A Co-Training-based Algorithm Using Confidence Values to Select Instances

Karliane M. O. Vale, Flavius L. Gorgônio, Yago N. Araújo
Dept. of Computing and Technology
Federal University of Rio Grande do Norte
Caicó, Brazil
{karliane, flavius}@dct.ufrn.br, yagonobreg@gmail.com

Arthur C. Gorgônio, Anne Magály de P. Canuto
Dept. of Informatics and App. Mathematics
Federal University of Rio Grande do Norte
Natal, Brazil
gorgonioarthur@gmail.com, anne@dimap.ufrn.br

***Abstract*—Data classification tasks have been used in a wide range of problems in the Machine Learning field and the different learning paradigms (supervised, unsupervised or semi-supervised) define the task a computer can learn from a set of labeled and/or unlabeled data. This paper presents a study in the semi-supervised learning paradigm and proposes changes on the co-training algorithm in order to propose a confidence value procedure to include new instances in the labeled dataset. In order to evaluate the proposed method, an empirical analysis with 30 datasets has been conducted, with different characteristics, that were set up with different percentages of initially labeled instances. Each dataset was trained using four different classification algorithms (Naive Bayes, Decision tree, Ripper and *k*-NN) as basis for the co-training training procedure. The obtained results are promising and they indicate that, in most cases, the proposed method performs better than the co-training method originally proposed in the literature.**

***Index Terms*—Data classification, Semi-supervised learning, Co-training algorithm.**

I. Introduction

Machine learning is a research field that aims to develop computational algorithms to automate tasks and, consequently, to reduce the need for human and specialist intervention in tasks that can be automated. According to the type of data used in the machine learning process, we have three learning paradigms: unsupervised learning (in which all the instances are unlabeled), supervised learning (in which all the instances are labeled) and semi-supervised learning (in which some part of the instances is labeled) [1].

Data classification, a common task in machine learning that has been applied to a wide range of problems, uses primarily supervised learning algorithms. However, a natural limitation of such algorithms is that they need to have a set of labeled instances with a reasonable size in order to achieve a reasonable performance. In this case, the need for a large number of training instances is undeniably a critical problem. Given that the cost of manually labeling instances is usually high and it is a time-consuming process, one way to smooth out this problem is through a process of training classifiers with a small amount of labeled data and a large amount of unlabeled data, which is known as semi-supervised learning (SSL).

Semi-supervised learning algorithms use labeled instances to build their initial hypothesis and combine the information obtained from these instances in order to label the unlabeled instances. In other words, it is possible to use partially-supervised information to guide the learning process and to increase the amount of evidence of the target labels [2].

There are several semi-supervised learning models and algorithms described in the literature, including self-training, co-training and multi-view learning, mixture models, graph-based methods, and semi-supervised support vector machines [3]. The training process for semi-supervised learning algorithms is similar to any model. Initially, a classifier is trained from a reduced number of labeled instances. This classifier is then used to classify instances that have not yet been labeled, which are added to the training dataset. Finally, the classifier is retrained using the modified dataset and the whole procedure is repeated [1].

The two most used SSL algorithms are self-training and co-training and the labeling process for these algorithms is similar [4]. The main difference between co-training and other algorithm is that the former uses an approach based on the idea of multi-description. In its execution, training instances are described by two or more sets of disjoint attributes, i.e., through different describing views. Nevertheless, each view is sufficient to train one or more classifiers [5].

However, we have observed that the process of automatic assignment of labels is still a difficult task. The main question is related to the random choice of the unlabeled instances to be labeled. In order to smooth out the problem cited above, Rodrigues *et al.* [6] proposed a confidence parameter to guide the labeling process of the self-training method. According to the authors, the main idea is to minimize the randomness in which the instances are chosen during the labeling process.

In this paper, we will address the limitation of the co-training algorithm related to the selection criterion used to decide the instances to be labeled in each iteration. Our objective is to present a more robust extension of the co-training algorithm, named Co-Training with Fixed Threshold - CTFT, which uses a confidence threshold as a selection criterion of the unlabeled instances. The main advantage of CTFT is that, during the labeling process, this method selects only instances whose confidence in prediction are higher than or equal to the threshould, so that they are considered reliable. In order to evaluate the feasibility of the proposed strategy,

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

an empirical analysis has been conducted. In this analysis, the proposed algorithm (with threshold) was compared to the original co-training, using 30 classification datasets.

This paper is divided into seven sections as follows. Sections II and III describe the fundamental concepts and some research related to the subject of this paper, while a detailed description of the proposed method is illustrated in Section IV. In Section V, the methodology used in the empirical analysis is described. Section VI presents the results provided by the empirical analysis and some discussions about them. Finally, Section VII presents some conclusion.

II. Semi-supervised Learning

As the name itself suggests, semi-supervised learning is situated between the supervised and unsupervised learning paradigms, given that it works with partially labeled data [3]. The difference between these paradigms is related to the way in which the knowledge generalization process is carried out. While supervised learning uses instances whose classes (labels) are previously known, in unsupervised learning such classes are unknown [7]. Therefore, the definition of the supervised learning task consists of training classifiers based on previously known classification instances. However, in real-world classification tasks, you can find datasets in which only part of the data is labeled, while the rest has no labels. The semi-supervised learning mechanism proposes to treat data with such characteristics in order to achieve a better classification [8].

Semi-supervised learning considers the set D of instances as being divided into two sets: 1) the labeled data $\{D_L\} = \{(\mathbf{x}_i, y_i) | i = 1, \cdots, l\}$, so that $\mathbf{x}$ is an instance, y is the label for this instance $\mathbf{x}$ and l is the number of labeled instances; 2) the unlabeled data $\{D_U\} = \{(\mathbf{x}_j) | j = l+1, \cdots, l+u\}$, so that $\mathbf{x}$ is an instance and u is the amount of unlabeled instances. Often, it is assumed that $|D_U| \gg |D_L|$.

One of the advantages of semi-supervised learning is the potential to reduce the need for large amounts of labeled data in domains in which only a small set of labeled instances is available [9]. An additional advantage of this learning paradigm can be verified in scenarios in which an expert does not have a complete knowledge about the concept to be learned. In other words, this expert only has knowledge concerning some instances of a given dataset, having, therefore, some difficulties of labeling instances to increment the training dataset. Several researches have been developed addressing the semi-supervised learning, either to solve problems [10]–[12], or to create extensions or new algorithms [13]–[16]. In this work, the co-training algorithm has been modified in order to enable changes in the process of inclusion of new instances in the labeled dataset.

The co-training algorithm, like the other semi-supervised learning algorithms, increases the labeled set of data by iteratively classifying the set of unlabeled data and moving the most reliable prediction instances to the labeled set of data [17]. In this method, two complementary classifiers are generated simultaneously, fed with two different views of the set of attributes: $\mathbf{x}_i = [\mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)}]$, where $\mathbf{x}_i$ is an instance, $\mathbf{x}_i^{(1)}$ is an instance view, containing a subset of attributes of $\mathbf{x}_i$ and $\mathbf{x}_i^{(2)}$ is the other instance view, containing the remaining attributes of the $\mathbf{x}_i$, which have not been used in $\mathbf{x}_i^{(1)}$. Once each classifier provides its prediction, their outputs are combined using a voting strategy (or any other combination strategy). In parallel, after generating these two groups from the data, the labeled dataset needs to be increased. In order to do this, the prediction of the first classifier is used to increase the labeled set available for the second classifier, and vice versa [18]. In other words, the prediction of one classifier is included in the labeled set of the other classifier.

Figure 1 shows the flow of the labeling process for co-training. As it can be observed in this figure, at the beginning, two views are created, in which two supervised classifiers are generated using the two different views (C_1 and C_2) of the initially labeled data. The next step is to classify the unlabeled data, using classifier C_1 to label based on view 1 and C_2 to label based on view 2. Finally, the instances with highest confidence in each classifier are selected and added to the set of labeled data of the other classifier. This process is repeated until the set of unlabeled data is empty. Algorithm 1 describes the steps performed to carry out this process in the co-training algorithm.

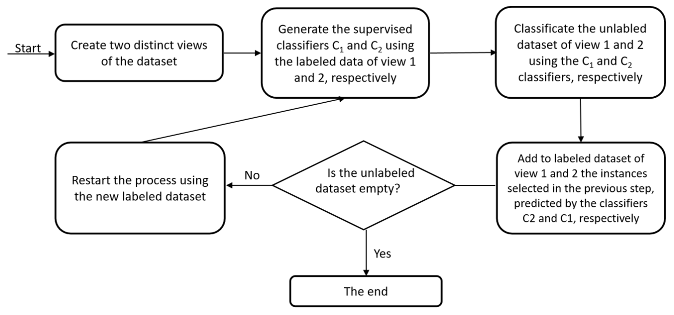

Start
Create two distinct views of the dataset
Generate the supervised classifiers C1 and C2 using the labeled data of view 1 and 2, respectively
Classificate the unlabled dataset of view 1 and 2 using the C1 and C2 classifiers, respectively
Add to labeled dataset of view 1 and 2 the instances selected in the previous step, predicted by the classifiers C2 and C1, respectively
Is the unlabeled dataset empty?
No
Restart the process using the new labeled dataset
Yes
The end

Fig. 1. Labeling process in co-training.

As it can be observed, in the co-training algorithm, two classification algorithms are used in order to provide the predictions of the unlabeled instances. Among the classification algorithms usually used in the literature, some of the most significant are Naive Bayes (NB), Decision tree (DT), rule based classification algorithm (ripper) and *k*-Nearest Neighbor (*k*-NN. The Naive Bayes method is a probabilistic classifier based on applying Bayes' theorem that assumes the attributes are conditionally independent [19]. The Decision tree classifier is based on a divide-and-conquer strategy to solve a decision problem [19]. The ripper algorithm is a propositional rule learning algorithm that performs efficiently on large noise datasets, naturally extends first-order representation and are competitive with Decision tree in generalization performance [20]. The *k*-NN algorithm classifies a new instance based on the class that appears most frequently among the closest instances and, to achieve this, it usually considers the Euclidean distance [21]. These four classification algorithms will be used in the empirical analysis to be performed in this paper.

Algorithm 1: Co-training algorithm

```
1  Input: labeled data $\{D_L\} = \{(x_i, y_i)|i = 1 \cdots l\}$,
     unlabeled data $\{D_U\} = \{(x_j)|j = l+1 \cdots l+u\}$, a
     learner named $k$.
2  begin
3      Every instance has two views $X_i = [X_i^{(1)}, X_i^{(2)}]$.
4      Initially, we have the training instances as
         $D_L^{(1)} = \{(X_i^{(1)}, Y_i), ..., (X_l^{(1)}, Y_l)\}$ and
         $D_L^{(2)} = \{(X_i^{(2)}, Y_i), ..., (X_l^{(2)}, Y_l)\}$ as well as
5      the unlabeled instances as
         $D_U^{(1)} = \{(X_i^{(1)}), ..., (X_l^{(1)})\}$ e
         $D_U^{(2)} = \{(X_i^{(2)}), ..., (X_l^{(2)})\}$
6      where,
7      $\{D_L\} = D_L^{(1)} \cup D_L^{(2)}$ e $D_L^{(1)} - D_L^{(2)} = \emptyset$
8      $\{D_U\} = D_U^{(1)} \cup D_U^{(2)}$ e $D_U^{(1)} - D_U^{(2)} = \emptyset$
9      repeat
10         Generate classifiers $C^{(1)}$ and $C^{(2)}$ from
             training data $D_L^{(1)}$ e $D_L^{(2)}$, respectively.
11         Classify the unlabeled data $D_U^{(1)}$ and $D_U^{(2)}$
             using the classifiers $C^{(1)}$ and $C^{(2)}$,
             respectively.
12         Add the first $k$ instances with highest
             confidence prediction classified by $C^{(1)}$ to set
             $D_L^{(2)}$.
13         Add the first $k$ instances with highest
             confidence prediction classified by $C^{(2)}$ to set
             $D_L^{(1)}$.
14         Remove those instances from the unlabeled
             dataset.
15     until $\{D_U\} = \emptyset$;
16 end
17 Output: labeled data
```

III. Related Work

In this section, some studies are described in which co-training is used with the goal of solving problems. In [10], for instance, a semi-supervised approach was proposed that adapts active learning to the co-training algorithm to classify hyperspectral images, automatically selecting new training samples of unlabeled pixels. The effectiveness of the proposed approach is validated using a probabilistic support vector machine classifier.

Research [11] is aimed at designing a semi-supervised learning model for the NER system (Indonesian Named Entity Recognition). The NER system aims to identify and to classify an entity based on its context, however few instances have a label. Thus, the semi-supervised learning model co-training was used to deal with unlabeled data in the NER learning process and to produce new labeled data that can be applied to improve a new NER classification system. Another study with co-training can be found in [12], in which the authors proposed a new approach for classifying students in an academic credit system, combining transfer learning and co-training. The resulting model can effectively predict the study status of a student enrolled in an educational program, using a classification model enhanced by transfer learning techniques and co-training in educational data from another program. In addition, this approach can address the scarcity of sets of educational data for early prediction of students with problems.

There are, additionally, some studies, similar to this one, which extend co-training or use it to create new algorithms. In research [13], for instance, Deep Co-training was created, which consists of a method based on deep learning (inspired by the structure of co-training). Deep Co-training trains several deep neural networks to be used with the different views necessary for the functioning of co-training and explores contradictory examples to encourage differences among the views, in order to prevent the neural networks to collapse with one another. In [14] a method named multi-co-training was proposed, which aimed at improving the performance of a document classification system. Documents are transformed using three document representation methods in order to increase the variety of the sets of attributes for classification.

In [15], a semi-supervised learning algorithm was introduced, which combines co-training with the support vector machine (SVM) classification algorithm. By means of an interactive learning procedure, the new final set of labeled data can be determined based on sets of unlabeled data, training two SVM classifiers. Additionally, in [16], the authors proposed a new object recovery algorithm based on co-training, named co-transduction. The objective of this work was to develop an algorithm to merge different measures of similarity for robust object recovery through a semi-supervised learning structure. Given two similarity measures and a shape to be queried, the proposed algorithm iteratively retrieves the most similar shapes using one measure and assigns them to a pool for the other measure to make a new classification and vice versa.

Finally, it is worthwhile to mention that the algorithm presented in this work was inspired by the extension proposed in [6], which used a confidence parameter in the self-training labeling process to minimize the inclusion of noise and to improve, in general, the accuracy of the classification tasks. Therefore, only instances whose output labels yielded by the classifier have confidence values above a confidence threshold are taken into account. In this sense, this confidence threshold was used to define the automatic label assignment in the semi-supervised learning process.

IV. The Proposed Co-training Method

The general steps of the extension of the co-training algorithm proposed in this work, named Co-Training with Fixed Threshold (CTFT), is shown in Figure 2. In this figure, the dashed rectangle in the upper right corner represents the main difference between this proposal and the original co-training. As in the original co-training, CTFT begins by creating two distinct views (1 and 2) of the data set. Then, two supervised classifiers (C_1: view 1 and C_2: view 2) are generated using the sets of labeled data as training set. As mentioned previously, these two classifiers are used to classify the unlabeled data. In the following step, the instances whose confidence value in the prediction is higher than or equal to a confidence threshold defined for each classifier are selected and labeled using the prediction delivered by the classifier output combination. Then, the unlabeled instances selected in the previous step, predicted by the classifiers C_2 and C_1, are added to the set of labeled data of both classifiers, using the co-training inclusion procedure (described in Section II). Finally, the process restarts using the new sets of labeled data and iterates until the sets of unlabeled data are empty or there are no instances whose confidence in the prediction is higher than or equal to the minimum accepted confidence.

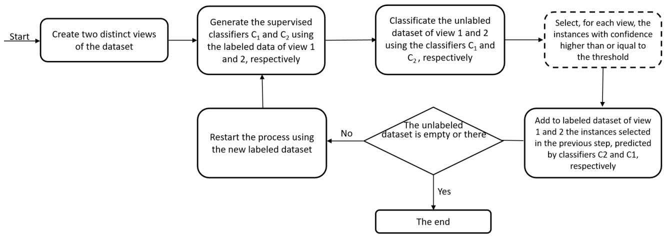

Start
Create two distinct views of the dataset
Generate the supervised classifiers C1 and C2 using the labeled data of view 1 and 2, respectively
Classificate the unlabled dataset of view 1 and 2 using the classifiers C1 and C2, respectively
Select, for each view, the instances with confidence higher than or iqual to the threshold
Add to labeled dataset of view 1 and 2 the instances selected in the previous step, predicted by classifiers C2 and C1, respectively
The unlabeled dataset is empty or there
No
Restart the process using the new labeled dataset
Yes
The end

Fig. 2. The general steps of Co-Training with Fixed Threshold (CTFT)

As already mentioned, in this work, we developed an extension of the co-training algorithm following the same methodology described in [6]. In other words, we included a new parameter, called confidence threshold, which is responsible for selecting instances in the labeling process. In the original co-training algorithm, the number of instances to be included in the labeling was also a parameter, k (steps 13 and 14 of Algorithm 1). However, it is a static parameter that always selects the same number of instances to be labeled and it is the same value for both classifiers. However, as classifiers C_2 and C_1 have different views of the same dataset, they may have different behavior (performance). Additionally, their behavior may change throughout the labeling process. Therefore, the use of a single static parameter may not be an efficient selection for the labeling process. Aiming at making this process more robust, we introduce the confidence threshold, which defines the accepted minimum confidence that an instance must have in order to be included in the labeled set. This confidence threshold (CT) is defined for each classifier and a different number of unlabeled instances may be selected to be included in the labeled set at each iteration. As a consequence, in this proposed algorithm, the labeling process may not label all unlabeled instances, but only those whose confidence thresholds are higher than the minimum accepted confidence.

Algorithm 2 presents the main steps of the CTFT. In this algorithm, the lines 12 and 13 represent the differences between this algorithm and the original co-training (Algorithm 1).

Algorithm 2: Co-training algorithm using a fixed threshold

```
1  Input: labeled data $\{D_L\} = \{(x_i, y_i)|i = 1 \cdots l\}$,
       unlabeled data $\{D_U\} = \{(x_j)|j = l+1 \cdots l+u\}$,
       minimal confidence $mc$.
2  begin
3      Each instance has two views $X_i = [X_i^{(1)}, X_i^{(2)}]$.
4      Initially we have training instances as
         $D_L^{(1)} = \{(X_i^{(1)}, Y_i), ..., (X_l^{(1)}, Y_l)\}$ and
         $D_L^{(2)} = \{(X_i^{(2)}, Y_i), ..., (X_l^{(2)}, Y_l)\}$ and
5      unlabeled data as $D_U^{(1)} = \{(X_i^{(1)}), ..., (X_l^{(1)})\}$ and
         $D_U^{(2)} = \{(X_i^{(2)}), ..., (X_l^{(2)})\}$
6      where,
7      $\{D_L\} = D_L^{(1)} \cup D_L^{(2)}$ and $D_L^{(1)} - D_L^{(2)} = \emptyset$
8      $\{D_U\} = D_U^{(1)} \cup D_U^{(2)}$ and $D_U^{(1)} - D_U^{(2)} = \emptyset$
9      repeat
10         Generate the classifiers $C^{(1)}$ and $C^{(2)}$ from
             training data $D_L^{(1)}$ and $D_L^{(2)}$, respectively.
11         Classify unlabeled data $D_U^{(1)}$ e $D_U^{(2)}$ using the
             classifiers $C^{(1)}$ e $C^{(2)}$, respectively.
12         Add to set $D_L^{(2)}$ the instances classified by
             $C^{(1)}$, whose prediction's confidence rate is
             greater than or equal to the minimum
             confidence threshold ($CR_1$) for the inclusion
             of new instances.
13         Add to set $D_L^{(1)}$ the instances classified by
             $C^{(2)}$, whose prediction's confidence rate is
             greater than or equal to the minimum
             confidence threshold ($CR_2$) for the inclusion
             of new instances.
14         Remove those instances from the set of
             unlabeled data.
15     until $\{D_U\} = \emptyset$ or $\nexists x_i \in \{D_U\} \mid conf(x_i) \geq mc$;
16 end
17 Output: labeled data
```

V. Experimental Methodology

An empirical analysis is conducted in order to validate the feasibility of the proposed co-training algorithm. In this analysis, we used 30 different classification datasets, obtained from different repositories: UCI Machine Learning (UCI) [22], Knowledge Extraction based on Evolutionary Learning (KEEL) [23], Kaggle Datasets [24] and GitHub [25]. Table I briefly describes the datasets used, in terms of the number of instances (#Inst), attributes (#Att) and classes (#Classes) in each dataset. In addition, it also indicates the type of data (Type), whether it be integer (I) and/or categorical (C) and/or real (R).

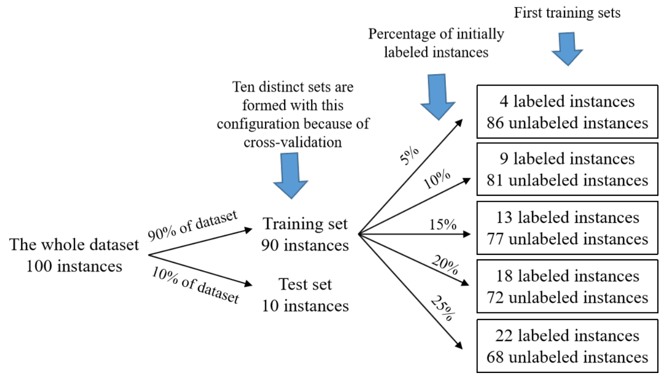
Figure 3 presents an example to illustrate the division of the dataset to be used in this analysis, considering the learning strategy used. As it can be observed in Figure 3, ten distinct sets of 90% of the total dataset for training and 10% for testing are created. This is due to the fact that we are applying a 10-fold stratified cross validation technique. It is common to have a sample of data for training and another independent sample, with different instances, for testing. As long as both samples

TABLE I
DATASETS

Dataset	***#Inst***	***#Att***	***#Class***	***Type***
Balance Scale	625	5	3	C
BTSC[1]	748	5	2	C
Bupa	345	7	2	C,I,R
Car Evaluation	1728	7	4	C
Cnae-9	1080	857	9	I
Connectionist Bench	208	60	2	R
Hill Valley With Noise	606	101	2	R
Image Segmentation	2310	19	7	R
Indian Liver Patient	583	10	2	I,R
Iris	150	4	3	R
KR vs KP[2]	3196	37	2	C
Leukemia	100	50	2	R
Mamographic Mass	961	6	2	I
Multiple Features	2000	649	10	R
Mushroom	8124	22	2	C
Musk	6598	168	2	I
Ozone Level Detection	2536	73	2	R
Pen Digits[3]	10992	16	10	I
Phishing Website	2456	30	3	I
Pima	768	9	2	I, R
Planning Relax	182	13	2	R
Seeds	210	7	3	R
Semeion	1593	256	10	I
Solar Flare	1389	10	3	C
SPECTF Heart	267	44	2	I
Tic-Tac-Toe Endgame	958	9	2	C
Twonorm	7400	21	2	R
Vehicle	946	18	4	I
Waveform	5000	40	3	R
Wilt	4839	6	2	R

[1]Blood Transfusion Service Center. [2]King-Rook vs King-Pawn.
[3]Pen-based recognition of handwritten digits.

are representative, the error rate in the test suite provides a good indication of performance [20].



First training sets
Percentage of initially labeled instances
Ten distinct sets are formed with this configuration because of cross-validation
The whole dataset 100 instances
90% of dataset
10% of dataset
Training set 90 instances
Test set 10 instances
5%
10%
15%
20%
25%
4 labeled instances 86 unlabeled instances
9 labeled instances 81 unlabeled instances
13 labeled instances 77 unlabeled instances
18 labeled instances 72 unlabeled instances
22 labeled instances 68 unlabeled instances

Fig. 3. An example of data division.

Once a dataset is divided into training and testing, the test set is separated and the training set is further divided into labeled and unlabeled sets. Since all datasets are originally labeled, it is possible to perform an extensive analysis of the impact of the size of the initially labeled set in the analysed semi-supervised methods. In this analysis, 5 different configurations are used, 5%, 10%, 15%, 20% and 25% of the data initially labeled. In other words, of the 90% of the instances selected for training, the process started with either 5%, 10%, 15%, 20% or 25% of the labeled data. In this way, it is possible to analyze the performance of the analysed methods as the amount of instances initially labeled increases. The choice of the labeled dataset was made randomly, but in a stratified way, respecting the same proportion of classes as the dataset original dataset.

After pre-processing the data, the training/test procedure starts. In this analysis, four classification algorithms well known in the literature are applied: Naive Bayes, decision tree, Ripper and k-NN. As explained earlier, these algorithms are chosen because of their popularity and their use in the machine learning domain. For all algorithms, we used the implementations of the Weka tool, available in the R language. More details on the operation of the classification algorithms and their implementation in the R language can be obtained in [26] and [27].

Additionally, it was necessary to choose a confidence threshold to be used throughout the labeling process. Therefore, we performed an initial analysis using different values, of which 95% obtained the better results. In order to validate the performance of the proposed methods from a statistical point of view, we applied the Wilcoxon test. Since this test is non-parametric, it is suitable for comparing the performance of two learning algorithms when applied to separate datasets. A more complete discussion of the Wilcoxon test is presented in [28].

VI. RESULTS AND DISCUSSION

In this section, we will present and analyse the results of the empirical analysis which evaluates the performance of the proposed method, CTFT. As previously explained, the results obtained from these methods will be compared with those obtained from the original co-training algorithm.

This analysis is divided into two parts and, in the next two subsections, we will present the obtained results for each part, obtained performance and statistical analysis, respectively.

A. *Performance Analysis*

Table II, that reports the obtained results, is organized as follows: the first column represents the name of the methods; columns 2 to 6 indicate the average accuracy and standard deviation obtained by the corresponding semi-supervised method, averaged over all 30 datasets, according to the percentage of initially labeled instances, namely: 5%, 10%, 15%, 20% and 25%, respectively. In addition, the CTFT accuracy levels which are higher than the original co-training - OCT are highlighted in bold. The horizontal sub-parts of this table present the average accuracy of each classification algorithm: Naive Bayes, Decision tree, Ripper and k-NN.

By analysing Table II, it is possible to notice that the method proposed in this work (CTFT) obtained higher accuracy than the original co-training (OCT) in all analysed cases. In addition, as it was expected, the accuracy of both methods increase as the percentage of initially labeled instances increases. When using 5% of instances in the labeling set, the initial training

TABLE II
AVERAGE ACCURACY OF ALL METHODS WITH NAIVE BAYES, DECISION TREE, RIPPER AND K-NN

Met.	% of inicially labeled instances				
	5%	**10%**	**15%**	**20%**	**25%**
	Naive Bayes				
OCT	59.90 ± 19.47	61.59 ± 18.89	62.75 ± 18.50	63.13 ± 18.67	63.89 ± 18.62
CTFT	**61.98 ± 17.72**	**64.34 ± 17.57**	**65.32 ±17.90**	**65.89 ± 18.26**	**66.81 ± 17.45**
	Decision tree				
OCT	59.63 ± 20.92	63.18 ± 20.10	65.90 ± 18.81	66.98 ± 18.22	68.43 ± 17.95
CTFT	**62.09 ± 19.99**	**68.66 ± 16.75**	**70.00 ± 15.74**	**71.03 ± 15.29**	**72.26 ± 14.79**
	Ripper				
OCT	59.26 ± 16.78	63.19 ± 15.64	65.07 ± 15.91	66.64 ± 15.77	68.65 ± 15.87
CTFT	**64.07 ± 16.33**	**67.92 ± 15.30**	**69.63 ± 14.80**	**70.38 ± 14.29**	**71.66 ± 14.12**
	***k*-NN**				
OCT	65.15 ± 17.88	69.04 ± 17.32	70.71 ± 17.05	72.25 ± 16.87	72.88 ± 17.01
CTFT	**70.28 ± 14.66**	**72.79 ± 14.06**	**74.41 ± 13.32**	**75.07 ± 13.27**	**75.66 ± 12.81**

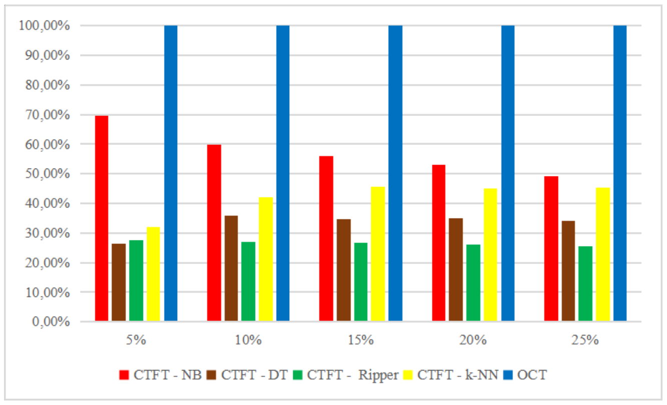


Fig. 4. Percentage of instances labeled in each method, separated by classifier

sets are usually very small and, as a consequence, this classifier will not be able to classify efficiently the unlabeled instances. However, the difference in performance when increasing from 5% to 25% the percentage of initially labeled instances was not huge for *k*-NN, for example.

It is important to emphasize that both methods (OCT and CTFT) have high standard variation values. This is due to the fact that the values presented in this table are averaged over 30 datasets. As these datasets represent different classification applications, the analysed methods tend to have different performances over these 30 datasets, leading to a high standard deviation.

There is also an important aspect to be highlighted in relation to the CTFT algorithm. As previously mentioned, due to the fact that this algorithm uses a fixed threshold, it does not label instances whose confidence rate is lower than the threshold initially set. As a consequence, the set of labeled data may only contain those instances whose prediction is reliable, positively affecting the prediction of the classifiers and, as a consequence, improving the effectiveness of the co-training testing process.

Figure 4 shows the average percentage of labeled instances for each classifier using the OCT and CTFT algorithms during the labeling process. When analyzing this figure, we can see that the CTFT algorithm labels between 30% and 70% of the instances of the set of initially unlabeled data. On the other hand, the OCT algorithm always labels the entire set of unlabeled data.

These results show that the inclusion of instances in the labeled set which are not reliable (low confidence) may deteriorate the performance of a semi-supervised method. Therefore, a more meticulous selection criterion can have a positive effect in the performance of these methods and this is the main motivation behind the proposal of CTFT.

B. Statistical Analysis

After a visual evaluation of the performance of each semi-supervised method, there is a need to perform a statistical analysis of the obtained results. As previously explained, the Wilcoxon test was used to compare the performance of different methods applied to different datasets. Table III presents the results (p-values) of the Wilcoxon test. This table is organized as follows: the first column represents the name of the analysed classification algorithms, namely: Naive Bayes (NB), Decision tree (DT), Ripper and *k*-NN; columns 2 to 6 indicate the obtained p-value, according to the percentage of initially labeled instances, namely: 5%, 10%, 15%, 20% and 25%, respectively. In this case, for each cell configuration, CTFT was compared to COT, using the Wilconxon test.

TABLE III
WILCOXON TEST´S RESULT FOR EACH CLASSIFICATION ALGORITHMS FOR EACH PERCENTAGE OF INITIALLY LABELED INSTANCES

Alg.	% of inicially labeled instances				
	5%	**10%**	**15%**	**20%**	**25%**
NB	7.98e-03	1.29e-06	1.07e-06	1.61e-08	3.51e-10
DT	1.07e-06	2.96e-15	8.66e-14	1.94e-11	8.25e-11
Ripper	8.85e-08	4.98e-08	1.34e-09	1.33e-09	1.31e-08
k-NN	1.52e-13	4.31e-10	1.03e-11	1.80e-05	3.45e-07

By analyzing the referred table, it is possible to observe that in all cases the p-value was lower than 0.05. Therefore, we can conclude that, from a statistical point of view, the proposed method provides higher accuracies than the original co-training.

VII. CONCLUSION

This work presented an extension of the co-training method, named Co-Training with Fixed Threshold (CTFT). As previously highlighted, the original co-training algorithm (OCT) has a limitation related to the selection criterion to define which instances will be labeled at each iteration. It tends to included not reliable instances and this can deteriorate the performance of this method. The purpose of our proposal was to smooth out this limitation by including a confidence threshold as the selection criterion to include new instances in the labeled set, thus improving the overall average accuracy of the proposed method, when compared to the approach used by the OCT.

The proposed algorithm was evaluated in an empirical analysis that gathered 30 databases with diverse characteristics.

In addition, four different classification algorithms and five different percentages of initially labeled instances were assessed, simulating situations that may occur in real-world databases. The results obtained demonstrated that CTFT significantly improves the average accuracy of OCT, as verified by the application of Wilcoxon statistical tests on the obtained results.

Despite the improvement obtained from the use of CTFT in relation to OCT, it is worth mentioning that, unlike the OCT algorithm, CTFT may not label all instances of the unlabeled data set. The behavior of OCT implies the possibility of including instances with low confidence (not reliable instances), which can negatively influence the prediction of this method. On the other hand, the CTFT procedure limits the inclusion of some instances in the set of labeled data. In this way, the training set will contain only those instances whose prediction is reliable, positively affecting the prediction of the classifiers.

References

[1] O. Chapelle, B. Scholkopf, and A. Zien, *Semi-Supervised Learning*. The MIT Press, 2006.

[2] J. Tanha, M. Van Someren, and H. Afsarmanesh, "Semi-supervised self-training for decision tree classifiers," *International Journal of Machine Learning and Cybernetics*, vol. 8, pp. 355–370, Feb 2017.

[3] X. Zhu and A. B. Goldberg, "Introduction to semi-supervised learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 3, no. 1, pp. 1–130, 2009.

[4] K. M. O. Vale, *A proposal to automate the process of labeling instances in semi-supervised learning algorithms (in Portuguese)*. PhD thesis, Federal Univ. of Rio Grande do Norte, Nov. 2019.

[5] E. Matsubara, M.-C. Monard, and G. Batista, "Multi-view semi-supervised learning: An approach to obtain different views from text datasets.," pp. 97–104, 01 2005.

[6] F. M. Rodrigues, A. de M Santos, and A. M. P. Canuto, "Using confidence values in multi-label classification problems with semi-supervised learning," in *The 2013 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, 2013.

[7] M. C. Monard and J. A. Baranauskas, *Sistemas Inteligentes: Fundamentos e Aplicações*, ch. Conceitos sobre Aprendizado de Máquina, p. 89 114. Manole, 2003.

[8] M. Wang, W. Fu, S. Hao, D. Tao, and X. Wu, "Scalable semi-supervised learning by efficient anchor graph regularization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, pp. 1864–1877, July 2016.

[9] A. Santos and A. Canuto, "Applying semi-supervised learning in hierarchical multi-label classification," *Expert Systems with Applications*, vol. 41, no. 14, pp. 6075 – 6085, 2014.

[10] S. Samiappan and R. J. Moorhead, "Semi-supervised co-training and active learning framework for hyperspectral image classification," in *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pp. 401–404, July 2015.

[11] B. Aryoyudanta, T. B. Adji, and I. Hidayah, "Semi-supervised learning approach for indonesian named entity recognition (ner) using co-training algorithm," in *2016 International Seminar on Intelligent Technology and Its Applications (ISITIA)*, pp. 7–12, July 2016.

[12] N. D. Hoang, V. T. N. Chau, and N. H. Phung, "Combining transfer learning and co-training for student classification in an academic credit system," in *2016 IEEE RIVF International Conference on Computing Communication Technologies, Research, Innovation, and Vision for the Future (RIVF)*, pp. 55–60, Nov 2016.

[13] S. Qiao, W. Shen, Z. Zhang, B. Wang, and A. Yuille, "Deep co-training for semi-supervised image recognition," in *The European Conference on Computer Vision (ECCV)*, September 2018.

[14] D. Kim, D. Seo, S. Cho, and P. Kang, "Multi-co-training for document classification using various document representations: Tf-idf, lda and doc2vec," *Information Sciences*, 2019.

[15] Y. Chen, T. Pan, and S. Chen, "Development of co-training support vector machine model for semi-supervised classification," in *2017 36th Chinese Control Conference (CCC)*, pp. 11077–11080, July 2017.

[16] X. Bai, B. Wang, C. Yao, W. Liu, and Z. Tu, "Co-transduction for shape retrieval," *IEEE Transactions on Image Processing*, vol. 21, pp. 2747–2757, May 2012.

[17] S. Sun, L. Mao, Z. Dong, and L. Wu, *Multiview Machine Learning*. Springer, 2019.

[18] A. Albalate and W. Minker, *Semi-Supervised and Unsupervised Machine Learning - Novel Estrategies*. Wiley, 2011.

[19] T. M. Mitchell, *Machine Learning*. McGraw-Hill, 1997.

[20] I. H. Witten, E. Frank, and M. A. Hall, *Data Mining: Practical Machine Learning Tools and Techniques*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 3rd ed., 2011.

[21] L. I. Kuncheva, *Combining Pattern Classifiers: Methods and Algorithms*. Wiley Publishing, 2nd ed., 2014.

[22] D. Dheeru and E. Karra Taniskidou, "UCI machine learning repository," 2017.

[23] J. Alcala-Fdez, A. Fernandez, J. Luengo, J. Derrac, and S. Garcia, "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.

[24] J. W. Smith, J. E. Everhart, W. C. Dickson, W. C. Knowler, and R. S. Johannes, "Using the ADAP learning algorithm to forecast the onset of diabetes mellitus," in *Proceedings of the Symposium on Computer Applications and Medical Care*, pp. 261–265, IEEE Computer Society Press, 1988.

[25] L. Breiman, "Bias, variance, and arcing classifiers," Tech. Rep. 460, Statistics Department, University of California at Berkeley, 1996.

[26] P. Cichosz, *Data Mining Algorithms: Explained Using R*. Wiley online library, Wiley, 2015.

[27] L. Torgo, *Data Mining with R: Learning with Case Studies, Second Edition*. Chapman & Hall/CRC, 2nd ed., 2017.

[28] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bulletin*, vol. 1, pp. 80–83, Dec. 1945.