FR3: A Fuzzy Rule Learner for Inducing Reliable Classifiers

Jens Hühn and Eyke Hüllermeier *

Department of Mathematics and Computer Scienes
Philipps-Universitt Marburg

This paper introduces a fuzzy rule-based classification method called FR3, which is short for Fuzzy Round Robin RIPPER. As the name suggests, FR3 builds upon the RIPPER algorithm, a state-of-the-art rule learner. More specifically, in the context of polychotomous classification, it uses a fuzzy extension of RIPPER as a base learner within a round robin scheme and, thus, can be seen as a fuzzy variant of the R3 learner that has recently been introduced in the literature. A key feature of FR3, in comparison with its non-fuzzy counterpart, is its ability to represent different facets of uncertainty involved in a classification decision in a more faithful way. FR3 thus provides the basis for implementing "reliable classifiers" that may abstain from a decision when not being sure enough, or at least indicate that a classification is not fully supported by the empirical evidence at hand. Besides, our experimental results show that FR3 outperforms R3 in terms of classification accuracy and, therefore, suggest that it produces predictions that are not only more reliable but also more accurate. The superb classification performance of FR3 is furthermore confirmed by comparing it to other state-of-the-art (fuzzy) rule learners.

1 Introduction

A close connection between classification learning, on the one side, and fuzzy preference modeling and decision making, on the other side, has recently been established by Hüllermeier and Brinker in [33]. The idea of their approach is to reduce a problem of polychotomous classification, involving m classes $\mathcal{L} = \{\lambda_1 \dots \lambda_m\}$, to a problem of decision making based on a fuzzy preference structure. Following a round robin scheme, their approach, called LVPC (Learning Valued Preferences for Classification), first trains an ensemble of binary models, one for every pair of classes. Then, given a query instance $\mathbf{x}$ with unknown class $\lambda(\mathbf{x})$, three fuzzy relations (in the form of $\{1 \dots m\} \times \{1 \dots m\} \to [0, 1]$ mappings) can be derived from the predictions of this ensemble. For every pair of labels (λ_i, λ_j), the corresponding entries in these relations express, respectively, a degree of

- **preference:** the degree to which the label λ_i is (strictly) preferred to λ_j as a classification for $\mathbf{x}$ (and vice versa);
- **conflict:** the degree to which λ_i and λ_j are in conflict with each other (as both of them are supported simultaneously as potential classifications);
- **ignorance:** the degree of ignorance reflecting to what extent neither λ_i nor λ_j is supported as a classification.

A final classification, or any other type of decision (e.g., to abstain or to gather additional information), can then be made on the basis of these relations.

A key feature of this approach is its ability to represent ignorance in a faithful way. In fact, even though many machine learning methods are able to reflect conflict in one way or the other, for example in terms

*(e-mail: {huehnj, eyke} @informatik.uni-marburg.de)

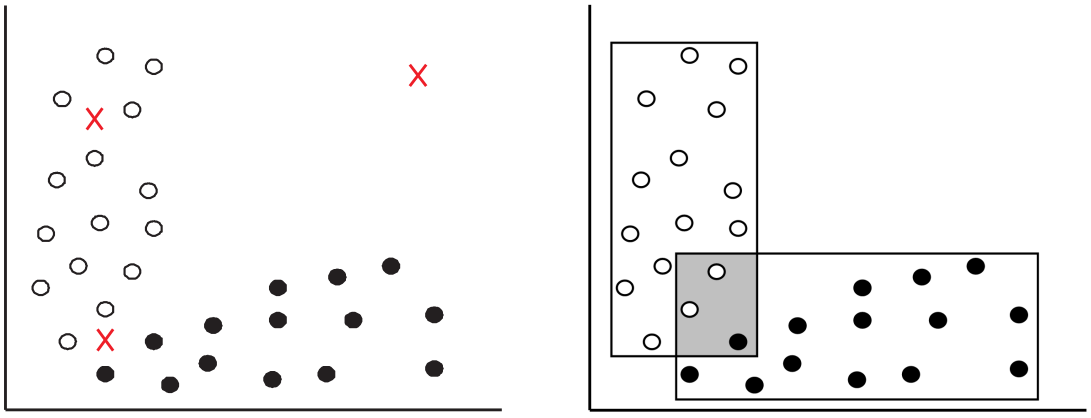
of probability distributions, the same is not true for ignorance. To illustrate the meaning of conflict and ignorance in the context of classification, consider the simple scenario shown in Figure 1a: Given observations from two classes, `black` and `white`, three new instances marked by a cross need to be classified. Obviously, given the current observations, the upper left instance can quite safely be classified as `white`. The case of the lower left instance, however, involves a high level of conflict, since both classes, `black` and `white`, appear plausible. The third situation is an example of ignorance: The upper right instance is located in a region of the instance space in which no observations have been made so far. Consequently, there is neither evidence in favor of class `black` nor in favor of class `white`.

It was already mentioned in [33] that rule-based classifiers are, in principle, ideally suited for implementing the pairwise models needed in LVPC. The main reason for this suitability is that, in contrast to standard discriminative classification methods (such as linear discriminant functions), rule-based models are able to represent conflict and, more importantly, ignorance in a natural way: A situation of conflict occurs if an instance $\mathbf{x}$ is simultaneously covered by two (or more) conflicting rules, while a situation of ignorance occurs if it is not covered by any rule; see Figure 1b.

In this regard, however, conventional rule-based classifiers can be criticized for at least two reasons: First, many approaches induce proper rules only for one class, typically the minority class, and add a default rule that predicts the other class in case no other rule applies. Thus, ignorance is eliminated in an artificial and arguably questionable way. In fact, note that this approach may come along with a high level of *extrapolation*, since the default class can be predicted in regions where it has never been observed before.

Second, since conventional (non-fuzzy) rules have "sharp boundaries", they produce an abrupt transition between support of a class and ignorance which is not very natural. Intuitively, the farther away an instance is located from the core of the closest rule, the higher the degree of ignorance should be. Or, stated differently, the support provided by a rule should decrease from "full" (inside the core) to "zero" in a gradual instead of an abrupt way.

To address these two issues, we propose to use fuzzy rules instead of conventional rules. More specifically, we develop a fuzzy extension of RIPPER [12], a state-of-the-art rule induction algorithm that produces accurate models in an efficient way. By using the fuzzy instead of the original version of RIPPER as a base learner within the round robin (all-pairs) decomposition scheme, we extend the R3 method proposed by Fürnkranz in [24]. Experimentally, it will be shown that our approach, called FR3, is not only able to reflect conflict and ignorance of a classification in a faithful way, but also outperforms R3 in terms of predictive accuracy. The superb classification performance of FR3 is furthermore confirmed by comparing it to the C4.5 decision tree learner [52] as well as a grid-based [8, 9] and an evolutionary fuzzy rule learner [27, 28].



(a) Observations from two classes (points) and new query instances (crosses).

(b) Regions of conflict (gray area) and ignorance (area not covered by any rule) in case of a rule-based model (rules indicated as rectangles).

Figure 1: Exemplary classification scenarios.

2 Outline of RIPPER

RIPPER was introduced in [12] as a successor of the IREP algorithm for rule induction [26]. Even though the key principles remained the same, RIPPER improves IREP in many details and is also able to cope with multi-class problems.

Consider a polychotomous classification problem involving m classes $\mathbb{L} \stackrel{\text{df}}{=} \{\lambda_1 \dots \lambda_m\}$. Suppose instances to be represented in terms of attributes A_i, $i = 1 \dots n$, which are either numerical (real-valued) or nominal, and let $\mathbb{D}_i$ denote the corresponding domains. Thus, an instance is represented as an n-dimensional attribute vector

$$\mathbf{x} = (x_1 \dots x_n) \in \mathbb{D} \stackrel{\text{df}}{=} \mathbb{D}_1 \times \dots \times \mathbb{D}_n.$$

A single RIPPER rule is of the form $r = \langle r_A \,|\, r_C \rangle$, consisting of an antecedent part r_A and a consequent part r_C. The antecedent part r_A is a conjunction of predicates (selectors) which are of the form $(A_i = v)$ for nominal and $(A_i \,\theta\, v)$ for numerical attributes, where $\theta \in \{\leq, =, \geq\}$ and $v \in \mathbb{D}_i$. The consequent part r_C is a class assignment of the form $(\text{class} = \lambda)$, where $\lambda \in \mathbb{L}$. A rule $r = \langle r_A \,|\, r_C \rangle$ is said to *cover* an instance $\mathbf{x} = (x_1 \dots x_n)$ if the attribute values x_i satisfy all the predicates in r_A.

RIPPER learns such rules in a greedy manner, following a separate-and-conquer strategy [23]. Prior to the learning process, the training data is sorted by class labels in ascending order according to the corresponding class frequencies. Rules are then learned for the first $m - 1$ classes, starting with the smallest one. Once a rule has been created, the instances covered by that rule are removed from the training data, and this is repeated until no instances from the target class are left. The algorithm then proceeds with the next class. Finally, when RIPPER finds no more rules to learn, a default rule (with empty antecedent) is added for the last (and hence most frequent) class.

Rules for single classes are learned until either all positive instances are covered or the last rule r that has been added was "too complicated". The latter property is implemented in terms of the total description length [53]: The stopping condition is fulfilled if the description length of r is at least d bits longer than the shortest description length encountered so far; Cohen suggests choosing $d = 64$.[1]

2.1 Learning Individual Rules

Each individual rule is learned in two steps. The training data, which has not yet been covered by any rule, is therefore split into a *growing* and a *pruning* set. In the first step, the rule will be specialized by adding antecedents which were learned using the growing set. Afterward, the rule will be generalized by removing antecedents using the pruning set.

Rule growing A new rule is learned on the growing data, using a propositional version of the FOIL algorithm [51, 54]. It starts with an empty conjunction and adds selectors until the rule covers no more negative instances, i.e., instances not belonging to the target class. The next selector to be added is chosen so as to maximize FOIL's information gain criterion (IG), which is a measure of improvement of the rule in comparison with the default rule for the target class:

$$\text{IG}_r \stackrel{\text{df}}{=} p_r \times \left(\log_2 \left(\frac{p_r}{p_r + n_r} \right) - \log_2 \left(\frac{p}{p + n} \right) \right),$$

where p_r and n_r denote, respectively, the number of positive and negative instances covered by the rule; likewise, p and n denote the number of positive and negative instances covered by the default rule.

Rule pruning The above procedure typically produces rules that overfit the training data. To remedy this effect, a rule is simplified so as to maximize its performance on the pruning data. For the pruning procedure the antecedents are considered in the order in which they were learned, and pruning actually means finding a position at which that list of antecedents is cut. The criterion to find that position is the rule-value metric:

$$V(r) \stackrel{\text{df}}{=} \frac{p_r - n_r}{p_r + n_r}$$

Therewith all antecedents that were learned after the antecedent which maximizes $V(r)$, will be pruned. Shorter rules are preferred in the case of a tie.

[1] Essentially, the description length of a rule depends on the number selectors in its antecedent part; see [52] for more details.

2.2 Rule Optimization

The rule set RS produced by the learning algorithm outlined so far, called IREP*, is taken as a starting point for a subsequent optimization process. This process re-examines the rules $r_i \in RS$ in the order in which they were learned. For each r_i, two alternative rules r_i' and r_i'' are created. The *replacement rule* r_i' is an empty rule, which is grown and pruned in a way that minimizes the error of the modified rule set $(RS \cup \{r_i'\}) \setminus \{r_i\}$. The *revision rule* r_i'' is created in the same way, except that it starts from r_i instead of the empty rule. To decide which version of r_i to retain, the MDL (Minimum Description Length [52]) criterion is used. Afterward, the remaining positives are covered using the IREP* algorithm.

The RIPPERk algorithm iterates the optimization of the rule set and the subsequent covering of the remaining positive examples with IREP* k times, hence the name RIPPER (Repeated Incremental Pruning to Produce Error Reduction).

2.3 Round Robin RIPPER

Round robin learning *aka* all-pairs or all-versus-all learning is a special decomposition technique that transforms a multi-class classification problem involving $m > 2$ classes $\mathbb{L} = \{\lambda_1 \dots \lambda_m\}$ into a number of *binary* problems. To this end, a separate model (base learner) $\mathcal{M}_{i,j}$ is trained for each *pair* of labels (λ_i, λ_j), $1 \leq i < j \leq m$. $\mathcal{M}_{i,j}$ is intended to separate the objects with label λ_i from those having label λ_j. If $(\mathbf{x}, \lambda_a) \in \mathbb{D} \times \mathbb{L}$ is an original training example (revealing that instance $\mathbf{x}$ has label λ_a), then $\mathbf{x}$ is considered as a *positive* example for all learners $\mathcal{M}_{a,j}$, $j > a$, and as a *negative* example for the learners $\mathcal{M}_{i,a}$, $i < a$.

At classification time, a query $\mathbf{x}$ is submitted to all $m(m-1)/2$ learners, and each prediction $\mathcal{M}_{i,j}(\mathbf{x})$ is typically interpreted as a vote for a label. Assuming models in the form of $[0,1]$-valued (scoring) classifiers, an output close to 1 indicates support of λ_i, whereas an output close to 0 is counted as evidence in favor of λ_j. The simplest classification strategy, then, is to predict the class label with the highest score in terms of the sum of (weighted) votes:

$$s_i \stackrel{\text{df}}{=} \sum_{1 \leq j \neq i \leq m} s_{i,j}, \tag{1}$$

where $s_{i,j} = \mathcal{M}_{i,j}(\mathbf{x})$ for $i < j$ and $s_{i,j} = 1 - \mathcal{M}_{j,i}(\mathbf{x})$ for $j < i$.

Even though the main purpose of decomposition techniques is to enable the application of methods that are inherently limited to binary classification, such as support vector machines, to polychotomous problems, round robin learning can be interesting even in the case where the models $\mathcal{M}$ can, in principle, handle multi-class problems in a direct way. The main reason is that the binary problems are often much simpler than the original m-class problem, so that models induced from data become more accurate and more stable. In particular, for the case of RIPPER, Fürnkranz [24, 25] showed that a Round Robin RIPPER (R3), i.e., an all-pairs classifier with RIPPER as a (binary) base learner, outperforms the original multi-class RIPPER.

Apart from that, the all-pairs decomposition technique is essential for the LVPC approach proposed in [33], namely for producing the (binary) relations that constitute a fuzzy preference structure (cf. Section 3.5).

3 Fuzzy Round Robin RIPPER

In this section, we introduce the Fuzzy Round Robin RIPPER (FR3) approach. This is done in two steps: First, we propose a fuzzy version of the basic RIPPER, called FRIPPER. In a second step, FRIPPER is then integrated as a base learner in a round robin learning scheme. FRIPPER modifies the original RIPPER algorithm in several ways, as will be detailed in the following subsections; here, we focus on the two-class case. The multi-class case, in connection with round robin learning, will then be addressed in the final subsection.

3.1 Learning Rules for Both Classes

A first modification of RIPPER concerns its use of default rules. As mentioned previously, using one class as a default prediction is disadvantageous with regard to reliable classification and, in particular, hinders a faithful representation of ignorance. Besides, this strategy comes along with a systematic bias in favor of those classes chosen as a default, namely the large ones, which also causes problems in round robin learning. This is why Fürnkranz, in his R3 algorithm, modifies RIPPER as follows: Each pairwise model actually

consists of two classifiers which take, respectively, the first and the second class as a default; the model output, then, is the average of the two predictions.

To represent ignorance, a classifier must be able to abstain, that is, to refrain from supporting any class. To achieve this, we also train two classifiers for every pairwise model. However, instead of averaging the two classifiers, we combine them by merging the respective proper rules, i.e., the non-default rules. To realize the difference, consider an instance not covered by any (proper) rule. While the support for both classes is 0 in our approach, it is, respectively, 1/2 when averaging the models (as the instance is covered by both default rules).[2]

The RIPPER algorithm can be divided into the building and the optimization phase. The rule building is done via the IREP* algorithm, which essentially consists of a propositional FOIL algorithm, the pruning strategy (cf. Section 2.1) and the stopping conditions. Interestingly, we found that the pruning strategies in IREP* have a negative influence on the performance of FR3. We therefore omitted the pruning step and instead learned the initial rule set on the whole training data directly. To explain this finding, note that, without pruning, IREP* produces more specific rules that better fit our general strategy to extrapolate in a "cautious" way. Moreover, small rules provide a better starting point for our fuzzification procedure, to be detailed in Section 3.3, in which rules can be made more general but not more specific.

In the optimization phase, the pruning was retained, as its deactivation was not beneficial. This is in agreement with the goal to minimize the MDL. The coverage of the remaining positive instances, which is again accomplished with IREP*, also benefited from omitting the pruning, just like IREP* in the building phase.

The new algorithm still applies pruning when it comes to creating the replacement and the revision rule. Here, the original pruning strategy is applied, except in case the pruning strategy tries to remove all antecedents from a rule, thereby generating a default rule. In this case, the pruning will be aborted, and the unpruned rule will be used for the MDL comparison in the optimization phase.

3.2 Representation of Fuzzy Rules

A selector constraining a numerical attribute A_i (with domain $\mathbb{D}_i = \mathbb{R}$) in a RIPPER rule can obviously be expressed in the form $(A_i \in I)$, where $I \subseteq \mathbb{R}$ is an interval: $I = (-\infty, v]$ if the rule contains a selector $(A_i \leq v)$, $I = [u, \infty)$ if it contains a selector $(A_i \geq u)$, and $I = [u, v]$ if it contains both (in the last case, two selectors are combined).

Essentially, a fuzzy rule is obtained through replacing intervals by fuzzy intervals, namely fuzzy sets with trapezoidal membership function. A fuzzy interval of that kind is specified by four parameters and will be written $I^F = (\phi^{s,L}, \phi^{c,L}, \phi^{c,U}, \phi^{s,U})$:

$$I^F(v) \stackrel{\text{df}}{=} \begin{cases} 1 & \phi^{c,L} \leq v \leq \phi^{c,U} \\ \frac{v-\phi^{s,L}}{\phi^{c,L}-\phi^{s,L}} & \phi^{s,L} < v < \phi^{c,L} \\ \frac{\phi^{s,U}-v}{\phi^{s,U}-\phi^{c,U}} & \phi^{c,U} < v < \phi^{s,U} \\ 0 & \text{else} \end{cases}$$

$\phi^{c,L}$ and $\phi^{c,U}$ are, respectively, the lower and upper bound of the core of the fuzzy set; likewise, $\phi^{s,L}$ and $\phi^{s,U}$ are, respectively, the lower and upper bound of the support. Note that, as in the non-fuzzy case, a fuzzy interval can be open to one side ($\phi^{s,L} = \phi^{c,L} = -\infty$ or $\phi^{c,U} = \phi^{s,U} = \infty$.) In fact, as will be seen later on, the fuzzy antecendents successively learned by FRIPPER are fuzzy half-intervals of exactly that kind.

A fuzzy selector $(A_i \in I_i^F)$ *covers* an instance $\mathbf{x} = (x_1 \ldots x_n)$ to the degree $I_i^F(x_i)$. A fuzzy rule r^F involving k selectors $(A_i \in I_i^F)$, $i = 1 \ldots k$, covers $\mathbf{x}$ to the degree $\mu_{r^F}(\mathbf{x}) = \min_{i=1\ldots k} I_i^F(x_i)$.

3.3 Rule Fuzzification

To obtain fuzzy rules, the idea is to fuzzify the final rules from our modified RIPPER algorithm. More specifically, using the training set $\mathbb{D}_\text{T} \subseteq \mathbb{D}$ for evaluating candidates, the idea is to search for the best fuzzy extension of each rule, where a fuzzy extension is understood as a rule of the same structure, but with intervals replaced by fuzzy intervals. Taking the intervals I_i of the original rules as the cores $[\phi_i^{c,L}, \phi_i^{c,U}]$

[2] Actually, our implementation works with only one classifier per model that contains learned rules for both classes. The important point here is that both R3 and our approach learn two rule sets for each pair of classes.

of the sought fuzzy intervals I_i^F, the problem is to find optimal bounds for the respective supports, i.e., to determine $\phi_i^{s,L}$ and $\phi_i^{s,U}$.

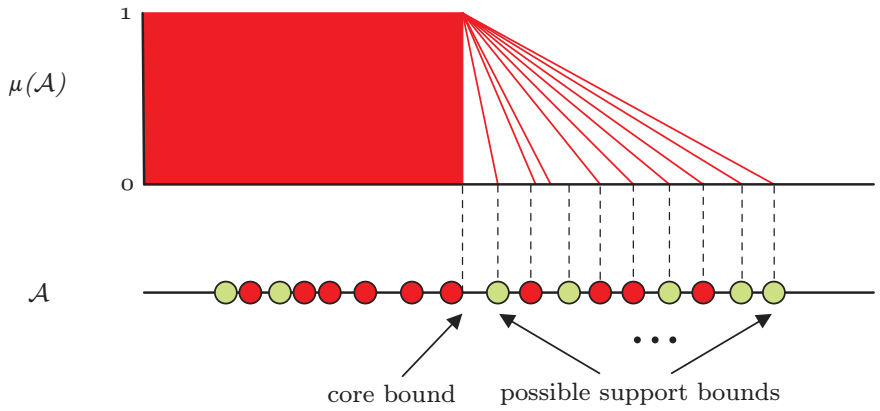


Figure 2: Examination of possible support bounds given a crisp antecedent.

For the fuzzification of a single antecedent ($A_i \in I_i$) it is important to consider only the relevant training data $D_{\mathrm{T}}^{(i)}$, i.e., to ignore those instances that are excluded by any other antecedent ($A_j \in I_j^F$), $j \neq i$:

$$\mathbb{D}_{\mathrm{T}}^{(i)} = \left\{\mathbf{x} \in \mathbb{D}_{\mathrm{T}} \mid I_j^F(x_j) > 0 \text{ for all } j \neq i\right\} \subseteq \mathbb{D}_{\mathrm{T}} \tag{2}$$

We partition $\mathbb{D}_{\mathrm{T}}^{(i)}$ into the subset of positive instances, $D_{\mathrm{T}^+}^{(i)}$, and negative instances, $D_{\mathrm{T}^-}^{(i)}$. To measure the quality of a fuzzification, the rule purity will be used:

$$\mathrm{pur} = \frac{p_i}{p_i + n_i}, \tag{3}$$

$$p_i \stackrel{\mathrm{df}}{=} \sum_{\mathbf{x} \in D_{\mathrm{T}^+}^{(i)}} \mu_{A_i}(\mathbf{x})$$

$$n_i \stackrel{\mathrm{df}}{=} \sum_{\mathbf{x} \in D_{\mathrm{T}^-}^{(i)}} \mu_{A_i}(\mathbf{x})$$

Rules are fuzzified in a greedy way, as shown by Algorithm 1. In each iteration, a fuzzification is computed for every antecedent, namely the best fuzzification in terms of (3). This is done by testing all values

$$\{x_i \mid \mathbf{x} = (x_1 \ldots x_k) \in D_{\mathrm{T}}^{(i)},\, x_i < \phi_i^{c,L}\}$$

as candidates for $\phi_i^{s,L}$ and, likewise, all values

$$\{x_i \mid \mathbf{x} = (x_1 \ldots x_k) \in D_{\mathrm{T}}^{(i)},\, x_i > \phi_i^{c,U}\}$$

as candidates for $\phi_i^{s,U}$ (see Figure 2). Ties are broken in favor of larger fuzzy sets, that is, larger distances from the core.

The fuzzification is then realized for the antecedent with the largest purity. This is repeated until all antecedent have been fuzzified. It is important to mention that there exists a trivial fuzzification which is always found, namely the one that sets the support bound to the first instance behind the core bound. Even though this fuzzification does not change the purity on the training data, it is meaningful when it comes to classifying new instances.

Note that the fuzzification of a single antecedent may change the relevant training data (2), which is hence recomputed in each iteration. In fact, each fuzzification may increase the number of covered instances, which in turn may also influence the rule purity. Furthermore, note that, after the complete antecedent part of a rule has been fuzzified, the whole procedure could in principle be repeated until convergence is

Algorithm 1 The antecedent fuzzification algorithm for a rule r

Let A be the set of numeric antecedents of r
while $A \neq \emptyset$ **do**
  $a_{\max} \leftarrow$ null {$a_{\max}$ denotes the antecedent with the highest purity}
  $\text{pur}_{\max} \leftarrow 0$ {$\text{pur}_{\max}$ is the highest purity value, so far}
  for $i \leftarrow 1$ to size(A) **do**
    compute the best fuzzification of $A[i]$ in terms of purity
    $\text{pur}_{A[i]} \leftarrow$ be the purity of this best fuzzification
    if $\text{pur}_{A[i]} > \text{pur}_{\max}$ **then**
      $\text{pur}_{\max} \leftarrow \text{pur}_{A[i]}$
      $a_{\max} \leftarrow A[i]$
    end if
  end for
  $A \leftarrow A \setminus a_{\max}$
  Update r with $a_{\max}$
end while

achieved (convergence is guaranteed, as purity can only increase in each iteration). We did not implement this option, however, as we observed that, except for very rare cases, convergence is already achieved after the first iteration.

To analyze the complexity of the above fuzzification procedure, note that, in each iteration, at most $|\mathbb{D}_T|$ instances (support bounds) are checked for every candidate attribute. Since the total number of iterations is bounded by the number of attributes, n, the overall complexity is $O(|\mathbb{D}|n^2)$.

3.3.1 Bounding Fuzzy Rules

Some fuzzy intervals may still be open to one side, which means that the corresponding rule has unbounded support. As this is not in agreement with our "cautious" extrapolation strategy, we finally close such intervals: If $\phi_j^{c,L} = -\infty$, this core bound is set to

$$\phi_j^{c,L} \;=\; \min\{x_j \mid \mathbf{x} = (x_1 \ldots x_k) \in D_{\mathrm{T}_+},\, \mu_{I_j^F}(\mathbf{x}) > 0\},$$

where D_{T_+} is the subset of positive instances in D_{T}. Moreover, the support bound $\phi_j^{s,L}$ is set to the minimal value in $\mathbb{D}_j$, the domain of attribute A_j.[3] This way, the core of the rule is restricted to the region in which positive examples have indeed been observed, while the support decreases as a linear function of the distance from this region. Analogous modifications are made in the case where $\phi_j^{c,U} = \infty$.

3.4 Classifier Output

Suppose that fuzzy rules $r_1^0 \ldots r_k^0$ and $r_1^1 \ldots r_\ell^1$ have been learned, respectively, for classes λ_0 and λ_1. For a new query instance $\mathbf{x}$, the supports of these classes are then given, respectively, by

$$\begin{array}{lll} s_0 & \stackrel{\text{df}}{=} & \max_{i=1\ldots k} \left(\mu_{r_i^0}(\mathbf{x}) \cdot \mathrm{CF}(r_i^0)\right) \\ s_1 & \stackrel{\text{df}}{=} & \max_{j=1\ldots \ell} \left(\mu_{r_j^1}(\mathbf{x}) \cdot \mathrm{CF}(r_j^1)\right) \;\;, \end{array} \tag{4}$$

where

$$\mathrm{CF}(r) = \frac{\sum_{\mathbf{x} \in \mathbb{D}_T^+ :\, \lambda(\mathbf{x}) = r_C} \mu(\mathbf{x})}{\sum_{\mathbf{x} \in \mathbb{D}_T} \mu(\mathbf{x})} \tag{5}$$

[3] More specifically, to avoid 0-memberships, we go beyond this point, as a rule of thumb by 50% of the width of $\mathbb{D}_j$.

is a measure of the confidence or validity of a rule.[4] From these two support degrees, the following values are derived, which constitute the output of the FRIPPER algorithm (in the two-class case):

$$\begin{array}{rcl} P(\lambda_0,\lambda_1) & = & s_0 - \min\{s_0,s_1\} \\ P(\lambda_1,\lambda_0) & = & s_1 - \min\{s_0,s_1\} \\ C(\lambda_0,\lambda_1) & = & \min\{s_0,s_1\} \\ I(\lambda_0,\lambda_1) & = & 1 - \max\{s_0,s_1\} \end{array} \tag{6}$$

$C(\lambda_0,\lambda_1)$ is the degree of *conflict*, namely the degree to which both classes are supported. Likewise, $I(\lambda_0,\lambda_1)$ is the degree of *ignorance*, namely the degree to which none of the classes is supported. Finally, $P(\lambda_0,\lambda_1)$ and $P(\lambda_1,\lambda_0)$ denote, respectively, the strict preference for λ_0 and λ_1. Note that at least one of these two degrees is zero, and that $P(\lambda_0,\lambda_1)+P(\lambda_1,\lambda_0)+C(\lambda_0,\lambda_1)+I(\lambda_0,\lambda_1) \equiv 1$. In passing, we also remark that (6) is actually a standard decomposition scheme, which is used in fuzzy preference modeling [20] to decompose a weak preference relation (here given by the support degrees s_0, s_1) into three parts: strict preference, indifference (which here corresponds to conflict), and indistinguishablity (here ignorance).

3.5 Round Robin Learning

Given a set of classes $\mathbb{L} = \{\lambda_1 \dots \lambda_m\}$, the FRIPPER algorithm as outlined above can be applied to each pair of labels $(\lambda_k, \lambda_\ell)$, thereby producing an ensemble of models $\mathcal{M}_{k\ell}$, $1 \leq k < \ell \leq m$. A query instance $\mathbf{x} \in \mathbb{D}$ is then submitted to each model. As explained in Section 3.4, the output of model $\mathcal{M}_{k\ell}$ is a quadruple $\mathcal{M}_{k\ell}(\mathbf{x}) = (p_{k\ell}, p_{\ell k}, c_{k\ell}, i_{k\ell})$, where $p_{k\ell}$ is the preference for λ_k in comparison with λ_ℓ, $p_{\ell k}$ the preference for λ_ℓ, $c_{k\ell}$ the corresponding degree of conflict, and $i_{k\ell}$ the degree of ignorance.

Thus, three relations (P, C, I) are obtained, a strict preference relation $P = (p_{k\ell})$, a conflict relation $C = (c_{k\ell})$, and an ignorance relation $I = (i_{k\ell})$; note that C and I are symmetric, so the entries in the relations are well-defined for all $1 \leq k \neq \ell \leq m$. These relations provide the basis for sophisticated classification and decision policies. For example, in the standard scenario where a single prediction is sought, the following classification rule could be used:

$$\lambda^* = \arg\max_{\lambda_k \in \mathbb{L}} \sum_{1 \leq \ell \neq k \leq m} p_{k\ell} + \frac{1}{2} \cdot c_{k\ell} + \frac{N_k}{N_k + N_\ell} \cdot i_{k\ell}, \tag{7}$$

where N_k is the number of examples from class λ_k in the training data (and hence an unbiased estimate of the class probability). This decision rule, that turned out to perform well in practice (cf. Section 4), evaluates each candidate label in terms of the sum of strict preferences over all other labels, distributes the corresponding degrees of conflict in a uniform way and the degrees of ignorance in proportion to the size of the classes (in other words, prior probabilities are used in the case of no further information).

Going beyond the conventional classification setting, a preference structure (P, C, I) can be especially useful in generalized settings in which, for example, more than one class can be predicted in cases of conflict, or a classification decision can be refused in cases of ignorance (cf. Section 4.4).

4 Experimental Results

To analyze the performance of our FR3 approach, we conducted several experimental studies under the WEKA 3.5.5 framework [61]. As a starting point, we used the RIPPER implementation of WEKA ("JRip"), both for re-implementing Fürnkranz's R3 and our FR3.

4.1 Classification Accuracy

In a first study, we compared RIPPER, R3, and FR3 with respect to classification accuracy. The minimum number of covered instances per antecedent was set to 2, and for the number of folds and the number of optimizations in RIPPER we used, respectively, values 3 and 2 (which is the default setting in WEKA and leads to RIPPER2). R3 was used with the weighted voting variant, i.e., the vote of a pairwise classifiers is weighted in terms of rule purity; in [25], this method was found to outperform binary (0/1) voting and Laplace weighted voting.

[4] See [35, 37] for an interesting discussion of the effect of rule weighing and advantages thereof.

Additionally, we also included the C4.5 decision tree learner [52] as a well-known benchmark classifier and, moreover, added two fuzzy rule-based classifiers from the KEEL suite [1]: The CHI algorithm is based on [8, 9] and uses rule weighing as proposed in [37].[5] The SLAVE algorithm makes use of genetic algorithms to learn a fuzzy classifier [27, 28].[6] Both algorithms are frequently used for experimental purposes (e.g., [19, 36, 13, 62]).

We collected 25 datasets from the UCI [3] and the STATLIB [45] repositories and from [5, 4, 30]; see Table 1 for an overview. Additionally, we created five data sets with data from a German meteorological institute (DWD).[7] In these data sets, the task is to predict the origin (one of the federal states in Germany) of a set of measurements (e.g., sunshine duration, temperature, ...). As our fuzzy extension is ineffective for nominal attributes, we only selected datasets having at least as many numeric as nominal attributes.

			# Attributes		
Dataset	# Inst..	# Classes	c	n	m
analcatdata-authorship	841	4	70	0	0
analcatdata-halloffame	1340	3	15	2	1
analcatdata-votesurvey	48	4	3	1	0
cars	406	3	6	1	2
collins	500	15	20	3	0
ecoli	336	8	7	0	0
eucalyptus	736	5	14	5	9
glass	214	6	9	0	0
iris	150	3	4	0	0
metStatCoordinates	4748	16	3	0	0
metStatRainfall	4748	16	12	0	0
metStatRST	336	12	3	0	0
metStatSunshine	422	14	12	0	0
metStatTemp	673	15	12	0	0
mfeat-factors	2000	10	216	0	0
mfeat-fourier	2000	10	76	0	0
mfeat-karhunen	2000	10	64	0	0
mfeat-morphological	2000	10	6	0	0
mfeat-zernike	2000	10	47	0	0
optdigits	5620	10	64	0	0
page-blocks	5473	5	10	0	0
pasture	36	3	21	1	0
pendigits	10992	10	16	0	0
segment	2310	7	19	0	0
squash-unstored	52	3	20	3	8
synthetic control	600	6	60	1	0
vehicle	846	4	18	0	0
vowel	990	11	10	2	0
waveform-5000	5000	3	40	0	0
wine	178	3	13	0	0

Table 1: Properties of the datasets used in the experiments: number of instances and classes, continuous (c) and nominal (n) attributes, and attributes with missing instances (m).

The experiments were conducted by randomly splitting each dataset into 2/3 for training and 1/3 for testing, and deriving the classification accuracy on the testing data for each learner. This procedure was repeated 100 times. Table 2 summarizes the results in terms of mean classification accuracies.[8]

[5] We used the following parameter setting: 3 fuzzy sets, product t-norm, maximum inference, and weighting scheme number 2 from [37].

[6] We used the following parameter setting: 5 fuzzy sets, 500 iterations without change, mutation probability 0.01, use weights, population size 100.

[7] Available at: `http://www.uni-marburg.de/fb12/kebi/research/repository`

[8] The classifier FR3-c, which also appears in the table, will be analyzed in Section 4.2.

The overall picture conveyed by the results is clearly in favor of FR3, which outperforms the other methods on most data sets. In particular, FR3 is better than C4.5 on 25 out of 30 datasets, better than CHI on all but two datasets, and better than SLAVE on all but one dataset. To analyze the differences between FR3, R3 and RIPPER more closely, we followed the two-step procedure recommended by Demšar [15]: First, a Friedman Test [21, 22] is conducted to test the null hypothesis of equal classifier performance. In case this hypothesis is rejected, which means that the classifiers' performance differs in a statistically significant way, a posthoc test is conducted to analyze these differences in more detail.

data set	FR3	R3	RIPPER	C4.5	CHI	SLAVE	FR3-c
analcatdata-authorship	**95.17**	94.37	93.05	93.50	71.60	91.87	94.66
analcatdata-halloffame	93.13	**93.22**	92.87	92.87	92.18	92.68	93.06
analcatdata-votesurvey	36.06	35.58	34.40	38.75	**40.19**	29.51	36.63
cars	81.48	79.52	75.93	**82.15**	68.96	70.68	80.93
collins	94.53	92.87	92.67	**96.10**	42.63	50.87	93.12
ecoli	**82.91**	82.46	80.57	81.35	77.43	81.03	82.04
eucalyptus	**64.25**	63.72	58.69	59.98	54.09	58.16	63.88
glass	**72.98**	69.61	63.18	66.69	61.39	61.83	71.13
iris	94.78	94.25	93.45	94.25	92.27	**94.92**	94.08
metStatCoordinates	**93.30**	92.85	92.04	92.87	46.79	58.77	92.97
metStatRainfall	**69.68**	68.13	60.66	59.47	24.51	29.35	68.86
metStatRST	43.22	**44.13**	36.08	38.60	25.24	42.02	43.30
metStatSunshine	**52.94**	51.17	44.48	46.78	37.93	28.83	52.25
metStatTemp	**57.38**	56.10	47.45	53.18	30.63	22.10	57.23
mfeat-factors	**93.35**	92.64	87.05	87.96	89.19	86.83	93.06
mfeat-fourier	**80.48**	79.26	71.37	74.42	69.27	73.49	80.16
mfeat-karhunen	**91.51**	89.70	79.13	80.20	82.55	78.37	91.13
mfeat-morphological	**72.31**	72.25	70.74	71.60	57.93	67.08	72.09
mfeat-zernike	**77.18**	76.13	67.58	69.11	72.37	68.26	77.05
optdigits	**96.22**	95.55	89.68	89.51	45.90	93.45	96.01
page-blocks	97.04	**97.13**	96.79	96.89	91.96	93.58	96.92
pasture-production	71.28	67.91	68.46	**73.67**	44.23	53.63	67.47
pendigits	**98.07**	97.48	95.54	95.92	97.45	87.26	97.67
segment	**96.97**	96.14	94.53	95.95	83.65	88.87	96.52
squash-unstored	75.52	74.92	71.74	**76.08**	70.56	65.56	75.02
synthetic control	**92.24**	90.78	82.85	90.00	68.33	89.23	90.89
vehicle	**72.78**	71.66	67.80	71.38	61.99	64.08	72.63
vowel	**84.03**	77.85	64.71	75.60	59.49	63.84	81.58
waveform	79.79	**80.58**	78.72	75.05	72.38	75.34	78.56
wine	92.70	92.63	90.02	91.22	**92.77**	92.46	91.84

Table 2: Estimation of classification accuracies in terms of averages on the testing data (best per dataset in bold).

The Friedman test is a non-parametric test which is based on the relative performance of classifiers in terms of their ranks: For each dataset, the methods to be compared are sorted according to their performance, i.e., each method is assigned a rank (in case of ties, average ranks are assigned); see Table 3. Let k be the number of classifiers and N the number of datasets. Let r_i^j be the rank of classifier j on dataset i, and $R_j = \frac{1}{N}\sum_{i=1}^{N} r_i^j$ the average rank of classifier j. Under the null-hypothesis, the Friedman statistic

$$\chi_F^2 = \frac{12N}{k(k+1)}\left[\sum_{j=1}^{k}(R_j)^2 - \frac{k\cdot(k+1)^2}{4}\right]$$

is asymptotically χ^2 distributed with $k-1$ degrees of freedom. If N and k are not large enough, it is recommended to use the following correction which is F-distributed with $(k-1)$ and $(k-1)(N-1)$ degrees

of freedom [34]:

$$\frac{(N-1)\cdot\chi_F^2}{N\cdot(k-1)-\chi_F^2} \tag{8}$$

In our case, the value of (8) is 149.82, while the critical value for the significance level $\alpha = 0.01$ is only 4.99. Thus, the null-hypothesis can quite safely be rejected, which means that there are significant differences in the classifiers' performance.

Dataset	FR3	R3	RIPPER
analcatdata-authorship	1	2	3
analcatdata-halloffame	2	1	3
analcatdata-votesurvey	1	2	3
cars	1	2	3
collins	1	2	3
ecoli	1	2	3
eucalyptus	1	2	3
glass	1	2	3
iris	1	2	3
metStatCoordinates	1	2	3
metStatRainfall	1	2	3
metStatRST	2	1	3
metStatSunshine	1	2	3
metStatTemp	1	2	3
mfeat-factors	1	2	3
mfeat-fourier	1	2	3
mfeat-karhunen	1	2	3
mfeat-morphological	1	2	3
mfeat-zernike	1	2	3
optdigits	1	2	3
page-blocks	2	1	3
pasture-production	1	3	2
pendigits	1	2	3
segment	1	2	3
squash-unstored	1	2	3
synthetic control	1	2	3
vehicle	1	2	3
vowel	1	2	3
waveform	2	1	3
wine	1	2	3
average (R)	1.13	1.90	2.97

Table 3: Ranks of the classifiers.

Given the result of the Friedman Test, we conducted the Nemenyi Test [48] as a posthoc test to compare classifiers in a pairwise manner. According to this test, the performance of two classifiers is significantly different if the distance of the average ranks exceeds the critical distance $CD_\alpha = q_{\alpha,k,\infty}\cdot\frac{1}{\sqrt{2}}$, where the q-value is taken from the Studentized Range Statistic [49]. The results of this test are summarized in Table 4: R3 is significantly better than RIPPER at the significance level $\alpha = 0.01$, which confirms the findings from [24]. More importantly, however, FR3 is even better than R3 at the same level.

4.2 The Effect of Fuzzification

The previous results have shown that FR3 is a significant improvement in comparison to RIPPER and R3. To explain this improvement, we conjecture that the scores produced by fuzzy rules are superior to those produced by conventional rules, which in turn is beneficial for the voting scheme that is used by the round robin learner to determine a prediction.

	FR3	R3	RIPPER
FR3		+	+
R3	−		+
RIPPER	−	−	

Table 4: Pairwise comparison between classifiers: + (−) indicates a better (worse) performance at a significance level of $\alpha = 0.01$.

To examine whether the fuzzification of rules is indeed the main factor, or whether the improvements should perhaps be attributed to other modifications, we conducted some additional experiments with a "crisp" variant of FR3, included in Table 2 under the name FR3-c. To optimize an interval as originally produced by RIPPER, this variant conducts a search process quite similar to the search for an optimal fuzzy interval (cf. Section 3.3). Instead of a trapezoid, however, it is again only allowed to use intervals, i.e., it simply tries to optimize the original decision boundary in terms of the rule's purity. When comparing FR3-c to R3, it loses 9 and wins 21 cases, which is less than the 26 wins achieved by FR3. The importance of the fuzzy rules become even more obvious when comparing FR3-c to FR3 itself. Here, the latter has a higher classification rate for all except two datasets.

From this analysis we conclude that the use of fuzzy rules is indeed essential for the superb performance of FR3.

4.3 Model Complexity

Since FR3 disables the pruning step in IREP*, it learns more specialized rules. Therefore, it is likely to produce models that are more complex, in terms of the number of rules and their lengths, than those produced by R3.

Indeed, FR3 produces more specific rules than R3 for all but one dataset, and also the average rule length (number of attributes in the antecedent part) of FR3 (1.78) is slightly larger than the average length for R3 (1.53); see Table 5 for detailed statistics. Likewise, FR3 uses more rules for all but one dataset, and again, the average number of rules is slightly higher for FR3 (3.00) than for R3 (2.31).

As can be seen, the improvement in performance comes at the cost of slightly more complex models, even though the average differences (less than one additional rule and about 0.3 additional attributes per rule) are admissible.

4.4 Representation of Uncertainty

The ability to represent uncertainty involved in a classification decision, in terms of measures of conflict and ignorance, is arguably one of the main advantages of FR3. To test whether FR3 does indeed provide a basis for implementing classifier that are more "reliable", we conducted another series of experiments in a setting of classification with reject option. Roughly speaking, the idea is that, if γ is a reliable index of classification uncertainty, then the value of γ should correlate with the probability to make a correct decision. Or, stated differently, when abstaining from the classification of all instances the γ-value of which exceeds a threshold t, the classification accuracy should improve on the remaining instances. The dependency between the threshold t and the classification accuracy is typically depicted in the form of so-called *accuracy-rejection* curves.

In our experiments, we tested two very simple uncertainty indexes (needless to say, various other indexes are conceivable) directly related to the two types of uncertainty reflected by FR3: γ_c is the degree of conflict between the top-class as suggested by FR3 (in terms of the score (7)) and the second-best class. Likewise, γ_i is the degree of ignorance between these two classes. Again, each dataset was randomly split, in proportion 2:1, for training and testing. This was repeated 100 times, and each instance (occurring in potentially many of the 100 test sets) was associated with its average γ-index.

The monotonicity expected of the dependence between rejection threshold t and classification accuracy is confirmed by the experimental results summarized in Table 6. Using γ_c, an improvement is obtained for all datasets, and γ_i leads to an improvement in all but one case. Typical accuracy-rejection curves are shown in Figure 3 (the plateaus in these curves are caused by the absence of instances with corresponding γ-values).

	rules per rule set		attributes per rule	
Dataset	FR3	R3	FR3	R3
analcatdata-authorship	2.51	2.20	1.76	1.54
analcatdata-halloffame	4.07	2.53	2.31	1.62
analcatdata-votesurvey	1.47	1.16	1.30	1.12
cars	4.28	3.14	1.86	1.64
collins	1.00	1.02	1.00	1.00
ecoli	1.45	1.20	1.34	1.12
eucalyptus	4.03	2.90	2.10	1.67
glass	1.64	1.37	1.46	1.23
iris	1.23	1.17	1.22	1.12
metStatCoordinates	1.51	1.40	1.36	1.29
metStatRainfall	4.35	3.37	2.37	2.01
metStatRST	1.68	1.40	1.35	1.22
metStatSunshine	1.59	1.35	1.42	1.23
metStatTemp	1.85	1.60	1.51	1.37
mfeat-factors	1.99	1.75	1.57	1.31
mfeat-fourier	2.69	2.14	1.93	1.54
mfeat-karhunen	2.91	2.56	1.87	1.63
mfeat-morphological	1.79	1.55	1.50	1.36
mfeat-zernike	3.30	2.63	2.09	1.70
optdigits	3.87	3.42	2.29	2.00
page-blocks	2.76	2.38	1.99	1.73
pasture-production	1.11	1.07	1.07	1.04
pendigits	3.77	3.41	2.31	2.07
segment	1.97	1.74	1.72	1.51
squash-unstored	1.33	1.20	1.19	1.10
synthetic control	1.45	1.52	1.36	1.15
vehicle	4.45	3.49	2.38	2.06
vowel	2.54	2.29	1.77	1.61
waveform	19.92	10.73	4.61	3.59
wine	1.58	1.47	1.39	1.18
average	3.00	2.31	1.78	1.53

Table 5: The number of rules per rule set and the number of attributes per rule.

In summary, these experiments clearly show that both measures of uncertainty derived by FR3, conflict and ignorance, are reliable indicators of the uncertainty involved in a classification decision.

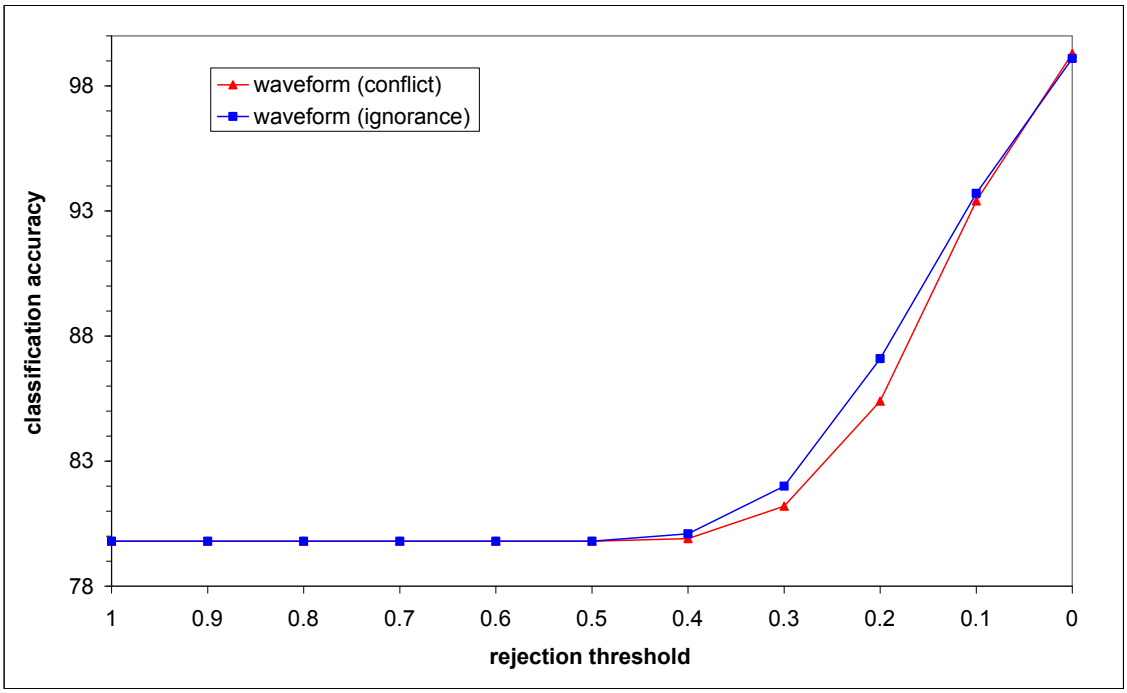


Figure 3: Accuracy-rejection curves for the dataset waveform.

5 Related Work

This section gives a brief overview of work in three related research fields, namely fuzzy rule learning, decomposition techniques for reducing multi-class to binary classification problems, and approaches to deal with issues of uncertainty and reliable classification.

There is a wealth of work on fuzzy rule learning, a comprehensive survey of which is clearly beyond the scope of this paper. The field can be roughly separated into several subfields: Firstly, there are fuzzy extensions of conventional rule induction techniques, such as covering algorithms [11]. Grid-based approaches, which proceed from fixed fuzzy partitions of the individual dimensions, are also quite popular [59]; they are less flexible but may have advantages with respect to interpretability. A well-known representative of this kind of approach is the CHI algorithm that we also used in our experiments [8, 9]. It proceeds from a fuzzy partition for each attribute and learns a rule for every grid cell. This is done by searching the training instance with maximal degree of membership in this cell (matching degree of the rule antecedent) and adopting the corresponding class attribute as the rule consequent. It is worth mentioning that fuzzy extensions of rule learning algorithms have not only been developed for the propositional case, but also for the case of first-order logic [18, 50, 56].

Secondly, several fuzzy variants of *decision tree learning*, following a separate-and-conquer strategy and producing rule sets of a special (hierarchical) structure, have been proposed [60].

Thirdly, *hybrid methods* that combine fuzzy set theory with other (soft computing) methodologies, notably evolutionary algorithms and neural networks, are especially important in the field of fuzzy rule learning. For example, evolutionary algorithms are often used to optimize ("tune") a fuzzy rule base or for searching the space of potential rule bases in a (more or less) systematic way [13]. One of these classifiers, which was also included in our experimental comparison, is the SLAVE classifier [27, 28]. It uses a genetic learning approach to create a fuzzy rule-based system by following a covering scheme. SLAVE represents each rule as a single chromosome. It uses an iterative approach, which means that the result of the genetic algorithm is not meant to cover all positive examples. Instead, the genetic algorithm is repeated until the iteratively generated set of rules is sufficient to represent the training set. Another interesting approach in this area is

	rejection threshold									
	1		0.6		0.2		0.1		0	
dataset	acc	cov	acc	cov	acc	cov	acc	cov	acc	cov
analcatdata-authorship (conflict)	95.2	100.0	95.2	100.0	96.2	94.9	98.1	85.9	99.4	54.9
analcatdata-authorship (ignorance)	95.2	100.0	95.2	100.0	96.4	96.0	97.8	89.4	99.3	60.6
analcatdata-halloffame (conflict)	93.1	100.0	93.1	100.0	94.1	97.8	96.0	92.5	98.4	76.3
analcatdata-halloffame (ignorance)	93.1	100.0	93.1	99.9	95.2	93.6	98.1	84.3		
analcatdata-votesurvey (conflict)	36.8	100.0	36.8	100.0	36.8	100.0	38.4	83.3		
analcatdata-votesurvey (ignorance)	36.8	100.0	35.5	87.5						
cars (conflict)	81.5	100.0	81.5	100.0	85.5	88.2	90.6	72.2	98.9	43.1
cars (ignorance)	81.5	100.0	81.5	100.0	88.0	75.4	95.9	55.7		
collins (conflict)	94.5	100.0	94.5	100.0	97.2	94.6	97.7	91.8	99.3	75.0
collins (ignorance)	94.5	100.0	94.5	100.0	97.2	94.6	97.7	91.8	99.7	70.6
ecoli (conflict)	83.1	100.0	83.1	100.0	84.8	93.8	87.1	85.4	94.2	55.4
ecoli (ignorance)	83.1	100.0	83.1	100.0	87.1	86.0	92.1	72.0		
eucalyptus (conflict)	64.5	100.0	64.5	100.0	66.2	91.6	70.8	65.9	89.5	21.7
eucalyptus (ignorance)	64.5	100.0	64.7	98.6	81.0	40.4	92.9	23.6	99.8	10.1
glass (conflict)	72.8	100.0	72.8	100.0	74.2	90.1	80.1	65.3	97.7	14.6
glass (ignorance)	72.8	100.0	72.8	99.5	78.6	73.2	89.2	40.8		
iris (conflict)	94.4	100.0	94.4	100.0	94.6	98.6	96.3	95.9	99.7	80.3
iris (ignorance)	94.4	100.0	94.4	100.0	95.2	97.3	96.9	93.9	99.8	28.6
metStatCoordinates (conflict)	93.3	100.0	93.3	99.9	95.4	94.7	97.2	88.6	99.6	70.9
metStatCoordinates (ignorance)	93.3	100.0	93.3	100.0	94.3	97.2	96.3	91.3	97.7	5.6
metStatRainfall (conflict)	69.7	100.0	69.7	100.0	75.9	77.2	86.6	46.7	96.3	13.3
metStatRainfall (ignorance)	69.7	100.0	69.7	100.0	78.6	68.8	89.0	39.8		
metStatRST (conflict)	43.4	100.0	43.4	100.0	43.9	96.7	49.5	71.4	57.1	8.9
metStatRST (ignorance)	43.4	100.0	43.5	99.7	53.5	27.7	58.1	6.0		
metStatSunshine (conflict)	52.6	100.0	52.6	100.0	55.0	86.5	64.6	42.7	86.6	3.8
metStatSunshine (ignorance)	52.6	100.0	52.6	100.0	61.0	55.5	72.4	19.4		
metStatTemp (conflict)	57.3	100.0	57.3	100.0	59.2	94.2	67.2	66.3	84.5	21.8
metStatTemp (ignorance)	57.3	100.0	57.4	99.9	70.1	57.5	82.4	29.4		
mfeat-factors (conflict)	93.3	100.0	93.3	99.9	95.1	94.5	97.5	85.5	99.6	58.3
mfeat-factors (ignorance)	93.3	100.0	93.3	100.0	94.8	95.9	96.9	87.6	99.7	23.0
mfeat-fourier (conflict)	80.4	100.0	80.4	99.9	84.8	87.1	92.7	62.0	99.6	32.2
mfeat-fourier (ignorance)	80.4	100.0	80.5	99.8	89.5	73.8	94.6	58.1	99.6	12.5
mfeat-karhunen (conflict)	91.5	100.0	91.5	99.9	94.8	87.2	97.8	71.6	99.9	32.9
mfeat-karhunen (ignorance)	91.5	100.0	91.5	100.0	94.0	92.5	96.4	77.5	99.4	8.4
mfeat-morphological (conflict)	72.2	100.0	72.2	99.9	73.4	95.5	76.1	85.3	91.6	46.2
mfeat-morphological (ignorance)	72.2	100.0	73.8	93.9	86.0	61.9	95.5	39.0	100.0	10.6
mfeat-zernike (conflict)	77.1	100.0	77.1	99.9	80.1	85.9	83.1	68.0	97.9	22.8
mfeat-zernike (ignorance)	77.1	100.0	80.2	91.7	90.1	71.9	95.2	58.5	99.2	6.0
optdigits (conflict)	96.2	100.0	96.3	99.9	98.1	93.8	99.1	86.3	99.9	62.7
optdigits (ignorance)	96.2	100.0	96.2	100.0	97.3	96.4	98.5	90.1	99.8	51.7
page-blocks (conflict)	97.1	100.0	97.1	100.0	97.7	98.5	98.5	96.2	99.4	88.3
page-blocks (ignorance)	97.1	100.0	97.1	100.0	97.7	98.4	98.7	95.4		
pasture-production (conflict)	71.6	100.0	71.6	100.0	72.3	91.7	72.8	72.2		
pasture-production (ignorance)	71.6	100.0	71.6	100.0	75.3	77.8	79.6	52.8		
pendigits (conflict)	98.1	100.0	98.1	99.9	99.0	96.5	99.4	92.2	99.9	73.3
pendigits (ignorance)	98.1	100.0	98.1	100.0	98.5	98.3	99.1	95.0	99.8	49.2
segment (conflict)	96.9	100.0	96.9	100.0	98.1	96.5	98.9	92.3	99.8	76.7
segment (ignorance)	96.9	100.0	96.9	100.0	98.0	97.0	98.8	93.1	99.9	40.7
squash-unstored (conflict)	74.8	100.0	74.8	100.0	74.6	96.2	78.7	80.8	76.7	25.0
squash-unstored (ignorance)	74.8	100.0	76.0	94.2						
synthetic control (conflict)	92.2	100.0	92.2	100.0	94.4	89.2	96.0	77.3	98.2	43.0
synthetic control (ignorance)	92.2	100.0	92.2	100.0	94.1	90.8	96.2	78.0	98.9	24.8
vehicle (conflict)	72.6	100.0	72.6	100.0	74.2	91.8	77.6	75.1	95.1	24.9
vehicle (ignorance)	72.6	100.0	74.0	94.0	95.1	45.7	98.5	37.7	100.0	2.6
vowel (conflict)	84.0	100.0	84.0	100.0	88.7	83.7	93.8	59.0	98.9	13.7
vowel (ignorance)	84.0	100.0	84.0	100.0	87.8	84.4	92.0	63.9	98.5	7.4
waveform (conflict)	79.8	100.0	79.8	100.0	85.4	77.9	93.4	50.1	99.3	14.6
waveform (ignorance)	79.8	100.0	79.8	100.0	87.1	75.9	93.7	55.8	99.1	24.4
wine (conflict)	92.8	100.0	92.8	100.0	95.1	92.1	97.3	83.7	99.7	52.2
wine (ignorance)	92.8	100.0	92.8	100.0	93.2	97.2	95.5	89.9	99.5	20.8

Table 6: Classification rates (acc) on the test set for different rejection thresholds and the coverage (cov) in terms of the percentage of non-rejected instances, both for conflict (using γ_c) and ignorance (using γ_i) as rejection criteria. For statistical reasons, results for less than 10 instances are not reported.

the one proposed in [14], which applies the idea of boosting [40] to the evolutionary learning of rule-based classifiers. *Neuro-fuzzy* methods [46, 47] encode a fuzzy system as a neural network and apply corresponding learning methods (like backpropagation). Fuzzy rules are then extracted from a trained network. A recent approach in this field is the SOTFN-SV algorithm which creates a Takagi-Sugeno (TS) rule base [38], using a TS-type fuzzy network combined with SVM learning techniques [57]. It contains five layers that consist of an input layer, an antecedent layer, a rule layer, a consequent layer, and finally an output layer. In contrast to previous SVM approaches [7, 10, 42, 43], SOTFN-SV learns the antecedents with a simplified version of the fuzzy clustering algorithm proposed in [39] and the consequences with a linear-kernel SVM.

Decomposition techniques for reducing multi-class to binary classification problems have been investigated quite extensively in recent years. Many standard decomposition schemes, including the all-pairs (round robin) and the one-against-rest scheme, are special cases of the more general approach of Error Correcting Output Codes (ECOC) [17] or, more precisely, their generalization that has been introduced in [2]. Even though ECOC allows for a more flexible decomposition of the original problem into simpler ones, the all-pairs approach has the advantage that it provides a fixed, domain-independent and non-stochastic decomposition with a good overall performance. In several experimental studies, including [2], it performed en par or better with competing decoding matrices. What is more important for us, however, is that the pairwise case produces binary relations as output, which is essential for the idea of LVPC, namely to connect classification learning with fuzzy preference modeling and decision making [33].

As mentioned repeatedly, our approach is most closely related to the R3 method by Fürnkranz [24] and, in fact, can be seen as direct (fuzzy) extension thereof. Fürnkranz also studied alternative decomposition schemes and found the pairwise approach (round robin) to be superior in terms of classification accuracy.

Issues of uncertainty and reliable classification have been addressed under various perspectives in the machine learning literature (e.g. [41, 58]) and remain to be an active area of research. Even though the focus is definitely on probabilistic methods, alternative frameworks for modeling and representing uncertainty have also been investigated [16, 32]. A distinction between different types uncertainty has been made, for example, in connection with reject options for nearest neighbor classification [31], where a distance reject (non-existence of neighbors close enough to the query) is distinguished from an ambiguity reject (existence of close neighbors from different classes). We are not aware, however, of a general and systematic treatment of the topic which goes beyond such special applications.

6 Concluding Remarks

In this paper, we have introduced a fuzzy rule-based classifier called Fuzzy Round Robin RIPPER (FR3). As opposed to conventional methods, FR3 carefully distinguishes between two sources of uncertainty in classification, namely conflict and ignorance, and, correspondingly, offers predictions of a more differentiated type: Against the background of the data seen so far, in conjunction with the underlying model assumptions, FR3 compares the potential decisions (class labels) in a pairwise manner and, for each pair, suggests to what extent one label is preferable to the other one, to what extent there is a conflict between these labels, and to what extent none of the two are supported. A prediction, or any other type of decision, can then be made on the basis of the fuzzy preference structure thus obtained.

Focusing on the core part of the method, namely the induction of the fuzzy preference structure, we have used relatively simple decision policies in this paper, both for standard classification (predicting a single class) and for classification with reject option. Nevertheless, developing suitable decision policies for different types of (generalized) classification problems is an important issue that we plan to address in future work. An interesting idea, for example, is to employ techniques from belief function theory, which not only offers suitable means for representing ignorance, but also operators for combining different sources of information [55].

Another interesting aspect concerns interpretability issues [6, 44]. Being able to understand a model produced by an inductive learner is desirable in general and become essential if the model is used, for example, for decision support [29]. Even though ensemble classifiers are usually judged critical from an interpretability point of view, we are actually not convinced that an FR3 prediction is necessarily less understandable than a prediction from a conventional (multi-class) fuzzy rule-based model. It is true that many pairwise models, as a whole, might be more difficult to capture than a single model. On the other hand, pairwise comparisons are known to play an important role in human decision making. Moreover, an FR3 prediction reduces complexity by providing information on two levels of abstraction: On the "relational

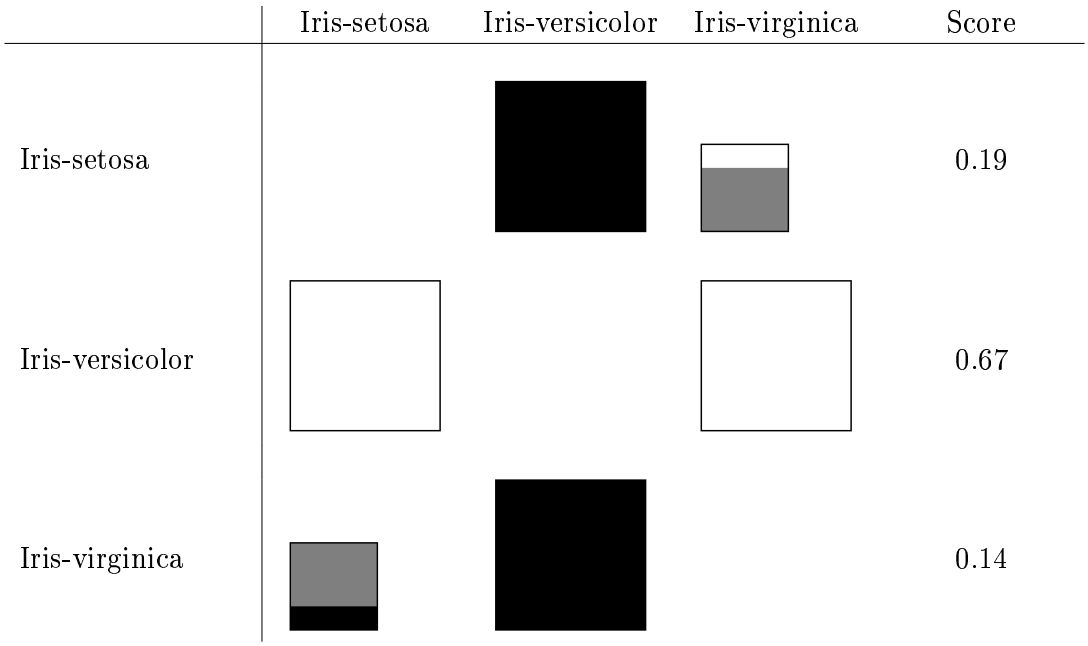


Figure 4: Graphical illustration of a preference structure predicted by FR3 for a query instance on the iris data. The size of a box is proportional to the degree of non-ignorance (1 minus ignorance). The size of the white (black) area is proportional to the degree of preference in favor of the row-class (column-class). The gray area shows the corresponding degree of conflict. The rightmost column shows the final score (7) for every class.

level", the preference structure gives a rough picture of the situation, including uncertainties and potential conflicts. Information on this level become especially comprehensible when being presented in a graphical form, as shown in Figure 4. If the need arises, each entry in the corresponding relations can then be "explained" by an underlying pairwise model. As an advantage, note that each pairwise model itself will typically be much simpler than a single polychotomous model, as it refers to only two instead of all classes simultaneously.

A Java implementation of FR3, running under the open-source machine learning toolkit WEKA, can be downloaded at: `http://www.uni-marburg.de/fb12/kebi/research/software`.

Acknowledgments

This research was supported by the German Research Foundation (DFG) and the Konrad-Adenauer-Foundation (KAS). The CHI and SLAVE classifiers were made available to us by the developers of the KEEL software (Alberto Fernández). We gratefully acknowledge this support. Finally, we like to thanks three anonymous reviewers for their helpful comments and useful suggestions.

References

[1] J. Alcalá-Fdez, L. Sánchez, S. García, M. del Jesus, S. Ventura, J. Garrell, J. Otero, C. Romero, J. Bacardit, V. Rivas, J. Fernández, and F. Herrera. *KEEL: A Software Tool to Assess Evolutionary Algorithms to Data Mining Problems*. Soft Computing (in press), 2008.

[2] E. Allwein, R. Schapire, and Y. Singer. Reducing multiclass to binary: a unifying approach for margin classifiers. *Journal of Machine Learning Research*, 1:113–141, 2001.

[3] A. Asuncion and D. Newman. UCI machine learning repository. `http://archive.ics.uci.edu/ml/index.html`, 2007.

[4] D. Barker. Dataset: Pasture production. `http://weka.sourceforge.net/wiki/index.php/Datasets`, 2007. Obtained on 20th of October 2007.

[5] B. Bulloch. Dataset: Eucalyptus soil conservation. `http://weka.sourceforge.net/wiki/index.php/Datasets`, 2007. Obtained on 20th of October 2007.

[6] J. Casillas, O. Cordon, F. Herrera, and L. Magdalena, editors. *Interpretability Issues in Fuzzy Modeling*. Studies in Fuzziness and Soft Computing. Springer-Verlag, Berlin, 2003.

[7] Y. Chen and J. Wang. Support vector learning for fuzzy rule-based classification systems. *IEEE Transactions on Fuzzy Systems*, 11(6):716–728, 2003.

[8] Z. Chi, J. Wu, and H. Yan. Handwritten numeral recognition using self-organizing maps and fuzzy rules. *Pattern Recognition*, 28(1):59–66, 1995.

[9] Z. Chi, H. Yan, and T. Pham. *Fuzzy Algorithms: With Applications to Image Processing and Pattern Recognition*. World Scientific Publishing Co., Inc., River Edge, NJ, USA, 1996.

[10] J. Chiang and P. Hao. Support vector learning mechanism for fuzzy rule-based modeling: a new approach. *IEEE Transactions Fuzzy Systems*, 12(1):1–12, 2004.

[11] I. Cloete and J. Van Zyl. Fuzzy rule induction in a set covering framework. *IEEE Transactions Fuzzy Systems*, 14(1):93–110, 2006.

[12] W. Cohen. Fast effective rule induction. In A. Prieditis and S. Russell, editors, *Proceedings of the 12th International Conference on Machine Learning, ICML*, pages 115–123, Tahoe City, CA, July 9–12, 1995. Morgan Kaufmann.

[13] O. Cordon, 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.

[14] M. del Jesus, F. Hoffmann, L. Navascues, and L.Sanchez. Induction of fuzzy-rule-based classifiers with evolutionary boosting algorithms. *IEEE Transactions on Fuzzy Systems*, 12(3):296–308, 2004.

[15] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

[16] T. Denoeux. A k-nearest neighbor classification rule based on Dempster-Shafer Theory. *IEEE Transactions on Systems, Man, and Cybernetics*, 25(5):804–813, 1995.

[17] T. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286, 1995.

[18] M. Drobics, U. Bodenhofer, and E. Klement. FS-FOIL: an inductive learning method for extracting interpretable fuzzy descriptions. *International Journal of Approximative Reasoning*, 32(2–3):131–152, 2003.

[19] A. Fernández, S. García, F. Herrera, and M. del Jesus. An analysis of the rule weights and fuzzy reasoning methods for linguistic rule based classification systems applied to problems with highly imbalanced data sets. In *Applications of Fuzzy Sets Theory*, volume 4578 of *Lecture Notes in Computer Science*, pages 170–178. Springer Berlin / Heidelberg, 2007.

[20] J. Fodor and M. Roubens. *Fuzzy preference modelling and multicriteria decision support*. Kluwer Academic Publishers, 1994.

[21] M. Friedman. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, 32(200):675–701, 1937.

[22] M. Friedman. A comparison of alternative tests of significance for the problem of m rankings. *The Annals of Mathematical Statistics*, 11(1):86–92, 1940.

[23] J. Fürnkranz. Separate-and-conquer rule learning. *Artificial Intelligence Review*, 13(1):3–54, February 1999.

[24] J. Fürnkranz. Round robin classification. *Journal of Machine Learning Research*, 2:721–747, 2002.

[25] J. Fürnkranz. Round robin ensembles. *Intelligent Data Analysis*, 7(5):385–403, 2003.

[26] J. Fürnkranz and G. Widmer. Incremental reduced error pruning. In *Proceedings of the 11th International Conference on Machine Learning, ICML*, pages 70–77, 1994.

[27] A. Gonzalez and R. Perez. Slave: a genetic learning system based on an iterative approach. *IEEE Transactions on Fuzzy Systems*, 7(2):176–191, 1999.

[28] A. Gonzalez and R. Perez. Selection of relevant features in a fuzzy genetic learning algorithm. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 31(3):417–425, 2001.

[29] A. Hamilton-Wright, D. Stashuk, and H. Tizhoosh. Fuzzy classification using pattern discovery. *IEEE Transactions on Fuzzy Systems*, 15(5):772–783, 2007.

[30] W. Harvey. Dataset: Squash harvest stored / unstored. `http://weka.sourceforge.net/wiki/index.php/Datasets`, 2007. Obtained on 20th of October 2007.

[31] M. Hellman. The nearest neighbor classification rule with a reject option. *Transactions on Systems, Man, and Cybernetics*, SMC-6:179–185, 1970.

[32] E. Hüllermeier. Possibilistic instance-based learning. *Artificial Intelligence*, 148(1–2):335–383, 2003.

[33] E. Hüllermeier and K. Brinker. Learning valued preference structures for solving classification problems. *Fuzzy Sets and Systems (to appear)*, 2008.

[34] R. Iman and J. Davenport. Approximations of the critical region of the friedman statistic. *Communications in Statistics*, 9(6):571–595, 1980.

[35] H. Ishibuchi and T. Nakashima. Effect of rule weights in fuzzy rule-based classification systems. *IEEE Transactions on Fuzzy Systems*, 9(4):506–515, 2001.

[36] H. Ishibuchi and T. Yamamoto. Performance evaluation of three-objective genetic rule selection. In *The 12th IEEE International Conference on Fuzzy Systems*, volume 1, pages 149–154, 2003.

[37] H. Ishibuchi and T. Yamamoto. Rule weight specification in fuzzy rule-based classification systems. *IEEE Transactions on Fuzzy Systems*, 13(4):428–436, 2005.

[38] C. Juang, S. Chiu, and S. Chang. A self-organizing ts-type fuzzy network with support vector learning and its application to classification problems. *IEEE Transactions on Fuzzy Systems*, 15(5):998–1008, 2007.

[39] C. Juang and C. Lin. An online self-constructing neural fuzzy inference network and its applications. *IEEE Transactions on Fuzzy Systems*, 6(1):12–32, 1998.

[40] M. Kearns. Thoughts on hypothesis boosting. ML class project, 1988.

[41] M. Kukar and I. Kononenko. Reliable classifications with machine learning. In *Proceedings of the European Conference on Machine Learning, ECML*, pages 219–231, 2002.

[42] C. Lin, C. Yeh, and C. Hsu. Fuzzy neural network classification design using support vector machine. In *Proceedings of the 2004 International Symposium on Circuits and Systems*, volume 5, pages V–724–V–727, 2004.

[43] C. Lin, C. Yeh, S. Liang, J. Chung, and N. Kumar. Support-vector-based fuzzy neural network for pattern classification. *IEEE Transactions on Fuzzy Systems*, 14(1):31–41, 2006.

[44] C. Mencar, G. Castellano, and A. Fanelli. On the role of interpretability in fuzzy data mining. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 15(5):521–537, 2007.

[45] M. Meyer and P. Vlachos. Statlib. `http://lib.stat.cmu.edu/`, 2007.

[46] S. Mitra and Y. Hayashi. Neuro-fuzzy rule generation: survey in soft computing framework. *IEEE Transactions on Neural Networks*, 11(3):748–768, 2000.

[47] D. Nauck, F. Klawonn, and R. Kruse. *Foundations of Neuro-Fuzzy Systems*. Wiley, Chichester, UK, 1997.

[48] P. Nemenyi. *Distribution-free multiple comparisons*. PhD thesis, Princeton University, 1963.

[49] D. Newman. The distribution of range in samples from a normal population, expressed in terms of an independent estimate of standard deviation. *Biometrika*, 31:20–30, 1939.

[50] H. Prade, G. Richard, and M. Serrurier. Enriching relational learning with fuzzy predicates. In *Proc. PKDD–03, European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 399–410, Cavtat-Dubrovnik, Croatia, 2003.

[51] J. Quinlan. Learning logical definitions from relations. *Machine Learning*, 5(3):239–266, 1990.

[52] J. Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann, 1993.

[53] J. Quinlan. MDL and categorial theories (continued). In *Proceedings of the 12th International Conference on Machine Learning, ICML*, pages 464–470, Lake Taho, California, 1995. Morgan Kaufmann.

[54] J. Quinlan and R. Cameron-Jones. Foil: A midterm report. In *Proceedings of the 6th European Conference on Machine Learning, ECML*, pages 3–20, London, UK, 1993. Springer-Verlag.

[55] B. Quost, T. Denoeux, and M. Masson. Pairwise classifier combination using belief functions. *Pattern Recognition Letters*, 28(5):644–653, 2007.

[56] M. Serrurier and H. Prade. Introducing possibilistic logic in ILP for dealing with exceptions. *Artificial Intelligence*, 171(16–17):939–950, 2007.

[57] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1999.

[58] V. Vovk, A. Gammerman, and G. Shafer. *Algorithmic Learning in a Random World*. Springer-Verlag, 2003.

[59] L. Wang and J. Mendel. Generating fuzzy rules by learning from examples. *Transactions on Systems, Man, and Cybernetics*, 22(6):1414–1427, 1992.

[60] T. Wang, Z. Li, Y. Yan, and H. Chen. A survey of fuzzy decision tree classifier methodology. In *Proceedings of the Second International Conference of Fuzzy Information and Engineering*, volume 40 of *Advances in Soft Computing*. Springer-Verlag, Berlin, Heidelberg, 2007.

[61] I. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Francisco, 2nd edition, 2005.

[62] M. Zolghadri and E. Mansoori. Weighting fuzzy classification rules using receiver operating characteristics (roc) analysis. *Inf. Sci.*, 177(11):2296–2307, 2007.