A New Approach to Fuzzy Random Forest Generation

Adriano Donato De Matteis, Francesco Marcelloni, Armando Segatori
Dipartimento di Ingegneria dell'Informazione
University of Pisa
Pisa, Italy
E-mail:francesco.marcelloni@unipi.it

***Abstract*—Random forests have proved to be very effective classifiers, which can achieve very high accuracies. Although a number of papers have discussed the use of fuzzy sets for coping with uncertain data in decision tree learning, fuzzy random forests have not been particularly investigated in the fuzzy community. In this paper, we first propose a simple method for generating fuzzy decision trees by creating fuzzy partitions for continuous variables during the learning phase. Then, we discuss how the method can be used for generating forests of fuzzy decision trees. Finally, we show how these fuzzy random forests achieve accuracies higher than two fuzzy rule-based classifiers recently proposed in the literature. Also, we highlight how fuzzy random forests are more tolerant to noise in datasets than classical crisp random forests.**

***Keywords*—Fuzzy Random Forests, Fuzzy Decision trees, Fuzzy Mutual Information**

I. Introduction

Decision trees are very popular classification methods due to a number of reasons [1], [2]. First of all, the classification accuracy achieved by decision trees is generally comparable to or higher than other well-known classification models. Second, decision tree induction algorithms require to tune a very small number of parameters. Third, decision trees are easy to interpret and comprehend. Several tree induction algorithms have been proposed in the literature since the seminal paper on ID3 by Quinlan [3].

Fuzzy decision trees are an extension of crisp decision trees to deal with uncertain data [4], [5]. Similar to a crisp decision tree, a fuzzy decision tree is a directed acyclic graph, in which each edge connects two nodes from parent node to child node. The node, which has no parent node, is called *root*, while the nodes, which have no child nodes, are called *leaves*. Each node in fuzzy decision trees represents a fuzzy subset rather than a crisp set as in classical crisp decision trees. The root coincides with the universe of discourse of each variable. All the child nodes generated from the same parent node constitute a fuzzy partition for the variable considered in the parent node.

Decision trees and fuzzy decision trees grow in a top-down way when we successively partition the training data into subsets having similar or the same output (class labels) [6]. Usually, the growth of the tree terminates when all data associated with a node belong to the same class. Most of the decision trees and fuzzy decision trees partition the training data into subsets by involving in this process only a single feature. Fuzzy decision trees proposed in the literature can be categorized into different types depending on the nature of the splitting mechanism [7]. In particular, we will focus on the fuzzy decision trees based on a generalization of ID3. The algorithms in this category apply fuzzy sets to quantify continuous attributes and then use the ID3 approach to construct the decision tree. Fuzzy entropy, information gain or gain ratio are used as a measure of attribute selection. Other algorithms exploit the minimal ambiguity (non-specificity) of a possibility distribution [8], the fuzzy Gini index [9], the maximum classification importance of attribute contributing to its consequent [10] and a normalized fuzzy Kolmogorov-Smirnov discrimination quality measure [11] to select the attribute used in the node splitting.

Generally, continuous attributes are partitioned before starting the fuzzy decision tree learning: each attribute is discretized by optimizing purposely-defined indexes [12], [13], [8]. In [14], the authors propose an interesting analysis on different combinations of discretization methods for dividing attribute domains into partitions and different approaches for defining membership functions on these partitions. To the best of our knowledge, a few papers have proposed algorithms which partition continuous attributes concurrently with the fuzzy tree generation. These algorithms exploit particular fuzzy partitions [9] or quite complex approaches [15], [16] . None of them, however, has been employed in the generation of fuzzy random forests.

A random forest is an ensemble learning method for classification, which constructs a multitude of decision trees and outputs the class that is the mode of the classes of the individual trees. The algorithm for inducing a random forest was developed by Leo Breiman [17]: in the algorithm, bagging is used in tandem with random attribute selection. Bagging creates diversity by constructing each classifier with a different set of examples, which are obtained from the original training dataset by re-sampling with replacement. At each node of each tree of the forest, a subset of the available attributes is randomly chosen and the best split available within these attributes is selected for that node. The number of attributes randomly chosen at each node is a parameter of the random forests. Random forests have proved to be very effective and have been applied in several different domains [18].

Fuzzy random forests were originally presented in Bonissone et al. [19], [20] and further discussed and applied in other papers [21], [22]. The fuzzy random forest learning exploits two algorithms. The first algorithm generates the forest by performing, for each tree, a random sampling with replacement on the available examples and calling the second algorithm. This algorithm is based on ID3 and builds a fuzzy tree by, first, selecting a random subset of attributes available at each node and, then, by choosing the best one to perform the split. Continuous attributes are partitioned before starting the random forest generation by using trapezoidal fuzzy sets. These partitions are obtained by using a complex method that adopts two steps. In the first step, a number of split points are determined by adopting the basic procedure for splitting nodes used in the classical C4.5 decision tree learning algorithm [23]. Then, in the second step, a genetic algorithm is employed to obtain the fuzzy sets that define the partitioning of the continuous attributes from the split points. When an unlabeled object has to be classified, all the fuzzy trees are activated and the outputs are combined. Different combination methods are proposed and experimented in [20]. Unfortunately, the results discussed in this paper cannot be used for a fair comparison. Indeed, the number of fuzzy decision trees employed in the fuzzy random forests change from one dataset to another and no explanation is provided on how this number should be chosen. Further, for each dataset, only the result of the best combination method on the test set is shown and this method is different from one dataset to another. Again, no explanation is given on how this method should be selected.

In this paper, we propose to generate the fuzzy partitions concurrently with the generation of the fuzzy decision trees, as generally carried out in the classical non-fuzzy decision tree generation when dealing with continuous attributes. When creating a decision node, we define a strong fuzzy partition consisting of three triangular fuzzy sets. The core of the intermediate fuzzy set coincides with the right and left extremes of the supports of the left and right fuzzy sets, respectively. Thus, the partition is completely determined by a unique point: the core of the intermediate fuzzy set. For each attribute in the set of randomly selected attributes for the node, we evaluate different positions of the core and select the position that provides the highest information gain. The information gain is computed by using the fuzzy entropy. The parent node is therefore split into three child nodes (one for each fuzzy set), which contain the objects of the parent node with membership values higher than 0.5 to the corresponding fuzzy set. The same continuous variable can be considered in different decision nodes from the root to a leaf: for each of these decision nodes we apply the same fuzzy partition. Since each of these nodes focuses on a specific interval of the universe of the continuous variable, this corresponds to carry out a zoom in on the specific interval. The learning of the fuzzy decision tree terminates when no node can be further split based on determined conditions.

We generate 100 fuzzy decision trees by randomly selecting different training sets through bagging. Then, we combine the results of the trees by adopting a very simple method: for each class, we add all the confidence values output by any leaf that contains training set instances of the class. The class characterized by the highest total confidence value is associated with the unlabeled object.

We tested the fuzzy random forests on 28 publicly available datasets. We compared the results with two well-known fuzzy classifiers recently proposed in the literature and with the random forest proposed by Breiman [17]. We show that our approach outperforms the fuzzy classifiers and is statistically equivalent to the random forest. Finally, we perturbed the datasets by adding noise in the class attribute [20]. In particular, we changed the class of the objects in the training set with probabilities 10% and 20%. We show that fuzzy random forests are more robust to noise than classical crisp random forests.

The paper is organized as follows. In Section II we introduce our approach to the learning of fuzzy decision trees. Sections III and IV describe how fuzzy random forests are generated and used in classification, respectively. In Section V we discuss the experimental results and Section VI draws some final conclusions.

II. The proposed Fuzzy Decision Tree Learning Algorithm

In this section, we introduce the fuzzy decision tree learning algorithm we use to generate the fuzzy random forest. Let $X = \{X_1, \dots, X_F\}$ be the set of input variables and X_{F+1} be the output variable. Since we consider classification problems, X_{F+1} is a discrete variable, which can assume values in $\{C_1, \dots, C_M\}$, where M is the number of possible classes. Let U_f, with $f = 1, ..., F$, be the universe of the f^{th} variable X_f. Let $TTR = \{(\mathbf{x}_1, x_{F+1,1}), \dots, (\mathbf{x}_N, x_{F+1,N})\}$ be the tree training set composed of N input-output pairs, with $\mathbf{x}_p = [x_{1,p} \dots, x_{F,p}]$ and $x_{F+1,p} \in \{C_1, \dots, C_M\}$. Input variables X_f can be continuous or categorical. Continuous variables need to be partitioned for generating the decision tree.

Unlike the fuzzy decision tree learning used in [20], we do not assume that continuous variables are partitioned before starting the tree learning, but we determine these fuzzy partitions during the tree generation. We aim to propose an approach that is easy to implement, is computationally light and guarantees to achieve accuracy values comparable with classical random forests. The ratio behind this approach is to explore specific zones of the input domain more and more in detail during the tree generation. In practice, we adopt a sort of zoom in on this specific zones. The “magnifying glass” is a strong fuzzy partition consisting of three triangular fuzzy sets. We adopt this partition because it is determined by just choosing a point, the core of the intermediate triangular fuzzy set.

Let S be the set of instances contained in a generic node of the tree. Further, for each instance $\mathbf{x}_p \in S$, let $\mu_S(\mathbf{x}_p)$ be the membership value of $\mathbf{x}_p$ to S. At the root of the tree, S coincides with TTR and $\mu_S(\mathbf{x}_p) = 1$ for each instance $\mathbf{x}_p \in S$. At each node, we select randomly a subset of the total number of input variables and partition the continuous variables contained in the set by using a strong partition with three triangular fuzzy sets. Fig. 1 shows an example of this partition. For the categorical variables, we simply consider all the possible values. Then, for each partition, we compute the fuzzy information gain (*FGain*) and choose the input variable with the highest value of FGain for splitting the node.

In classical crisp decision trees, the points can belong to only one of the subsets generated by partitioning the example set of the parent node. In fuzzy decision trees, with strong partitions, one point $\mathbf{x}_p$ can belong to two different fuzzy sets, for instance B_1 and B_2, with complementary membership degrees ($\mu_{B_1}(\mathbf{x}_i) = 1 - \mu_{B_2}(\mathbf{x}_i)$). To simplify the generation of the tree and to reduce its deepness, we consider in the child nodes only the examples with membership degree higher than 0.5. Thus, for each fuzzy set in the partition shown in Fig.1, we create a node and transfer to this node the examples which belong to the $\alpha - cut$, with $\alpha = 0.5$, of the fuzzy set. We verified that this choice simplifies the generation of the fuzzy decision tree without affecting the final accuracy.

More formally, for each attribute X_f, we sort the attribute values $x_{f,1} \ldots, x_{f,N_S}$ for the set S. Let l_f and u_f be the lower and upper bounds of the universe in X_f of the points contained in S. To determine the optimal cut-point t_f for variable X_f, we pose t_f in correspondence of the f-th coordinate (except l_f and u_f) of each point of the universe. For each possible candidate, we define a strong fuzzy partition of the universe by using three triangular fuzzy sets, namely $A_{f,1}$, $A_{f,2}$ and $A_{f,3}$ as shown in Fig. 1. The cores of $A_{f,1}$, $A_{f,2}$ and $A_{f,3}$ coincide with l_f, t_f and u_f, respectively.

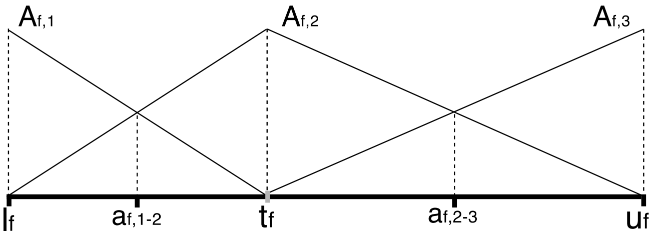

Fig. 1. An example of fuzzy partition of continuous variables used for splitting nodes

Then, we compute the points $a_{f,1-2}$ and $a_{f,2-3}$, where the membership function (MF) of $A_{f,1}$ intersects the MF of $A_{f,2}$, and the MF of $A_{f,2}$ intersects the MF of $A_{f,3}$, respectively. More precisely,

$$a_{f,1-2} = \frac{l_f + t_f}{2} \tag{1}$$

$$a_{f,2-3} = \frac{t_f + u_f}{2} \tag{2}$$

Let S_1, S_2 and S_3 be the subsets of examples in S with values of input variable X_f lower than or equal to $a_{f,1-2}$, larger than $a_{f,1-2}$ and lower than $a_{f,2-3}$, and larger than or equal to $a_{f,2-3}$, respectively.

We recall that the cardinality of a fuzzy set is defined as

$$|S| = \sum_{i=1}^{N_S} \mu_S(\mathbf{x}_i) \tag{3}$$

where N_S is the number of objects in S.

From this definition, we compute the cardinality of S_j, $j = 1, 2, 3$ as:

$$|S_j| = \sum_{i=1}^{N_j} \mu_{S_j}(\mathbf{x}_i) = \sum_{i=1}^{N_j} TN(\mu_{A_{f,j}}(x_{f,i}), \mu_S(\mathbf{x}_i)) \tag{4}$$

where N_j is the number of values of X_f (crisp cardinality) in the set S_j, $\mu_{S_j}(\mathbf{x}_i) = TN(\mu_{A_{f,j}}(x_{f,i}), \mu_S(\mathbf{x}_i))$ is the membership degree of example $\mathbf{x}_i$ to set S_j, $\mu_{A_{f,j}}(x_{f,i})$ is the membership degree of example $\mathbf{x}_i$ to fuzzy set $A_{f,j}$, $\mu_S(\mathbf{x}_i)$ is the membership degree of example $\mathbf{x}_i$ to set S (for the root of the decision tree, $\mu_S(\mathbf{x}_i) = 1$) and the operator TN is a T-norm. In the experiments, we adopted the *product* as T-norm.

To compute the fuzzy information gain, we exploit the weighted fuzzy entropy [14]. Let $FEnt(S_j)$, $j = 1, 2, 3$, be the fuzzy entropy of S_j defined as:

$$FEnt(S_j) = \sum_{m=1}^{M} -\frac{|S_{j,C_m}|}{|S_j|} log_2(\frac{|S_{j,C_m}|}{|S_j|}) \tag{5}$$

where S_{j,C_m} is the set of examples in S_j with class label equal to C_m. Then, the weighted fuzzy entropy $WFEnt(t_f; S)$ is computed as:

$$WFEnt(t_f; S) = \sum_{j=1}^{3} \frac{|S_j|}{|S|} FEnt(S_j) \tag{6}$$

The fuzzy information gain $FGain$ for variable X_f is defined as:

$$FGain(A, t_f; S) = FEnt(S) - WFEnt(A, t_f; S). \tag{7}$$

As in the discretization approaches used in crisp decision trees, a continuous variable can be considered in several decision nodes in the same path from the root to a leaf. In each node, we apply the same fuzzy partition shown in Fig. 1 to the universe of the set of objects that belong to the node with membership value higher than 0. For instance, let us assume that continuous variable X_f is used in a decision node to partition the universe $[l_f, u_f]$. The partition generates three child nodes, which contain objects with values of variable X_F in $\left[l_f, \frac{l_f+t_f}{2}\right]$, $\left[\frac{l_f+t_f}{2}, \frac{t_f+u_f}{2}\right]$ and $\left[\frac{t_f+u_f}{2}, u_f\right]$, respectively. Let us suppose that in the path generated from the first child node another decision node considers variable X_f. Then, a new partition is generated by considering the universe $\left[l_f, \frac{l_f+t_f}{2}\right]$. In practice, the new partition is devoted to analyze in detail a specific subset of the initial X_f domain. This process corresponds to perform a zoom in on specific intervals of the variables.

In the case of categorical variables, we split the node into a number of child nodes equal to the number of possible values for the variable. In the computation of the FGain, we compute the cardinality as:

$$|S_l| = \sum_{i=1}^{n_l} TN(1, \mu_S(x_i)) \tag{8}$$

where S_l is the set of values which correspond to the l-th categorical value.

The learning algorithm builds the fuzzy decision tree by splitting each node until one of the following termination conditions is met:

1) the node does not contain at least three examples;
2) the node contains examples which belong to the same class;

3) in case of datasets with only categorical variables, all the variables have been used in the path and therefore no variable can be used to perform the split;
4) the value of FGain is lower than a threshold ϵ. In the experiments, we adopted $\epsilon = 10^{-6}$;
5) in the case of continuous variables, at least two subsets generated by splitting the parent node contain a minimum number s of examples. In the experiments, we fixed s to 1.

Fig. 2 summarizes the learning algorithm used to generate the fuzzy decision trees. In our experiments, we set the number G of randomly selected variables used to generate child nodes to 5.

```
FuzzyDecisionTreeLearning(in: TTR; out: FuzzyTree)
begin
Let $S = TTR$ and $\mu_S(\mathbf{x}_p) = 1$ for each example $\mathbf{x}_p$ in S;
  1) Select $G$ input variables from the set $F$;
  2) For each variable in $G$
       • If the variable is continuous
           ○ For each possible cut point $t_f$
               ▪ create a strong fuzzy partition with three triangular fuzzy sets;
               ▪ compute the FGain using formula 7;
           ○ select the fuzzy partition with the highest FGain;
       • If the variable is categorical compute the FGain by using formula 7;
  3) choose the attribute with the highest value of FGain;
  4) split the node in child nodes according to possible outputs of the chosen variable;
  5) compute the membership values of each object to the child nodes;
  6) Repeat steps 1-5 until nodes can be split.
end
```

Fig. 2. The fuzzy decision tree learning algorithm

We do not label each leaf node with just one class, as typically made in crisp decision trees (for instance, adopting a majority voting strategy). Rather, each leaf node is labeled with all the classes that have at least one example in the leaf node: each class has a weight proportional to the number of training examples of that class in the node. More formally, we compute the weight w_m associated with each class C_m in the leaf node as:

$$w_m = \frac{|S_{C_m}|}{|S|}. \tag{9}$$

where S_{C_m} is the set of examples in S with class label equal to C_m.

III. Fuzzy Random Forest Learning

The algorithm used for learning the fuzzy random forests follows the Breiman's methodology: each fuzzy decision tree is constructed to the maximum size and without pruning. We recall that random forests have two stochastic elements: (1) bagging employed for the selection of the datasets used as input for each tree; and (2) the set of attributes considered as candidates for each node split. These randomizations increase the diversity of the trees and significantly improve their overall predictive accuracy when their outputs are combined.

Let TR be the training set used for the generation of the random forest. For learning each decision tree of the forest, we create a tree training set TTR of size $|TR|$ by randomly sampling TR with replacement. Then, we apply the algorithm described in Fig. 2 to TTR for generating the fuzzy decision tree. We generate V trees, where V is a parameter fixed by the user (in our experiments, we set V to 100).

Fig. 3 shows the algorithm used for the generation of the fuzzy random forest.

```
FuzzyRandomForestLearning(in: TR, V; out: Fuzzy Random Forest)
begin
Repeat the following steps $V$ times
  1) generate $TTR$ by randomly sampling with replacement $TR$;
  2) call the FuzzyDecisionTreeLearning function in Fig. 2 passing TTR and V as input;
  3) insert the fuzzy decision tree output from the function into the tree ensemble;
end
```

Fig. 3. The fuzzy random forest learning algorithm

IV. Classification

Given an unlabeled example $\widehat{x}$, in the classification phase, each tree of the forest outputs a list of possible classes with associated a confidence value. The confidence value is computed as sum of the activation degrees determined by any leaf node of the tree for that class. The class activation degree AD_m is calculated as the product between the weight w_m associated with the class C_m in the leaf node and the membership degree of $\widehat{x}$ to the leaf node. We recall that this degree is computed as the product of the membership values of $\widehat{x}$ to all the nodes in the path from the root to the leaf node: the membership values are computed by considering the overall fuzzy set associated with the node and not only the part with membership values higher than 0.5, as in the learning phase. Each activated leaf node produces a list of class activation degrees, which are summed up to compute the confidence value for that class. Thus, each tree outputs a list of pairs composed by the name of the class and the corresponding confidence value. More formally, for each class C_m, $m = 1, \ldots, M$, the confidence value CV_m is computed as

$$CV_m = \sum_{v=1}^{V} \sum_{\forall l} w_m \cdot \mu_l(\widehat{x}) \tag{10}$$

where $\mu_l(\widehat{x})$ is the membership value of $\widehat{x}$ to leaf node l of the fuzzy decision tree.

Several methods have been proposed in the literature to combine the outputs of different decision trees which compose a random forest. In the framework of fuzzy random forest, an interesting analysis has been performed in [20]. We experimented all the methods discussed in [20] and realized that the method, which guaranteed the best performance in terms of accuracy, simply sums all the corresponding confidence values for all the V lists generated by the trees and outputs the class corresponding to the highest total confidence value.

V. Experimental Results

In the first experiment, we tested our random forest on twenty-eight classification datasets extracted from the KEEL repository[1]. As shown in Table I, the datasets are characterized by different numbers of input variables (from 3 to 60), input/output instances (from 80 to 7200) and classes (from 2 to 7). For the datasets CLE, DER, HEP, MAM, and WIS, we removed the instances with missing values. The number of instances in the table refers to the datasets after the removing process.

TABLE I. Datasets used in the experiments (sorted for increasing numbers of input variables).

Dataset	# Instances	# Variables	# Classes
Haberman (HAB)	306	3	2
Hayes-roth (HAY)	160	3	3
Iris (IRI)	150	4	3
Mammographic (MAM)	830	5	2
Newthyroid (NEW)	215	5	3
Tae (TAE)	151	5	3
Bupa (BUP)	345	6	2
Appendicitis (APP)	106	7	2
Pima-532 (PIM-532)	532	7	2
Pima (PIM)	768	8	2
Glass(GLA)	214	9	6
Saheart (SAH)	462	9	2
Wisconsin (WIS)	683	9	2
Contraceptive (CON)	1473	9	3
Cleveland (CLE)	297	13	5
Heart (HEA)	270	13	2
Wine (WIN)	178	13	3
Smoking (SMO)	2855	13	3
Australian (AUS)	690	14	2
Vehicle (VEH)	846	18	4
Bands (BAN)	365	19	2
Hepatitis (HEP)	80	19	2
Image Segmentation (IMA)	2310	19	7
Thyroid (THY)	7200	21	3
Wdbc (WDB)	569	30	2
Dermatology (DER)	358	34	6
Ionosphere (ION)	351	34	2
Sonar (SON)	208	60	2

For each dataset, we performed a ten-fold cross-validation and executed three trials for each fold with different seeds for the random function generator (30 trials in total). All the results presented in this section are obtained by using the same folds for all the algorithms.

We carried out two experiments. In the first experiment, we aimed to highlight how fuzzy random forests can achieve values of classification accuracy considerably higher than other fuzzy classification approaches. In the second experiment, we analyzed the behavior of crisp and fuzzy random forests with data affected by noise. We intended to evaluate whether the fuzziness could help to manage noise.

A. First Experiment

In the first experiment, we compare the results generated by our fuzzy random forests with the fuzzy classifiers generated by two state-of-the-art fuzzy rule-based learning algorithms, namely FURIA [25] and PAES-RCS [26].

FURIA (Fuzzy Unordered Rules Induction Algorithm) was introduced in [25] as an extension of the RIPPER algorithm [27]. The main extensions regard: i) the use of fuzzy rather than crisp rules , ii) the exploitation of unordered rather than ordered rule sets, and iii) the introduction of a novel rule stretching method in order to manage uncovered examples. The descriptions of both FURIA and RIPPER can be found in [25] and [27], respectively.

PAES-RCS is an approach based on a multi-objective evolutionary algorithm to learn concurrently the rule and data bases of fuzzy rule-based classifiers. The learning process is performed by selecting a set of rules from the set of candidate rules and a set of conditions for each selected rule. This hybrid scheme was denoted as rule and condition selection (RCS) in [26]. To generate the set of candidate rules, the training set is pre-processed by transforming each continuous variable into a categorical and ordered variable. To this aim, a pre-defined fuzzy partition of each input variable is exploited. Then, the well-known C4.5 algorithm [6] is applied to the transformed training set for generating a decision tree. Finally, the set of candidate fuzzy rules is extracted from the decision tree: each rule corresponds to each path from the root to a leaf node. During the multi-objective evolutionary process, PAES-RCS generates the rule bases of the fuzzy rule-based classifiers by using the RCS approach and concurrently learns the MF parameters of the linguistic terms used in the rules. Accuracy and interpretability are measured in terms of percentage of correct classification and total number of antecedent conditions of the rules in the rule base, respectively. In [26], the authors have proved that PAES-RCS generates fuzzy rule-based classifiers with accuracy and complexity statistically comparable to, and sometimes better than, the ones generated by other state-of-the-art MOEA-based approaches.

Table II shows, for each dataset, the average classification rates calculated on the test set for the fuzzy random forests, FURIA and the most accurate solution of PAES-RCS after 50,000 fitness evaluations, respectively. We observe that the table considers only the datasets shown in [26], where PAES-RCS was proposed.

The analysis of Table II highlights that the classification rates achieved by the fuzzy random forests are on average higher than the ones obtained by the other two algorithms. To statistically validate this observation, for each algorithm, we generate a distribution consisting of the mean values of the accuracy of solutions on the test set by using all the datasets. Then, we apply the Friedman test in order to compute a ranking among the distributions [28], and the Iman and Davenport test [29] to evaluate whether there exists a statistical difference among the distributions. If the Iman and Davenport p-value

[1] available at http://sci2s.ugr.es/keel/datasets.php[24]

TABLE II. AVERAGE CLASSIFICATION RATES OBTAINED ON THE TEST SETS BY OUR FUZZY RANDOM FORESTS, FURIA AND THE MOST ACCURATE SOLUTIONS GENERATED BY PAES-RCS

Dataset	**Fuzzy Random Forest**	**FURIA**	**PAES-RCS**
HAB	70.72	**75.44**	72.65
HAY	80.88	83.13	**84.03**
IRI	**95.33**	94.66	**95.33**
MAM	83.80	**83.89**	83.37
NEW	**97.27**	96.30	95.35
TAE	**62.55**	43.08	60.81
BUP	**72.20**	69.02	68.67
APP	**87.73**	85.18	85.09
PIM	**76.48**	74.62	74.66
GLA	**75.13**	72.41	72.13
SAH	70.51	69.69	**70.92**
WIS	**97.13**	96.35	96.46
CLE	58.36	56.20	**59.06**
HEA	**83.89**	80.00	83.21
WIN	**97.28**	96.60	93.98
AUS	**86.00**	85.22	85.80
VEH	**75.38**	71.52	64.89
BAN	**70.89**	64.65	67.56
HEP	**89.58**	84.52	83.21
WDB	96.01	**96.31**	95.14
DER	**97.64**	95.24	95.43
ION	92.01	91.75	90.40
SON	79.93	**82.14**	77.00
Mean	**82.46**	80.34	80.66

is lower than the level of significance α (in the experiments $\alpha = 0.05$), we can reject the null hypothesis and affirm that there exist statistical differences between the multiple distributions associated with each approach. Otherwise, no statistical difference exists. If there exists a statistical difference, we apply a post-hoc procedure, namely the Holm test [30]. This test allows detecting effective statistical differences between the control approach, i.e. the one with the lowest Friedman rank, and the remaining approaches.

In Table III we show the Friedman rank and the Iman and Davenport p-value for each algorithm. We observe that the statistical hypothesis of equivalence is rejected. Thus, we have to apply the Holm post-hoc procedure considering our fuzzy random forests as control algorithm (associated with the lowest rank and in bold in the Table). As shown in Table IV, we observe that the fuzzy random forests statistically outperform both FURIA and PAES-RCS.

TABLE III. RESULTS OF THE NON-PARAMETRIC STATISTICAL TESTS ON THE CLASSIFICATION RATES COMPUTED ON THE TEST SET FOR THE FUZZY RANDOM FORESTS, FURIA AND THE MOST ACCURATE SOLUTIONS GENERATED BY PAES-RCS

Algorithm	*Friedman rank*	*Iman and Davenport p-value*	*Hypothesis*
Fuzzy Random Forest	1.413	0.00134950	Rejected
FURIA	2.2609		
PAES-RCS	2.3261		

TABLE IV. HOLM POST HOC PROCEDURE FOR $\alpha = 0.05$

i	algorithm	z-value	p-value	alpha/i	Hypothesis
2	PAES-RCS	3.096281	0.00196	0.025	Rejected
1	FURIA	2.875118	0.004039	0.05	Rejected

B. Second experiment

To verify whether fuzzy random forests, thanks to the use of fuzziness, are more robust than crisp random forests to noise, we first executed the classical and fuzzy random forests on all the datasets in Table I. As regards crisp random forest, we adopted the classical implementation proposed by Breiman [17]. Then, we perturbed the datasets by adding noise to the label data. In particular, we adopted the procedure proposed in [20]: we changed the class of the objects in the training set with probabilities 10% and 20%.

Table V shows the average classification rates achieved by classical and fuzzy random forests on the original datasets. We can observe that the classification rates are similar, also if fuzzy random forests achieve on average a classification rate higher than crisp random forests. Table VI shows the average classification rates on the test sets achieved by classical and fuzzy random forests trained by employing training sets with classes randomly changed with probabilities 10% and 20%, respectively. We observe that with the increase of the noise, the differences between the mean values of the average classification rates tend to increase.

TABLE V. AVERAGE CLASSIFICATION RATES OBTAINED ON THE TEST SETS BY FUZZY AND CRISP RANDOM FORESTS.

Dataset	**Fuzzy RF**	**Crisp RF**
HAB	70.72	**71.26**
HAY	**80.88**	80.38
IRI	95.33	**95.40**
MAM	**83.80**	81.24
NEW	**97.27**	95.01
TAE	**62.55**	49.79
BUP	72.20	**74.38**
APP	**87.73**	86.48
PIM-532	**78.45**	77.75
PIM	**76.48**	75.41
GLA	75.13	**77.45**
SAH	**70.51**	70.11
WIS	**97.13**	96.94
CON	**51.86**	51.73
CLE	**58.36**	57.36
HEA	**83.89**	83.04
WIN	**97.28**	97.15
SMO	**65.01**	63.29
AUS	**86.00**	85.72
VEH	**75.38**	74.71
BAN	70.89	**72.73**
HEP	**89.58**	88.73
IMA	96.20	**97.81**
THY	97.30	**99.59**
WDB	**96.01**	95.87
DER	**97.64**	97.15
ION	92.01	**93.09**
SON	79.93	**82.90**
Mean	**81.63**	81.16

To statistically validate this observation, we applied a non-parametric test, namely the Wilcoxon signed-rank test for pairwise comparison of two sample means [31], on the results obtained by the two ensembles on the original dataset, and on the datasets randomly changed with probabilities 10% and 20%, respectively.

Table VII shows the results of the Wilcoxon test. Here, $R+$ and $R-$ represent the ranks corresponding to the fuzzy and crisp random forests, respectively. We observe that the p-values for the original datasets and the datasets changed with probability 10% are higher than the level of significance $\alpha =$

TABLE VI. AVERAGE CLASSIFICATION RATES OBTAINED ON THE TEST SETS BY FUZZY AND CRISP RANDOM FORESTS TRAINED BY USING DATASETS WITH THE CLASSES RANDOMLY CHANGED WITH PROBABILITIES 10% AND 20%, RESPECTIVELY.

Dataset	**Class Noise**			
	10%		**20%**	
	Fuzzy RF	**Crisp RF**	**Fuzzy RF**	**Crisp RF**
HAB	**71.98**	71.08	**71.78**	69.10
HAY	77.13	**78.25**	71.63	**72.50**
IRI	94.27	**95.20**	**92.73**	88.67
MAM	**82.31**	80.04	**78.14**	76.35
NEW	**97.09**	94.63	**94.12**	90.92
TAE	**54.41**	47.02	**53.10**	44.27
BUP	**70.20**	70.06	**67.84**	66.35
APP	**89.00**	85.39	**88.79**	85.21
PIM-532	**78.81**	77.30	**76.99**	75.27
PIM	**75.72**	75.12	**75.12**	74.20
GLA	75.96	**76.94**	71.15	**75.49**
SAH	**71.24**	68.97	**69.71**	66.26
WIS	**96.70**	95.98	**96.03**	95.13
CON	**50.16**	49.11	**49.25**	48.42
CLE	**58.14**	56.59	57.60	**57.76**
HEA	**82.41**	81.59	**79.04**	76.56
WIN	**96.49**	96.19	91.09	**92.16**
SMO	**64.35**	60.99	**60.50**	56.49
AUS	**84.96**	84.77	**81.86**	80.09
VEH	74.22	**74.24**	71.01	**71.80**
BAN	70.44	**71.45**	69.36	**71.49**
HEP	82.71	**85.08**	**82.49**	82.27
IMA	96.08	**97.32**	**95.90**	95.58
THY	96.83	**99.59**	96.26	**99.37**
WDB	95.58	**95.70**	**95.20**	94.28
DER	97.53	**97.57**	96.70	**97.52**
ION	88.88	**91.20**	86.11	**86.88**
SON	79.81	**80.83**	75.75	**76.85**
Mean	**80.48**	79.94	**78.40**	77.40

TABLE VII. RESULTS OF THE WILCOXON SIGNED-RANK TEST ON THE ACCURACY OF BOTH FUZZY AND CRISP RANDOM FORESTS FOR $\alpha = 0.05$

Comparison	$R+$	$R-$	p-value	Hypothesis
Without noise				
Fuzzy RF vs Crisp RF	240	166	$\geq$ 0.2	Not Rejected
10% noise				
Fuzzy RF vs Crisp RF	247	159	$\geq$ 0.2	Not Rejected
20% noise				
Fuzzy RF vs Crisp RF	291	115	0.04512	Rejected

0.05. Thus, the null hypothesis is not rejected. In the case of datasets changed with probability 20%, the p-values are lower than $\alpha = 0.05$ and therefore the null hypothesis is rejected. This result confirms that fuzzy random forests are less sensitive to noise data than crisp random forests.

VI. CONCLUSIONS

In this paper, we have proposed a novel approach to the generation of a fuzzy decision tree in the context of fuzzy random forests. Although random forests have proved to be very accurate classifiers, they have not been extensively studied in the fuzzy community: only a few works have proposed approaches for generating fuzzy random forests and discussed their application to benchmark datasets. These approaches however build fuzzy partitions of the continuous attributes before starting the execution of the learning algorithm. Unlike these approaches, we create the fuzzy partitions during the generation of the tree by adopting an approach which iteratively zoom in on specific intervals of the universe.

We have shown that our fuzzy random forests outperform two recently proposed fuzzy rule-based classifier. Further, we have discussed how the use of fuzziness allows increasing the capability of random forests to manage noisy datasets with respect to classical crisp random forests.

REFERENCES

[1] J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques*, 3rd ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2012.

[2] X. Wang, X. Liu, W. Pedrycz, and L. Zhang, "Fuzzy rule based decision trees," *Pattern Recognition*, vol. 48, no. 1, pp. 50 – 59, 2015.

[3] J. Quinlan, "Induction of decision trees," *Machine Learning*, vol. 1, no. 1, pp. 81–106, 1986.

[4] C. Janikow, "Fuzzy decision trees: Issues and methods," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 28, no. 1, pp. 1–14, 1998.

[5] Y.-l. Chen, T. Wang, B.-s. Wang, and Z.-j. Li, "A survey of fuzzy decision tree classifier," *Fuzzy Information and Engineering*, vol. 1, no. 2, pp. 149–159, 2009.

[6] J. R. Quinlan, *C4.5: Programs for Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993.

[7] X. Liu, X. Feng, and W. Pedrycz, "Extraction of fuzzy rules from fuzzy decision trees: An axiomatic fuzzy sets (afs) approach," *Data & Knowledge Engineering*, vol. 84, no. 0, pp. 1 – 25, 2013.

[8] Y. Yuan and M. J. Shaw, "Induction of fuzzy decision trees," *Fuzzy Sets and Systems*, vol. 69, no. 2, pp. 125 – 139, 1995.

[9] B. Chandra and P. Varghese, "Fuzzy sliq decision tree algorithm," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 38, no. 5, pp. 1294–1301, 2008.

[10] X.-Z. Wang, D. Yeung, and E. Tsang, "A comparative study on heuristic algorithms for generating fuzzy decision trees," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 31, no. 2, pp. 215–226, 2001.

[11] X. Boyen and L. Wehenkel, "Automatic induction of fuzzy decision trees and its application to power system security assessment," *Fuzzy Sets and Systems*, vol. 102, no. 1, pp. 3 – 19, 1999.

[12] R. Weber, "Fuzzy-id3: a class of methods for automatic knowledge acquisition," in *Proc. 2nd Internat. Conf. on Fuzzy Logic & Neural Networks*, 1992, pp. 265–268.

[13] ——, "Automatic knowledge acquisition for fuzzy control applications," in *Proc. Internat. Symp. on Fuzzy Systems*, 1992, pp. 9–12.

[14] M. Zeinalkhani and M. Eftekhari, "Fuzzy partitioning of continuous attributes through discretization methods to construct fuzzy decision tree classifiers," *Information Sciences*, vol. 278, pp. 715–735, 2014.

[15] C. Z. Janikow, "A genetic algorithm method for optimizing fuzzy decision trees," *Information Sciences*, vol. 89, no. 34, pp. 275 – 296, 1996.

[16] A. Myles and S. Brown, "Induction of decision trees using fuzzy partitions," *Journal of Chemometrics*, vol. 17, no. 10, pp. 531–536, 2003.

[17] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001, cited By 8751.

[18] Z.-H. Zhou, *Ensemble Methods: Foundations and Algorithms*, 1st ed. Chapman & Hall/CRC, 2012.

[19] P. Bonissone, J. Cadenas, M. Del Carmen Garrido, and R. Daz-Valladares, "Fundamentals for design and construction of a fuzzy random forest," *Studies in Fuzziness and Soft Computing*, vol. 249, pp. 23–42, 2010.

[20] P. Bonissone, J. Cadenas, M. Carmen Garrido, and R. Andrs Daz-Valladares, "A fuzzy random forest," *International Journal of Approximate Reasoning*, vol. 51, no. 7, pp. 729–747, 2010.

[21] J. Cadenas, M. Garrido, R. Martnez, and P. Bonissone, "Extending information processing in a fuzzy random forest ensemble," *Soft Computing*, vol. 16, no. 5, pp. 845–861, 2012.

[22] J. Cadenas, M. Garrido, R. Martnez, D. Pelta, and P. Bonissone, "Using a fuzzy decision tree ensemble for tumor classification from gene expression data," in *IJCCI 2013 - Proceedings of the 5th International Joint Conference on Computational Intelligence*, 2013, pp. 320–331.

[23] J. Quinlan, "Improved use of continuous attributes in c4.5," *Journal of Artificial Intelligence Research*, vol. 4, pp. 77–90, 1996.

[24] J. Alcal-Fdez, A. Fernndez, J. Luengo, J. Derrac, and S. Garca, "Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework," *Multiple-Valued Logic and Soft Computing*, vol. 17, no. 2-3, pp. 255–287, 2011.

[25] J. Huhn and E. Hullermeier, "FURIA: an algorithm for unordered fuzzy rule induction," *Data Mining and Knowledge Discovery*, vol. 19, no. 3, pp. 293–319, 2009.

[26] M. Antonelli, P. Ducange, and F. Marcelloni, "A fast and efficient multi-objective evolutionary learning scheme for fuzzy rule-based classifiers," *Information Sciences*, vol. 283, pp. 36–54, 2014.

[27] W. W. Cohen, "Fast effective rule induction," in *In Proceedings of the 12th International Conference on Machine Learning*. Morgan Kaufmann, 1995, pp. 115–123.

[28] M. Friedman, "The use of ranks to avoid the assumption of normality implicit in the analysis of variance," *Journal of the American Statistical Association*, vol. 32, no. 200, pp. 675–701, 1937.

[29] R. L. Iman and J. H. Davenport, "Approximations of the critical region of the Friedman statistic," *Communications in Statistics - Theory and Methods Part A*, vol. 9, pp. 571–595, 1980.

[30] S. Holm, "A simple sequentially rejective multiple test procedure," *Scandinavian Journal of Statistics*, vol. 6, pp. 65–70, 1979.

[31] D. J. Sheskin, *Handbook of Parametric and Nonparametric Statistical Procedures*, 4th ed. Chapman & Hall/CRC, 2007.