IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received September 18, 2019, accepted September 30, 2019, date of publication October 8, 2019, date of current version October 23, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2946256

Paraconsistent Random Forest: An Alternative Approach for Dealing With Uncertain Data

GABRIELA W. FAVIEIRO AND ALEXANDRE BALBINOT
Electrical Engineering Department, Universidade Federal do Rio Grande do Sul, Porto Alegre 90035-190, Brazil

Corresponding author: Gabriela W. Favieiro (gabi.favieiro@gmail.com)

This work was supported in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior—Brasil (CAPES) under Grant 001.

ABSTRACT Pattern recognition algorithms have introduced increasingly sophisticated solutions. However, many datasets are far from perfect; for example, they may include inconsistencies and have missing data, which may interfere with the classification process. Thus, the use of paraconsistent logic can provide a compelling quantitative analysis approach in classification algorithms because it deals directly with inaccurate, inconsistent and incomplete data. Paraconsistent logic is considered a nonclassical logic, which enables the processing of contradictory signals in its theoretical structure without invalidating the conclusions. In this context, the proposed approach aggregates the power of hybrid classifiers, the low noise susceptibility of the random forest approach and the robustness of paraconsistent logic to provide an intelligent treatment of contradictions and uncertainties in datasets. The proposed method is called paraconsistent random forest. The computational results demonstrated that paraconsistent random forest could classify several databases with satisfactory accuracy in comparison with state-of-the-art methods, namely, LDA, KNN, and SVM. Regarding imperfect datasets, the proposed approach significantly outperforms most of these methods in terms of prediction accuracy.

INDEX TERMS Decision trees, hybrid classifier, pattern recognition, paraconsistent logic, random forest.

I. INTRODUCTION

Despite the sophisticated solutions that are introduced by pattern recognition algorithms, many datasets are far from perfect; e.g., many datasets contain inconsistencies and missing data, which may interfere with the classification process, as in the case of biological signals [1]–[4]. For instance, one of the many issues reported in the area of myoelectric prosthesis control consists that the acquired data is not reliable to represent the motion classes because of incomplete information and signal contaminants. These data inconsistencies interfere in the motion classification, leading to long and continuous training sessions [2], [4]–[10]. Hence, it is important to employ an algorithm that can take these elements into account in order to diminish the added error when confronting incomplete or inconsistent data, being able to maintain a stable classification without the need of continuous interventions. However, many proposed solutions are based in adaptive methods that continuously check the signal quality and performs an under demand automatic retraining of the classification model, that can be very time consuming.

Thus, a machine learning algorithm that is robust enough to signal degradations could possibly increase the clinical and commercial impact of myoelectric prosthesis. A robust algorithm could also be beneficial in others contexts that is necessary to deal with uncertain data [11]–[13].

Via the use of nonclassical logic, one can incorporate an additional step for improving the robustness and accuracy of the decision-making method. Paraconsistent logic (PL) is considered a nonclassic logic because it enables the treatment of contradictory signals on their theoretical structure in a nontrivial manner without invalidating the conclusions [14]. Paraconsistent logic has been applied in diverse areas, such as image analysis, medical disease diagnosis, and character and voice recognition [11], [15]–[18]. These studies reported that PL is highly suitable for pattern recognition because it can deal with imprecise, inconsistent and paracomplete data. Similar approaches, which differ in terms of topology, were also used to handle nondeterministic scenarios [19]–[21].

Another well-known algorithm is the random forest (RF). Random forest can be considered an extension of decision trees because this methodology uses several uncorrelated decision trees for decision making, thereby increasing the accuracy with low noise susceptibility, especially when

The associate editor coordinating the review of this manuscript and approving it for publication was Jihwan P. Choi.

dealing with stochastic signals [22]. When comparing RF to traditional statistical methods, it shows advantages when dealing with a high number of nonlinear iterations between independent variables and discrete target values. Also, it offers other advantages such as comprehensibility and measures of the variables importance [23]. RF is also an efficient algorithm that can handle large datasets [22]–[24] and has a proving record in dealing with noisy data [25], [26].

In the literature, although several algorithms based on nonclassical logic, such as fuzzy logic [13], [19], [21], [25], [27]–[31], have been proposed to deal with noise or lack of information, the hybridization of the random forest and paraconsistent logic has not yet been carried out. The absence of such kind of algorithms represents an interesting knowledge gap in the area of artificial intelligence. Therefore, the contribution of this work is the proposal of a novel classifier that is based on an ensemble of paraconsistent decision trees. The primary objective is to merge paraconsistent logic with random forest in a manner that utilizes the full capacity of PL for dealing with inconsistencies, the comprehensibility that is generated by decision trees and the low noise susceptibility of the random forest approach for use in contexts involving non-ideal data.

Additionally, a hybridization of the techniques of random forest and paraconsistent trees can expand its representative influence and, as a consequence, its applicability. It has advantages that enable the algorithm to deal more efficiently with the noisy and incomplete data that are typical of advanced real-world applications in order to mitigate the effects of theses contaminants on the classification accuracy, on which this type of algorithm can excel compared to others.

II. THEORETICAL BACKGROUND

A. DECISION TREES

Decision tree is a well-known algorithm in the literature for several reasons [22], [25], [32]–[34]: First, the generated model is highly comprehensible, which facilitates understanding [25]. Second, its accuracy, especially if used with an ensemble of decision trees, is comparable or superior to other well-known algorithms in the area of pattern recognition [22], [25]. Another important consideration is that decision trees have very few parameters that require adjustment, which renders the model less complicated to define.

A widely used technique is the ID3 algorithm, which was proposed by Quinlan [32], in which the system can handle symbolic domains. Additionally, ID3 assumes discrete domains with small cardinalities. This may help increase the comprehensibility of the generated model, compared to, for instance, the CART algorithm. CART creates numerous thresholds for partitioning numerical data, thereby reducing the comprehensibility of the generated model due to extensive data partitioning [33]. However, although ID3 is easier to understand, prior data partitioning is required. Both methods use recursive partitioning to learn and create a discriminative model, in which the details are represented in a tree.

Decision tree is also widely used as a base classifier in ensembles. For creating an ensemble, several techniques are available [35]. One of the most famous is random forest [22], which combines two techniques to generate a set of uncorrelated decision trees. Random forest uses the bagging technique, in which each classifier is created with a distinct data set that is obtained from the training data set via resampling with replacement. An adaptation of the randomization technique that was proposed by Dietterich [35] is also used in random forest, in which the 20 best attributes are selected in each node and one is randomly selected to split it.

Nonetheless, in supervised learning, one of the most important aspects is the training set, whose classes are known in advance. The main strategy of supervised learning is to create a set of rules that can determine the class of any object from a set of attributes [32]. However, the available attributes cannot always deal with object ambiguity. For instance, if the training data contain two objects with identical values, it is impossible to distinguish them, even if these objects belong to different classes; hence, there is a contradiction in the database. Therefore, the use of binary categories may not be the best option, depending on the type of signal. In this case, the use of nonclassical logic can improve the robustness to the decision algorithm if there may be contradictions in the system [36].

B. PARACONSISTENT LOGIC

Most common technologies are based on classical logic, namely, most of the machine decision basis is binary. However, this strictly binary topology does not accord with the human decision basis, which does not always result in a binary output. This complexity was the motivation behind nonclassical logic research since they could provide a superior logical decision in situations that involve, for example, uncertain, ambiguous, or contradictory information [21].

Paraconsistent logic was proposed approximately in 1910 by J. Lukasiewicz and N.A Vasilév independently. They envisioned the possibility of a logic that predicted contradictions in its basic structure [37]. However, only in 1954 was the initial system of paraconsistent logic (PL) developed, which included propositional calculations and all logic levels, by da Costa [38]. Over the years, paraconsistent logic continued to evolve, which enabled the manipulation of inconsistent systems without eliminating contradictions [37]. For instance, a PL can be considered a discriminating logical system that deals with data contradictions and uncertainties [19].

The paraconsistent annotated logic propositions correspond to the type p(μ, λ), wherein p is a proposition and their evidence degrees (or annotation constants) are μ and $\lambda \in [0, 1] \subset \Re$, where μ denotes the favorable evidence degree of p and λ the contrary evidence degree of p [17].

The certainty degree and the uncertainty degree are defined as:

$$G_c = (\mu - \lambda) \tag{1}$$

$$G_{un} = (\mu + \lambda - 1) \tag{2}$$

TABLE 1. Values of G_c and G_{un} and the corresponding logic states.

(G_c, G_{un})	Logic State
(1.0, 0.0)	True
(-1.0, 0.0)	False
(0.0, 1.0)	Inconsistent
(0.0, -1.0)	Paracomplete

We analyze the lattice with $G_c, G_{un} \in [-1, +1] \subset \mathfrak{R}$, and the four extreme logic states that are presented in Table 1 with the corresponding values of G_c and G_{un}.

$G_{un}(\mu, \lambda)$ represents the distance of the annotation constant (μ, λ) from the inconsistent or paracomplete state. Similarly, $G_c(\mu, \lambda)$ represents the distance of the annotation constant (μ, λ) from the true or false state [11].

The real certainty degree (G_{rc}) is obtained through the values of G_c and G_{un}. The projection of the ordered pair (G_c, G_{un}) onto the certainty degree axis attenuates the effects of their contradiction [18]. Then, the real certainty degree is based on (3) if $G_C > 0$ or (4) if $G_C < 0$:

$$G_{rc} = 1 - \sqrt{(1 - |G_C|)^2 + (G_{un})^2}, \quad \text{for } G_C > 0 \quad (3)$$

$$G_{rc} = \sqrt{(1 - |Gc|)^2 + (G_{un})^2} - 1, \quad \text{for } G_C < 0 \quad (4)$$

To create a paraconsistent network, in which the output of a node can be used as an input to another, G_{rc} must be normalized. The resultant real evidence degree (μ_{ER}) is based on the normalized value of G_{rc} and is defined as (5):

$$\mu_{ER} = \frac{G_{cr} + 1}{2} \quad (5)$$

The paraconsistent analysis nodes (PANs) are based on paraconsistent logic fundamentals. The PANs consist of an algorithm for treating and controlling signs of imprecise and contradictory information [7], [8]. Contrary to the theory of probability, favorable evidence and contrary evidence are not directly related. For instance, in the theory of probability, the favorable evidence of an event C is specified by p(C), which indicates the probability of occurrence of C. Thus, the probability of belief –C is 1-p(C). This relation is not considered valid in paraconsistent annotated logic [39]. Therefore, it is possible to obtain the real evidence degree (μER) from two input evidence items that are related to the same proposition, which are denoted as μ_1 and μ_2 ($\lambda_2 = 1 - \mu_2$), as described in Algorithm 1.

Thus, a random forest with a paraconsistent decision tree as a base classifier is proposed. Among the various ensemble techniques that are based on decision trees, random forest was chosen because, similar to boosting, it generates the best results in terms of accuracy and noise susceptibility [3]. In addition, a multiple classifier system provides, as an advantage, improvements in the results in terms of noise resistance compared to an individual classifier, as concluded in [8]. Moreover, the use of paraconsistent decision trees can provide the random forest with the advantages of this type of technique, which were discussed previously: an increased noise resistance and extended applicability to uncertain or vague contexts. In the present study, paraconsistent logic is used such that the pattern recognition system is considered to have an intrinsic certainty degree that considers the contradictions and uncertainties of the system [40].

Algorithm 1 Paraconsistent Analysis Node (PAN) Algorithm

```
Function PAN(in: input evidence degree $\mu_1$, where $0 \leq \mu_1 \leq 1$, and input evidence degree $\mu_2$, where $0 \leq \mu_2 \leq 1$; out: $\mu E$)
    1. Calculate the unfavorable input evidence degree:
          $\lambda_2 = 1 - \mu_2$
    2. Calculate the degree of contradiction:
          $\mu_{ctr} = \frac{(\mu+\lambda)}{2}$
    3. Calculate the certainty degree:
          $G_C = \mu - \lambda$
    4. Calculate the uncertainty degree:
          $G_{un} = (\mu + \lambda) - 1$
    5. Calculate the vector that is formed by $G_c$ and $G_{un}$:
          $D = \sqrt{(1 - |G_C|)^2 + (G_{un})^2}$
    6. Determine the quadrant to which $G_c$ belongs to and calculate the projected value $G_{rc}$ of the pair [$G_c$, $G_{un}$] on the $G_c$ axis:
          If ($G_c > 0$):   $G_{rc} = 1 - D$
          If ($G_c < 0$):   $G_{rc} = D - 1$
          If ($G_c == 0$):  $G_{rc} = 0$
    7. Calculate the output value:
          $\mu E = \frac{(G_{rc}+1)}{2}$
end
```

III. PARACONSISTENT RANDOM FOREST: AN ENSEMBLE THAT IS BASED ON PARACONSISTENT DECISION TREES

The proposed method is a random forest of paraconsistent decision trees, namely, an ensemble of classifiers that are based on Breiman's methodology. The method is referred to as paraconsistent random forest (PRF) in the text.

As a precondition for the PRF classification system, the input data must be partitioned into multiple domains with a corresponding pertinence degree. To calculate the inconsistency among samples, it is essential for the divided partitions to be fuzzy, instead of crispy, sets. Nonetheless, considering that the data are fuzzy, a data sample can belong to more than one partition; hence, it is possible to deal with this type of inconsistency in a nontrivial manner based on the principles of paraconsistent logic.

A. DOMAIN PARTITION PREPROCESSING

The fuzzy C-means algorithm (FCM) is used to obtain the domain partition of the input dataset by creating clusters from the inputs and defining Gaussian membership functions that are based on the calculated clusters. In fuzzy C-means clustering, each data item belongs to one or more clusters with a pertinence degree that is specified by membership grades that vary between 0 and 1. This method, which was developed by Dunn in 1973 and improved by Bezdek in 1981, is commonly used in pattern recognition [41]. It is based on the minimization of the following objective function:

$$J_m = \sum_{i=1}^{N} \sum_{j=1}^{C} u_{ij}^m \left\| x_i - c_j \right\|^2, \quad 1 \leq m < \infty \tag{6}$$

where m is any real number that is greater than 1, u_{ij} is the degree of membership of x_i to cluster j, x_i is the ith d-dimensional measured data item, and c_j is the d-dimensional center of the cluster. The data are partitioned via the iteration of the function that is specified above and the membership u_{ij} and the cluster centers c_j are updated by:

$$u_{ij} = \frac{1}{\sum_{k=1}^{c} \left(\frac{\|x_i - c_j\|}{\|x_i - c_k\|} \right)^{\frac{2}{m-1}}}, \quad c_j = \frac{\sum_{i=1}^{N} u_{ij}^m \cdot x_i}{\sum_{i=1}^{N} u_{ij}^m} \tag{7}$$

The termination criterion for the iteration depends on the condition $max_{ij}\left\{\left|u_{ij}^{(k+1)} - u_{ij}^{k}\right|\right\} < \varepsilon$, where ε is a termination criterion that is between 0 and 1 and k is the number of iteration steps. The process of iteration converges to a local minimum of the objective function J_m.

As discussed above, the data belong to each defined cluster according to the representation of a membership function. These membership functions are responsible for the fuzzy behavior of the algorithm, in which the same data do not belong exclusively to a well-defined cluster. In this sense, the membership function indicates that each datum may belong to several clusters with various values of the membership coefficient [41].

Another important consideration is that this research is focused not on the fuzzy domain partitioning method, but on the paraconsistent random forest algorithm. The fuzzy domain partition is only used as a preprocessing stage.

B. PARACONSISTENT RANDOM FOREST ALGORITHM

Breiman [22] proposed a random forest in which the decision trees can get as deep as the data permit and no pruning is used. However, to construct each tree with low data correlation, Breiman's methodology includes two stochastic elements during the tree construction. The first stochastic element is found in the dataset input selection, where a technique called bagging is used to select data for each tree randomly. The second stochastic element is formed by randomly selecting, at each node, a small group of input variables for splitting [22], [25]. These randomizations can provide a superior solution in terms of prediction accuracy due to the diversity that is generated by the various trees, as described in Algorithm 2.

Moreover, the random forest involves two parameters: the number of trees and the number of subset attributes for the node split. In the paraconsistent random forest, the size of the subset is a fixed value that is based on the number of features ($\sqrt{F^N}$), as proposed by Breiman [22]. Thus, the number of trees is the only significant parameter in the paraconsistent random forest method.

Algorithm 2 Paraconsistent Random Forest

```
Function PRF (in: samples E, number of trees T, domain partition of the samples DP; out: PRF)
  1. For t = 1 to T
    1.1. Et = Bagging (E)
    1.2. ParaconsistentTrees(t) = ParaconsistentDecisionTree (Et, DP)
  2. PRF = ParaconsistentTrees
end
```

C. PARACONSISTENT DECISION TREE ALGORITHM

The paraconsistent decision tree algorithm is based on a modification of the ID3 algorithm, which was proposed by Quinlan [32]. Typically, in the ID3 algorithm, an attribute is divided into one or more categories and the training example memberships to these partitioned sets are binary, namely, a sample either belongs or does not belong to that partition [33]. However, in the proposed modification, the partitioned sets are fuzzy and, therefore, elements of nonclassical logic must be employed. Paraconsistent logic is employed to deal with the inconsistencies of the fuzzy domain partition. Apart from the adaptations in the inference procedures and the domain partition, the basic elements for building a decision tree are respected. The following are the notations that are used to describe the proposed algorithm:

- E represents the set of samples that are used to build the tree, where E^N represents the samples that belong/lead to node N;
- C represents the set of classification classes;
- F^N denotes the feature set that leads to node N;
- The set of features in node N is represented by $F^N = \{F_1, F_2, F_3, \ldots F_i\}$;
- P^N and I^N denote the total example certainty degree and total information gain, respectively, of node N;
- $I^{S_{Fi}}$ represents the calculated standard information for feature i (F_i);
- $G_i^N = I^N - I^{S_{Fi}}$ represents the calculated information gain for feature i (F_i) in node N;
- P_k^N represents the propagated sample certainty degree for class C_k in node N;
- $X^N = \{X_j^N\}$ is the set of propagated certainty degrees of the samples j that belongs to node N;
- μf_k^c denotes the degree of membership on a specified fuzzy partition.

The algorithm must initialize the variables X^{Root} and I^{Root} to 1, as the root node represents the entire description space of a tree [33]. Additionally, the root node contains all training samples of the tree, which were previously selected in the bagging step, as specified in Algorithm 2 [22]. In each iteration, the algorithm obtains a random subset of the available attributes to be analyzed and selects the best attribute for the current node split. To select the best attribute, a popular method is to maximize the information gain [25]. This mechanism is computationally simple, as it assumes independence of the attributes [33]. Moreover, to calculate the information gain ($\mathrm{I^N}$) of the node, it uses equations (8) and (9):

$$I^N = -\sum\nolimits_{k=1}^{|C|} \left(\frac{P_k^N}{P^N} \cdot \log_2 \frac{P_k^N}{P^N} \right), \quad P^N = \sum\nolimits_{k=1}^{|C|} P_k^N \tag{8}$$

$$P_k^N = \sum\nolimits_{j=1}^{|E|} X_j^N \tag{9}$$

In the standard ID3 algorithm, the P_k^N function counts the total number of samples that belong to class k. If the partition is binary, a sample can belong or not to the attribute partition (X_j^N can be 0 or 1). Therefore, the value of P_k^N corresponds to the total number of samples j that satisfy the conditions that lead to node N.

However, to use a fuzzy domain partition, changes are required in the algorithm if a sample can belong to one or more clusters [33]. For the proposed method, the influence value of each sample X_j^N is calculated according to the paraconsistent certainty degree of the previous node and the degree of membership of the sample to fuzzy partition μf_k^c via the PAN algorithm (1). Therefore, the value of X_j^N corresponds to the certainty degree with which sample j satisfies the conditions that lead to node N. The function is defined as follows (10):

$$X_j^N = PAN(X_j^{N-1}, \mu f_k^c) \tag{10}$$

As discussed previously, paraconsistent logic accepts that the data from an attribute may not always be available, as is the case of many practical applications. In the case of a missing feature ($\mu f_k^c = NaN$), $X_j^N = PAN(X_j^{N-1}, \mu f_k^c)$ is 0.5, which represents the unknown value according to the principals of paraconsistent logic [24].

After the node information gain I^N is determined, the algorithm selects a random subset of features of size $\sqrt{F^N}$ for a possible node split. Due to the randomness, not all attributes are considered for each split. However, the attributes that are not selected in a split can be used by other splits in the same tree once the algorithm has recursively selected only features that have yet to be used.

To select the optimal feature for splitting the node, the standard information gain for each feature ($I^{S_{Fi}}$) must be calculated. The standard information gain is calculated via the same approach as the node information gain ($\mathrm{I^N}$), but with a restriction that the information is calculated only for the selected feature $\mathrm{F_i}$.

Therefore, the feature that maximizes equation (11) represents the most substantial information gain that is calculated in that subset and that feature is selected for splitting the node.

$$G_i^N = I^N - I^{S_{Fi}} \tag{11}$$

Subsequently, the child node conditions are generated from the domain partition of the selected feature. Thus, the samples that belong in the parent node are partitioned into child nodes according to those conditions [33]. Additionally, if there is no training sample to support a partition, the child node will not be generated [32].

Consequently, the decision tree is built with each node that represents the selected feature on which the domain partition was split and the terminal nodes (or leaf nodes) that represent the class label of the final subset of this branch [32]. The decision tree growth is terminated (a leaf node is obtained) if one of the following conditions is satisfied:

- A node is pure, namely, all remaining samples belong to the same class. In this case, the node becomes a leaf and is labeled with the corresponding class of the samples.
- There are no available features for the split node, but the samples still do not belong to the same class. Then, the node is converted into a leaf and labeled using majority voting to select the class to which more samples are associated in the subset.

Algorithm 3 Paraconsistent Decision Tree

```
Function ParaconsistentDecisionTree (in: E, DP, X)
    1. Define $X^{N=root} = 1$, $I^{N=Root} = 1$
    2. If leaf node (termination criteria)
        2.1. Leaf Node Class = $C_K$
        2.2. end
    3. Calculate the information gain of the node ($I^N$)
    4. Randomly select a subset of features ($\sqrt{F^N}$) for possible node splitting
        4.1. Calculate the standard information gain of each selected feature $F_i$ ($I^{S_{Fi}}$)
    5. Select the feature $F_i$ for node splitting that maximizes the equation $G_i^N = I^N - I^{S_{Fi}}$
    6. Split the node using the selected attribute ($F_i$). Identify all feature domain partitions that contain an example of set $E^N$
    7. Create the subnodes with all valid feature domain partitions (DPs) from Feature $F_i$
    8. ParaconsistentDecisionTree (in: $E^N$, DP, X): recursive function
end
```

In Algorithm 3, the tree only considers the features that are necessary for obtaining a path; thus, the tree is not required to use all available features.

Furthermore, the proposed paraconsistent random forest can combine the major concepts of Breiman's random forest with paraconsistent logic, as described in Algorithms 2 and 3.

D. PARACONSISTENT DECISION TREE EVALUATION

The paraconsistent tree evaluation procedure is almost the same as that of ID3. The main difference is that a sample traverses every path of the tree and all leaves generate a certainty degree that is associated with the corresponding class.

The certainty degree of each leaf is calculated from the propagation degree of the path using the function X_j^N, which was described previously. Therefore, each leaf has an associated class, which is associated with a calculated certainty degree, as described in Algorithm 4.

At this stage, the evaluation of a sample generates several certainty degrees and associated classes. To reach the final decision and to define the winning class of the forest, it is necessary to define one or more decision-making approaches, which will be presented in the following.

Algorithm 4 Paraconsistent Decision Tree Evaluation

```
Function ParaconsistentDecisionEvaluation (in: e, DP, X, out: leaf)
    1. Define $X^{N=root} = 1$
    2. If the node is a leaf:
        2.1. leaf(N) = [$C_k$, $X_j^N$]
        2.2. end
    3. $X_j^{N+1} = PAN(X_j^N, \mu f_k^c)$
    4. ParaconsistentDecisionEvaluation (in: e, DP, $X^{N+1}$): recursive function
end
```

E. DECISION-MAKING APPROACHES

Based on the paraconsistent decision tree evaluation that is described above, every leaf *i* of every tree *t* has a corresponding class (C_k) with a certainty degree (u_{ER}), which can be represented by the vector leaf(i,t)=[C_k, u_{ER}]. To calculate the final classification that is assigned to a sample, it is necessary to define two strategies: Strategy (1), which regards the forest as a unique classificatory system, and Strategy (2), which considers the individual classification of each tree in the forest prior to the final class decision-making.

Strategy (1): Considering the certainty degree values of all leaves in the forest, for identifying the class with the highest associated degree, two approaches are proposed:

- Average forest (AF): Calculate the average certainty degree for each class by considering the values of all leaves in the forest and choosing the class according to *arg max*, as follows:

$$\text{AvgU}_{er}(c_K) = \text{average (leaf } (C_k == k));$$

$$\text{ForestClass} = \text{argmax}(\text{AvgU}_{er});$$

- Paraconsistent forest (PF): Calculate the certainty degree for each class via the paraconsistent extractor algorithm (ParaExtr$_{ctr}$), which considers the values of all leaves in the forest, and choose the class according to *arg max*, as follows:

$$\text{AvgU}_{er}(c_K) = \text{ParaExtr}_{ctr}(\text{leaf } (C_k == k));$$

$$\text{ForestClass} = \text{argmax}(\text{AvgU}_{er});$$

The ParaExtr$_{ctr}$ algorithm, which was proposed by [42], is an algorithm for removing the effects of contradictions from uncertain knowledge databases that employ PANs. Considering that the PRF algorithm provides a list of certainty degrees G_μ for each associated class that represent the calculated values for each possible path of the trees, the ParaExtr$_{ctr}$ algorithm selects the maximum and minimum evidence degrees (μ_{max} and μ_{min}) for composing the PAN input. The primary objective of PANs is to extract the contradiction effect by calculating the resulting evidence degree between inputs μ ($\mu = \mu_{max}$) and λ ($\lambda = 1 - \mu_{min}$). The two evidence degrees from G_μ are replaced by the resultant evidence degree (μ_{R1}). This process is repeated until only one evidence degree remains in the evidence degree list G_μ [16], as represented in Algorithm 5. At the end of this process, each class will correspond to a unique recognition evidence degree (***μM***), which is the representative value of all considered leaves.

Algorithm 5 Paraextr $_{ctr}$ Algorithm [42]

```
Function ParaExtr(in: $G_\mu$, out:$\mu M$)
    1. Receive an input list of certainty degrees
            $G_\mu = (\mu E_1, \mu E_2, \mu E_3, \mu E_4, \mu E_5 \ldots)$
    2. Find from the list the maximum value
            $\mu_{max} = \max(G_\mu)$
    3. Find from the list the minimum value
            $\mu_{min} = \min(G_\mu)$
    4. Calculate the resulting evidence degree using the min and max values as inputs of a PAN
            $\mu_{R1} = PAN(\mu_{max}, \mu_{min})$
    5. Substitute the identified max and min values and add the calculated resulting evidence degrees
            $G_\mu = G_\mu - \mu_{max} - \mu_{min} + \mu_{R1}$
    6. Return to step 2 and continue until the length of $G_\mu$ is 1.
    7. The output value is the last remaining evidence degree from the list
            $\mu M = G_\mu$
end
```

Strategy (2): This strategy determines the winning class for each tree (using the *arg max* approach) and selects the global forest classification using the majority voting technique.

- Average tree with majority voting (ATMV): Calculate the average certainty degree for each class by considering the values of all leaves in each tree. Choose the winning class of the tree according to *arg max*. Finally, majority voting is used to select a unique class of the forest by considering the individual classification of each tree.
- Paraconsistent tree with majority voting (PTMV): Calculate the certainty degree for each class using the paraconsistent extractor by considering the values of all leaves in each tree. First, it is necessary to choose the winning class of the tree according to *arg max*.Then, define a single classification of the forest via majority voting based on the single classification of each tree in the forest.

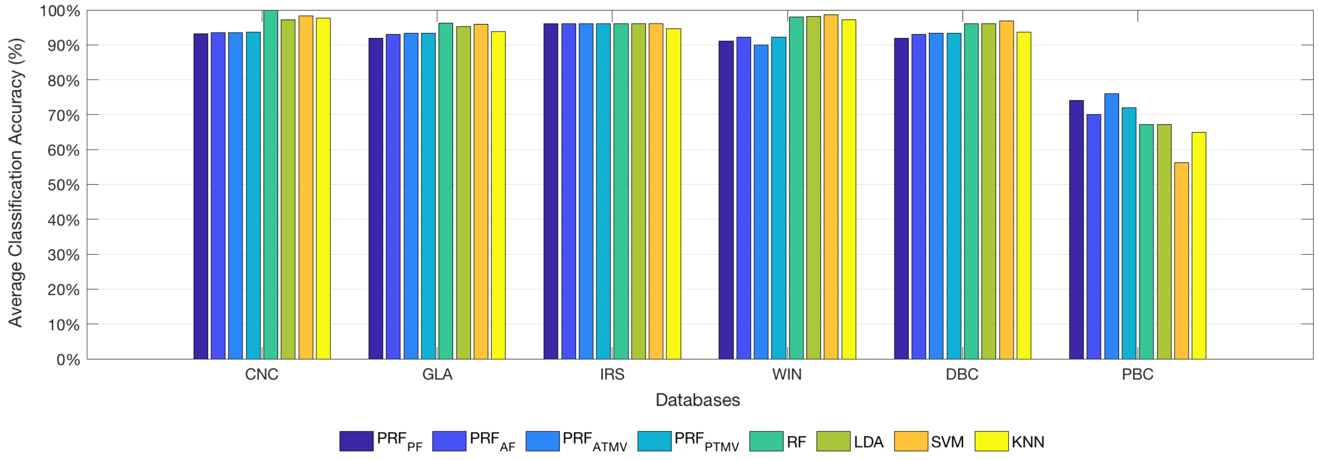

FIGURE 1. **Average classification accuracies for the selected datasets and classification methods.**

TABLE 2. **Dataset characteristics.**

Dataset	Abbr.	# instances	# features	# classes
Iris Plant	IRS	150	4	3
Breast Cancer Wisconsin (Prognostic)	PBC	198	34	2
Breast Cancer Wisconsin (Diagnostic)	DBC	569	32	2
Glass Identification	GLA	214	10	6
Wine	WIN	178	13	3
Cancer	CNC	216	4000	2

IV. RESULTS AND DISCUSSION

The test methodology is used to obtain several results on various datasets, which are used to evaluate the accuracies of the proposed PRF and all proposed decision-making algorithms. Additionally, the test experiments utilize a dataset degradation approach to evaluate the accuracy and behavior when dealing with imperfect data, for example, missing and noisy values. Moreover, the PRF algorithm is compared to well-known algorithms in the area of pattern recognition classification, such as RF, SVM, LDA, and KNN. The datasets that are selected for the test methodology belong to the UCI repository [43], which are commonly used for benchmarking, and their characteristics are listed in Table 2.

Table 2 lists the dataset name, the corresponding abbreviation ‘‘abbr’’, and other essential characteristics, namely, the total numbers of features, classes, and samples. The use of well-known datasets is important because it facilitates comparison with other algorithms in the scientific community. During the construction of the PRF trees, no pruning was used, namely, the trees were constructed to their maximum size. Additionally, the forest size for all datasets was defined empirically with a fixed value of 100 trees.

Moreover, the accuracy of the PRF algorithm was calculated for all decision-making algorithms that are proposed for the paraconsistent random forest. The average accuracy is plotted in Figure 1, along with those of other well-known algorithms on six selected databases. According to the evaluation of the results of the PRF ensemble against those that were obtained by a series of classifiers and multiclassifiers, the PRF ensemble is an efficient classifier, with similar averages to those that are obtained with well-known algorithms. The exception is the GLA database, on which the PRF method obtained inferior classification results, with a difference of $\sim 5\%$. This difference could be explained by the challenge of generating a suitable fuzzy domain partition using the simple method of fuzzy C-means.

A. EFFECT OF ABSENT DATA

To further evaluate the performance of the PRF in dealing with inconsistent data, a percentage $\alpha\%$ of missing values were introduced in the datasets. Considering that each dataset contains E samples and each sample has M attributes, $\alpha\%(E*M)$ values were randomly selected among all the uniformly distributed attributes. Each selected value, which corresponds to an example and to an attribute, was substituted by NaN. The classification models were created using the complete available information and the missing data were introduced only in the testing datasets. The objective of this evaluation is to determine how well the algorithms behave when dealing with incomplete datasets after the modeling, since this can occur in real-world applications.

To visualize the impact in terms of the prediction accuracy as a function of the defined percentage of missing values $\alpha\%$, the decrease in the percentage of the average classification accuracy was computed as follows (12):

$$\%decreaseaccuracy = AO - AM \qquad (12)$$

where AM is the average classification accuracy for the modified dataset with the addition of missing data and

TABLE 3. Average decrease in accuracy when introducing missing values.

	Algorithm	% decrease in accuracy					
		α% of added missing values					
		5%	**10%**	**30%**	**50%**	**70%**	**90%**
I R S	**PRF**$_{\text{PF}}$	0.00%	2.82%	7.05%	12.68%	22.54%	21.13%
	PRF$_{\text{AF}}$	1.39%	5.55%	9.72%	11.11%	20.83%	20.83%
	PRF$_{\text{ATMV}}$	**0.00%**	**2.78%**	5.55%	11.11%	20.83%	**19.45%**
	PRF$_{\text{PTMV}}$	1.39%	4.17%	**4.17%**	**8.33%**	**19.45%**	22.22%
P B C	**PRF**$_{\text{PF}}$	0.00%	1.35%	4.05%	**0.00%**	**0.00%**	8.11%
	PRF$_{\text{AF}}$	1.43%	5.71%	1.43%	10.00%	10.00%	8.57%
	PRF$_{\text{ATMV}}$	1.32%	2.63%	7.89%	11.84%	14.47%	11.84%
	PRF$_{\text{PTMV}}$	**0.00%**	**0.00%**	**1.39%**	12.50%	11.11%	**5.56%**
D B C	**PRF**$_{\text{PF}}$	0.76%	0.76%	1.52%	2.68%	8.02%	6.49%
	PRF$_{\text{AF}}$	2.26%	1.51%	1.88%	5.66%	12.08%	9.81%
	PRF$_{\text{ATMV}}$	0.38%	0.00%	0.00%	**-0.76%**	**4.51%**	**3.38%**
	PRF$_{\text{PTMV}}$	**0.00%**	**-0.38%**	**-0.38%**	0.38%	6.39%	4.89%
G L A	**PRF**$_{\text{PF}}$	2.02%	-2.05%	10.19%	10.19%	26.52%	22.45%
	PRF$_{\text{AF}}$	**-2.13%**	**2.13%**	6.38%	10.64%	31.91%	17.02%
	PRF$_{\text{ATMV}}$	0.00%	-1.91%	**-5.76%**	1.93%	**7.69%**	**13.48%**
	PRF$_{\text{PTMV}}$	1.86%	-1.86%	3.69%	**0.00%**	29.63%	22.22%
W I N	**PRF**$_{\text{PF}}$	**0.00%**	**0.00%**	**3.33%**	**3.33%**	**11.11%**	**23.33%**
	PRF$_{\text{AF}}$	0.00%	4.87%	8.54%	14.63%	19.51%	26.82%
	PRF$_{\text{ATMV}}$	0.00%	2.41%	8.44%	13.25%	21.69%	25.30%
	PRF$_{\text{PTMV}}$	2.47%	3.70%	6.18%	11.11%	11.11%	24.69%
C N C	**PRF**$_{\text{PF}}$	0.00%	4.87%	8.54%	14.63%	19.51%	26.82%
	PRF$_{\text{AF}}$	0.00%	2.41%	8.44%	13.25%	21.69%	25.30%
	PRF$_{\text{ATMV}}$	2.47%	3.70%	**6.18%**	11.11%	**11.11%**	24.69%
	PRF$_{\text{PTMV}}$	**0.00%**	**1.20%**	8.44%	**7.22%**	12.05%	**16.86%**

AO is the average classification accuracy for the original dataset.

Table 3 lists the average decreases in accuracy for the PRF algorithms with the four decision basis variations. In the worst-case scenario, with 90% imperfect values in the testing dataset, the maximum decrease in accuracy was only 19.45% in the IRIS database via the PRF$_{\text{ATMV}}$ approach. Considering only 5% missing values, the proposed algorithm could maintain the average accuracy with no significant decrease in classification performance.

Likewise, the decrease in accuracy results were also compared with those of other well-known classifiers in the area of pattern recognition, as presented in Table 6. In most cases, 5% of missing values resulted in a significant decrease in the prediction accuracy for methods SVM, KNN and LDA, which varied from 8% to 45% increase in the classification error.

The proposed paraconsistent random forest outperforms other algorithms in dealing with missing values. The PRF classification accuracy is not affected by the 5% missing values and with 90% invalid data, the classification error increases by 3%-18.67%, depending on the database. The increase in the error is larger when the database has more classes, as in the Wine and Iris Plant datasets.

Analyzing Figure 2, for algorithms SVM, KNN, and LDA, 5% replacement of the data already has a severe impact on the classification performance, with an accuracy reduction that ranges from 8.26%-45%. For 30% invalid data, the reduction in accuracy for these algorithms reaches a rate of at least 22.02%. In comparison to the proposed method, the impact of the addition of invalid values only becomes noticeable from 30%, for which the maximum error was approximately 8%. However, on the databases (GLA, DBC, and PBC), the decrease in the classification performance remains low up to 90% invalid data and varies between 3-6.42% for the proposed algorithm

TABLE 4. Algorithm comparison of the average decrease in accuracy when introducing missing values.

	Algorithm	% decrease in accuracy					
		α% of added missing values					
		5%	**10%**	**30%**	**50%**	**70%**	**90%**
I R S	PRF	0.00%	2.78%	4.17%	8.33%	19.45%	19.45%
	RF	0.00%	1.42%	2.82%	7.05%	12.68%	18.32%
	LDA	15.28%	29.17%	45.83%	59.72%	61.11%	63.89%
	SVM	15.28%	30.55%	45.83%	59.72%	61.11%	63.89%
	KNN	14.09%	29.58%	45.07%	59.15%	60.57%	63.38%
P B C	PRF	0.00%	0.00%	1.39%	0.00%	0.00%	5.56%
	RF	-3.92%	8.50%	12.91%	21.24%	23.37%	25.49%
	LDA	22.22%	25.49%	25.49%	25.49%	25.49%	25.49%
	SVM	11.11%	11.11%	11.11%	11.11%	11.11%	11.11%
	KNN	11.15%	23.99%	22.97%	22.97%	22.97%	22.97%
D B C	PRF	0.00%	-0.38%	-0.38%	-0.76%	4.51%	3.38%
	RF	0.36%	0.36%	0.73%	1.81%	5.45%	8.36%
	LDA	27.74%	32.85%	34.67%	34.67%	34.67%	34.67%
	SVM	28.26%	32.97%	35.14%	35.14%	35.14%	35.14%
	KNN	27.71%	32.21%	32.95%	32.95%	32.95%	32.95%
G L A	PRF	0.00%	-1.91%	-5.76%	1.93%	7.69%	13.48%
	RF	0.00%	-12.96%	-5.55%	-11.12%	0.00%	11.10%
	LDA	13.64%	33.33%	40.91%	43.95%	42.43%	42.43%
	SVM	15.88%	27.34%	46.04%	44.45%	44.45%	44.45%
	KNN	17.46%	26.99%	38.10%	39.69%	39.69%	39.69%
W I N	PRF	0.00%	0.00%	3.33%	3.33%	11.11%	23.33%
	RF	0.00%	1.20%	8.44%	7.22%	12.05%	16.86%
	LDA	30.68%	39.77%	54.55%	57.96%	59.09%	59.09%
	SVM	46.19%	59.18%	71.24%	72.17%	72.17%	72.17%
	KNN	31.03%	39.08%	54.03%	57.47%	58.62%	58.62%
C N C	PRF	0.00%	1.20%	6.18%	7.22%	11.11%	16.86%
	RF	0.00%	0.00%	3.33%	3.33%	11.11%	23.33%
	LDA	30.68%	39.77%	54.55%	57.96%	59.09%	59.09%
	SVM	46.19%	59.18%	71.24%	72.17%	72.17%	72.17%
	KNN	31.03%	39.08%	54.03%	57.47%	58.62%	58.62%

TABLE 5. Average decrease in accuracy when introducing noisy values.

	Algorithm	% decrease in accuracy					
		α% of added missing values					
		5%	**10%**	**30%**	**50%**	**70%**	**90%**
C N C	**PRF**$_{\text{PF}}$	**-0.61%**	0.61%	2.45%	3.07%	6.44%	12.88%
	PRF$_{\text{AF}}$	0.31%	0.00%	2.45%	2.45%	5.81%	10.40%
	PRF$_{\text{ATMV}}$	0.31%	0.61%	2.45%	**1.83%**	**4.89%**	**9.79%**
	PRF$_{\text{PTMV}}$	0.00%	**0.00%**	**1.22%**	2.74%	7.01%	13.72%
G L A	**PRF**$_{\text{PF}}$	0.38%	**1.53%**	4.96%	13.36%	16.79%	18.32%
	PRF$_{\text{AF}}$	0.75%	2.26%	5.66%	13.58%	16.60%	18.87%
	PRF$_{\text{ATMV}}$	0.38%	2.26%	4.14%	10.90%	14.66%	18.42%
	PRF$_{\text{PTMV}}$	**0.00%**	1.88%	**3.76%**	**9.77%**	**12.78%**	**16.54%**
I R S	**PRF**$_{\text{PF}}$	2.78%	5.56%	5.56%	6.94%	18.06%	30.56%
	PRF$_{\text{AF}}$	2.78%	**5.56%**	**5.56%**	**6.94%**	**18.06%**	31.94%
	PRF$_{\text{ATMV}}$	**2.78%**	6.94%	9.72%	15.28%	20.83%	**30.56%**
	PRF$_{\text{PTMV}}$	2.78%	5.56%	12.50%	15.28%	26.39%	37.50%
W I N	**PRF**$_{\text{PF}}$	0.00%	1.22%	3.66%	13.41%	20.73%	17.07%
	PRF$_{\text{AF}}$	0.00%	6.02%	7.23%	18.07%	24.10%	15.66%
	PRF$_{\text{ATMV}}$	**-1.23%**	2.47%	**1.23%**	9.88%	18.52%	16.05%
	PRF$_{\text{PTMV}}$	1.20%	**1.20%**	2.41%	**7.23%**	**14.46%**	**12.05%**
D B C	**PRF**$_{\text{PF}}$	0.76%	3.82%	7.25%	9.16%	18.32%	18.70%
	PRF$_{\text{AF}}$	1.13%	3.77%	7.55%	10.94%	19.25%	19.25%
	PRF$_{\text{ATMV}}$	1.13%	**2.26%**	6.02%	9.40%	15.41%	15.79%
	PRF$_{\text{PTMV}}$	**0.00%**	2.26%	**4.14%**	**7.14%**	**15.04%**	**14.66%**
P B C	**PRF**$_{\text{PF}}$	1.35%	**-4.05%**	0.00%	-2.70%	-2.70%	-2.70%
	PRF$_{\text{AF}}$	-2.86%	-1.43%	**-1.43%**	**-10.00%**	**-8.57%**	**-8.57%**
	PRF$_{\text{ATMV}}$	-1.32%	0.00%	3.95%	0.00%	0.00%	0.00%
	PRF$_{\text{PTMV}}$	**-4.17%**	2.78%	0.00%	-5.56%	-4.17%	-5.56%

B. EFFECT OF NOISE

Equally to the experiment of missing data, the performance of the PRF method was also evaluated considering the presence of noise in the dataset. To emulate noisy data, a percentage a% of the dataset was substituted by uniformly distributed random values.

Table 5 lists the average decreases in accuracy for the PRF algorithms with the four decision basis variations for the experiment with added noise. In the worst-case scenario, with 90% noisy values in the testing dataset, the maximum decrease in accuracy was only 30.56% in the IRS database via

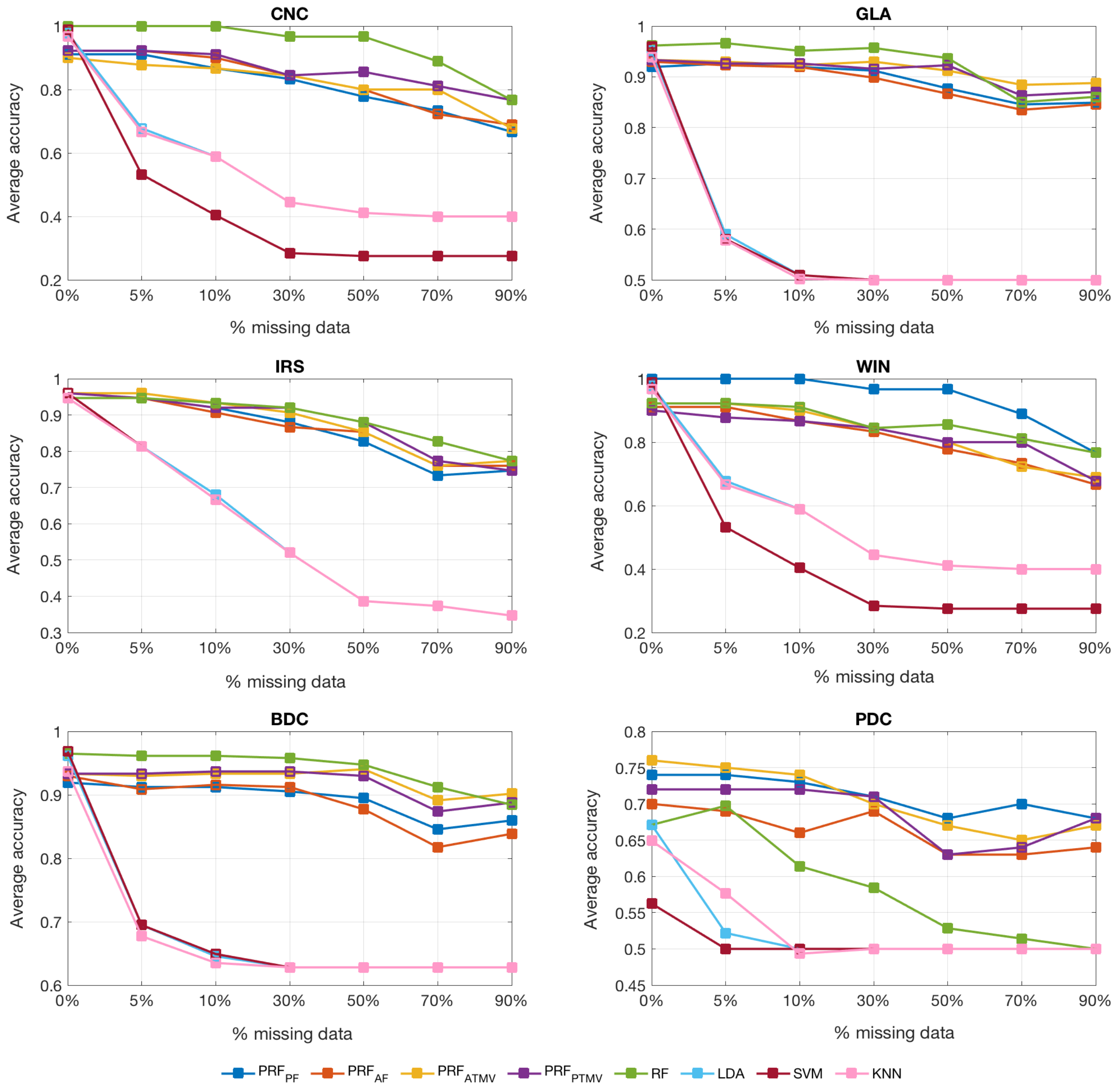

FIGURE 2. Average classification accuracy graph comparison for the PRF algorithms and LDA, SVM and KNN with the addition of missing data.

the PRF_{ATMV} approach. Considering only 5% noisy values the proposed algorithm could maintain the average accuracy with a maximum decrease in classification performance of 2.78% in the IRS database. Besides, the introduction of up to 30% of noise occasioned a maximum of 5% increase in the classification error for all datasets.

Table 6 list the decrease in accuracy in comparison to other well-known classifiers in the presence of noise. It is noticeable that a 5% noise introduction causes a significant impact in the classification prediction for the methods SVM, KNN and LDA with an increase of the classification error varying from 3% to 40%, depending on the database. The PRF classification accuracy is not affected by the 5% noisy values and considering 90% noisy data, the classification error increases by 3%-30.56%, depending on the database. Hence, it is possible to affirm that the paraconsistent random forest outperforms other algorithms in dealing with noise.

Figure 3 presents the overall classification accuracy for the experiment when introducing noise. The PRF algorithm has at least similar accuracies comparing to the RF method considering up to 30% noisy data. Besides, above 30% of added noise, the PRF presented a less susceptibility to noise with superior classification accuracies in comparison to the RF method. Also, analyzing the classification accuracy of the methods LDA KNN and SVM for a 5% noise addition it is noticeable a considerable reduction in the classification performance, with an increase of the classification error up to 40%.

TABLE 6. Algorithm comparison of the average decrease in accuracy when introducing noise.

	Algorithm	% decrease in accuracy					
		α% of added missing values					
		5%	**10%**	**30%**	**50%**	**70%**	**90%**
CNC	PRF	0.31%	0.61%	2.45%	1.83%	4.89%	9.79%
	RF	4.00%	9.43%	25.14%	38.29%	48.29%	56.00%
	LDA	18.82%	33.24%	54.41%	60.29%	56.18%	62.94%
	SVM	22.09%	38.08%	57.27%	63.66%	64.53%	64.53%
	KNN	21.93%	37.43%	57.02%	63.45%	63.74%	64.62%
GLA	PRF	0.38%	2.26%	4.14%	10.90%	14.66%	18.42%
	RF	0.47%	0.07%	15.00%	27.09%	39.78%	43.55%
	LDA	32.23%	43.03%	47.18%	43.91%	52.04%	50.21%
	SVM	38.93%	48.32%	48.47%	47.89%	47.89%	47.89%
	KNN	31.23%	36.81%	45.53%	46.11%	46.71%	46.41%
IRS	PRF	2.78%	6.94%	9.72%	15.28%	20.83%	30.56%
	RF	2.78%	5.56%	12.50%	15.28%	26.39%	37.50%
	LDA	13.89%	19.44%	40.28%	45.83%	51.39%	50.00%
	SVM	12.50%	23.61%	45.83%	55.56%	61.11%	62.50%
	KNN	9.86%	19.72%	47.89%	53.52%	56.34%	54.93%
WIN	PRF	-1.23%	2.47%	1.23%	9.88%	18.52%	16.05%
	RF	-0.95%	0.76%	10.68%	29.30%	40.45%	46.12%
	LDA	30.66%	39.06%	62.08%	58.77%	51.04%	49.91%
	SVM	25.92%	45.92%	65.07%	66.20%	66.20%	66.20%
	KNN	30.48%	38.57%	57.05%	64.67%	61.90%	62.38%
DBC	PRF	1.13%	2.26%	6.02%	9.40%	15.41%	15.79%
	RF	1.09%	3.28%	16.79%	33.94%	43.07%	54.74%
	LDA	35.40%	50.00%	51.46%	51.46%	46.35%	46.35%
	SVM	29.71%	34.42%	35.14%	35.14%	35.14%	35.14%
	KNN	40.82%	51.69%	61.42%	59.55%	59.18%	60.30%
PBC	PRF	-1.32%	0.00%	3.95%	0.00%	0.00%	0.00%
	RF	12.25%	9.15%	9.97%	16.18%	0.98%	19.44%
	LDA	3.59%	32.03%	21.24%	33.66%	24.67%	14.38%
	SVM	11.11%	7.41%	11.11%	11.11%	11.11%	11.11%
	KNN	20.27%	22.64%	18.58%	22.97%	22.97%	19.76%

Moreover, according to the trend lines in Figure 2 and 3, the four variations of the proposed algorithm follow a similar tendency among themselves in the presence of inconsistencies. In addition, in terms of the average classification accuracy, the decision algorithm that obtains the best average result is the *average tree with majority voting* (PRF_{ATMV}), which is widely used in ensemble decision trees. The only difference from the ATMV method is the need to first calculate an average for each decision tree since in this approach there is no single path to be covered. In this respect, the ATMV algorithm is also more straightforward computationally because it requires fewer iterations compared to the proposed decision algorithms in the decision making. The computation time of the $\text{ParaExtr}_{\text{ctr}}$ algorithm [16] directly depends on the size of the tree; hence, making a final decision with the forest can increase the execution time substantially without necessarily improving the classification accuracy.

Furthermore, the structure of the tree is heavily dependent on the fuzzy domain partition; hence, the definition of the fuzzy sets can directly affect the accuracy of the created tree. Therefore, further research into fuzzy domain partitioning may improve the result of the algorithm. However, the improved performance in dealing with inconsistent data, represented by noise and missing values, is observed in the plots in Figure 2 and 3, in which the PRF has a less abrupt slope compared to the other algorithms.

C. STATISTICAL ANALYSIS

To validate the experiments, an analysis of variance (ANOVA) and multiple comparisons were used. The ANOVA provides a statistical test for determining whether the differences in a factor's means are significant or not. In the design of three-factor experiments, the mean accuracy is regarded as the response variable. In relation to the controllable factors, the classification methods, the percentage of inconsistent data and the database were employed. Consider the completely randomized three-factor experiment with underlying model (13):

$$y_{ijk} = \mu + \alpha_i + \beta_j + \gamma_k + \alpha\beta_{ij} + \alpha\gamma_{ik} + \beta\gamma_{jk} + \varepsilon_{ijk} \quad (13)$$

where μ is the overall mean effect, α_i is the ith effect of factor A (classification methods), β_j is the jth effect of factor B (percentage of inconsistent data), γ_k is the kth effect of factor C (database) and ε_{ijk} is a random error component, which follows a normal distribution with zero mean and variance σ^2.

A confidence level of 95% was considered for the F-test. According to the analysis of variance for a three-factor experiment, all the main effects and their combination are significant. Thus, there is a distinction among the average accuracies of the classification methods. Further analyzing the difference among the means using the Tukey method with 95% confidence, we conclude that the paraconsistent random forest methods have significantly outperformed classic algorithms LDA, KNN and SVM when introducing missing data in the dataset, as presented in Table 7. The RF method obtained similar average accuracy when compared to the PRF method for the same scenario. As detailed previously, when data is absent, the evidence degree for the missing attribute is defined as 0.5 according to the principles of paraconsistent logic; thus, for every missing feature the attributed value is the same for the PRF method. However, in the random forest approach, the solution for dealing with missing data is to return the most probable class considering the larger number of training samples that reached that specific node. According to the analysis of the means, both approaches obtained average accuracies without significant distinction; hence, both methods PRF and RF present a satisfactory solution for dealing with missing data with an accuracy at least 20% higher in relation to the methods LDA, SVM and KNN.

Table 8 lists the comparison of the means (Tukey method) with 95% of confidence among the classification methods considering the introduction of noise in the database. It's possible to affirm that the PRF method and their variations were less susceptible to noise in terms of classification accuracy compared to the algorithms RF, SVM KNN and LDA.

Moreover, the statistical analysis revealed that there are no significant distinctions among the means for all four PRF decision basis variations when introducing missing and noisy values in the dataset. Thus, as there are no significant differences in the classification accuracy, the best decision method for PRF is ATMV because of its lower computational complexity, as discussed previously.

D. DISCUSSION

In the literature, others methods have been proposed with the objective to reducing noise susceptibility, such as fuzzy

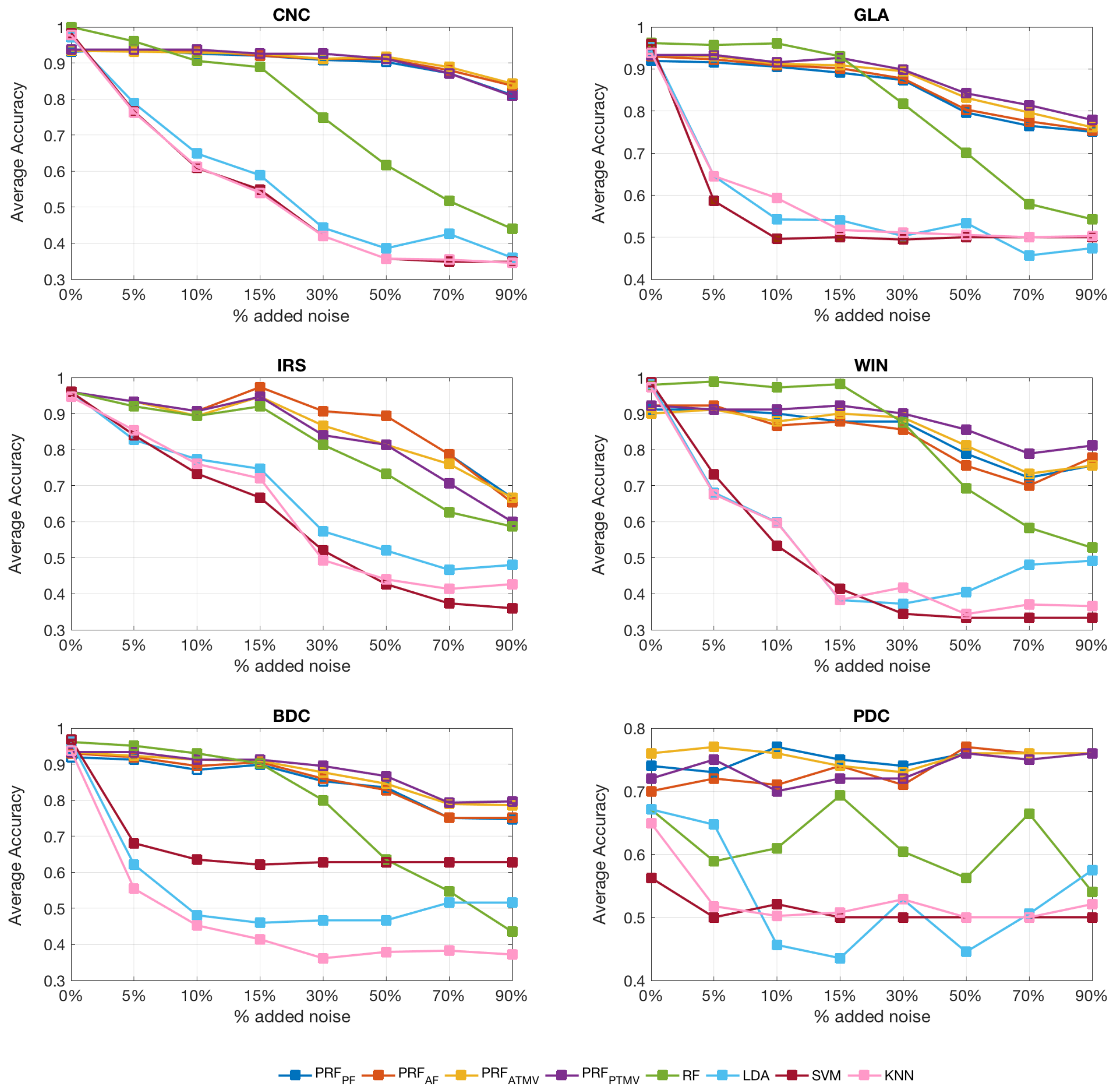

FIGURE 3. Average classification accuracy graph comparison for the PRF algorithms and LDA, SVM and KNN with the addition of noise.

TABLE 7. Comparison of the means with 95% of confidence among the classification methods considering the missing values.

Methods	Samples	Means	Group†	
RF	42	85,79%	A	
PRF_{ATMV}	42	85,16%	A	
PRF_{PTMV}	42	84,65%	A	
PRF_{PF}	42	84,56%	A	
PRF_{AF}	42	82,40%	A	
LDA	42	60,31%		B
KNN	42	59,73%		B
SVM	42	58,11%		B

† Means that do not share a letter are significantly distinct.

TABLE 8. Comparison of the means with 95% of confidence among the classification methods considering the noisy values.

Methods	Samples	Means	Group†		
PRF_{ATMV}	48	85,30%	A		
PRF_{PTMV}	48	85,17%	A		
PRF_{PF}	48	84,59%	A		
PRF_{AF}	48	84,44%	A		
RF	48	76,41%		B	
LDA	48	57,80%			C
KNN	48	56,88%			C
SVM	48	54,87%			C

† Means that do not share a letter are significantly distinct.

random forest (FRF) [25]. FRF uses a fuzzy decision tree approach and outperforms well-known methods on data with missing values. Comparing the results that are obtained with the proposed PRF method with those of FRF, on databases IRS and WIN, PRF outperformed FRF [25] in data classification with missing values, as presented in Table 9.

A larger distinction can be observed with 30% missing data, in which the PRF method obtained a decrease in

TABLE 9. Algorithm comparison with the fuzzy random forest method.

Dataset	Method	Original classification accuracy	% decrease in accuracy: α% of missing values 5%	% decrease in accuracy: α% of missing values 30%
IRS	PRF	96,00%	0,00%	5,33%
	FRF [25]	97,33%	1,23%	16,71%
WIN	PRF	92,22%	0,00%	7,78%
	FRF [25]	97,88%	4,41%	14,21%

accuracy of approximately 5-7% in comparison to the FRF method, for which the decrease was approximately 14-16%. However, the FRF method obtained slightly better results in the classification of the original data. This comparison supports the hypothesis that paraconsistent random forest is a suitable algorithm in contexts that involve the treatment of incomplete data. Another difference between FRF and PRF is that the fuzzy random forest was trained and tested with the modified dataset. This distinction can lead to a reduced decrease in accuracy once the method has been presented with data of this type during model creation.

Moreover, a few methods have been proposed exploiting paraconsistent logic in pattern recognition applications. One of them is the paraconsistent neural networks (PNN), a type of ANN based on paraconsistent logic. PNN have been employed with success in the recognition of MICR (Magnetic Ink Character Recognition) characters commonly used in bank checks [16]. According to the authors, the PNN outperformed other studies in the area with an accuracy of 97.8%. PNN has also being applied, with promising results, to aid diagnosis of Alzheimer disease by analyzing EEG signals [11]. Besides the PNN, the paraconsistent logic is also employed as a network of interconnected nodes (PANnet) in the treatment of Raman spectroscopy data to support diagnosis of skin cancer [21]. The PANnet was able to classify Raman spectroscopy data with a better discrimination in comparison to conventional statistical process. However, despite the promising results achieved by the use of paraconsistent logic in pattern recognition, all the cited researches used classification models that were manually defined and application specific; hence, it was not possible to perform a benchmark with a standard dataset for comparison between distinct approaches of the paraconsistent logic.

Nonetheless, a discriminative paraconsistent machine (DPM) was proposed by [19]. Contrary to the other methods presented in the literature, DPM is a versatile method based on a supervised training discriminative model that incorporates paraconsistency criteria. According to the authors, the DPMs can be applied in innumerous fields, but it lacks the evaluation of the algorithm with a standard database for benchmarking purposes. Furthermore, the DPM was applied to the classification of pathologies in voice data obtaining similar accuracies to the traditional method SVM.

It is important to emphasize that the use of paraconsistent logic in pattern recognition application is still recent in the scientific community. Hence, it is difficult to perform a more detailed comparison with the proposed method PRF, since many researches are focused in the application and do not execute an evaluation of the algorithm using standardized dataset.

Furthermore, it is clearly observed from the literature that one of the possible applications that could benefit from the PRF method is the classification of biosignals, as the motion recognition thought myoelectric signals. Among the reported issues in the motion classification is the model degradation in the presence of signal contaminants. To overcame this concern, several approaches were proposed as automatic model retraining [4], [10], [45] and signal replacement by mathematical models [7]. However, these solutions are not always viable to commercial use since they demand a high computational time and memory usage. Hence, the PRF seems to be an interesting alternative for its low parametrization and its superior results in dealing with uncertain data.

V. CONCLUSION

In this paper, the ID3 algorithm was modified to enable it to deal with inconsistent datasets by applying paraconsistent logic and random forests. The new approach is called paraconsistent random forest. The proposed method combines the well-known decision trees with the robustness of the paraconsistent logic in dealing with inconsistent/incomplete datasets. It was demonstrated that a hybridization of random forest and paraconsistent trees presents advantages that enable the algorithm to be more efficient when dealing with the noisy, incomplete data that are typical of advanced practical applications.

A complete algorithm was presented, starting from data partitioning, building the paraconsistent tree and application of the random forest approach to infer the winning class with an associated certainty degree. Moreover, various methods for combining the outputs of each paraconsistent tree in the ensemble into a single classification were defined. Those methods are based on algorithms that are frequently used in the literature to obtain the final decision in ensembles and on paraconsistent logic.

In the results section, several experimental procedures were employed to analyze the feasibility and robustness of the method. The results were compared with those of other algorithms and missing and noisy data were incorporated into the selected datasets. The PRF ensemble outperformed the other methods on imperfect datasets. PRF was also compared to FRF, which is an algorithm that was proposed in the literature and also aims at solving the problem of imperfect datasets. PRF outperformed FRF in dealing with missing information in the dataset. Thus, according to the results, the PRF ensemble presented itself as a promising versatile classifier with superior capability to deal with inconsistent inputs.

Moreover, new studies are suggested for evaluating the proposed algorithm in contexts where it is necessary to deal with uncertain data. For instance, in algorithms applied to biological signals, new studies on the applicability of paraconsistent random forest may consider the inherent inconsistencies that are due to the presence of high levels of noise and the lack of adequate signal acquisition, thereby resulting in superior classification performance and increased robustness.

REFERENCES

[1] A. J. Young, L. J. Hargrove, and T. A. Kuiken, "Improving myoelectric pattern recognition robustness to electrode shift by changing interelectrode distance and electrode configuration," *IEEE Trans. Biomed. Eng.*, vol. 59, no. 3, pp. 645–652, Mar. 2012.

[2] Y. Gu, D. Yang, Q. Huang, W. Yang, and H. Liu, "Robust EMG pattern recognition in the presence of confounding factors: Features, classifiers and adaptive learning," *Expert Syst. Appl.*, vol. 96, pp. 208–217, Apr. 2018.

[3] Q. Huang, D. Yang, L. Jiang, H. Zhang, H. Liu, and K. Kotani, "A novel unsupervised adaptive learning method for long-term electromyography (EMG) pattern recognition," *Sensors*, vol. 17, no. 6, p. 1370, 2017.

[4] V. H. Cene, G. Favieiro, and A. Balbinot, "Upper-limb movement classification based on sEMG signal validation with continuous channel selection," in *Proc. 37th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBC)*, vol. 37, Aug. 2015, pp. 486–489.

[5] A. A. Adewuyi, L. J. Hargrove, and T. A. Kuiken, "An analysis of intrinsic and extrinsic hand muscle EMG for improved pattern recognition control," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 24, no. 4, pp. 485–494, Apr. 2016.

[6] C. Ahmadizadeh, L.-K. Merhi, B. Pousett, S. Sangha, and C. Menon, "Toward intuitive prosthetic control: Solving common issues using force myography, surface electromyography, and pattern recognition in a pilot case study," *IEEE Robot. Autom. Mag.*, vol. 24, no. 4, pp. 102–111, Dec. 2017.

[7] K. De O. A. De Moura and A. Balbinot, "Virtual sensor of surface electromyography in a new extensive fault-tolerant classification system," *Sensors*, vol. 18, no. 5, p. 1388, May 2018.

[8] P. McCool, G. D. Fraser, A. D. C. Chan, L. Petropoulakis, and J. J. Soraghan, "Identification of contaminant type in surface electromyography (EMG) signals," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 22, no. 4, pp. 774–783, Jul. 2014.

[9] S. Muceli, N. Jiang, and D. Farina, "Extracting signals robust to electrode number and shift for online simultaneous and proportional myoelectric control by factorization algorithms," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 22, no. 3, pp. 623–633, May 2014.

[10] O. W. Samuel, M. G. Asogbon, Y. Geng, A. H. Al-Timemy, S. Pirbhulal, N. Ji, S. Chen, P. Fang, and G. Li, "Intelligent EMG pattern recognition control method for upper-limb multifunctional prostheses: Advances, current challenges, and future prospects," *IEEE Access*, vol. 7, pp. 10150–10165, 2019.

[11] J. M. Abe, H. F. S. Lopes, and K. Nakamatsu, "Paraconsistent neurocomputing and brain signal analysis," *Vietnam J. Comput. Sci.*, vol. 1, no. 4, pp. 219–230, Nov. 2014.

[12] E. J. Rechy-Ramirez and H. Hu, "Bio-signal based control in assistive robots: A survey," *Digit. Commun. Netw.*, vol. 1, no. 2, pp. 85–101, Apr. 2015.

[13] M. S. Roodposhti, J. Aryal, H. Shahabi, and T. Safarrad, "Fuzzy Shannon entropy: A hybrid GIS-based landslide susceptibility mapping method," *Entropy*, vol. 18, no. 10, p. 343, 2016.

[14] W. Carnielli and M. E. Coniglio, *Paraconsistent Logic: Consistency, Contradiction and Negation*, 1st ed. Cham, Switzerland: Springer, 2016.

[15] B. Dunin-Kęplicz, A. Strachocka, A. Szałas, and R. Verbrugge, "Paraconsistent semantics of speech acts," *Neurocomputing*, vol. 151, pp. 943–952, Mar. 2015.

[16] S. Souza, J. M. Abe, and K. Nakamatsu, "MICR automated recognition based on paraconsistent artificial neural networks," *Procedia Comput. Sci.*, vol. 22, pp. 1083–1091, Dec. 2013.

[17] J. M. Abe, "Paraconsistent logics and applications," in *Proc. Int. Workshop Soft Comput. Appl.*, vol. 4, 2010, pp. 11–18.

[18] J. I. da Silva Filho, "Treatment of uncertainties with algorithms of the paraconsistent annotated logic," *J. Intell. Learn. Syst. Appl.*, vol. 4, no. 2, pp. 144–153, 2012.

[19] R. C. Guido, S. Barbon Jr., R. D. Solgon, K. C. S. Paulo, L. C. Rodrigues, I. N. da Silva, and J. P. L. Escola, "Introducing the discriminative paraconsistent machine (DPM)," *Inf. Sci.*, vol. 221, pp. 389–402, Feb. 2013.

[20] S. B. Júnior, R. C. Guido, and L. S. Vieira, "A neural-network approach for speech features classification based on paraconsistent logic," in *Proc. 11th IEEE Int. Symp. Multimedia*, San Diego, CA, USA, Dec. 2009, pp. 567–570.

[21] J. I. Da Silva Filho, C. V. Nunes, D. V. Garcia, M. C. Mario, F. Giordano, J. M. Abe, M. Tadeu T. Pacheco, and L. Silveira, Jr., "Paraconsistent analysis network applied in the treatment of Raman spectroscopy data to support medical diagnosis of skin cancer," *Med. Biol. Eng. Comput.*, vol. 54, no. 10, pp. 1453–1467, Oct. 2016.

[22] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.

[23] S. Schülter, S. Lessmann, and S. Voß, "A case study of random forest in predictive data mining," *Wirtschaftsinformatik*, vol. 2, pp. 319–328, Jan. 2009.

[24] H. Buhrman and R. de Wolf, "Complexity measures and decision tree complexity: A survey," *Theor. Comput. Sci.*, vol. 288, pp. 21–43, Oct. 2002.

[25] P. Bonissone, J. M. Cadenas, M. C. Garrido, and R. A. Díaz-Valladares, "A fuzzy random forest," *Int. J. Approx. Reason.*, vol. 51, no. 7, pp. 729–747, 2010.

[26] M. Atzori, A. Gijsberts, I. Kuzborskij, S. Elsig, A.-G. M. Hager, O. Deriaz, C. Castellini, H. Müller, and B. Caputo, "Characterization of a benchmark database for myoelectric movement classification," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 23, no. 1, pp. 73–83, Jan. 2015.

[27] A. B. Ajiboye and R. F. Weir, "A heuristic fuzzy logic approach to EMG pattern recognition for multifunctional prosthesis control," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 13, no. 3, pp. 280–291, Sep. 2005.

[28] G. W. Favieiro and A. Balbinot, "Adaptive neuro-fuzzy logic analysis based on myoelectric signals for multifunction prosthesis control," in *Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc.*, Boston, MA, USA, vol. 33, Aug./Sep. 2011, pp. 7888–7891.

[29] M. Khezri and M. Jahed, "A neuro–fuzzy inference system for sEMG-based identification of hand motion commands," *IEEE Trans. Ind. Electron.*, vol. 58, no. 5, pp. 1952–1960, May 2011.

[30] F. H. Y. Chan, Y. S. Yang, F. K. Lam, Y.-T. Zhang, and P. A. Parker, "Fuzzy EMG classification for prosthesis control," *IEEE Trans. Rehabil. Eng.*, vol. 8, no. 3, pp. 305–311, Sep. 2000.

[31] G. W. Favieiro, K. O. A. Moura, and A. Balbinot, "Novel method to characterize upper-limb movements based on paraconsistent logic and myoelectric signals," in *Proc. 38th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBC)*, Orlando, FL, USA, Aug. 2016, pp. 395–398.

[32] J. R. Quinlan, "Induction of decision trees," *Mach. Learn.*, vol. 1, no. 1, pp. 81–106, 1986.

[33] C. Z. Janikow, "Fuzzy decision trees: Issues and methods," *IEEE Trans. Syst. Man, Cybern. B, Cybern.*, vol. 28, no. 1, pp. 1–14, Feb. 1998.

[34] M. Umanol, H. Okamoto, I. Hatono, H. Tamura, F. Kawachi, S. Umedzu, and J. Kinoshita, "Fuzzy decision trees by fuzzy ID3 algorithm and its application to diagnosis systems," in *Proc. IEEE Int. Fuzzy Syst. Conf.*, Orlando, FL, USA, Jun. 1994, pp. 2113–2118.

[35] T. G. Dietterich, "An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization," *Mach. Learn.*, vol. 40, pp. 139–157, Aug. 2000.

[36] P. Cobreros, "Paraconsistent vagueness: A positive argument," *Synthese*, vol. 183, no. 2, pp. 211–227, Nov. 2011.

[37] J. I. da Silva Filho, J. M. Abe, and G. L. Torres, *Inteligência Artificial com redes de Análise Paraconsistentes*, 1st ed. Rio de Janeiro, Brazil: LTC, 2008.

[38] N. C. A. da Costa, "On the theory of inconsistent formal systems," *Notre Dame J. Formal Logic*, vol. 15, no. 4, pp. 497–510, Oct. 1974.

[39] F. Enembreck, B. C. Avila, and R. Sabourin, "Decision tree-based paraconsistent learning," in *Proc. 19th Int. Conf. Chilean Comput. Sci. Soc. (SCCC)*, 1999, pp. 43–52.

[40] R. Ertola, F. Esteva, T. Flaminio, L. Godo, and C. Noguera, "Paraconsistency properties in degree-preserving fuzzy logics," *Soft Comput.*, vol. 19, no. 3, pp. 531–546, 2014.

[41] S. L. Chiu, "Fuzzy model identification based on cluster estimation," *J. Intell. Fuzzy Syst.*, vol. 2, no. 3, pp. 267–278, 1994.

[42] J. I. Da Silva Filho, "Paraconsistent algorithm extractor of contradiction effects—Paraextrctr," (in Portuguese), *Seleção Doc.*, vol. 15, no. 4, pp. 21–25, 2009.

[43] D. Dua and C. Graff, "UCI machine learning repository," School Inf. Comput. Sci., Univ. California, Irvine, CA, USA, 2019. [Online]. Available: https://archive.ics.uci.edu/ml/citation_policy.html

[44] A. Phinyomark, F. Quaine, S. Charbonnier, C. Serviere, F. Tarpin-Bernard, and Y. Laurillau, "EMG feature evaluation for improving myoelectric pattern recognition robustness," *Expert Syst. Appl.*, vol. 40, no. 12, pp. 4832–4840, 2013.

[45] X. Zhang, H. Huang, and Q. Yang, "Real-time implementation of a self-recovery EMG pattern recognition interface for artificial arms," in *Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBC)*, Osaka, Japan, vol. 35, Jul. 2013, pp. 5926–5929.

[46] X. Zhang and H. Huang, "A real-time, practical sensor fault-tolerant module for robust EMG pattern recognition," *J. Neuroeng. Rehabil.*, vol. 12, no. 1, 2015, Art. no. 18.

GABRIELA W. FAVIEIRO was born in Porto Alegre, Brazil, in 1986. She received the B.E. degree in computer engineering and the M.Sc. degree in electrical engineering from the Federal University of Rio Grande do Sul (UFRGS), Brazil, in 2010 and 2012, respectively, where she is currently pursuing the Ph.D. degree in electrical engineering. Her research interests include biomedical signal processing, instrumentation, and artificial intelligence, mainly focused on upper-limb myoelectric signal processing, with several published articles in the area.

ALEXANDRE BALBINOT received the B.E. degree in electrical engineering from PUCRS, in 1993, the Specialist degree in clinical engineering from UFRGS, in 1995, the M.S. degrees in engineering (biomedical-biomechanical instrumentation area) from PUCRS, in 1997, and UFRGS, in 1998, respectively, and the Ph.D. degree in engineering from UFRGS, in 2001. He is currently an Associate Professor with the EE Department, UFRGS, where he has been coordinating research in areas of instrumentation, design and analysis of experimental data, biomedical instrumentation, signal processing, assistive technology, and high-performance sports machines. He is also a Federal Consultant for diverse governmental agencies and a Reviewer for several scientific journals.