Evolving Fuzzy Rules with UCS: Preliminary Results

Albert Orriols-Puig[1], Jorge Casillas[2], and Ester Bernadó-Mansilla[1]

[1]Grup de Recerca en Sistemes Intel·ligents
Enginyeria i Arquitectura La Salle
Universitat Ramon Llull
Quatre Camins 2, 08022 Barcelona, Spain
{aorriols,esterb}@salle.url.edu
[2]Dept. Computer Science and Artificial Intelligence
University of Granada
18071, Granada, Spain
casillas@ugr.es

Abstract. This paper presents Fuzzy-UCS, a Michigan-style Learning Fuzzy-Classifier System designed for supervised learning tasks. Fuzzy-UCS combines the generalization capabilities of UCS with the good interpretability of fuzzy rules to evolve highly accurate and understandable rule sets. Fuzzy-UCS is tested on a large collection of real-world problems, and compared to UCS and three highly-used machine learning techniques: the decision tree C4.5, the support vector machine SMO, and the fuzzy boosting algorithm Fuzzy LogitBoost. The results show that Fuzzy-UCS is highly competitive with respect to the four learners in terms of performance, and that the fuzzy representation permits a much better understandability of the evolved knowledge. These promising results of the online architecture of Fuzzy-UCS allow for further research and application of the system to new challenging problems.

1 Introduction

Michigan-style Learning Classifier Systems (LCSs) [19] are online machine learning techniques that use Genetic Algorithms (GAs) [19,18] to evolve a rule-based knowledge. Among the different uses, several LCSs have been designed for performing supervised learning tasks [34,4,3]. Typically, LCSs deal with numerical attributes by means of evolving a set of interval-based rules that cooperate to predict the output of new unlabeled examples. Although the competence of LCSs in terms of accuracy has been widely shown, this excellence has been hindered by a poor interpretability of the evolved rule sets, which typically consist of large sets of overlapping interval-based rules that can hardly be read by human experts.

During the last decade, the interest in Fuzzy Rule-Based Systems (FRBSs) [11] has increased since they provide a robust, flexible, and powerful methodology to deal with *noisy*, *imprecise*, and *incomplete* data. Besides, the fuzzy representation allows for a better interpretability of the classification models. This has led to the first analyses and designs of *Learning Fuzzy-Classifier Systems* (LFCSs). Since the introduction of the first LFCS [30], several new architectures have been proposed [23,17,32,6,7,9], which have been mostly applied to reinforcement learning and control tasks. One of the first proposals of LFCS for pattern classification was presented in [20]. These first successful steps toward the design of competent LFCS warrants for further investigation, especially in the supervised learning paradigm.

In this paper, we address the problem of interpretability in LCSs, and propose Fuzzy-UCS, an online accuracy-based LFCS architecture that works under a supervised learning paradigm. We depart from the UCS classifier system, which has been shown to be highly competitive with respect to some of the most used machine learning techniques [3]. We introduce a linguistic fuzzy representation to UCS, and redesign most of its components to permit the system to deal with fuzzy rules. With the inclusion of fuzzy rules, we seek for a better interpretability of the evolved knowledge, as well as a reduction in the search space, while maintaining a performance similar to the one obtained with an interval-based representation. Moreover, we also prepare the system to be able to deal with vague and uncertain data.

The remaining of this paper is organized as follows. Section 2 deeply explains the proposed Fuzzy-UCS architecture, especially focusing on the differences from the original UCS. In Sect. 3, we analyze the behavior of Fuzzy-UCS on a large collection of real-world problems, and compare the performance an interpretability of the models evolved by Fuzzy-UCS to those created by UCS and three other machine learning techniques: C4.5, SMO, and Fuzzy LogitBoost. Finally, Sect. 4 concludes, highlights the differential traits of Fuzzy-UCS, and enumerates new opportunities that will be addressed as further work.

2 Description of Fuzzy-UCS

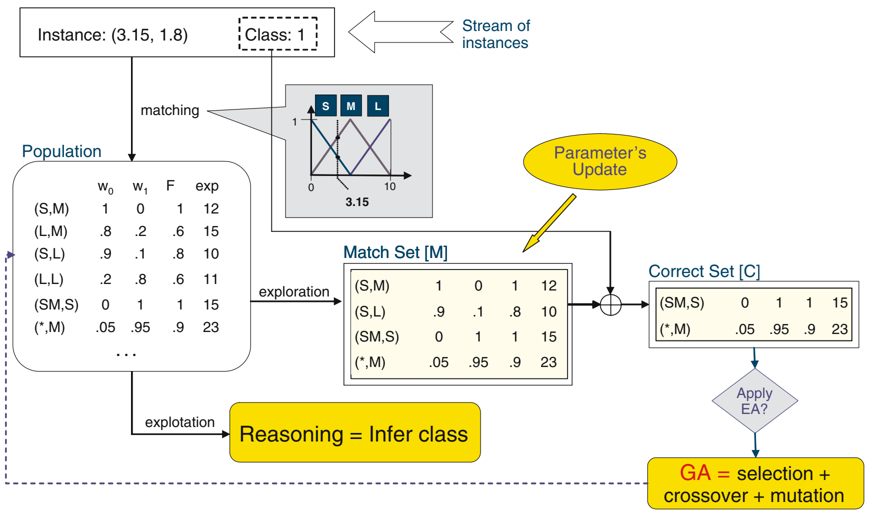
Figure 1 schematically shows the learning process of Fuzzy-UCS. The learner works in two different modes: training or *exploration* mode and testing or *exploitation* mode. During explore, Fuzzy-UCS evaluates online the quality of the rule-based knowledge, and evolves it by means of a GA. During test, Fuzzy-UCS uses the rules to infer the output of a given input instance. The different components of the system are detailed as follows.

2.1 Representation

Fuzzy-UCS evolves a *population* [P] of classifiers, where each classifier consists of a *linguistic fuzzy rule* and a set of parameters. The fuzzy rule is represented as follows:

$$\textbf{IF } x_1 \text{ is } \widetilde{A_1^k} \text{ and } \cdots \text{ and } x_n \text{ is } \widetilde{A_n^k} \textbf{ THEN } c^k \textbf{ WITH } F^k \tag{1}$$

where each input variable x_i is represented by a disjunction (T-conorm operator) of linguistic terms $\widetilde{A_i^k} = \{A_{i1} \vee ... \vee A_{in_i}\}$. In our experiments, all the input



	w_0	w_1	F	exp
(S,M)	1	0	1	12
(L,M)	.8	.2	.6	15
(S,L)	.9	.1	.8	10
(L,L)	.2	.8	.6	11
(SM,S)	0	1	1	15
(*,M)	.05	.95	.9	23
. . .				

(S,M)	1	0	1	12
(S,L)	.9	.1	.8	10
(SM,S)	0	1	1	15
(*,M)	.05	.95	.9	23

(SM,S)	0	1	1	15
(*,M)	.05	.95	.9	23

Fig. 1. Schematic illustration of Fuzzy-UCS

variables share the same semantics. The variables are defined by triangular-shaped fuzzy membership functions (see examples of these semantics with three and five linguistic terms per variable in Fig. 2).

The consequent of the rule internally maintains one weight for each of the m classes $\{w_1^k, \cdots, w_m^k\}$. Each weight w_j^k indicates the soundness with which the rule k predicts class j for an example that fully matches this rule. These weights are incrementally updated with the learning interaction (see Sect. 2.3), and serve to calculate the class of the rule. That is, the class c^k predicted by the rule is the class that has associated the weight with maximum value.

Each classifier has four main parameters: a) the fitness F, which estimates the accuracy of the rule, b) the correct set size cs, which averages the sizes of the correct sets in which the classifier has participated, c) the experience exp, which reckons the contributions of the rule to classify the input instances, and d) the numerosity num, which counts the number of copies of the classifier in the population. All these parameters are updated online as specified in Sect. 2.3.

To implement this representation, we propose to use a binary coding for the antecedent of the rule. That is, a one-valued allele indicates that the corresponding linguistic term is used in this variable. The class predicted by the rule is codified as an integer, and the fitness as a float number. For instance, if we have three linguistic labels {S [small], M [medium], L [large]} for each input and two possible classes $\{c_1, c_2\}$, the fuzzy rule

$$\textbf{IF } x_1 \text{ is } S \text{ and } x_2 \text{ is } \{S \text{ or } L\} \textbf{ THEN } c_1 \textbf{ WITH } 0.8 \tag{2}$$

is encoded as: $[100|101||c1|0.8]$.

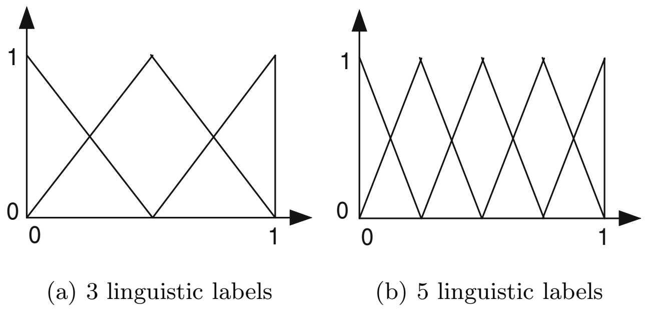


Fig. 2. Representation of a fuzzy partition for a variable with (a) three and (b) five triangular-shaped membership functions

2.2 Performance Component

The performance component of Fuzzy-UCS is inherited from UCS and adapted to deal with the new fuzzy representation. UCS learns under a supervised learning scheme. Given an input example e with its associated class c, UCS creates the *match set* [M] with all the classifiers in [P] that *match* the input instance. Then, [M] is used differently depending on whether the system is running on explore or on exploit mode. In explore mode, UCS forms the *correct set* [C], which consists of all the classifiers in [M] that advocate the class of the input example. If [C] is empty, *covering* is triggered. In exploit mode, the best action selected from the vote (weighted by fitness) of all classifiers in [M] is returned as the predicted output.

Fuzzy-UCS follows this process, but the role of the matching and the inference processes changes as they are adapted to deal with linguistic terms. In the following, the matching degree calculation and the phases followed in explore mode are detailed. The inference mechanism used during test is explained in Sect. 2.5.

Calculation of the matching degree. The *matching degree* $\mu_{A^k}(e)$ of a rule k with an example e is computed as follows. For each variable x_i of the rule, we compute the membership degree for each of its linguistic terms, and aggregate them by means of a T-conorm (disjunction). Then, the matching degree of the rule is determined by the T-norm (conjunction) of the matching degree of all the input variables. In our implementation, we used a *bounded sum* ($min\{1, a + b\}$) as T-conorm and the *product* ($a \cdot b$) as T-norm. Note that we used a bounded sum instead of other typical operators for the T-conorm to emulate the *don't care* used in the crisp representation. That is, with the bounded sum, a variable will have no influence if all its linguistic terms are set to '1'.

Creation of the match set. Given the input e, all the classifiers with a *matching degree* greater than zero form the *match set*.

Creation of the correct set. Next, the *correct set* [C] is created with all the classifiers in [M] that advocate the class c. If there is not any rule in [C] that matches e with the maximum matching degree, the *covering operator* is triggered. We create the rule that matches e with maximum degree as follows. For each variable x_i, the linguistic term with maximum matching with e_i is activated. Then, the rule is generalized by setting each linguistic term to '1' with probability $P_{\#}$. The parameters of the classifier are initialized to: F=1, *exp*=0, *num*=1, and *cs* is set to the size of [C]. Finally, this rule is introduced in the population if there is not any rule in [C] with the same matching degree.

2.3 Parameters Update

At the end of each learning iteration, the parameters of all the classifiers that belong to the match set are updated according to their matching degree with the input example e of class c. First, the experience of each rule is incremented according to the current matching degree:

$$exp_{t+1}^k = exp_t^k + \mu_{A^k}(e) \tag{3}$$

Then, for each class j, we compute the sum of correct matchings cm_j of each classifier k:

$$cm_{j_{t+1}}^k = cm_{j_t}^k + m(k, j) \tag{4}$$

where

$$m(k, j) = \begin{cases} \mu_{A^k}(e) & \text{if j=c} \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

That is, we compute separately the sum of matching degrees of every rule with the examples of different classes. Next, the weight of each class is computed as:

$$w_{j_{t+1}}^k = \frac{cm_{j_{t+1}}^k}{exp_{t+1}^k} \tag{6}$$

For example, if a rule k only matches examples of class j, the weight w_j^k will be 1 and the remaining weights 0. Rules that match instances of more than one classes will have the corresponding weights ranging from 0 to 1. In all cases, the sum of all the weights is 1.

Then, the fitness is computed from the class weights with the aim of favoring classifiers that match instances of only one class. For this purpose, we compute the fitness as follows:

$$F_{t+1}^k = w_{max_{t+1}}^k - \sum_{j|j \neq max} w_{j_{t+1}}^k \tag{7}$$

The equation selects the weight w_{max}^k with maximum value and substract the values of the other weights. Note that this formula can result in classifiers with zero or negative fitness (for example, if the class weights are equal). Finally, the correct set size cs^k is calculated as the arithmetic average of the sizes of all the correct sets to which the classifier has belonged.

2.4 Discovery Component

Similarly to UCS, Fuzzy-UCS uses a *genetic algorithm* for discovering new promising rules. The GA is applied on a correct set if the average time since its last application on the classifiers that form this correct set is greater than θ_{GA}. In this case, two parents are selected from [C] with probability proportional to their fitness and the matching degree of their fitness. That is,

$$p_{sel}^k = \frac{(F^k)^\nu \cdot \mu_{A^k}(e)}{\sum_{i \in [C] | F^i \geq 0} (F^i)^\nu \cdot \mu_{A^k}(e)} \tag{8}$$

where $\nu > 0$ is a constant that fixes the pressure toward maximally accurate rules (in our experiments, we set ν=10). Rules with negative fitness are not considered for selection. The fitness of young classifiers is decreased since they receive a minimum number of updates. That is, if $exp_k < \theta_{exp}$, the fitness of the classifier k is multiplied by $1/\theta_{exp}$.

Next, the parents are crossed and mutated with probabilities P_χ and P_μ respectively. The consequent of the rule and the parameters of the offspring are initialized as in covering.

The crossover operator crosses the rule antecedent by two points selected randomly. This could result in classifiers containing variables with no linguistic terms, which would indicate that the rule is not applicable. If this is the case, we copy a linguistic term from one of the parents. Crossover does not modify the consequent of the rule.

The mutation operator randomly decides if a variable has to be mutated. If a variable is selected, three types of mutation can be applied: *expansion*, *contraction*, or *shift*. Expansion chooses a linguistic term not represented in the corresponding variable and adds it to this variable; thus, it can only be applied on variables that do not have all the linguistic terms. Contraction is the opposite process: is removes a linguistic term from one variable; so, it can only be applied if the variable has more than one linguistic term. Shift changes a linguistic term for its immediately inferior or superior.

Finally, the new offspring are inserted into the population. First, each offspring is checked for subsumption with its parents. If either of the parents is enough experienced ($exp > \theta_{sub}$), highly accurate ($F > F_0$), and more general than the offspring, its numerosity is incremented. If the offspring cannot be subsumed by any of its parents, the same process is used to find a subsumer in [C]. If there is no subsumer, the offspring is inserted in the population. A classifier is deleted if the population is full; in this case, each classifier is given a deletion probability of

$$P_{del}^k = \frac{d^k}{\sum_{\forall j \in [P]} d_j} \tag{9}$$

where

$$d_k = \begin{cases} cs^k \cdot num^k \cdot \frac{\overline{F}}{F^k} & \text{if } exp^k > \theta_{del} \text{ and } (F^k)^\nu < \delta \overline{F} \\ cs^k \cdot num^k & \text{otherwise} \end{cases} \tag{10}$$

where $\overline{F}$ is the average fitness of classifiers in [P], and δ and θ_{del} are two parameters set by the user ($0 < \delta < 1$ and $\theta_{del} > 0$). Thus, this method gives higher deletion probabilities to numerous classifiers that belong to large correct sets; moreover, it also penalizes experienced classifiers with low fitness.

2.5 Fuzzy-UCS in Test Mode

Fuzzy-UCS aims at obtaining a highly accurate rule set of minimum size. To obtain high accuracy, we need to define an effective reasoning method that infers the output class from the final population. To obtain a reduced rule set, some reduction strategies may be applied to remove classifiers that are not important for the reasoning. In the following, we discuss two reasoning approaches which lead to two different rule set reduction mechanisms.

Class Inference. In test mode, given a new unlabeled instance e, several rules can match (with different degrees) this instance, each having a certain fitness F^k. Thus, a reasoning process needs to be applied to decide the output. Here, we propose two fuzzy-inference approaches:

1. *Weighted average inference.* In this approach, all rules vote to infer the output. Each rule k emits a vote v_k for class j it advocates, where
$$v_k = F^k \cdot \mu_{A^k}(e) \tag{11}$$
The votes for each class j are added:
$$\forall j : vote_j = \sum_{k|c^k=j}^{N} v_k \tag{12}$$
and the most-voted class is returned as the output. Note that this strategy is analogous to the inference scheme of UCS.
2. *Action winner inference.* This approach proposes to select the rule k that maximizes $F^k \cdot \mu_{A^k}(e)$, and choose the class of the rule as output [20]. Thus, the knowledge of overlapping rules is not considered in this inference scheme.

Ruleset Reduction. At the end of the learning, the population is reduced to obtain a minimum set of rules with the same training accuracy as the original rule set. The reduction strategy depends on the type of inference used.

1. *Reduction based on weighted average.* Under the weighted average inference, the final population is reduced by removing all the rules that a) are not experienced enough ($exp < \theta_{exploit}$) or b) have zero or negative fitness.
2. *Reduction based on action winner.* If action winner inference is used, it is only necessary to maintain the rules that maximize the prediction vote for each training example. Thus, after training, this reduction scheme infers the output for each training example. The rule that maximizes the vote v_j for each example is copied to the final population.

Next section compares Fuzzy-UCS with the two inference and reduction mechanisms to UCS and other machine learning techniques. For notation, these schemes will be addressed as: weighted average inference (wavg), and action winner inference (awin).

3 Experimentation

In this section, we analyze the competence of Fuzzy-UCS in classification tasks. The main goal of data classification is to obtain highly accurate models that provide comprehensible explanations for human experts. For this purpose, we compare the performance and rule set interpretability of Fuzzy-UCS with respect to UCS and three other well-known machine learning techniques: the decision tree C4.5 [25], the support vector machine SMO [24], and the boosting algorithm based on a fuzzy representation Fuzzy LogitBoost [22]. In the following, we first detail the methodology followed in the comparison and then present the results obtained.

3.1 Methodology

Experimentation Problems. We selected a collection of twenty real-world problems. This collection includes problems with different characteristics (see Table 1) which may pose particular challenges to the different learning techniques. All these problems were obtained from the UCI repository [5], except for *tao*, which was chosen from a local repository [4].

Machine Learning Techniques Included in the Comparison. We compared Fuzzy-UCS with the two inference mechanisms to four machine learning techniques: UCS, C4.5, SMO, and Fuzzy LogitBoost. UCS [3] is the learning classifier system from which Fuzzy-UCS was derived; so, we want to analyze whether the fuzzy representation permits to achieve similar performance and improves the interpretability of the rule sets evolved by UCS. C4.5 [25] is a decision tree that enhances ID3 by introducing methods to deal with continuous variables and missing values. C4.5 is one of the most used learners in the realm of pattern classification, since it usually results in accurate tree-based models which are quite interpretable by human experts. SMO [24] is a widely-used implementation of *support vector machines* [31]; SMO implements the *Sequential Minimization Algorithm* to solve the dual problem. We included both C4.5 and SMO into the analysis to compare Fuzzy-UCS to two top-notch machine learning techniques. Fuzzy LogitBoost [22] is a boosting algorithm that iteratively invokes a genetic algorithm to extract simple fuzzy rules that are combined to decide the output of new examples. We selected Fuzzy LogitBoost to be in the comparison since it is a modern method which relies on statistics theory, and so, it is a good representative of fuzzy learners. Table 2 summarizes the main characteristics of the learners.

C4.5 and SMO were run using WEKA [35]. For Fuzzy LogitBoost, we used KEEL [2]. For UCS, we ran our own code. All the open source methods were

Table 1. Properties of the data sets. The columns describe: the identifier of the data set (Id.), the name of the data set (dataset), the number of instances (#Inst), the total number of features (#Fea), the number of real features (#Re), the number of integer features (#In), the number of nominal features (#No), the number of classes (#Cl), and the proportion of instances with missing values (%Miss)

Id.	**dataset**	**#Inst**	**#Fea**	**#Re**	**#In**	**#No**	**#Cl**	**%Miss**
ann	Annealing	898	38	6	0	32	5	0
aut	Automobile	205	25	15	0	10	6	22.4
bal	Balance	625	4	4	0	0	3	0
bpa	Bupa	345	6	6	0	0	2	0
cmc	Contraceptive method choice	1473	9	2	0	7	3	0
col	Horse colic	368	22	7	0	15	2	98.1
gls	Glass	214	9	9	0	0	6	0
h-c	Heart-c	303	13	6	0	7	2	2.3
h-s	Heart-s	270	13	13	0	0	2	0
irs	Iris	150	4	4	0	0	3	0
pim	Pima	768	8	8	0	0	2	0
son	Sonar	208	60	60	0	0	2	0
tao	Tao	1888	2	2	0	0	2	0
thy	Thyroid	215	5	5	0	0	3	0
veh	Vehicle	846	18	18	0	0	4	0
wbcd	Wisc. breast-cancer	699	9	0	9	0	2	2.3
wdbc	Wisc. diagnose breast-cancer	569	30	30	0	0	2	0
wne	Wine	178	13	13	0	0	3	0
wpbc	Wisc. prognostic breast-cancer	198	33	33	0	0	2	2
zoo	Zoo	101	17	0	1	16	7	0

Table 2. Summary of the main characteristics of the learners included in the comparison: C4.5, SMO, UCS, and Fuzzy LogitBoost (Bst)

	Paradigm	**Knowledge Rep. and Inference Method**
C4.5	Decision-tree induction	Decision-tree. *Inference*: class given by the corresponding leaf.
SMO	Neural networks (support vector machines)	Weights of the support vector machines. *Inference:* The class is determined by the decision function represented by the SVM.
UCS	Michigan-style GBML	Population of intervalar rules with a fitness value. *Inference:* The output is the most voted class among the matching classifiers.
Bst	Statistical Learning Theory and GBML	Population of linguistic fuzzy rules with a strength per class. *Inference:* The output is the most voted class among the matching classifiers.

configured with the parameters values recommended by default, with the following exceptions. The model for SMO was selected by choosing the kernel that maximized the global accuracy with respect to the other learners. That is, we ran SMO with polynomial kernels of order 1, 3, 5, and 10, and with a Gaussian kernel. Then, we ranked the results obtained with the three polynomial kernels and chose the model that maximized the average rank: SMO with polynomial kernels of order 3. We followed a similar strategy to select the maximum population size of Fuzzy LogitBoost, for which we did not find a set of recommended values in the literature. We tried population sizes of N={8,25,50,100} for all the data sets, and provide the results of N=50 since they permitted to achieve, in general, higher performance ratios than N=8 and N=25, and did not significantly differ from the results obtained with N=100.

UCS was configured with the following parameters (see [3,21] for notation details): $numIter$=100,000, N=6400, $acc_0 = 0.99$, ν=10, $\{\theta_{GA}, \theta_{del}, \theta_{sub}\}$=50, P_χ=0.8, P_μ=0.04, δ=0.1, $P_\#$=0.6. These are typical configuration parameters for UCS. No extra experimentation was made to improve the performance results, so that the reported results could be even improved with a further tuning of configuration parameters. Similar configuration parameter values were chosen for Fuzzy-UCS, that is: $numIter$=100,000, N=6400, $F_0 = 0.99$, $\nu = 10$, $\{\theta_{GA}, \theta_{del}, \theta_{sub}\} = 50$, $\theta_{exploit} = 10$, $P_\chi = 0.8$, $P_\mu = 0.04$, δ=0.1, and $P_\# = 0.6$. Both fuzzy based learners, i.e., Fuzzy-UCS and Fuzzy LogitBoost, used five linguistic terms per variable defined by triangular-shaped membership functions.

Comparison Metrics. The data models built by each learner were evaluated in terms of performance and interpretability. We measured the performance of the method with the test accuracy, i.e., the proportion of correct classification on previously unseen examples. To obtain reliable estimates of test accuracies, we ran the experiments on a ten-fold cross validation [29]. For the stochastic methods, the results provided correspond to the average of ten runs with different random seeds.

The comparison of the interpretability of the models is more complicated since the methods included in the comparison use different knowledge representations. Whilst UCS, Fuzzy-UCS, and Fuzzy LogitBoost use a rule-based representation, C4.5 builds a decision tree, and SMO represents the data model with the weights (ranging from 0 to 1) of a support vector machine. For this reason, we gathered some indicators of the model sizes, i.e., number of rules for the rule-based systems, number of leaves for the trees, and number of weights for the support vector machines. In the next section, we qualitatively discuss the advantages and disadvantages of the different representations.

Statistical Analysis. We followed the methodology pointed in [12] to statistically analyze the differences in performance among learners. As suggested by the author, we avoided to use any parametric statistical test since they require that the data satisfy several strong conditions. Instead, all the statistical analysis is based on non-parametric tests.

We first applied a multi-comparison statistical procedure to test whether all the learning algorithms performed the same on average. Specifically, we used the Friedman test [15,16], the non-parametric equivalent to the analysis of variance test ANOVA [14]. As indicated in [12], if the Friedman test rejects the hypothesis that all the learners perform the same on average, several post-hoc tests can be used to detect significant differences between groups of learners. As our aim was to compare Fuzzy-UCS to the other learners (that is, one control learner against the others) we used the non-parametric Bonferroni-Dunn test [13]. The Bonferroni-Dunn test defines that one learner performs significantly differently than a control learner if the corresponding average rank differs by, at least, a critical difference CD, which is computed as

$$CD = q_{\alpha}\sqrt{\frac{n_{\ell}(n_{\ell}+1)}{6n_{ds}}} \tag{13}$$

where n_{ℓ} is the number of learners in the comparison, n_{ds} is the number of data sets, and q_{α} is the critical value based on the Studentized range statistic [28]. We illustrate the results of this test by showing the group of learners that perform equivalently to the control learner.

The Bonferroni-Dunn test is said to be conservative, especially as the number of learners increases or the number of data sets decreases, so that it may not detect significant differences although they actually exist. Nonetheless, we use this test in the first stage of our analysis since it permits to detect groups of learners that truly perform differently from other learners. We latter apply pairwise comparisons to detect further significant differences between learners that belong to the same group, assuming the risk of increasing the error of rejecting null hypotheses when they are actually true. We used the non-parametric Wilcoxon signed-ranks test [33] for pairwise comparisons, and provide the approximative p-values computed as indicated in [28].

3.2 Results

Comparison of the Performance. Table 3 shows the average performance obtained with the six learners on the twenty real-world problems. The last two rows of the table supply the average rank and the position of each learner in the ranking. The procedure to calculate the ranks is the following. For each data set, we ranked the learning algorithms according to their performance; the method with the highest accuracy was the first in the ranking, while the learner with the poorest results holds the last position of the ranking. If a group of learners had the same accuracy, we assigned the average rank of the group to each of those learners.

The experimental results evidence the competitiveness of Fuzzy-UCS with respect to the other machine learning techniques, especially when all the rules are used in the inference process. That is, Fuzzy-UCS with weighted average inference is the third method of the ranking; it is only outperformed by SMO with polynomial kernels and UCS, the method from whom Fuzzy-UCS was inspired.

Table 3. Comparison of the accuracy rate of Fuzzy-UCS with weighted average (wavg) and action winner (awin) inference schemes to UCS, C4.5, SMO, and Fuzzy LogitBoost (LBoost). The two last rows of the table report the average rank of each learning algorithm (Rnk), and its position in the rank (Pos).

	UCS	**C4.5**	**SMO**	**LBoost**	**Fuzzy-UCS**	
					wavg	**awin**
ann	99.05	98.90	99.34	76.20	98.85	97.39
aut	77.41	80.94	78.09	32.63	74.42	67.42
bal	77.32	77.42	91.20	88.30	88.65	84.40
bpa	67.59	62.31	59.97	64.46	59.82	59.42
cmc	50.27	52.62	48.75	51.10	51.72	49.67
col	96.26	85.32	75.59	63.06	85.01	82.46
gls	70.04	66.15	66.15	68.18	60.65	57.21
h-c	79.72	78.45	78.59	62.09	84.39	82.62
h-s	74.63	79.26	78.89	59.33	81.33	80.78
irs	95.40	94.00	92.67	95.33	95.67	95.47
pim	74.61	74.23	76.70	71.84	74.88	74.11
son	76.49	71.07	85.52	53.38	80.78	73.71
tao	87.00	95.92	84.22	91.73	81.71	83.02
thy	95.13	94.91	88.91	97.08	88.18	89.49
veh	71.40	71.14	83.30	37.25	67.68	65.35
wbcd	96.28	94.99	96.42	94.12	96.01	95.73
wdbc	95.96	94.40	97.58	62.74	95.20	94.61
wne	96.13	93.89	97.75	85.02	94.12	94.86
wpbc	69.40	71.61	81.25	76.35	76.06	76.05
zoo	96.78	92.81	97.83	41.89	96.50	94.78
Rnk	***2.80***	***3.63***	***2.68***	***4.55***	***3.20***	***4.15***
Pos	***2***	***4***	***1***	***6***	***3***	***5***

These results indicate that, even though the granularity limitations that the linguistic fuzzy representation may impose compared to an interval-based representation, Fuzzy-UCS presents a similar performance than UCS. Besides, note that Fuzzy-UCS with weighted average achieves a better average rank than C4.5, one of the most used machine learning techniques. Similarly, Fuzzy-UCS with weighted average also outperforms Fuzzy LogitBoost, indicating that Fuzzy-UCS can evolve more accurate fuzzy rules than Fuzzy LogitBoost.

The average rank worsens when only the information of the best rules is used for inferring the class of new input instances. That is, Fuzzy-UCS with action winner inference holds the fifth position of the ranking. These results confirm the advantages of combining the information of all the fuzzy rules in the inference process [10]. However, the next section shows that Fuzzy-UCS with action winner results in much more reduced rule sets than Fuzzy-UCS with weighted average, which opens an accuracy-interpretability trade-off. Moreover, Fuzzy-UCS with action winner also outperforms the other fuzzy learner, Fuzzy LogitBoost.

We analyzed statistically the results to identify significant differences among learners. Friedman test for multiple comparisons rejected the null hypothesis

Table 4. Pairwise comparison of the performance of the six learners by means of a Wilcoxon signed-ranks test. The above diagonal contains the approximate p-values. The below diagonal shows a symbol $\oplus$ / $\ominus$ if the method in the row significantly outperforms/degrades the method in the column at a significance level of .05 and +/=/− if there is no significant difference and performs better/equal/worse.

	UCS	**C4.5**	**SMO**	**LogitBoost**	**wavg**	**awin**
UCS		.2043	.6542	.0072	.4330	.0674
C4.5	-		.4209	.0111	.7938	.3905
SMO	+	+		.0072	.1672	.0400
LogitBoost	$\ominus$	$\ominus$	$\ominus$		.0100	.0276
wavg	-	+	-	$\oplus$		.0032
awin	-	=	$\ominus$	$\oplus$	$\ominus$	

that the six learners performed the same on average with $p = 0.006$. To evaluate the differences between Fuzzy-UCS with the two inference schemes and the other learners, we applied the Bonferroni-Dunn test at $\alpha = 0.10$. Figure 3 places each method according to its average rank. Furthermore, it connects with a line the methods that perform equivalently to (1) Fuzzy-UCS with weighted average inference, and (2) Fuzzy-UCS with action winner inference. The statistical analysis indicates that Fuzzy-UCS with weighted average significantly outperforms Fuzzy LogitBoost. Besides, Fuzzy-UCS with action winner performs equivalently to all the methods except for SMO.

As the Bonferroni test is said to be conservative [28], so that it may not detect all the significant differences, we complemented the statistical study by compar-

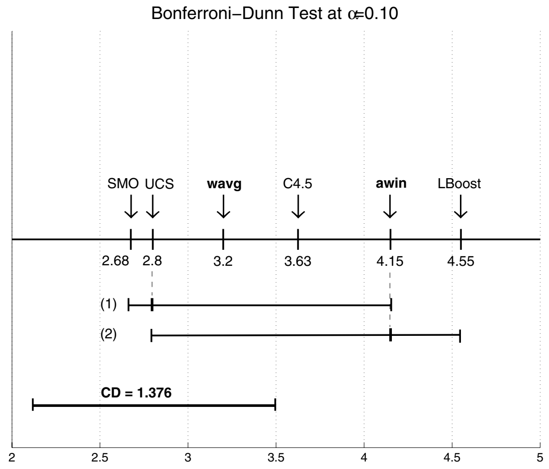


Fig. 3. Comparisons of one leaner against the others according to a Bonferroni-Dunn test at a significance level of 0.10. All the learners are compared to two different control groups: (1) Fuzzy-UCS with weighted average and (2) Fuzzy-UCS with action winner. The methods connected are those that perform equivalently to the control learner.

ing each pair of learners. Note that with this approach we are increasing the risk of rejecting hypothesis that are actually true [12]; nonetheless, we do not base the statistical analysis on pairwise comparisons, but use them to complement the conclusions drawn above. The above diagonal of Table 4 shows the approximate p-values of the pairwise comparison according to a Wilcoxon signed-ranks test. The below diagonal indicates with the symbols $\oplus$ and $\ominus$ that the method in the row significantly outperforms/degrades the performance obtained with the method in the column. Similarly, the symbols + and - are used to denote a non-significant improvement/degradation. The symbol = indicates that each method outperforms and degrades the other the same number of times.

The pairwise analysis confirms the conclusions pointed by the Bonferroni-Dunn test, and it detects additional significant differences among learners. The test supports that Fuzzy-UCS with weighted average inference does not degrade the performance of any other learner and significantly improves Fuzzy LogitBoost and Fuzzy-UCS with action winner inference. On the other hand, it indicates that Fuzzy-UCS with action winner inference outperforms Fuzzy LogitBoost; besides, it degrades the results obtained by SMO and Fuzzy-UCS with weighted average. Finally, note that Fuzzy-UCS with action winner inference is statistically equivalent to C4.5; that is, it outperforms and degrades the results of C4.5 in the same number of the data sets.

Comparison of the Interpretability. The analysis made in the previous section highlights that Fuzzy-UCS is able to evolve highly accurate models, especially when the weighted average inference is used. Herein, we analyze the readability of the models evolved by Fuzzy-UCS, and qualitatively compare them to the models built by the other learners. To evaluate the readability of the final model, we consider two aspects: (i) the intrinsic interpretability of the model, i.e., if the knowledge representation can be easily understood by human experts; and (ii) the size of the model.

First, we compare the type of rules evolved by Fuzzy-UCS to the other knowledge representations. Figure 4 depicts partial examples of the models created by each learner for the two-dimensional problem *tao*. Fuzzy-UCS evolves a set of linguistic fuzzy rules with a fitness value (see Fig. 4(a)). UCS creates a population of interval-based rules; the lower and upper bounds of the intervals can take any valid value for the attribute (see Fig. 4(b)). C4.5 builds a decision tree where each node establishes a decision over one variable (see Fig. 4(c)). SMO creates $\binom{n}{2}$ support vector machines (where n is the number of classes of the classification problem), each one consisting of a set of continuous weights ranging from 0 to 1 (see Fig. 4(d)). Fuzzy LogitBoost evolves a set of linguistic fuzzy rules (see Fig. 4(e)). These rules are similar to the ones of Fuzzy-UCS, with the following two exceptions. In Fuzzy-UCS, rule's variables can take a disjunction of linguistic terms; differently, in Fuzzy LogitBoost the variables are represented by a single linguistic term. Furthermore, the rules of Fuzzy-UCS predict a single class with a certain fitness, whilst the rules of Fuzzy LogitBoost maintain a weight for each class, and use these weights in the inference process.

if x is XL **then** blue **with** F=1.00
if x is XS **then** red **with** F=1.00
if x is {XS or S} **and** y is {XS or S} **then** red **with** F=0.87
...

(a) Fuzzy-UCS

if x is [-6.00, -0.81] **and** y is [-6.00, 0.40] **then** red **with** acc= 1.00
if x is [2.84, 6.00] **and** y is [-5.26, 4.91] **then** blue **with** acc =1.00
if x is [-6.00, -0.87] **and** y is [-6.00, 0.74] **then** red **with** acc =1.00
...

(b) UCS

```
x <= -2.75
|  x <= -3.25: red (308.0)
|  x > -3.25
|  |   y <= 1.75: red (55.0)
|  |   y > 1.75
|  |  |  x <= -3: red (11.0/1.0)
|  |  |  x > -3
|  |  |  |     y <= 4.25: blue (6.0)
|  |  |  |     y > 4.25: red (4.0)
                     ...
```

(c) C4.5

```
-     1.000     * <0.229 0.875 > * X]
-     0.298     * <0.708 0.437 > * X]
                      ...
```

(d) SMO

if x is L **and** y is L **then** blue **with** -5.42 **and** red **with** 0.0
if x is M **and** y is XS **then** blue **with** 2.21 **and** red **with** 0.0
if x is M **and** y is XL **then** blue **with** -2.25 **and** red **with** 0.0
...

(e) Fuzzy LogitBoost

Fig. 4. Examples of part of the models evolved by (a) Fuzzy-UCS, (b) UCS, (c) C4.5, (d) SMO, and (e) Fuzzy LogitBoost for the two-dimensional tao problem

This first analysis permits us to draw several observations concerning the readability of the representation itself. In the following, we distinguish between three types of learners depending on the inherent interpretability of their knowledge representation:

1. SMO presents the least readable knowledge representation, since the weights of the support vector machines provide poor information for human experts. Thus, SMO is not suitable when the interpretability of the models is important.
2. The trees created by C4.5 and the rules evolved by UCS can be easily interpreted by human experts given models of moderate size. However, note that both the decision nodes and the rules codify the decision over input variables with continuous numbers. This permits to precisely define the decision boundaries, but may hinder the interpretability of the models, especially when the number of nodes or rules increases.
3. Fuzzy-UCS and Fuzzy LogitBoost use a linguistic representation. As the input variables are represented by linguistic terms, individual rules can be read more easily than interval-based rules and decision trees. Nonetheless, linguistic terms imply a discretization of the feature space, which may result in a loss of precision to define the class boundaries.

Next, we analyze the sizes of the models evolved by the different learners. For this purpose, we used the following metrics. For rule-based systems, i.e., Fuzzy-UCS, UCS, and Fuzzy LogitBoost, we counted the number of rules evolved. For the tree-based system, i.e., C4.5, we reckoned the number of leaves, since each path from the root of the tree to one leaf can be regarded as a complex rule. Note that these measures are not directly comparable due to the differences in the knowledge representations. However, we use these metrics to qualitatively analyze the differences among learners.

Table 5 depicts the model sizes of the rule-based and tree-based systems. These results show that:

1. Fuzzy-UCS with weighted average inference evolves smaller rules sets than UCS in most of the problems; besides, the fuzzy rule representation is more readable. However, the still large amount of rules may hinder the interpretability of the final rule set.
2. Fuzzy-UCS with action winner inference creates a much smaller rule sets than UCS and Fuzzy-UCS with weighted average. These big differences permit to significantly improve the readability of the final rule set, at a cost of slightly decreasing the test performance as shown in the previous section.
3. Fuzzy LogitBoost creates rule sets of moderate size. Actually, the size of the rule sets is determined by a configuration parameter. In our experiments,

Table 5. Average sizes of the models built by UCS, C4.5, Fuzzy LogitBoost (LBoost), and Fuzzy-UCS with weighted average and action winner inference

	UCS	**C4.5**	**LBoost**	**Fuzzy-UCS**	
				wavg	**awin**
ann	4410	38	50	2769	75
aut	4064	44	50	3872	114
bal	1712	45	50	1212	114
bpa	2603	25	50	1440	73
cmc	3175	162	50	1881	430
col	3446	5	50	4135	154
gls	3013	24	50	2799	62
h-c	2893	29	50	3574	113
h-s	3499	17	50	3415	117
irs	634	5	50	480	18
pim	3225	19	50	2841	192
son	5999	14	50	3042	178
tao	609	36	50	111	19
thy	1283	8	50	1283	37
veh	4601	69	50	3732	332
wbcd	1799	12	50	3130	138
wdbc	5079	11	50	5412	276
wne	3413	5	50	3686	95
wpbc	5078	12	50	3772	156
zoo	1244	11	50	773	16

we set the maximum population size to 50 since it maximized the average performance rank of the algorithm. Smaller population sizes could be set for particular problems without loss of performance. However, we are interested in robust methods that do not present a high dependency on their configuration parameters. For this reason, we used the same parameters in all the runs, and did not search for the best configuration for each specific data set (in fact, we did not follow this approach for any learner).

The rule sets evolved by Fuzzy LogitBoost are slightly smaller than those evolved by Fuzzy UCS. However, note that the rules created by LogitBoost maintain a weight per each class (see Fig. 4(e)), whilst Fuzzy-UCS's rules only maintain a single fitness value (see Fig. 4(a)). Furthermore, Fuzzy-UCS evolves different number of rules depending on the intrinsic complexities of the domain, while Fuzzy LogitBoost needs to know beforehand the number of rules to be created.

4. C4.5 evolves trees of moderate size. The number of leaves is sligthly inferior than the number of rules of Fuzzy-UCS with action winner.

The whole analysis provided in this section showed the competitivity of Fuzzy-UCS with respect to the other machine learning techniques. In terms of performance, Fuzzy-UCS with weighted average inference was one of the best methods in the ranking. In terms of interpretability, we showed that Fuzzy-UCS with action winner inference resulted in rule sets much more reduced than those evolved by UCS—and Michigan-style Learning Classifier Systems in general—at the cost of slightly decreasing the test performance. This evidenced an accuracy-interpretability trade-off: the more information is used in the inference, the more accurate the prediction of the class of test examples is. The next section emphasizes the advantages of the online learning architecture and the many opportunities that Fuzzy-UCS offers, which are left for further work.

4 Conclusions and Further Work

In this paper, we proposed Fuzzy-UCS, a Michigan-style Learning Fuzzy-Classifier System for supervised learning. Fuzzy-UCS was derived from UCS with the aim of achieving more understandable models. For this reason, we replaced the interval-based rule representation of UCS with a linguistic fuzzy representation, and designed two inference mechanisms that offer different levels of rule set reduction.

We tested the system on a large collection of real-world problems, and compared the performance and interpretability of the models evolved by Fuzzy-UCS—with the two types of inference—to the models created by UCS, and three other machine learning techniques: the decision tree C4.5, the support vector machine SMO, and the statistic classifier based on a fuzzy representation Fuzzy LogitBoost. The results highlighted the competence of Fuzzy-UCS, which was able to evolve highly accurate and interpretable rule sets. Moreover, it showed an accuracy-interpretability trade-off. As expected, the more information is used for the inference process, the highest the accuracy of the models is; however, the readability of these models also worsens, since there are more rules in the final populations. In further work, deeper research will be conducted to understand more carefully the actual limitations imposed by the linguistic representation and the inference scheme used, and fuzzy approximative representations will be studied.

Besides its good behavior, Fuzzy-UCS presents two differential traits with respect to the other learners in the comparison, which lead to several opportunities concerning different challenges of data mining. Fuzzy-UCS, and in general Michigan-style LCSs, evolve the rule set incrementally. This differs from many learners which go several times through all the data set to create the classification model. Due to the online architecture, Fuzzy-UCS could be applied to two topics of increasing interest: (i) learning from large data sets [8], and, (ii) learning from data streams [1]. Furthermore, the use of fuzzy logic allows Fuzzy-UCS to be adapted for learning from vague and uncertain data, which is frequent in real-world classification problems [26,27]. The research on these three topics is left as further work.

Acknowledgements

The authors thank the support of *Enginyeria i Arquitectura La Salle*, Ramon Llull University, as well as the support of *Ministerio de Educación y Ciencia* under projects TIN2005-08386-C05-01 and TIN2005-08386-C05-04, and *Generalitat de Catalunya* under grants 2005FI-00252 and 2005SGR-00302.

References

1. Aggarwal, C.: Data Streams: Models and Algorithms. Springer, Heidelberg (2007)
2. Alcalá-Fdez, J., del Jesus, M.J., Garrell, J.M., Herrera, F., Herbás, C., Sánchez, L.: Proyecto KEEL: Desarrollo de una herramienta para el análisis e implementación de algoritmos de extracción de conocimiento evolutivos. In: Aguilar, J.S., Giráldez, R., Riquelme, J.C. (eds.) Tendencias de la Minería de Datos en España, Red Española de Minería de Datos y Aprendizage, pp. 413–424 (2004)
3. Bernadó-Mansilla, E., Garrell, J.M.: Accuracy-Based Learning Classifier Systems: Models, Analysis and Applications to Classification Tasks. Evolutionary Computation 11(3), 209–238 (2003)
4. Bernadó-Mansilla, E., Llorà, X., Garrell, J.M.: XCS and GALE: a comparative study of two learning classifier systems on data mining. In: Lanzi, P.L., Stolzmann, W., Wilson, S.W. (eds.) IWLCS 2001. LNCS (LNAI), vol. 2321, pp. 115–132. Springer, Heidelberg (2002)
5. Blake, C.L., Merz, C.J.: UCI Repository of ML Databases: University of California (1998), `http://www.ics.uc.edu/~mlearn/MLRepository.html`
6. Bonarini, A.: Evolutionary Learning of Fuzzy rules: competition and cooperation. In: Pedrycz, W. (ed.) Fuzzy Modelling: Paradigms and Practice, pp. 265–284. Kluwer Academic Press, Norwell (1996)
7. Bonarini, A., Trianni, V.: Learning fuzzy classifier systems for multi-agent coordination. Information Sciences: an International Journal 136(1-4), 215–239 (2001)

8. Bottou, L., Le Cun, Y.: On-line learning for very large data sets: Research Articles. Applied Stochastic Models in Business and Industry 21(2), 137–151 (2005)
9. Casillas, J., Carse, B., Bull, L.: Fuzzy-XCS: a michigan genetic fuzzy system. IEEE Transactions on Fuzzy Systems 15(4), 536–550 (2007)
10. Cordón, O., del Jesús, M.J., Herrera, F.: Analyzing the Reasoning Mechanisms in Fuzzy Rule Based Classification Systems. Mathware & Soft Computing 5, 321–332 (1998)
11. Cordón, O., Herrera, F., Hoffmann, F., Magdalena, L.: Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knowledge Bases. In: Advances in Fuzzy Systems—Applications and Theory, vol. 19, World Scientific, Singapore (2001)
12. Demšar, J.: Statistical Comparisons of Classifiers over Multiple Data Sets. Journal of Machine Learning Research 7, 1–30 (2006)
13. Dunn, O.J.: Multiple Comparisons among Means. Journal of the American Statistical Association 56, 52–64 (1961)
14. Fisher, R.A.: Statistical Methods and Scientific Inference, 2nd edn. Hafner Publishing Co, New York (1959)
15. Friedman, M.: The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance. Journal of the American Statistical Association 32, 675–701 (1937)
16. Friedman, M.: A Comparison of Alternative Tests of Significance for the Problem of m Rankings. Annals of Mathematical Statistics 11, 86–92 (1940)
17. Furuhashi, T., Nakaoka, K., Uchikawa, Y.: Suppression of excess fuzziness using multiple fuzzy classifier systems. In: Proceedings of the 3th IEEE International Conference on Fuzzy Systems, pp. 411–414. Morgan Kaufmann, San Francisco (1994)
18. Goldberg, D.E.: Genetic Algorithms in Search, Optimization & Machine Learning, 1st edn. Addison Wesley, Reading (1989)
19. Holland, J.H.: Adaptation in Natural and Artificial Systems. The University of Michigan Press (1975)
20. Ishibuchi, H., Nakashima, T., Murata, T.: Performance evaluation of fuzzy classifier systems for multidimensional pattern classification problems. IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics 29(5), 601–618 (1999)
21. Orriols-Puig, A., Bernadó-Mansilla, E.: A Further Look at UCS Classifier System. In: GECCO 2006: Genetic and Evolutionary Computation Conference workshop program, Seattle, W.A., USA, 08–12 July 2006. ACM Press, New York (2006)
22. Otero, J., Sánchez, L.: Induction of descriptive fuzzy classifiers with the logitboost algorithm. Soft Computing 10(9), 825–835 (2006)
23. Parodi, A., Bonelli, P.: A new approach to fuzzy classifier systems. In: Proceedings of the 5th International Conference on Genetic Algorithms, pp. 223–230. Morgan Kaufmann, San Francisco (1993)
24. Platt, J.: Fast Training of Support Vector Machines using Sequential Minimal Opt. In: Advances in Kernel Methods - Support Vector Lear. MIT Press, Cambridge (1998)
25. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, San Francisco (1995)
26. Sánchez, L., Couso, I.: Advocating the use of Imprecisely Observed Data in Genetic Fuzzy Systems. IEEE Transactions on Fuzzy Systems(forthcoming, 2007)

27. Sánchez, L., Couso, I., Casillas, J.: Modeling vague data with genetic fuzzy systems under a combination of crisp and imprecise criteria. In: Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Multicriteria Decision Making, pp. 346–353 (2007)
28. Sheskin, D.J.: Handbook of Parametric and Nonparametric Statistical Procedures. Chapman & Hall, Boca Raton (2000)
29. Dietterich, T.G.: Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. Neural Comp. 10(7), 1895–1924 (1998)
30. Valenzuela-Radón, M.: The Fuzzy Classifier System: A Classifier System for Continuously Varying Variables. In: 4th ICGA, pp. 346–353. Morgan Kaufmann, San Francisco (1991)
31. Vapnik, V.: The Nature of Statistical Learning Theory. Springer, Heidelberg (1995)
32. Velasco, J.: Genetic-based on-line learning for fuzzy process control. International Journal of Intelligent Systems 13, 891–903 (1998)
33. Wilcoxon, F.: Individual comparisons by ranking methods. Biometrics 1, 80–83 (1945)
34. Wilson, S.W.: Get Real! XCS with Continuous-Valued Inputs. In: Learning Classifier Systems. From Foundations to Applications. LNCS (LNAI), pp. 209–219. Springer, Berlin (2000)
35. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, 2nd edn. Morgan Kaufmann, San Francisco (2005)