2.000	0.000
vehicle	15.2	0.002	0.118	0.030	0.031	0.015	0.034	0.027	0.005	1.0	0.155	0.155	0.155	0.000	2.000	2.000	2.000	0.000
waveform	19.3	0.001	0.001	0.001	0.000	0.000	0.643	0.214	0.371	1.0	0.108	0.108	0.108	0.000	2.000	2.000	2.000	0.000
wine	54.5	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.8	0.010	0.010	0.010	0.000	2.000	2.000	2.000	0.000
yeast	15.5	0.156	0.282	0.188	0.034	0.109	0.134	0.123	0.007	1.0	0.246	0.246	0.246	0.000	2.000	2.000	2.000	0.000

Table 5: Statistics of the Pareto front approximations with the original objectives.

	2e								
		Obj. 1 - δ				Obj. 2 - Complx			
	Car.	Min.	Max.	Avg.	Dev.	Min.	Max.	Avg.	Dev.
abalone	120.5	0.000	0.000	0.000	0.000	2.000	2.000	2.000	0.000
breast	86.1	0.000	0.000	0.000	0.000	2.000	2.000	2.000	0.000
glass	102.3	0.000	0.000	0.000	0.000	2.000	2.000	2.000	0.000
heart	94.5	0.000	0.000	0.000	0.000	2.000	2.000	2.000	0.000
ionosphere	89.0	0.000	0.000	0.000	0.000	2.000	2.000	2.000	0.000
magic	136.9	0.000	0.000	0.000	0.000	2.000	2.000	2.000	0.000
optdigits	145.7	0.000	0.000	0.000	0.000	2.000	2.000	2.000	0.000
pblocks	132.5	0.000	0.000	0.000	0.000	2.000	2.000	2.000	0.000
pendigits	136.7	0.000	0.000	0.000	0.000	2.000	2.000	2.000	0.000
phoneme	137.1	0.000	0.000	0.000	0.000	2.000	2.000	2.000	0.000
pima	124.6	0.000	0.000	0.000	0.000	2.000	2.000	2.000	0.000
sat	133.8	0.000	0.000	0.000	0.000	2.000	2.000	2.000	0.000
segment	124.8	0.000	0.000	0.000	0.000	2.000	2.000	2.000	0.000
sonar	99.0	0.000	0.000	0.000	0.000	2.000	2.000	2.000	0.000
spambase	132.1	0.000	0.000	0.000	0.000	2.000	2.000	2.000	0.000
texture	142.2	0.000	0.000	0.000	0.000	2.000	2.000	2.000	0.000
vehicle	126.9	0.000	0.000	0.000	0.000	2.000	2.000	2.000	0.000
waveform	135.9	0.000	0.000	0.000	0.000	2.000	2.000	2.000	0.000
wine	73.3	0.000	0.000	0.000	0.000	2.000	2.000	2.000	0.000
yeast	131.4	0.000	0.000	0.000	0.000	2.000	2.000	2.000	0.000

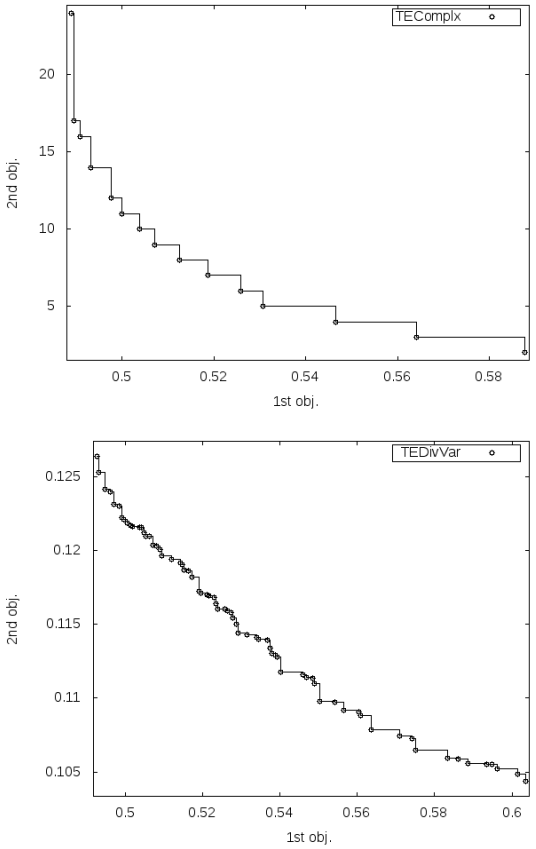

Fig. 3. The Pareto front approximation obtained for *abalone* using fitness function 2a (objective 1 stands for training error and objective 2 for complexity) on the top, and fitness function 2b (Objective 1 stands for training error and objective 2 for variance) on the bottom.

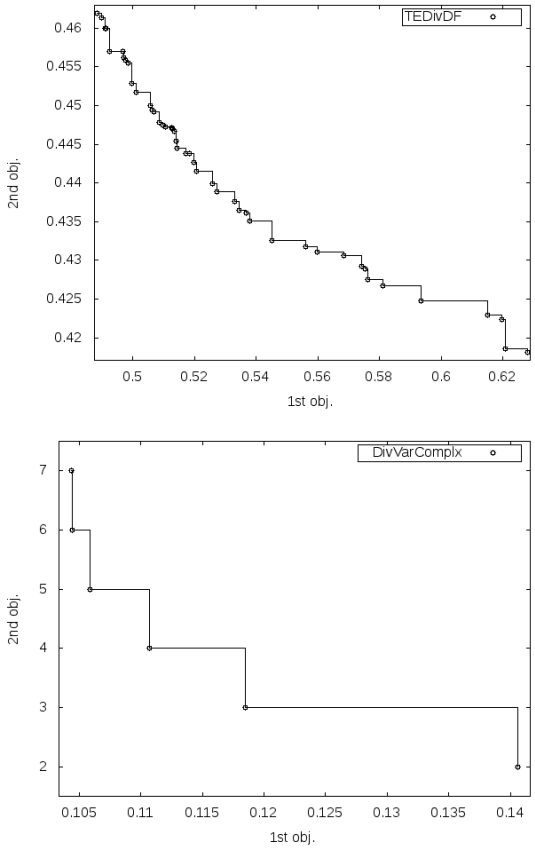

Fig. 4. The Pareto front approximation obtained for *abalone* using fitness function 2c (objective 1 stands for training error and objective 2 for double fault) on the top, and fitness function 2d (Objective 1 stands for variance and objective 2 for complexity) on the bottom.

It can be noticed that the ranges of the objectives vary depending on the dataset given. The training error, which is the first objective of 2a, 2b, and 2c, converges to 0 for several datasets, while growing up to 0.605 overall. The complexity, which is the second objective of 2a, 2d, and 2e lays in the range between 2 and 21, thus showing significant reduction obtained by the multiobjective genetic component classifier selection method (recall that the original number of classifiers is 50). Besides, the variance, being the second objective of 2b and the first of 2d, obtains a minimum value equal to 0 and grows up to 1.202.

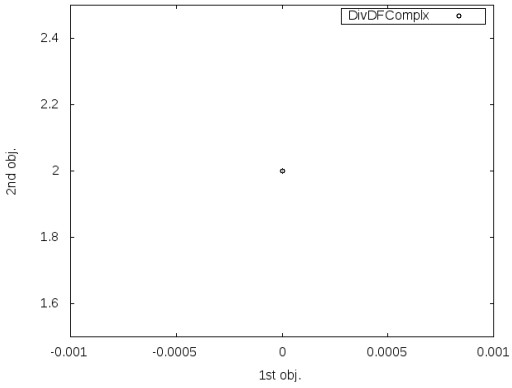

Fig. 5. The Pareto front approximation obtained for *abalone* using fitness function 2e (objective 1 stands for double fault and objective 2 for complexity).

The double fault, which is the second objective of 2c and the first of 2e, lays in the range between 0 and 4.722. In addition, the standard deviation values of the training error vary for each fitness function, 2a, 2b, and 2c variants, depending on the dataset and it is hard to point the one being the most stable. On the other hand, considering the standard deviation of complexity, the lowest values are obtained by both 2d and 2e variants, thus showing the already mentioned deceptive behavior. The same results were obtained for diversity measures.

The combination of double fault and complexity seems to be a special case to be considered. It is a deceptive measure because either the two objectives are not conflicting or there is a specific limit condition where an optimal tradeoff is obtained. For all the datasets, a single optimum of the biobjective function was generated where complexity obtained value 2 and double fault took value 0. Such pair of values could be reasonable, since double fault is a pair-wise metric and in general tends to small ensemble sizes, the same as complexity. Nevertheless, it clearly shows this biobjective fitness function definition is not appropriate for the considered learning task.

A similar case is the 2d variant combining variance with complexity, as for almost all datasets, 19 ouf of 20, it obtained complexity equal to 2. Besides, apart from two datasets, abalone and optdigits, it obtained cardinality equal or close to 1, as already mentioned.

5.2.2. Perfomance analysis of the five variants in the two global learning objectives

Since the individual objectives of the Pareto front approximations presented in the previous subsection are not comparable, we have considered two common objectives, namely test error and complexity, in order to compare the results obtained by the five fitness functions proposed. Notice that, these are the actual learning goals that will be considered by the designer in order to choose the final fuzzy MCS structure.

The characteristic values of the Pareto front approximations of the two global learning goals are presented in Tables 6 and 7. The structure of these tables is similar to the ones in the previous subsection. The results are gathered for all of the folds out of 5x2cv and averaged. We show the statistics for each dataset and each fitness function in a different row. For each fitness function we show cardinality (called *Car.*) and for each objective we present minimum (called *Min.*), maximum (called *Max.*), mean (called *Avg.*), and standard deviation (called *Dev.*) of the averaged 5x2cv values.

Table 6. Statistics of the Pareto front approximations with the global learning objectives.

			Obj. 1 - test error				Obj. 2 - Complx			
		Car.	Min.	Max.	Avg.	Dev.	Min.	Max.	Avg.	Dev.
aba	2a	14.5	0.746	0.777	0.758	0.009	2.0	17.7	8.8	4.8
	2b	58.6	0.741	0.763	0.752	0.005	9.3	25.0	17.2	4.2
	2c	29.3	0.745	0.783	0.758	0.009	2.0	16.9	8.5	4.4
	2d	8.3	0.752	0.791	0.767	0.013	2.0	9.4	5.7	2.6
	2e	120.5	0.759	0.803	0.781	0.009	2.0	2.0	2.0	0.0
bre	2a	7.0	0.038	0.060	0.048	0.008	2.0	3.0	2.7	0.5
	2b	6.3	0.037	0.057	0.046	0.007	1.0	3.7	2.5	1.1
	2c	6.0	0.037	0.051	0.045	0.005	3.0	3.4	3.1	0.2
	2d	1.3	0.050	0.051	0.051	0.001	2.0	2.0	2.0	0.0
	2e	86.1	0.037	0.091	0.061	0.012	2.0	2.0	2.0	0.0
gla	2a	8.9	0.286	0.390	0.330	0.033	2.0	6.8	4.4	1.6
	2b	11.1	0.288	0.396	0.331	0.033	1.0	8.9	4.0	2.3
	2c	8.3	0.283	0.403	0.333	0.041	2.1	7.2	4.3	1.9
	2d	1.1	0.360	0.363	0.361	0.002	2.0	2.0	2.0	0.0
	2e	102.3	0.305	0.517	0.400	0.046	2.0	2.0	2.0	0.0
hea	2a	8.8	0.172	0.233	0.202	0.021	2.0	4.5	3.4	1.0
	2b	9.5	0.170	0.231	0.200	0.020	1.0	5.6	3.0	1.4
	2c	6.8	0.178	0.235	0.203	0.021	2.2	4.8	3.3	1.0
	2d	1.1	0.201	0.203	0.202	0.001	2.0	2.0	2.0	0.0
	2e	94.5	0.155	0.299	0.224	0.031	2.0	2.0	2.0	0.0
ion	2a	6.8	0.144	0.187	0.164	0.015	2.0	2.8	2.5	0.4
	2b	5.2	0.145	0.191	0.166	0.020	1.0	3.0	2.0	0.9
	2c	57.9	0.126	0.170	0.148	0.010	13.4	28.1	20.9	3.6
	2d	1.2	0.156	0.160	0.158	0.003	2.0	2.0	2.0	0.0
	2e	89.0	0.129	0.248	0.181	0.026	2.0	2.0	2.0	0.0
mag	2a	4.3	0.132	0.144	0.136	0.005	2.0	6.6	4.2	2.0
	2b	11.1	0.132	0.143	0.136	0.003	1.0	8.2	4.1	2.1
	2c	8.1	0.132	0.145	0.136	0.005	2.0	6.8	4.0	1.8
	2d	1.0	0.146	0.146	0.146	0.000	2.0	2.0	2.0	0.0
	2e	136.9	0.142	0.159	0.150	0.003	2.0	2.0	2.0	0.0
opt	2a	142.0	0.655	0.703	0.678	0.009	2.0	2.0	2.0	0.0
	2b	196.7	0.625	0.641	0.633	0.003	18.0	37.0	25.9	3.3
	2c	196.7	0.625	0.641	0.633	0.003	18.0	37.0	25.9	3.3
	2d	145.7	0.654	0.704	0.678	0.009	2.0	2.0	2.0	0.0
	2e	145.7	0.654	0.704	0.678	0.009	2.0	2.0	2.0	0.0
pbl	2a	8.2	0.028	0.035	0.031	0.003	2.0	11.2	6.0	3.3
	2b	15.8	0.027	0.034	0.030	0.002	1.0	8.8	4.1	2.1
	2c	10.8	0.027	0.038	0.031	0.003	2.0	10.9	5.6	3.0
	2d	1.0	0.034	0.034	0.034	0.000	2.0	2.0	2.0	0.0
	2e	132.5	0.031	0.047	0.038	0.003	2.0	2.0	2.0	0.0
pen	2a	17.2	0.016	0.032	0.020	0.004	2.0	9.3	5.8	2.2
	2b	22.7	0.016	0.034	0.022	0.005	1.0	11.3	4.4	2.5
	2c	83.0	0.014	0.018	0.016	0.001	15.8	26.7	21.5	2.5
	2d	1.0	0.032	0.032	0.032	0.000	2.0	2.0	2.0	0.0
	2e	136.7	0.029	0.042	0.035	0.003	2.0	2.0	2.0	0.0
pho	2a	7.8	0.125	0.152	0.133	0.009	2.0	9.8	6.0	2.9
	2b	14.9	0.127	0.151	0.135	0.007	1.0	8.8	4.3	2.1
	2c	10.0	0.125	0.160	0.136	0.011	2.0	10.0	5.2	2.7
	2d	1.0	0.153	0.153	0.153	0.000	2.0	2.0	2.0	0.0
	2e	137.1	0.144	0.183	0.162	0.008	2.0	2.0	2.0	0.0

In general, the highest cardinality is obtained by the 2e variant, which combines double fault and

complexity objectives. However, we have considered it as a deceptive fitness function in the previous subsection. On the other hand, the 2d variant, which is composed of variance and complexity objectives, almost always provides cardinality equal to 1 and was also categorized as deceptive. Thus, both variants are tricky and do not constitute good approach to provide high quality Pareto front approximations.

Table 7. Statistics of the Pareto front approximations with the global learning objectives.(cont.)

			Obj. 1 - test error				Obj. 2 - Complx			
		Car.	Min.	Max.	Avg.	Dev.	Min.	Max.	Avg.	Dev.
pim	2a	10.6	0.235	0.291	0.256	0.018	2.0	10.2	5.6	2.9
	2b	19.3	0.233	0.296	0.261	0.016	1.0	11.6	4.4	2.7
	2c	11.7	0.236	0.290	0.257	0.016	2.0	11.0	5.8	3.0
	2d	1.0	0.277	0.277	0.277	0.000	2.0	2.0	2.0	0.0
	2e	124.6	0.231	0.318	0.274	0.018	2.0	2.0	2.0	0.0
sat	2a	13.2	0.102	0.130	0.110	0.008	2.0	16.6	8.0	4.6
	2b	38.0	0.101	0.132	0.111	0.007	1.0	17.2	6.3	3.8
	2c	2.8	0.102	0.104	0.103	0.001	20.2	22.7	21.3	1.4
	2d	1.0	0.129	0.129	0.129	0.000	2.0	2.0	2.0	0.0
	2e	133.8	0.119	0.140	0.129	0.004	2.0	2.0	2.0	0.0
seg	2a	14.2	0.029	0.048	0.036	0.006	2.0	6.6	4.8	1.5
	2b	12.8	0.029	0.051	0.038	0.006	1.0	6.8	3.4	1.6
	2c	69.9	0.027	0.037	0.032	0.002	14.4	26.1	19.8	2.9
	2d	1.1	0.047	0.047	0.047	0.000	2.0	2.0	2.0	0.0
	2e	124.8	0.037	0.070	0.052	0.006	2.0	2.0	2.0	0.0
son	2a	7.9	0.203	0.284	0.245	0.031	2.0	3.3	2.9	0.6
	2b	7.5	0.217	0.289	0.252	0.026	1.0	3.7	2.5	1.0
	2c	4.5	0.213	0.271	0.242	0.022	2.8	3.3	3.0	0.1
	2d	1.1	0.269	0.274	0.272	0.003	2.0	2.0	2.0	0.0
	2e	99.0	0.188	0.406	0.292	0.047	2.0	2.0	2.0	0.0
spa	2a	7.8	0.057	0.072	0.061	0.005	2.0	9.4	5.1	2.7
	2b	17.3	0.056	0.070	0.061	0.004	1.0	9.8	4.7	2.4
	2c	2.4	0.056	0.058	0.057	0.001	11.9	13.8	12.8	1.1
	2d	1.1	0.072	0.072	0.072	0.000	2.0	2.0	2.0	0.0
	2e	132.1	0.065	0.090	0.077	0.005	2.0	2.0	2.0	0.0
tex	2a	18.3	0.032	0.065	0.040	0.009	2.0	8.0	5.6	1.9
	2b	22.3	0.033	0.067	0.043	0.009	1.0	9.2	4.2	1.9
	2c	99.8	0.028	0.035	0.032	0.001	17.2	30.4	23.8	3.0
	2d	1.0	0.067	0.067	0.067	0.000	2.0	2.0	2.0	0.0
	2e	142.2	0.058	0.087	0.071	0.005	2.0	2.0	2.0	0.0
veh	2a	17.0	0.257	0.302	0.275	0.012	2.0	13.4	7.7	3.5
	2b	24.9	0.255	0.316	0.282	0.015	1.0	13.4	5.4	3.2
	2c	15.2	0.260	0.308	0.278	0.014	2.0	14.1	7.0	3.6
	2d	1.0	0.307	0.307	0.307	0.000	2.0	2.0	2.0	0.0
	2e	126.9	0.270	0.350	0.310	0.017	2.0	2.0	2.0	0.0
wav	2a	22.8	0.148	0.195	0.158	0.011	2.0	21.0	10.3	5.5
	2b	53.3	0.146	0.199	0.163	0.012	1.0	23.1	7.1	4.8
	2c	19.3	0.146	0.152	0.149	0.002	24.0	29.3	26.5	1.8
	2d	1.0	0.197	0.197	0.197	0.000	2.0	2.0	2.0	0.0
	2e	135.9	0.181	0.213	0.196	0.006	2.0	2.0	2.0	0.0
win	2a	5.3	0.051	0.100	0.076	0.019	2.0	2.1	2.1	0.0
	2b	4.6	0.054	0.091	0.074	0.014	1.0	2.3	1.6	0.7
	2c	54.5	0.018	0.063	0.038	0.013	15.2	33.8	24.4	4.6
	2d	1.8	0.065	0.076	0.071	0.006	2.0	2.0	2.0	0.0
	2e	73.3	0.037	0.231	0.125	0.046	2.0	2.0	2.0	0.0
yea	2a	9.9	0.404	0.453	0.423	0.015	2.0	11.1	5.9	3.0
	2b	28.1	0.396	0.467	0.424	0.016	1.0	10.9	4.8	2.6
	2c	15.5	0.405	0.472	0.426	0.019	2.0	12.5	6.3	3.0
	2d	1.0	0.445	0.445	0.445	0.000	2.0	2.0	2.0	0.0
	2e	131.4	0.421	0.495	0.458	0.014	2.0	2.0	2.0	0.0

The other variants, 2a, 2b, and 2c, combining training error with complexity, variance, and double fault, respectively, seem to be performing well.

Having an insight to the results, we may emphasize the 2c variant, as the one having the lowest *Avg.* test error value for 10 out of 20 cases (+2 ties). Notice that the 2a variant is just slightly worse than the other two approaches. Considering the complexity, we may see that the 2e variant obtained the lowest value in all of the cases, while the 2d variant was not much worse with 19 out of 20 best values (19 ties with the 2e variant). The 2a and 2b variants obtained only one best result (both ties with the 2d and 2e variants), placing the 2c variant in the last position with no best value obtained.

Table 8. Comparison of Pareto fronts using HVR measure.

	2a	2b	2c	2d	2e
aba	**0.9973**	0.5126	**0.9973**	0.9961	0.9962
bre	0.6632	**0.9955**	0.3321	0.6627	0.6644
gla	0.8455	**0.9867**	0.8314	0.8376	0.8469
hea	0.6582	**0.9858**	0.5915	0.6564	0.6625
ion	0.9437	**0.9796**	0.5294	0.9416	0.9464
mag	0.9323	**0.9988**	0.9324	0.9300	0.9307
opt	**0.9952**	0.3335	0.3335	**0.9952**	**0.9952**
pbl	0.8555	**0.9983**	0.8555	0.8547	0.8553
pen	0.9609	**0.9992**	0.4307	0.9580	0.9587
pho	0.9267	**0.9978**	0.9266	0.9224	0.9241
pim	0.8700	**0.9944**	0.8700	0.8650	0.8730
sat	0.9554	**0.9988**	0.1738	0.9510	0.9528
seg	0.9483	**0.9982**	0.3295	0.9452	0.9472
son	0.6544	**0.9797**	0.3927	0.6492	0.6597
spa	0.9071	**0.9978**	0.1542	0.9047	0.9060
tex	0.9587	**0.9983**	0.3518	0.9525	0.9542
veh	0.8523	**0.9940**	0.8520	0.8459	0.8521
wav	0.9638	**0.9984**	0.2068	0.9554	0.9585
win	0.9240	**0.9893**	0.1066	0.9213	0.9265
yea	0.9315	**0.9947**	0.9311	0.9256	0.9301
avg.	0.8450	**0.8920**	0.5299	0.8415	0.8870
dev.	0.2202	0.2682	0.3263	0.2194	0.1058

Table 8 presents the results using the HVR metric. Besides, we have used box-plots based on the *C* metric that calculates the dominance degree of the Pareto front approximations of every pair of algorithms (see Fig. 6). Each rectangle contains ten box-plots representing the distribution of the *C* values for a certain ordered pair of fitness functions in the 20 problem instances (from abalone to yeast in alphabetical order). Each box refers to algorithm *A* in the corresponding row and algorithm *B* in the corresponding column and gives the fraction of *B* covered by *A* $(C(A,B))$. Consider for instance the top right boxplots, which represent the fraction of solutions of the 2e variant, considering the joint optimization of double fault and complexity, covered by the non-dominated sets produced by the 2a vari-

Figure 6: Comparison of the Pareto fronts using C-measure by means of box-plots.

ant, composed of the training error and complexity measures. In each box, the minimum and maximum values are the lowest and highest lines, the upper and lower ends of the box are the upper and lower quartiles, and a thick line within the box shows the median.

Our analysis of the HVR measure clearly points out the best performing fitness function for the final learning goal. The 2b variant, considering the joint optimization of training error and variance, obtained the highest value in 18 out of 20 cases. Nevertheless, it provided some instability for two remaining datasets and thus a high standard deviation value. For abalone and optdigits it obtained the worst values among the five fitness function designs. Function 2a, optimizing training error and complexity, obtained two ties. Even being a deceptive fitness function, the 2d, which joins variance and complexity as objectives, and 2e variant, which combines double fault and complexity, obtained one tie as well as fitness function 2c, jointly optimizing training error and double fault.

Concerning the average value on the twenty datasets considered, the order is 2b, 2e, 2a, 2d, 2c. It is a surprising fact that the two deceptive functions are not located in the two last positions but are able to overcome almost all variants for 2e, and one other variant for 2d. It seems that, although they are not able to derive a diverse set of solutions, the selected fuzzy MCSs obtained show a good performance in the global learning objectives tradeoff curve. Anyway, from all the latter analysis we may conclude that the 2b fitness function seems to be the best performing variant.

The analysis of the C-measure (see Fig. 6) highlights the 2a and the 2b variants, which clearly outperform the other fitness functions, especially the latter. When comparing these two approaches between them, the 2b fitness function obtains better results. Notice that, comparisons with either 2d or 2e variants provide deceptive results due to the small number of solutions contained in their Pareto front approximations.

Hence, considering the information provided by the two multiobjective metrics we may clearly draw the conclusion that the 2b fitness function is the best performing approach.

Finally, in order to complement the latter analysis, the aggregated Pareto fronts will be represented graphically for three of the datasets: abalone, waveform, and magic (see Figs. 7, 8, and 9, respectively) in order to allow an easy visual comparison of the performance of the different variants of the fitness functions.

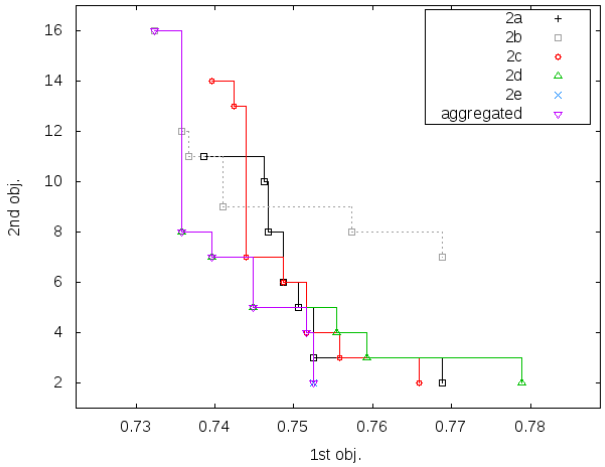

Fig. 7. The Pareto front approximations obtained for *abalone* using the five fitness functions. Objective 1 stands for test error and objective 2 for complexity. The pseudo-optimal Pareto front is also drawn for reference.

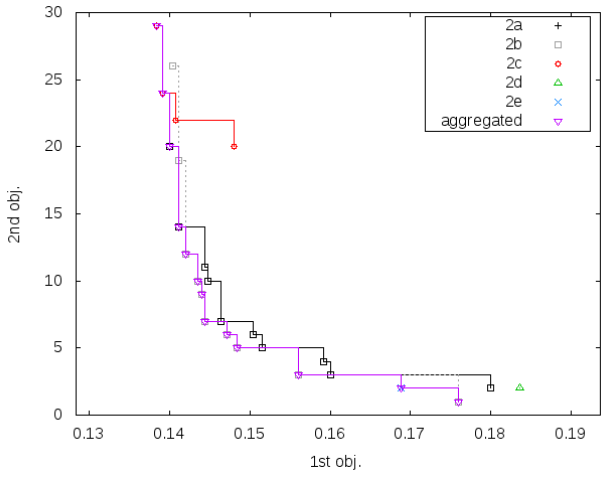

Fig. 8. The Pareto front approximations obtained for *waveform* using the five fitness functions. Objective 1 stands for test error and objective 2 for complexity. The pseudo-optimal Pareto front is also drawn for reference.

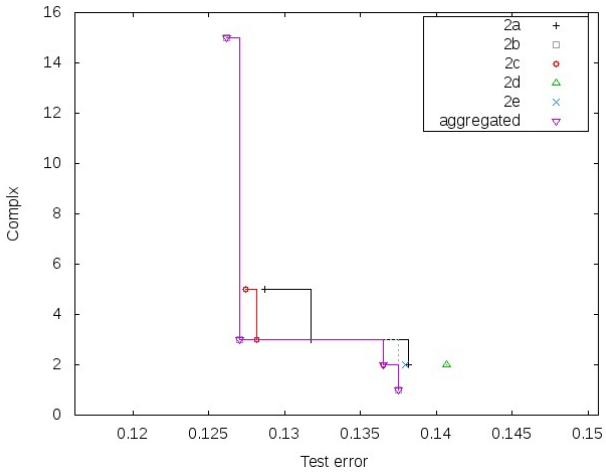

Fig. 9. The Pareto front approximations obtained for *magic* using the five fitness functions. Objective 1 stands for test error and objective 2 for complexity. The pseudo-optimal Pareto front is also drawn for reference.

It can be seen how the graphs corroborate the conclusions of the metrics analysis: the 2b fitness function obtained the worst HVR metric value for the abalone dataset, while it obtained the best performance, which is close to the pseudo-optimal Pareto front, when looking at the waveform and magic datasets. Notice how, the 2e variant obtains a single nondominated solution in the three problems but in all of them this solution is included in the pseudo-optimal Pareto front, thus justifying the good values obtained in the HVR and C metrics. That is not the case for function 2d whose small number of non-dominated solutions are usually far from the pseudo-optimal front. The bad performance of the 2c variant is also clearly observed.

5.3. Analysis and comparison of single solutions selected from the obtained Pareto front approximations

In this section, we aim to analyze the final performance of our proposal by imitating the process a human designer will develop in order to select a desired FURIA-based fuzzy MCS structure from those available in the obtained accuracy-complexity non-dominated fronts.

From each Pareto front approximation we have selected four different solutions, the one having the best value in the first objective in the considered fitness function, the one with the best value in the second objective in the considered fitness function, the one with the best tradeoff value, and the one with the best test error value. The tradeoff solution is selected as follows: we compute 1000 random weights $w1 \in [0,1]$, take the average value of the aggregation function of both objectives $Obj1$ and $Obj2$: $(w1 * Obj1 + (1-w1) * Obj2)$, and select the solution with the highest aggregated value. For each solution we present the values of three global learning objectives, Training error (*Tra*), test error (*Tst*), and complexity (*Cmpl*) in Tables 9 and 10. The average and standard deviation value for each of four different solutions in the 20 problems is also presented in Table 11. We do not show the two diversity measures values, since that was not the final learning objective. Note that we used diversity measures combined with accuracy and complexity measures in order to improve the accuracy-complexity tradeoff in the obtained FURIA-based fuzzy MCS.

From the results obtained we may draw following conclusions:

- The best performance in terms of test accuracy was obtained by the 2c variant. It outperforms the other approaches in 6 out of 20 cases (+7 ties) and also obtains the best average value. The 2b approach was one step behind obtaining 5 best results (+6 ties) and the second best average value. Of course, the 2d and 2e variants obtained the worst results, since they directly do not include accuracy in the objective space.
- Considering the complexity criterion the best results were obtained by 2a, 2d, and 2e variants. For all datasets they obtained the lowest number of classifiers equal to 2 and the lowest average value. Note that we discard 2b as the best value, which obtained number of classifiers equal to 1 15 times, since it is not considered as a MCS, but as a single classifier.
- It is rather hard to point a single approach finding the best accuracy-complexity tradeoff. The 2d and 2e approaches should be rather discarded, since for all of the solutions out of all datasets they provided the same complexity (equal to 2). Although the 2c fitness function provided the best averaged

Table 9: Statistics of four single solutions selected from the Pareto fronts.

		Best of 1st obj.			Best of 2nd obj.			Best tradeoff			Best test		
		Tra	Tst	Cmpl	Tra	Tst	Cmpl	Tra	Tst	Cmpl	Tra	Tst	Cmpl
aba	2a	0.505	0.751	17.700	0.605	0.776	2.000	0.605	0.776	2.000	0.512	0.746	14.400
	2b	0.509	0.750	22.300	0.599	0.756	9.400	0.509	0.750	22.000	0.536	0.741	18.600
	2c	0.506	0.750	16.800	0.622	0.780	2.000	0.506	0.750	20.000	0.511	0.745	13.900
	2d	0.599	0.756	9.400	0.689	0.791	2.000	0.689	0.791	2.000	0.606	0.752	8.000
	2e	0.611	0.773	2.000	0.611	0.773	2.000	0.611	0.773	2.000	0.649	0.759	2.000
bre	2a	0.000	0.039	3.000	0.007	0.054	2.000	0.007	0.054	2.000	0.001	0.038	2.900
	2b	0.000	0.045	3.600	0.009	0.045	1.000	0.000	0.045	6.000	0.005	0.037	2.700
	2c	0.000	0.038	3.000	0.000	0.038	3.000	0.000	0.038	3.000	0.000	0.037	3.000
	2d	0.014	0.051	2.000	0.014	0.051	2.000	0.014	0.051	2.000	0.014	0.050	2.000
	2e	0.010	0.052	2.000	0.010	0.052	2.000	0.010	0.052	2.000	0.023	0.037	2.000
gla	2a	0.004	0.309	6.800	0.113	0.364	2.000	0.113	0.364	2.000	0.015	0.286	5.600
	2b	0.007	0.325	8.700	0.113	0.359	1.000	0.063	0.348	15.000	0.052	0.288	4.700
	2c	0.005	0.300	7.200	0.162	0.397	2.100	0.066	0.323	7.000	0.021	0.283	5.500
	2d	0.142	0.360	2.000	0.142	0.360	2.000	0.142	0.360	2.000	0.142	0.360	2.000
	2e	0.133	0.372	2.000	0.133	0.372	2.000	0.133	0.372	2.000	0.193	0.305	2.000
hea	2a	0.001	0.184	4.500	0.050	0.199	2.000	0.050	0.199	2.000	0.013	0.172	3.700
	2b	0.001	0.197	5.500	0.057	0.215	1.000	0.001	0.198	3.000	0.030	0.170	2.900
	2c	0.000	0.185	4.800	0.064	0.214	2.200	0.000	0.185	5.000	0.010	0.178	3.900
	2d	0.074	0.201	2.000	0.074	0.201	2.000	0.074	0.201	2.000	0.074	0.201	2.000
	2e	0.059	0.204	2.000	0.059	0.204	2.000	0.059	0.204	2.000	0.100	0.155	2.000
ion	2a	0.003	0.153	2.800	0.008	0.149	2.000	0.008	0.149	2.000	0.006	0.144	2.500
	2b	0.003	0.162	3.000	0.027	0.169	1.000	0.003	0.162	3.000	0.013	0.145	2.300
	2c	0.004	0.126	18.700	0.004	0.126	18.700	0.004	0.126	16.000	0.004	0.126	18.700
	2d	0.024	0.156	2.000	0.024	0.156	2.000	0.024	0.156	2.000	0.024	0.156	2.000
	2e	0.014	0.162	2.000	0.014	0.162	2.000	0.014	0.162	2.000	0.034	0.129	2.000
mag	2a	0.097	0.133	6.600	0.113	0.144	2.000	0.113	0.144	2.000	0.097	0.132	5.600
	2b	0.098	0.132	8.200	0.107	0.143	1.000	0.100	0.135	3.000	0.098	0.132	7.400
	2c	0.097	0.133	6.600	0.115	0.144	2.000	0.108	0.140	2.000	0.098	0.132	5.600
	2d	0.118	0.146	2.000	0.118	0.146	2.000	0.118	0.146	2.000	0.118	0.146	2.000
	2e	0.116	0.145	2.000	0.116	0.145	2.000	0.116	0.145	2.000	0.119	0.142	2.000
opt	2a	0.401	0.686	2.000	0.401	0.686	2.000	0.401	0.686	2.000	0.452	0.655	2.000
	2b	0.175	0.632	30.200	0.175	0.632	30.200	0.175	0.632	37.000	0.198	0.625	26.000
	2c	0.175	0.632	30.200	0.175	0.632	30.200	0.175	0.632	37.000	0.198	0.625	26.000
	2d	0.400	0.685	2.000	0.400	0.685	2.000	0.400	0.685	2.000	0.451	0.654	2.000
	2e	0.400	0.685	2.000	0.400	0.685	2.000	0.400	0.685	2.000	0.451	0.654	2.000
pbl	2a	0.006	0.029	11.200	0.016	0.035	2.000	0.016	0.035	2.000	0.007	0.028	6.800
	2b	0.007	0.028	8.600	0.015	0.032	1.000	0.010	0.030	3.000	0.009	0.027	4.800
	2c	0.006	0.028	10.900	0.020	0.037	2.000	0.013	0.034	3.000	0.007	0.027	7.600
	2d	0.017	0.034	2.000	0.017	0.034	2.000	0.017	0.034	2.000	0.017	0.034	2.000
	2e	0.016	0.033	2.000	0.016	0.033	2.000	0.016	0.033	2.000	0.020	0.031	2.000
pen	2a	0.000	0.017	9.300	0.010	0.031	2.000	0.010	0.031	2.000	0.000	0.016	8.200
	2b	0.000	0.017	11.300	0.011	0.033	1.000	0.005	0.024	2.000	0.000	0.016	8.700
	2c	0.000	0.014	21.800	0.000	0.014	21.800	0.000	0.014	24.000	0.000	0.014	21.800
	2d	0.011	0.032	2.000	0.011	0.032	2.000	0.011	0.032	2.000	0.011	0.032	2.000
	2e	0.010	0.032	2.000	0.010	0.032	2.000	0.010	0.032	2.000	0.013	0.029	2.000
pho	2a	0.058	0.126	9.800	0.086	0.151	2.000	0.086	0.151	2.000	0.059	0.125	9.000
	2b	0.059	0.127	8.800	0.083	0.150	1.000	0.059	0.127	9.000	0.061	0.127	7.600
	2c	0.058	0.126	10.000	0.097	0.160	2.000	0.080	0.145	9.000	0.059	0.125	9.400
	2d	0.089	0.153	2.000	0.089	0.153	2.000	0.089	0.153	2.000	0.089	0.153	2.000
	2e	0.090	0.152	2.000	0.090	0.152	2.000	0.090	0.152	2.000	0.097	0.144	2.000

test accuracy, it should also be skipped, as this good performance is obtained at the cost of the highest complexity, with a very significant difference. Thus, the two left fitness functions are 2a and 2b. The first one provides rather lower complexity on average, whereas the latter one obtains better averaged accuracy with a slightly higher complexity.

To conclude, let us try to have an insight into the influence of the relation between the two objectives and the final success in the learning problem. Combination of diversity measures with complexity tends to produce small ensemble sizes. The two fitness functions obtained ensembles composed of only two classifiers. Consequently, these ensembles do not have a high quality, they obtained rather low accuracy. Combination of the training error with complexity seems to be a good way to look for the good balance between performance and the number of classifiers in the ensemble. However, it seems to overfit quite often. Our last proposal was a combination of training error with two diversity measures. Such two fitness functions, 2b and 2c, lead to finally selected high quality ensembles with a good accuracy-complexity tradeoff, as the ensembles are kept quite small in most of the cases. Thus, we may conclude that the combination of training error and diversity measures for genetic classifier selection in FURIA-based fuzzy classifiers leads to the obtaining

Table 10: Statistics of four single solutions selected from the Pareto fronts.(cont.)

		Best of 1st obj.			Best of 2nd obj.			Best tradeoff			Best test		
		Tra	Tst	Cmpl	Tra	Tst	Cmpl	Tra	Tst	Cmpl	Tra	Tst	Cmpl
pim	2a	0.027	0.245	10.200	0.087	0.283	2.000	0.087	0.283	2.000	0.033	0.235	6.900
	2b	0.024	0.245	11.600	0.100	0.285	1.000	0.070	0.267	2.000	0.041	0.233	6.800
	2c	0.021	0.247	11.000	0.103	0.287	2.000	0.054	0.266	12.000	0.033	0.236	7.000
	2d	0.103	0.277	2.000	0.103	0.277	2.000	0.103	0.277	2.000	0.103	0.277	2.000
	2e	0.090	0.279	2.000	0.090	0.279	2.000	0.090	0.279	2.000	0.128	0.231	2.000
sat	2a	0.010	0.103	16.600	0.049	0.130	2.000	0.049	0.130	2.000	0.013	0.102	11.600
	2b	0.011	0.103	17.200	0.049	0.132	1.000	0.011	0.103	15.000	0.012	0.101	14.600
	2c	0.010	0.102	21.400	0.010	0.102	21.600	0.010	0.102	17.000	0.010	0.102	21.200
	2d	0.053	0.129	2.000	0.053	0.129	2.000	0.053	0.129	2.000	0.053	0.129	2.000
	2e	0.050	0.130	2.000	0.050	0.130	2.000	0.050	0.130	2.000	0.060	0.119	2.000
seg	2a	0.000	0.031	6.600	0.012	0.047	2.000	0.012	0.047	2.000	0.001	0.029	5.500
	2b	0.000	0.033	6.700	0.015	0.043	1.000	0.004	0.036	6.000	0.002	0.029	5.100
	2c	0.000	0.027	17.600	0.000	0.027	17.600	0.000	0.027	20.000	0.000	0.027	17.600
	2d	0.017	0.047	2.000	0.017	0.047	2.000	0.017	0.047	2.000	0.017	0.047	2.000
	2e	0.013	0.048	2.000	0.013	0.048	2.000	0.013	0.048	2.000	0.019	0.037	2.000
son	2a	0.000	0.212	3.300	0.037	0.253	2.000	0.037	0.253	2.000	0.005	0.203	3.000
	2b	0.000	0.228	3.700	0.062	0.252	1.000	0.000	0.228	3.000	0.015	0.217	3.200
	2c	0.000	0.222	3.300	0.011	0.232	2.800	0.000	0.222	3.000	0.002	0.213	3.000
	2d	0.064	0.274	2.000	0.064	0.274	2.000	0.064	0.274	2.000	0.065	0.269	2.000
	2e	0.042	0.262	2.000	0.042	0.262	2.000	0.042	0.262	2.000	0.081	0.188	2.000
spa	2a	0.015	0.058	9.400	0.033	0.071	2.000	0.033	0.071	2.000	0.016	0.057	7.600
	2b	0.015	0.057	9.800	0.029	0.070	1.000	0.015	0.057	11.000	0.018	0.056	6.800
	2c	0.015	0.056	12.800	0.015	0.056	12.200	0.015	0.056	11.000	0.015	0.056	12.800
	2d	0.034	0.072	2.000	0.034	0.072	2.000	0.034	0.072	2.000	0.034	0.072	2.000
	2e	0.034	0.072	2.000	0.034	0.072	2.000	0.034	0.072	2.000	0.040	0.065	2.000
tex	2a	0.000	0.033	8.000	0.020	0.064	2.000	0.020	0.064	2.000	0.000	0.032	7.300
	2b	0.000	0.034	9.200	0.021	0.062	1.000	0.001	0.035	14.000	0.001	0.033	7.800
	2c	0.000	0.028	23.200	0.000	0.028	23.200	0.000	0.028	21.000	0.000	0.028	23.200
	2d	0.024	0.067	2.000	0.024	0.067	2.000	0.024	0.067	2.000	0.024	0.067	2.000
	2e	0.021	0.062	2.000	0.021	0.062	2.000	0.021	0.062	2.000	0.025	0.058	2.000
veh	2a	0.002	0.267	13.400	0.099	0.290	2.000	0.099	0.290	2.000	0.011	0.257	9.500
	2b	0.003	0.272	13.400	0.104	0.303	1.000	0.058	0.289	2.000	0.023	0.255	7.500
	2c	0.002	0.271	14.000	0.118	0.302	2.000	0.058	0.287	7.000	0.015	0.260	8.600
	2d	0.112	0.307	2.000	0.112	0.307	2.000	0.112	0.307	2.000	0.112	0.307	2.000
	2e	0.107	0.300	2.000	0.107	0.300	2.000	0.107	0.300	2.000	0.139	0.270	2.000
wav	2a	0.001	0.150	21.000	0.059	0.194	2.000	0.059	0.194	2.000	0.003	0.148	17.400
	2b	0.002	0.149	23.100	0.067	0.192	1.000	0.010	0.159	7.000	0.003	0.146	18.700
	2c	0.001	0.146	26.200	0.001	0.146	26.400	0.001	0.146	29.000	0.001	0.146	26.400
	2d	0.064	0.197	2.000	0.064	0.197	2.000	0.064	0.197	2.000	0.064	0.197	2.000
	2e	0.061	0.194	2.000	0.061	0.194	2.000	0.061	0.194	2.000	0.072	0.181	2.000
win	2a	0.000	0.052	2.100	0.001	0.054	2.000	0.001	0.054	2.000	0.000	0.051	2.100
	2b	0.000	0.072	1.700	0.006	0.057	1.000	0.000	0.072	2.000	0.004	0.054	1.400
	2c	0.000	0.021	17.400	0.000	0.021	17.400	0.000	0.021	17.000	0.000	0.018	18.700
	2d	0.007	0.066	2.000	0.007	0.066	2.000	0.007	0.066	2.000	0.007	0.065	2.000
	2e	0.007	0.058	2.000	0.007	0.058	2.000	0.007	0.058	2.000	0.022	0.037	2.000
yea	2a	0.156	0.406	11.100	0.250	0.452	2.000	0.250	0.452	2.000	0.158	0.404	10.100
	2b	0.158	0.410	10.900	0.254	0.464	1.000	0.158	0.410	11.000	0.188	0.396	7.100
	2c	0.156	0.412	12.500	0.282	0.467	2.000	0.156	0.412	13.000	0.159	0.405	10.800
	2d	0.281	0.445	2.000	0.281	0.445	2.000	0.281	0.445	2.000	0.281	0.445	2.000
	2e	0.260	0.453	2.000	0.260	0.453	2.000	0.260	0.453	2.000	0.291	0.421	2.000

Table 11: A comparison of the averaged performance of the four single solutions selected from the obtained Pareto sets.

		Best of 1st obj.			Best of 2nd obj.			Best tradeoff			Best test		
		Tra	Tst	Cmpl	Tra	Tst	Cmpl	Tra	Tst	Cmpl	Tra	Tst	Cmpl
2a	avg.	0.064	0.199	8.770	0.103	0.221	2.000	0.103	0.221	2.000	0.047	0.193	7.085
	dev.	0.140	0.207	5.373	0.152	0.210	0.000	0.152	0.210	0.000	0.116	0.202	4.135
2b	avg.	0.054	0.201	10.875	0.095	0.220	2.880	0.063	0.205	8.800	0.056	0.191	8.235
	dev.	0.120	0.200	7.337	0.134	0.202	6.698	0.118	0.201	8.752	0.122	0.197	6.411
2c	avg.	0.053	0.193	14.470	0.090	0.210	10.660	0.062	0.198	13.800	0.047	0.189	13.235
	dev.	0.119	0.203	7.681	0.148	0.214	10.212	0.118	0.204	9.518	0.116	0.200	7.878
2d	avg.	0.112	0.223	2.370	0.117	0.225	2.000	0.117	0.225	2.000	0.083	0.221	2.300
	dev.	0.150	0.206	1.655	0.166	0.211	0.000	0.166	0.211	0.000	0.064	0.202	1.342
2e	avg.	0.107	0.223	2.000	0.107	0.223	2.000	0.107	0.223	2.000	0.000	0.200	2.000
	dev.	0.153	0.210	0.000	0.153	0.210	0.000	0.153	0.210	0.000	0.000	0.203	0.000

good results and is a promising approach. Nevertheless, this is a quite subjective and user-dependent decision.

5.4. Comparison between EMO-selected and non-selected FURIA-based fuzzy MCSs

This subsection presents a final benchmarking of the performance of NSGA-II combined with FURIA-

Table 12: A comparison of the NSGA-II FURIA-based fuzzy MCSs against static FURIA-based MCS.

NSGA-II combined with FURIA-based MCSs.																				
	aba	bre	gla	hea	ion	mag	opt	pbl	pen	pho	pim	sat	seg	son	spa	tex	veh	wav	win	yea
test err.	**0.741**	**0.037**	0.283	**0.170**	**0.126**	**0.132**	**0.625**	**0.027**	**0.014**	**0.125**	**0.231**	**0.101**	**0.027**	**0.188**	**0.056**	**0.028**	**0.255**	**0.146**	**0.018**	**0.396**
fitness func.	2b	2b	2c	2b	2c	2a	2b	2c	2c	2c	2e	2b	2c	2e	2b	2c	2b	2c	2c	2b
nr of cl.	18.6	2.7	5.5	2	18.7	5.6	26	4.8	21.8	9	2	14.6	17.6	2	6.8	23.2	7.5	18.7	18.7	7.1
FURIA-based MCSs algorithms Small ensemble sizes.																				
	aba	bre	gla	hea	ion	mag	opt	pbl	pen	pho	pim	sat	seg	son	spa	tex	veh	wav	win	yea
test err.	0.753	**0.037**	0.313	0.178	0.134	0.136	0.628	0.028	0.015	0.136	0.235	0.105	0.035	0.198	0.061	0.036	0.276	0.156	0.036	0.408
nr of cl.	10	10	7	7	7	7	10	10	10	10	10	10	10	10	10	10	10	10	10	10
FURIA-based MCSs algorithms. Ensemble size 50.																				
	aba	bre	gla	hea	ion	mag	opt	pbl	pen	pho	pim	sat	seg	son	spa	tex	veh	wav	win	yea
test err.	0.748	0.041	0.287	0.182	0.145	0.135	0.630	0.028	0.016	0.135	0.241	0.102	0.034	0.226	0.059	0.031	0.275	0.149	0.035	0.400
C4.5 ensembles with bagging. Small ensemble sizes.																				
	aba	bre	gla	hea	ion	mag	opt	pbl	pen	pho	pim	sat	seg	son	spa	tex	veh	wav	win	yea
test err.	0.772	0.043	0.306	0.194	0.149	0.134	0.697	0.03	0.028	0.131	0.253	0.112	0.042	0.247	0.067	0.051	0.289	0.193	0.097	0.415
nr of cl.	10	7	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
Random forests. Small ensemble sizes.																				
	aba	bre	gla	hea	ion	mag	opt	pbl	pen	pho	pim	sat	seg	son	spa	tex	veh	wav	win	yea
test err.	0.777	0.041	**0.282**	0.211	0.14	0.134	0.695	0.031	0.016	0.119	0.264	0.104	0.034	0.239	0.06	0.04	0.269	0.185	0.048	0.438
nr of cl.	7	7	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10

based fuzzy MCSs. The aim of this work is to propose an advanced OCS framework, embedding NSGA-II with five different two-objective fitness function designs into FURIA-based fuzzy MCS. By doing so, we would like to obtain a good accuracy-complexity tradeoff. We present a comparison between the best results obtained from the genetically selected FURIA-based fuzzy MCSs, against these ensembles obtained with a fixed ensemble size. FURIA-based fuzzy MCSs are comprised by 7 or 10 classifiers, the small ensemble sizes providing the best results in our previous contribution [17], and with 50 classifiers, the original structure of the EMO-selected fuzzy MCSs in the previous sections. We also compare it with two state-of-the-art algorithms, random forests [38] and bagging C4.5 MCSs [54], comprised by 7 or 10 classifiers [17].

Table 12 presents test errors for all the datasets. It may be clearly seen that our new approach obtained the best performance overall. It outperformed the others in 18 out of 20 cases (+1 tie). Considering complexity, EMO-selected fuzzy MCSs keep a reasonably low number of classifiers, obtaining value 2 (3 times) in the best case and value 26 (for the optdigits dataset) in the worst case. Comparing to the original small ensemble sizes it is enough to increase the amount of classifier up to 2.5 times in the worst case in order to obtain good performance. Notice that in 11 out of 20 cases the EMO-selected fuzzy MCSs obtained the lowest complexity of the five MCS design variants considered. Thus, we may draw the conclusion that NSGA-II combined with FURIA-based fuzzy MCSs is a good approach to obtain high quality, well performing ensembles with a good accuracy-complexity tradeoff, when dealing with high dimensional datasets.

6. Conclusions and future works

In this study, we proposed a methodology in which a bagging approach together with a feature selection technique is used to train Fuzzy Unordered Rules Induction Algorithm (FURIA) in order to obtain a Fuzzy Multiclassifier System. We used a single winner-based method on top of the base classifiers. We proved that a single FURIA classifier performs well and is able to provide instability capabilities to some extent. Then, we tested FURIA Multiclassifier Systems with bagging, feature selection, and combination of both of them. By using the said techniques, we aimed to obtain FURIA-based fuzzy MCSs properly dealing with high dimensional data.

We have conducted comprehensive experiments over 21 datasets taken from the UCI machine learning repository. It turned out that the combination of bagging with FURIA turned out to be the best from among proposed ones. This approach provided promising results in comparison to the two state-of-the-art algorithms.

One of the next steps we will consider in the future line is to employ a search algorithm such like single and multiobjective metaheuristics, greedy al-

gorithms, local search, etc. to look for an optimal size of the ensemble. The other way to follow is to try to combine classifiers in a dynamic manner, in a way that a classifier or a set of them is responsible just for a particular data region. Furthermore, we would like to study the influence of other parameters (FURIA parameters, MCS parameters, etc.).

References

1. L. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms*. Wiley, 2004.
2. S.J. Verzi, G.L. Heileman, and M. Georgiopoulos. Boosted ARTMAP: Modifications to fuzzy ARTMAP motivated by boosting theory. *Neural Networks*, 19(4):446–468, 2006.
3. W. Pedrycz and K.C. Kwak. Boosting of granular models. *Fuzzy Sets and Systems*, 157(22):2934–2953, 2006.
4. H. Takahashi and H. Honda. A new reliable cancer diagnosis method using boosted fuzzy classifier with a SWEEP operator method. *Journal of Chemical Engineering of Japan*, 38(9):763–773, 2005.
5. J. Canul-Reich, L. Shoemaker, and L.O. Hall. Ensembles of fuzzy classifiers. In *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 1–6, London (UK), 2007.
6. Y. Nojima and H. Ishibuchi. Genetic rule selection with a multi-classifier coding scheme for ensemble classifier design. *International Journal of Hybrid Intelligent Systems*, 4(3):157–169, 2007.
7. C. Marsala. Data mining with ensembles of fuzzy decision trees. In *IEEE Symposium on Computational Intelligence and Data Mining*, pages 348–354, Nashville (USA), 2009.
8. P. P. Bonissone, J. M. Cadenas, M. C. Garrido, and R. A. Díaz-Valladares. A fuzzy random forest. *International Journal of Approximate Reasoning*, 51(7):729–747, 2010.
9. O. Cordón, A. Quirin, and L. Sánchez. A first study on bagging fuzzy rule-based classification systems with multicriteria genetic selection of the component classifiers. In *Third International Workshop on Genetic and Evolving Fuzzy Systems (GEFS)*, pages 11–16, Witten-Bommerholz (Germany), 2008.
10. O. Cordón 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 Intelligent Systems*, 7(1):45–64, 2010.
11. K. Trawiński, A. Quirin, and O. Cordón. Bi-criteria genetic selection of bagging fuzzy rule-based multiclassification systems. In *IFSA/EUSFLAT Conf.*, pages 1514–1519, 2009.
12. K. Trawiński, A. Quirin, and O. Cordón. On the combination of accuracy and diversity measures for genetic selection of bagging fuzzy rule-based multiclassification systems. *International Conference on Intelligent Systems Design and Applications (ISDA)*, 0:121–127, 2009.
13. D. Partridge and W.B. Yates. Engineering multiversion neural-net systems. *Neural Computation*, 8(4):869–893, 1996.
14. O. Cordón, F. Herrera, F. Hoffmann, and L. Magdalena. *Genetic Fuzzy Systems. Evolutionary Tuning and Learning of Fuzzy Knowledge Bases*. World Scientific, 2001.
15. O. Cordón, F. Gomide, F. Herrera, F. Hoffmann, and L. Magdalena. Ten years of genetic fuzzy systems: Current framework and new trends. *Fuzzy Sets and Systems*, 141(1):5–31, 2004.
16. F. Herrera. Genetic fuzzy systems: taxonomy, current research trends and prospects. *Evolutionary Intelligence*, 1(1):27–46, January 2008.
17. K. Trawiński, O. Cordón, and A. Quirin. On designing fuzzy multiclassifier systems by combining furia with bagging and feature selection. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 19(4):589–633, 2011.
18. J. C. Hühn and E. Hüllermeier. FURIA: an algorithm for unordered fuzzy rule induction. *Data Mining and Knowledge Discovery*, 19(3):293–319, 2009.
19. J. C. Hühn and E. Hüllermeier. An analysis of the FURIA algorithm for fuzzy rule induction. In *Advances in Machine Learning I*, pages 321–344. 2010.
20. T. Ho. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998.
21. R. Battiti. Using mutual information for selecting features in supervised neural net learning. *IEEE Transactions on Neural Networks*, 5(4):537–550, 1994.
22. T.A. Feo and M.G.C. Resende. Greedy randomized adaptive search procedures. *Journal of Global Optimization*, 6:109–133, 1995.
23. H. Ishibuchi, T. Nakashima, and M. Nii. *Classification and Modeling with Linguistic Information Granules: Advanced Approaches to Linguistic Data Mining (Advanced Information Processing)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2004.
24. C.A. Coello, G.B. Lamont, and D.A. Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems, 2nd Edition*. Springer, 2007.
25. Ludmila I. Kuncheva and Christopher J. Whitaker. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning*, 51(2):181–207, 2003.
26. D. Ruta and B. Gabrys. Classifier selection for major-

ity voting. *Information Fusion*, 6(1):63–81, 2005.
27. A. Tsymbal, M. Pechenizkiy, and P. Cunningham. Diversity in search strategies for ensemble feature selection. *Information Fusion*, 6(1):83–98, 2005.
28. K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6:182–197, 2002.
29. E.M. Dos Santos, R. Sabourin, and P. Maupin. Single and multi-objective genetic algorithms for the selection of ensemble of classifiers. In *International Joint Conference on Neural Networks (IJCNN)*, pages 3070–3077, Vancouver, 2006.
30. E.M. Dos Santos, R. Sabourin, and P. Maupin. A dynamic overproduce-and-choose strategy for the selection of classifier ensembles. *Pattern Recognition*, 41(10):2993–3009, 2008.
31. T.G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization. *Machine Learning*, 40(2):139–157, 2000.
32. R.E. Banfield, L.O. Hall, K.W. Bowyer, and W.P. Kegelmeyer. A comparison of decision tree ensemble creation techniques. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(1):173–180, 2007.
33. D. Optiz and R. Maclin. Popular ensemble methods: an empirical study. *Journal of Artificial Intelligence Research*, 11:169–198, 1999.
34. L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.
35. R. Schapire. The strength of weak learnability. *Machine Learning*, 5(2):197–227, 1990.
36. Z.H. Zhou. Ensembling local learners through multimodal perturbation. *IEEE Transactions of Systems, Man, and Cybernetics, Part B: Cybernetics*, 35(4):725–735, 2005.
37. L. Xu, A. Krzyzak, and C.Y. Suen. Methods of combining multiple classifiers and their application to handwriting recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, 22(3):418–435, 1992.
38. L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.
39. M.J. del Jesus, F. Hoffmann, L.J. Navascues, and L. Sánchez. Induction of fuzzy-rule-based classifiers with evolutionary boosting algorithms. *IEEE Transactions on Fuzzy Systems*, 12(3):296–308, 2004.
40. L. Sánchez and J. Otero. Boosting fuzzy rules in classification problems under single-winner inference. *International Journal of Intelligent Systems*, 22(9):1021–1034, 2007.
41. H. Takahashi and H. Honda. Lymphoma prognostication from expression profiling using a combination method of boosting and projective adaptive resonance theory. *Journal of Chemical Engineering of Japan*, 39(7):767–771, 2006.
42. C. Z. Janikow. Fuzzy decision trees: issues and methods. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 28(1):1–14, 1998.
43. J.J. Aguilera, M. Chica, M.J. del Jesus, and F. Herrera. Niching genetic feature selection algorithms applied to the design of fuzzy rule based classification systems. In *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 1794–1799, London (UK), 2007.
44. Y. Nojima and H. Ishibuchi. Designing fuzzy ensemble classifiers by evolutionary multiobjective optimization with an entropy-based diversity criterion. In *International Conference on Hybrid Intelligent Systems and Conference on Neuro-Computing and Evolving Intelligence*, Auckland, New Zealand, 2006.
45. H. Ishibuchi and Y. Nojima. Evolutionary multiobjective optimization for the design of fuzzy rule-based ensemble classifiers. *International Journal of Hybrid Intelligent Systems*, 3(3):129–145, 2006.
46. R. Schapire, Y. Freund, P. Bartlett, and W. Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. In *International Conference on Machine Learning*, pages 322–330, Nashville (USA), 1997.
47. B.V. Dasarathy and B.V. Sheela. A composite classifier system design: Concepts and methodology. *Proceedings of IEEE*, 67(5):708–713, 1979.
48. L.S. Oliveira, M. Morita, R. Sabourin, and F. Bortolozzi. Multi-objective genetic algorithms to create ensemble of classifiers. *Lecture Notes in Computer Science*, 3410:592–606, 2005.
49. W. W. Cohen. Fast effective rule induction. In *In Proceedings of the Twelfth International Conference on Machine Learning*, pages 115–123. Morgan Kaufmann, 1995.
50. H. Ishibuchi, T. Nakashima, and T. Morisawa. Voting in fuzzy rule-based systems for pattern classification problems. *Fuzzy Sets and Systems*, 103(2):223–238, 1999.
51. O. Cordón, M.J. del Jesus, and F. Herrera. A proposal on reasoning methods in fuzzy rule-based classification systems. *International Journal of Approximate Reasoning*, 20:21–45, 1999.
52. P. Panov and S. Džeroski. Combining bagging and random subspaces to create better ensembles. In *IDA'07: Proceedings of the 7th international conference on Intelligent data analysis*, pages 118–129, Berlin, Heidelberg, 2007. Springer-Verlag.
53. E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, 3:257–271, 1999.
54. J. R. Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., San Fran-

cisco, CA, USA, 1993.
55. T.G. Dietterich. Approximate statistical test for comparing supervised classification learning algorithms. *Neural Computation*, 10(7):1895–1923, 1998.
56. J. Knowles and D. Corne. On metrics for comparing nondominated sets. In *2002 Congress on Evolutionary Computation CEC '02*, volume 1, pages 711–716, 2002.