Possibilistic Approach For Novelty Detection In Data Streams

Tiago Pinho da Silva
Instituto de Ciências Matemáticas e de Computação
Universidade de São Paulo
São Carlos, SP, Brazil
tpinho@usp.br

Heloisa de Arruda Camargo
Departamento de Computação
Universidade Federal de São Carlos
São Carlos, SP, Brazil
heloisacamargo@ufscar.br

***Abstract*—In many real-world applications data arrive continuously, in the form of streams. Such data can be used for the acquisition of knowledge by machine learning methods. In data streams learning, novelty detection is a relevant topic, which aims to identify the emergence of a new concept or a drift in the known concept in real time. Most approaches in the literature that focus on the novelty detection problem, make assumptions that limit the method usefulness. For instance, some methods are designed lying on the supposition that labeled data will be available at some time in the stream, while others restrict the proposed algorithm to one-class problems. Some recent approaches aim to overcome the limitations mentioned, considering multiclass problems and unlabeled data streams. In addition, there are also proposals that explore concepts of fuzzy set theory to add more flexibility to the learning process, although restricted to labeled streams. In this paper, we propose a method for novelty detection in data streams called Possibilistic Fuzzy multiclass Novelty Detector for data streams (*PFuzzND*). Our methods generates models based on a proposed summarization structures named *SPFMiC* (Supervised Possibilistic Fuzzy Micro-Cluster), which provides flexible class boundaries, allowing the identification of different types of novel information, i.e, novel classes, extension of classes or outliers more efficiently. Experiments show that our approach is promising in dealing with the changes in data streams and presents improvements in comparison to the non-fuzzy version.**

I. Introduction

In data stream learning, a large amount of continuously generated data is used as source for the automatic acquisition of useful knowledge by machine learning methods. Furthermore, in real-world scenarios a Data Stream (DS) can be infinite in size and can change its statistical distribution over time [1]. Hence, classical machine learning methods require the development of mechanisms to be able to deal with DSs particularities.

Among the possible tasks under data streams, Novelty Detection (ND) is the primary focus of this paper. It can be seen as a classification task, which aims to identify new or unknown circumstances not seen before. It is an important subject of study, especially in DSs, where new concepts can appear, disappear or evolve over time. Therefore, ND in DSs makes it possible to distinguish the novel concepts from noise data, where novel concepts may be characterized as new concepts or changes in known concepts [1].

In the DS scenario, ND has received increasing attention from the scientific world [2]. Different works approach ND in DSs as a one-class classification task, in which the known concept is represented only by the normal class. Hence, the examples not explained by the model are then classified as unknown or novelty [3], [4]. However, this taks can be viewed as more general [5], i.e, in ND problems, it is more likely that the known concept may be composed by different classes. Furthermore, novel classes may appear in the course of time, which results in a phenomenon known as concept evolution [1]. Moreover, the distribution of known classes can change, resulting in a phenomenon known as concept drift. Thus, the decision model must evolve to represent the novel classes and the changes in the known classes. Following this idea, some works have recently been proposed [4]–[8]. The present work contributes with algorithms for detecting novelty in a multiclass classification task, which is regarded as a challenging task, but also a more realistic scenario for many real applications.

In order to obtain more flexible learning models considering the changes of the data, researchers have recently developed methods using concepts of the fuzzy set theory [9]. Resonating with the search for more flexible learning, we propose a method called Possibilistic Fuzzy multiclass Novelty Detector for data streams (*PFuzzND*), which is an extension of our previos method named Fuzzy multiclass Novelty Detector for data streams (*FuzzND*) [10]. Our proposals adopts a summarization structure named Supervised Possibilistic Fuzzy Micro-Clusters (SPFMiC). In addition, we used a fuzzy clustering similarity measure FR [11] to compare novel detected patterns with the ones already included in the current model. Based on the comparison results, the algorithm decides whether the patterns represent a novel class or is related to a pattern already detected. The experiments demonstrated that the proposed methods can deal better with the changes in the DS, reacting more precisely to them, and presents more stable results over the data stream in relation to similar methods in the literature.

The paper is organized as follows: Section II discusses related works that concern ND problem and fuzzy approaches to data stream. In Section III we describe the proposed method, *PFuzzND*. In Section IV we discuss the experiments settings to evaluate our proposals. Evaluation results and conclusions are provided respectively in Section V and Section VI.

II. Related Work

Concerning ND in DS learning, it is usually assumed that a labeled set is initially available and the unlabeled data arrive continuously in the form of streams. In addition, ND algorithms should consider that data may change over time, so the learning of a generic model from DS can be quite challenging.

In DS learning, ND approaches usually are divided into two steps [2], called *Framework Offline-Online*. During the first step (*offline*), it is assumed the availability of a labeled set which correspond to the normal concept. A normal concept can be seen as a normal situation, i.e, non-fraudulent credit card transactions. The labeled set is then used to generate the initial model, which is able to recognize the normal concept and will be used for the ND task. In the second step (*online*), the unlabeled data arriving from the stream, are classified incrementally as normal or unknown, in order to update the initial model and detect novel concepts. One of the advantages of this method is the instantaneous results generation, due to the model adaption that occurs incrementally. Also, the ND procedure may be executed independently from the classification procedure.

There are mainly two approaches for ND in DS [2]: one-class and multiclass. In the one-class approach an initial model is generated based on examples from one class, usually a class that represents the normal concept. In the *online* phase, different methods consider clusters of examples not explained by the current model (also named as unknown examples) to build a Novelty Pattern (NP) [3], [5], which is defined as a pattern identified from cohesive groups of similar examples marked with the unknown profile by the classification model [5]. A NP can indicate a change in the known concepts, named concept drift, or the appearing of a new one, named concept evolution [1].

In contrast, in the multiclass approach, it is assumed that the normal concept can be composed by a set of different classes [4]–[8], which can be viewed as a more likely approach, concerning DSs. However, most works limit the emergence of NP in one at a time, i.e., examples from different classes cannot appear interchangeably. In addition, some of these works update the decision model assuming that the true label for all instances will be available at a particular timestamp (time of arrival of the examples), characterizing an unrealistic assumption regarding DSs scenarios.

Recent works are being developed in order to handle this disadvantages [5], [8]. The AnyNovel [8] algorithm is mainly composed of two components: the Cohesion Validation Component (CVC) and OBSERVER. The CVC component aggregates dependent examples, representing the same concept, and those that can not be explained are stored in a temporary buffer. The OBSERVER monitors the evolution of the DS by analyzing the examples, which can be from different novel classes aggregated in the temporary buffer. Although this method can detect emergence of multiples novel classes it requires the true label for uncertain examples.

The *MINAS* algorithm, [5], which is the base for the proposal we present in this paper, applies unsupervised learning approaches in order to identify concept evolution in a multi-class scenario, thus not requiring the true label of examples to update the model. MINAS classifies new incoming instances as the known classes, known NPs or unknown. Unknown examples are the ones located outside the decision boundaries of all micro-clusters that composes the decision model of the method. The declared unknown examples are stored in a short-term memory. When the short-term memory reaches a certain amount of examples, they are clustered in order to discover cohesive groups, that may represent a NP, extension of a known class or extension of a known NP. This method considers the appearance of multiples NPs. Although efficient in detecting NPs, *MINAS* is very sensitive to noisy data and data scale, what causes a decrease in its accuracy.

A. Fuzzy Set Theory in Data Stream

The growth of DSs mining propels the search for methods to create more dynamic and flexible models, that can better deal with changes in data. DSs mining models that use fuzzy set theory have recently been proposed towards solving this issue, with promising results [12]–[14].

The ND task, in particular, have also been addressed by using concepts of fuzzy set theory. The data stream classifier *eClass* [15] is based on an evolving Fuzzy Rule-Based (FRB) classifier system of Takagi–Sugeno (eTS) type. This approach is divided in two phases, the prediction and evolution phases. During the first phase, the class label of an example is predicted; in the second phase, however, the true class label is required and used as supervisory information to update the classifier. Although this method can adapt to the emergence of novel classes, it is based on an unrealistic assumption, since labeling all the DS is time consuming. Also, this type of information may arrive late and not be useful to the current state of the DS.

I our previous work we proposed the *FuzzND* (Fuzzy multiclass Novelty Detector for data streams) [10], which is a fuzzy generalization for the *offline-online* framework presented in *MINAS* [5]. In the *offline* step a decision model is learned from a labeled set of examples using the Fuzzy C-Means

(FCM) [16] clustering algorithm. Later, during the *online* step, new unlabeled examples that come from the data stream are classified, incrementally, in one of the known classes of the model or as unknown by taking into consideration the membership values. Intermittently, the unknown examples are clustered in order to look for cohesive groups which are incorporated in the decision model as novel patterns. This method, however, can not handle situations where only one class appear for a time period, and the use of memberships is not ideal to detect outliers [17].

Although showing promising results, the fuzzy algorithms mentioned before still presents challenges to be addressed. In light of this problems we propose the method named *PFuzzND*, which will be presented in the next section.

III. Possibilistic Fuzzy Multiclass Novelty Detector

The proposed method called *PFuzzND* is based on the well-known clustering method *Possibilistic Fuzzy C-Means* (PFCM) [18], which uses the FCM memberships and PCM typicalities in order to create models less sensitive to initialization and parameter choices, and can best describe *outliers*.

In order to provide statistics for the calculation of the typicalities, the Supervised Possibilistic Fuzzy Micro-Cluster (SPFMiC) was proposed. A SPFMiC with fuzzification parameter m and typicality parameter n, for a set of examples $e_1, ..., e_N$ d-dimensional, with membership values $\mu_1, ..., \mu_N$, values of typicality $\gamma_1, ..., \gamma_N$, *timestamps* $t_1, ..., t_N$ and centroid c is defined as the vector $(M^e, T^e, \overline{CF1^e_\mu}, \overline{CF1^e_t}, SSD^e, N,\ t, class\_id)$. Where, M^e is the linear sum of the examples memberships powered by m, T^e is the linear sum of the examples typicalities powered by n, $\overline{CF1^e_\mu}$ is the linear sum of the examples weighted by their memberships. $\overline{CF1^e_\gamma}$ is the linear sum of the examples weighted by their typicalities. SSD^e is the quadratic sum of distances from the examples to the micro-cluster prototype weighted by the example's membership, N is the number of examples associated to the micro-cluster, t is the ordinary mean of timestamps for points associated to the SPFMiC, and $class\_id$ is the class associated to the micro-cluster.

1) Offline Step: Concerning the proposed method, Algorithm 1 shows the *offline* step, that requires as input the data stream DS, the FCM parameter m, the number of clusters by class k_class and the labeled set that is going to be used to calculate the first micro-clusters $init\_points$.

In Algorithm 1, lines 4 and 3 are methods that were revised to comply to fuzzy set theory concepts. At the beginning, for each class of the labeled set, the set of corresponding points is given as entry for the FCM clustering algorithm (Step 3). The clusters of each set returned by the FCM are stored in variable $class\_cluster$ and lately summarized in a supervised fuzzy micro-cluster structure in the function *summarize* (Step 4). The decision model is defined as the set of SPFMiCs found for all different classes (Step 5).

Algorithm 1 PFuzzND - *Offline* Step Based on [10]

```
Require: $DS, m, k\_class, init\_points$
1: $model \leftarrow \emptyset$
2: for each class $C_i \in init\_points$ do
3:     $class\_clusters \leftarrow$ FCM$(init\_points_{class=C_i}, m, k\_class)$
4:     $class\_SPFMiC \leftarrow$ SUMMARIZE$(class\_clusters)$
5:     $model \leftarrow model \cup class\_SPFMiC$
6: end for
```

2) Online Step: Algorithm 2 describes the *online* step, where the input parameters $init\_\theta$ is an initial threshold for classifying and processing the examples. The $\theta_{adapter}$ and θ_{class} parameters correspond to adaptation thresholds for the classification step. Examples labeled as unknown are stored in a short-term memory ($short\_memory$). T is the minimum amount of unknown examples in the $short\_memory$ for the novelty detection procedure to be executed. Parameter P is a time threshold related to the forgetfulness of older SPFMiCs and ts is a time limit corresponding to the removal of older unknown examples in $short\_memory$. max_mic_{class} corresponds to the maximum of micro-groups per class. The parameters α, β, K and n are required for the calculation of typicalities, as well as being used to define the centroid of the SPFMiCs.

Algorithm 2 PFuzzND - *Online* Step

```
Require: $DS, init\_\theta, \theta_{adapt}, \theta_{class}, m, T, P, ts$
Require: $max\_mic_{class}, \alpha, \beta, K, n, dec_new$
1: $short\_memory \leftarrow \emptyset$
2: $all\_tip_{max} \leftarrow init\_\theta$
3: while !ISEMPTY$(DS)$ do
4:     $x \leftarrow$ NEXT$(DS)$
5:     $all\_m \leftarrow$ MEMBERSHIP$(x, model, m)$
6:     $all\_t \leftarrow$ TYPICALLITY$(x, model, n, k, \beta)$
7:     $(max\_class, max\_tip) \leftarrow$ MAX$(all\_t)$
8:     if $max\_tip \geq$ MEAN$(all\_tip_{max}) - \theta_{adapt}$ then
9:         $all\_tip_{max} \leftarrow all\_tip_{max} \cup max\_tip$
10:        $x.class \leftarrow max\_class$
11:        UPDATE$(model_{class=max\_class}, x, all\_m, all\_t)$
12:    else if $max\_tip \geq$ MEAN$(all\_tip_{max}) - \theta_{class}$ then
13:        CREATE_SFMIC$(model, x, max\_class, dec_new)$
14:        if $|model_{class=max\_class}| > max\_mic_{class}$ then
15:            $model \leftarrow$ REMOVE_SPFMiC$(model_{class=max\_class})$
16:        end if
17:    else
18:        $x.class \leftarrow unknown$
19:        $short\_memory \leftarrow short\_memory \cup x$
20:        if $|short\_memory| \geq T$ then
21:            $model \leftarrow$ NOVELTY_DETECTION$(short\_memory, model)$
22:        end if
23:    end if
24:    $current\_time \leftarrow x.time$
25:    $model \leftarrow$ REMOVE_SPFMiC$(model, P)$
26:    $short\_memory \leftarrow$ REMOVE_UNKNOWN$(short\_memory, ts)$
27: end while
```

In Algorithm 2, for each example x that arrives from the stream, the algorithm calculates the memberships and typicalities of x concerning all SPFMiCs present in the model (Steps 5-6). Typicality values will be used to decide whether x will receive the label of an existing class or will be classified as unknown. This process is done by verifying if the highest typicality is greater or equal to the mean of the maximum typicalities of all the previous examples labeled as belonging to a class known (stored in all_tip_{max}), minus an adaptation threshold θ_{adapt}, whose function is to ensure that slightly below-average examples can be classified with the known labels (Step 8). When the first example is being processed, its highest typicality value will be compared to the value of the initial parameter $init\_\theta$.

If x is labeled as belonging to an existing class C_i, i.e., its maximum typicity is greater than or equal to the mean of the maximum typicalities of previous examples minus an adaptive threshold θ_{adapt}. Its maximum typicality is added to all_tip_{max} to update the mean of the previous maximum typiccalities (Steps 9), and x is associated with the SPFMiCs of class C_i (Steps 11). Otherwise, if x is greater than or equal to the average of previous maximum typicalities minus an adaptation threshold θ_{class} (Step 12), x corresponds to a potential extension of class C_i and a new SPFMiC is created for this class with x as the prototype (Step 13).

3) Novelty Detection Step: Whenever a new example is marked with the unknown profile, *PFuzzND* verifies if there is a minimal number of examples in the short-term memory ($short\_memory$). If so, *PFuzzND* executes a novelty detection procedure in an unsupervised fashion which requires as input the structure $short\_memory$ containing the unknown examples, the FCM fuzzification parameter m and the number of clusters k_short.

Algorithm 3 PFuzzND - Novelty Detection Step

```
Require: $short\_memory, model, m, k\_short$
 1: $temp\_clusters \leftarrow$ FCM($short\_memory, m, k\_short$)
 2: for each $cluster\ temp\_c_i \in temp\_cluster$ do
 3:     if VALIDATE($temp\_c_i$) then
 4:         $SPFMiC_{NP} \leftarrow$ SUMARIZING($temp\_c_i$)
 5:         for each $SPFMiC_{class=NP\#}\ NP_i \in model$ do
 6:             $FR \leftarrow$ SIMILARITY($NP_i, temp\_c_i$)
 7:             $all\_FR \leftarrow all\_FR \cup FR$
 8:         end for
 9:         $(label_{max}, max\_fr) \leftarrow$ MAX($all\_FR$)
10:         if $max\_fr \geq \phi$ then
11:             $SPFMiC_{NP}.label \leftarrow label_{max}$
12:             $model \leftarrow SPFMiC_{NP}$
13:         else
14:             $label_{new} \leftarrow$ NEW_NP_LABEL($model$)
15:             $SPFMiC_{NP}.label \leftarrow label_{new}$
16:             $model \leftarrow SPFMiC_{NP}$
17:         end if
18:     end if
19: end for
```

In Algorithm 3 the first step is the application of FCM [16] on the examples in the short-term memory ($short\_memory$), producing k_short clusters (Step 1). *PFuzzND* evaluates each one of the clusters to decide if it is valid, by checking if its fuzzy silhouette coefficient [19] is greater than 0 and if it has a representative number of examples (Step 3).

Whenever a cluster is validated, it represents a Novelty Pattern (NP). The next step is to summarize the valid cluster and check if it will be assigned a new label generated by the method, or the label of the closest NP already labelled before. The new labels are defined as NP#id, where #id is a unique number that represents the NP. In order to assign a label to a new NP, *PFuzzND* determines the similarity between the valid cluster representing the new NP and the other NP clusters already in the model, by calculating the fuzzy clustering similarity FR introduced in [11] (Steps 5-8). If the maximum similarity between the new NP cluster and the other NP clusters is greater than a ϕ parameter chosen by the user, the new NP cluster receives the same label as the NP cluster to which it has the maximum similarity (Steps 11-12). Otherwise, a new label is created and associated to the new NP cluster (Steps 14-16). If a cluster is not validated, it is discarded, and its examples remain in the short-term memory ($short\_memory$) until the model decides to remove them.

Unlike the *FuzzND* method that uses class compatibility through *fuzzy* memberships to classify new examples in order to identify *outliers*, in *PFuzzND* the use of typicalities allows a better description of these types of examples during the classification step.

IV. Experiments

In order to verify the advantages of the possibilistic fuzzy-based approaches proposed here, we compare the results of *PFuzzND* against the results obtained from the *MINAS* and *FuzzND* algorithms. The proposed approache was implemented using the R language, assisted by packages, namely stream [20] and e1071 [21]. Each experiment was executed 5 times, due to the randomness of the clustering algorithms used in the *offline* phase and in the novelty detection procedure.

Next, we explain with more details the datasets and evaluation metrics used in the experiments.

A. Datasets

The evaluation of the methods was done using synthetic datasets named MOA3 [5], RBF [22] and SynEDC. Table I presents details of each dataset, columns #Instances and #Attributes indicate the total number of examples and attributes for each dataset respectively. The total number of classes and number of classes available during the offline phase are indicated at columns #Classes and #Classes (Offline) respectively. The difference between the values of these two last columns is the number of novel classes that the algorithms are expected to detect. The specificities for the experiments executions are described in subsection IV-B.

TABLE I
DATA SETS USED IN EXPERIMENTS

Identifier	#Instances	#Attributes	#Classes	#Classes (Offline)
MOA3	100,000	4	4	2
SynEDC	400,000	54	20	7
RBF	48,588	2	5	3

B. Algorithms

1) FuzzND: Concerning the *offline* phase, parameters m and k_class were set to 2 and 4 respectively. Regarding the *online* phase, parameters were defined as described in Table II. Parameter $init\_\theta_{class}$ was defined as 0.3, 0.3 and 0.2 for the datasets MOA3, RBF and SynEDC respectively, due to the high dimension of SynEDC. These parameter values were chosen for the *offline* and *online* phases because they have led to the best results in preliminary experiments.

TABLE II
FUZZND PARAMETERS - *Online* PHASE

Parameter	Value
$init\_\theta$	0.3
$init\_\theta_{class}$	*
θ_{adapt}	0.1
T	40
P	500
ts	200

In regards to the novelty detection step, the parameters m and k_short were defined as 2 and 4 respectively. The parameters used in the experiments for the proposed method outlined in this work were chosen empirically.

2) PFuzzND: Regarding the *PFuzzND* method, empirically determined results analyzes demonstrated a sensitivity of the algorithm to different datasets. Thus, the parameters k_class, k_short, ts, min_weight and the thresholds $init\_\theta$ and θ_{adapt} were selected differently, for each dataset, according to Table III. The common parameters T,P and ts were set with the same values as in *FuzzND*, to ensure a valid comparison.

TABLE III
PFUZZND PARAMETERS BY DATASET

	$init\_\theta$	θ_{class}	θ_{adapt}	k_class	k_short	ts	min_weight
MOA	0.95	0.80	0.00	4	4	200	25
RBF	0.95	0.80	0.00	4	4	200	25
SynEDC	0.90	0.58	0.00	8	8	200	25

3) MINAS: Concerning *MINAS*, the common parameters T,P and ts were set with the same values as in *PFuzzND* and *FuzzND*, to ensure a valid comparison. In addition, the number of micro clusters in the offline and novelty detection step were set similarly to *PFuzzND* algorithm Table III. The threshold for novelty detection was set to 1.1 and the threshold strategy chosen was the *strategy 1*, as defined to their default values listed in [5].

C. Evaluation

Experiments were evaluated using the incremental confusion-matrix proposed by [23], which evolves incrementally as soon as a new data point is classified. This matrix is composed by rows and columns that represent known classes, unknown examples and NPs. In this approach, NPs may be associated with a known class, thus being considered an extension of this class, or not being associated to any known class, thereby constituting a potential emergent class. It is important to state that a novel class may be represented by more than one NP, while a NP can represent only one class.

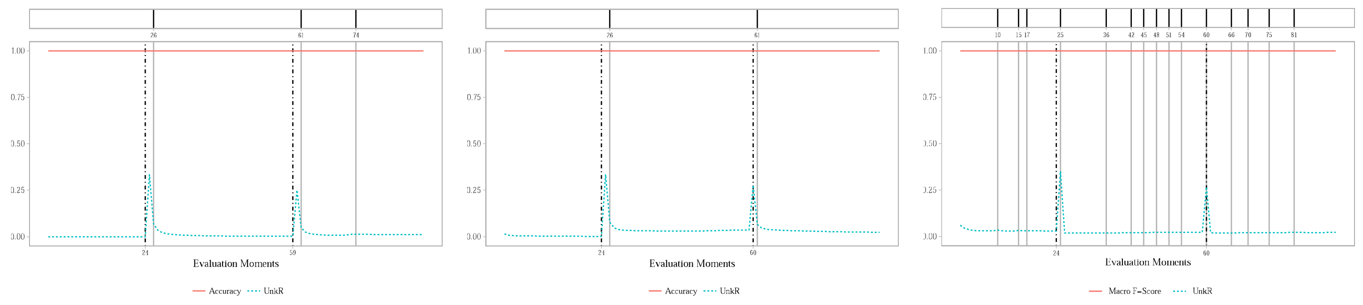
To observe the results of each algorithm through time, we presented the evaluation results in each 1000 interval of points defined. At this evaluation moments, it was calculated the Macro F-Score metric considering only the examples classified with a NP label or a class label. Moreover, the rate of examples classified as unknown by the algorithm was determined by the unknown rate (UnkR) measure [5], defined in Equation 1. In this equation, $\#C$ is the total number of classes, $\#UnkR_i$ is the number of examples of class C_i classified as unknown, and Exc_i is the total number of examples from class C_i. The results are presented in Section V.

$$Unkr = \frac{1}{\#C}\left(\sum_{i=1}^{\#C}\frac{\#UnkR_i}{\#Exc_i}\right) \tag{1}$$

V. RESULTS

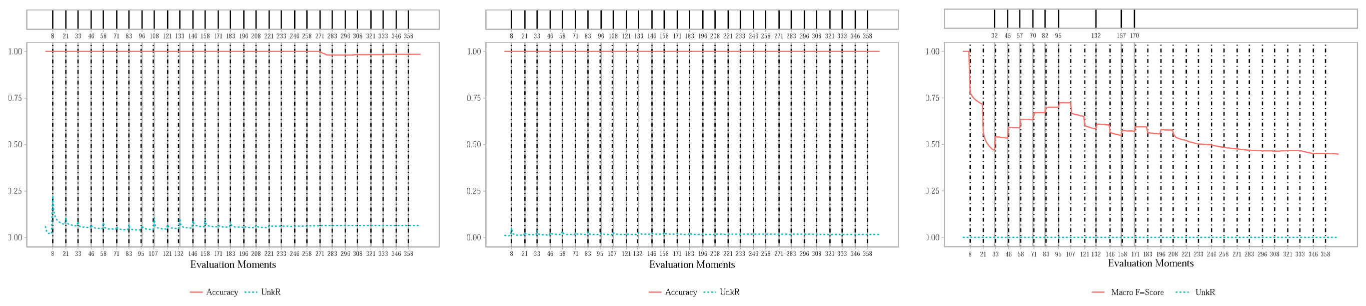
According to the experiment settings described in Section IV 90, 370, 48 evaluation moments were defined for the datasets MOA3, SynEDC and RBF, respectively. Figures 1, 2 and 3 present the Macro F-Score and UnkR results for the evaluation moments in each dataset for all algorithms. In these Figures, the vertical dotted lines indicate the evaluation moments when examples of the novel classes appeared, while the vertical continuous lines represent the evaluation moments where a NP was detected by the methods.

The reason behind the results in Fig. 1, Fig. 2 and Fig. 3 is related to the crisp characteristics of the micro-clusters in MINAS, in which an example must be inside a micro-cluster radius to be classified as the micro-cluster's class. In this scenario, examples that represent an extension of known classes are first classified as unknown, to be detected later in the ND procedure as a NP. Concerning DS, some applications require models that can adapt faster since changes may occur in a regular way. Furthermore, the use of crisp micro-clusters makes the learning process more sensitive to noise. On the other hand, *FuzzND* and *PFuzzND* consider fuzzy values, which make the knowledge acquisition process more flexible, what can be a relevant contribution considering non-static data. Besides, the proposed method can detect class extensions in



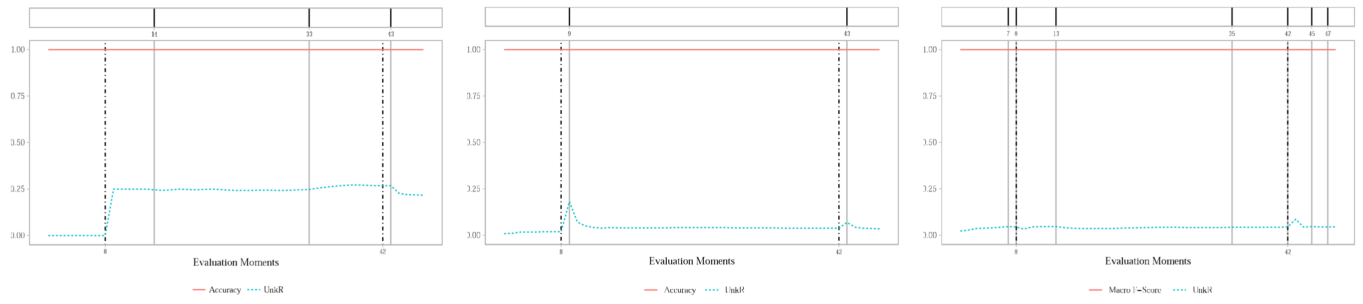
(a) Algorithm: FuzzND (b) Algorithm: PFuzzND (c) Algorithm: MINAS

Fig. 1. Macro F-Score and Unknown Rate measures for each evaluation moment in MOA3 dataset.



(a) Algorithm: FuzzND (b) Algorithm: PFuzzND (c) Algorithm: MINAS

Fig. 2. Macro F-Score and Unknown Rate measures for each evaluation moment in SynEDC dataset.



(a) Algorithm: FuzzND (b) Algorithm: PFuzzND (c) Algorithm: MINAS

Fig. 3. Macro F-Score and Unknown Rate measures for each evaluation moment in RBF dataset.

the classification step, and adapt the model during the online phase.

All three methods present results for the Macro F-Score measure in the dataset RBF (Fig. 3) and MOA3 (Fig. 1) equal to 1 for all evaluation moments, considering only the examples explained by the model. Only dataset SynEDC (Fig. 2) shows different results, and will be explained later.

Regarding MOA3 dataset results (Fig. 1), *MINAS* algorithm was able to detect the new classes that emerged along the DS at moments 25 and 60 as evidenced by the peaks UnkR measure and reduction of the same after such moments, along with the constant results of Macro F-Score equal to 1, which means that the method was able to classify correctly all the examples that was able to explain. On the other hand, produces more NPs detections, which may indicate a reaction to the extensions of classes or NPs, that are identified during the ND procedure.

As for the *FuzzND* method (Fig. 1a), the results demonstrate a similar response to the emergence of new classes, in addition to being able to produce results of Macro F-score constant and equal to 1. In general, the method was able to detect the appearance of new classes, evidenced by the increase of the UnkR measure and its immediate reduction after detection of a new NP at moments 26 and 61. Because the method can detect class and NP extensions during the classification step, the adaptation occurs faster than in MINAS method, which makes the method less susceptible to the production of NPs unrelated to the emergence of new classes. However, there may still be detections of NPs that do not refer to the appearance of new classes, but in this dataset this characteristic occurs only once.

Concerning the *PFuzzND* algorithm, it performed similarly to the others methods (Fig. 1b). However, it was able to detect only NPs related to the emergence of new classes. *PfuzzND* is also able to detect class or NP extensions during the classification process, which explains the low UnkR values during almost all evaluation moments, except for the appearance of new classes. The detection of NPs related only to the emergence of new classes demonstrates the best performance of the *PFuzzND* method over the other two. However, it is worth mentioning that its parameters were adjusted in order to provide the results obtained.

The SynEDC dataset has a non-stationary behavior, therefore, it suffer changes over time. It contains the emergence of new classes during the evaluation moments (158, 146, 132, 121, 107, 95, 83, 71, 58, 46, 33, 21, 8). In addition, in the evaluation moments (171, 183, 196, 208, 221, 233, 246, 258, 271, 283, 296, 308, 321, 333, 346, 358) it presents reappearance of old classes, which can be characterized as recurrent classes, i.e., classes that arise along the DS between large time intervals [2].

Regarding the results, the algorithm *MINAS* (Fig. 2c) did not detect all the new classes that emerged along the DS causing the reduction of the Macro F-Score measure. Among the 14 new classes that emerged until the moment of evaluation 171 the method was able to detect a total of 8. In addition, after the moment 171 the presence of peaks in the UnkR measure indicates that the method may have associated valid groups of unknown examples to recurrent classes. However, since the detection of new classes in the previous moments was not totally effective, the method did not improve its performance in the classification process, evidenced by the moderate reduction of the Macro F-Score measure over time.

The *FuzzND* and *PFuzzND* methods obtained very similar results (Fig. 2a and Fig. 2b). Compared to the *MINAS* algorithm, both methods were able to detect all the new classes that emerged along the DS. It is important to note that the proposed methods do not recognize recurring classes as the *MINAS* algorithm, thus, assuming that recurring classes are new classes. In general, both methods were able to react quickly to the emergence of new classes. However, the *PFuzzND* method showed smaller peaks of the UnkR measurement, especially at the beginning of the DS, in addition to maintaining the Macro F-Score measurement equal to 1 throughout the DS, unlike the *FuzzND* method which shows a reduction of this measure at the end of the DS. These results, however, need to be better evaluated in order to identify if there is a statistical difference between them.

The RBF dataset has a stationary behaviour, i.e., it does not present changes over time. However, it contains the emergence of new classes during the evaluation moments 8 and 42.

In the dataset RBF (Fig. 3), *MINAS* presents a behavior similar to the MOA3 dataset, dectecting more NPs than the number of novel classes that emerged over time. However, since this dataset is stationary, most of the NPs detected indicate misclassification of the examples belonging to the known classes or known NPs as unknown. As for *FuzzND* (Fig. 3a), however, we highlight the low values of the UnkR measure until the moment of evaluation 8, thus inferring that the method obtained a good initial model. In addition, as in the Macro F-Score results of *MINAS*, *FuzzND* was also able to correctly classify all the examples it could explain, a fact evidenced by the constant values of Macro F-Score equal to 1. However, this method presented a considerable delay for the detection of NPs related to the appearance of the first new class, which caused the increase in the UnkR measure that was moderately reduced after the detection of NPs and its incorporation in the model at the evaluation moments 14 and 33. Despite the increase in the UnkR measure, *FuzzND* was able to adapt correctly to the emergence of new classes and to detect a number of NPs close to the number of new classes that appeared in the DS.

In respect to the *PFuzzND* method (Fig. 3b), the best results were obtained in relation to the other algorithms, being able to adapt quickly to the emergence of new classes and to maintain the UnkR measure with values close to 0. In addition, it classified correctly all examples not labeled as unknown. Thus, the algorithm *PFuzzND* was able to better represent the distribution of the data in this experiment.

VI. Final Considerations

DSs represent a domain of problems that tends to increase, accompanying the technological evolution. Strategies for extracting knowledge from DSs are a trend within Machine Learning research. The unpredictable characteristics of this domains generate difficulties in the learning process, encouraging the search for flexible learning, for example, by integrating concepts of fuzzy set theory. Proposals based on fuzzy concepts aim to collaborate for the flexibility and adaptability of the knowledge learned.

In the ND task for DSs by means of fuzzy techniques, most of the proposals are limited to specific domains and consider the existence of only two classes (Normal and Abnormal). On the other hand, most of the approaches for multiclass ND in DSs do not represent the uncertanties that may occur over time, which can be detrimental in a context in which the distribution of the data is changeable over time, causing inadequate representations.

This work presents a new approache for multiclass ND in DSs. We introduced flexibility in the learning process through the application of possibilistic fuzzy set theory concepts to construct and manage a summarization structure for the *Offline-Online Framework*. The main contributions can be described as: Definition of supervised possibilistic fuzzy summarization structures SPFMiC, which offers quality representations for DSs. In addition, we proposed the novelty detection algorithm *PFuzzND* that was able to better react to changes in synthetic datasets and produce comparable results in real data.

The experiments held in this work were made with the purpose to validate our proposals and highlight its advantages with respect to the similar methods in the literature. Thus, there is still room for further investigations such as: the development of new strategies to reduce the number of parameters and decrease the methods parameters sensitivity. The study of different classification methods in order to better define decision surfaces in complex datasets, for example, that may show classes overlapping. Moreover, the experiments defined to evaluate the proposal involve only the method that served as basis, as described herein, there are other ND approaches in DSs, including using fuzzy techniques. Therefore, comparisons of the new approaches with other existing techniques are essential. There should also be concern about the use of evaluation metrics, since the different approaches were evaluated with different metrics.

The experiments show that the proposals are comparable, or better to the fuzzy base method, and are advantageous for general representation and identification of novelties in DSs. Moreover, the results obtained are an indication that the proposals are significant and contribute to the future development of extensions and new techniques based on the proposals presented in this work. Finally, the possibilistic fuzzy approache contribute to the advancement of research communities in flexible learning and DSs learning. .

Acknowledgment

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

References

[1] J. Gama, *Knowledge discovery from data streams*. CRC Press, 2010.

[2] E. R. Faria, I. J. Gonçalves, A. C. de Carvalho, and J. Gama, "Novelty detection in data streams," *Artificial Intelligence Review*, vol. 45, no. 2, pp. 235–269, 2016.

[3] E. J. Spinosa, F. de Leon, A. Ponce, and J. Gama, "Novelty detection with application to data streams," *Intelligent Data Analysis*, vol. 13, no. 3, pp. 405–422, 2009.

[4] T. Al-Khateeb, M. M. Masud, L. Khan, C. Aggarwal, J. Han, and B. Thuraisingham, "Stream classification with recurring and novel class detection using class-based ensemble," in *Data Mining (ICDM), 2012 IEEE 12th International Conference on*. IEEE, 2012, pp. 31–40.

[5] E. R. de Faria, A. C. P. de Leon Ferreira, J. Gama *et al.*, "Minas: multiclass learning algorithm for novelty detection in data streams," *Data Mining and Knowledge Discovery*, vol. 30, no. 3, pp. 640–680, 2016.

[6] M. Masud, J. Gao, L. Khan, J. Han, and B. M. Thuraisingham, "Classification and novel class detection in concept-drifting data streams under time constraints," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 6, pp. 859–874, 2011.

[7] M. M. Masud, Q. Chen, L. Khan, C. Aggarwal, J. Gao, J. Han, and B. Thuraisingham, "Addressing concept-evolution in concept-drifting data streams," in *Data Mining (ICDM), 2010 IEEE 10th International Conference on*. IEEE, 2010, pp. 929–934.

[8] Z. S. Abdallah, M. M. Gaber, B. Srinivasan, and S. Krishnaswamy, "Anynovel: detection of novel concepts in evolving data streams," *Evolving Systems*, vol. 7, no. 2, pp. 73–93, 2016.

[9] I. Škrjanc, J. A. Iglesias, A. Sanchis, D. Leite, E. Lughofer, and F. Gomide, "Evolving fuzzy and neuro-fuzzy approaches in clustering, regression, identification, and classification: A survey," *Information Sciences*, vol. 490, pp. 344–368, 2019.

[10] T. P. Silva, L. Schick, P. A. Lopes, and H. A. Camargo, "A fuzzy multiclass novelty detector for data streams," in *2018 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE, 2018, in press.

[11] X. Xiong, K. L. Chan, and K. L. Tan, "Similarity-driven cluster merging method for unsupervised fuzzy clustering," in *Proceedings of the 20th conference on Uncertainty in artificial intelligence*. AUAI Press, 2004, pp. 611–618.

[12] D. Leite, P. Costa, and F. Gomide, "Evolving granular neural network for semi-supervised data stream classification," in *Neural Networks (IJCNN), The 2010 International Joint Conference on*. IEEE, 2010, pp. 1–8.

[13] S. Hashemi and Y. Yang, "Flexible decision tree for data stream classification in the presence of concept change, noise and missing values," *Data Mining and Knowledge Discovery*, vol. 19, no. 1, pp. 95–131, 2009.

[14] A. Isazadeh, F. Mahan, and W. Pedrycz, "Mflexdt: multi flexible fuzzy decision tree for data stream classification," *Soft Computing*, vol. 20, no. 9, pp. 3719–3733, 2016.

[15] P. P. Angelov and X. Zhou, "Evolving fuzzy-rule-based classifiers from data streams," *IEEE Transactions on Fuzzy Systems*, vol. 16, no. 6, pp. 1462–1475, 2008.

[16] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*. Springer US, 1981.

[17] R. Krishnapuram and J. M. Keller, "The possibilistic c-means algorithm: insights and recommendations," *IEEE transactions on Fuzzy Systems*, vol. 4, no. 3, pp. 385–393, 1996.

[18] N. R. Pal, K. Pal, J. M. Keller, and J. C. Bezdek, "A possibilistic fuzzy c-means clustering algorithm," *IEEE transactions on fuzzy systems*, vol. 13, no. 4, pp. 517–530, 2005.

[19] R. J. Campello and E. R. Hruschka, "A fuzzy extension of the silhouette width criterion for cluster analysis," *Fuzzy Sets and Systems*, vol. 157, no. 21, pp. 2858–2875, 2006.

[20] M. Hahsler, M. Bolanos, and J. Forrest, *stream: Infrastructure for Data Stream Mining*, 2016, r package version 1.2-3. [Online]. Available: https://CRAN.R-project.org/package=stream

[21] D. Meyer, E. Dimitriadou, K. Hornik, A. Weingessel, and F. Leisch, *e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien*, 2015, r package version 1.6-7. [Online]. Available: https://CRAN.R-project.org/package=e1071

[22] Computational Intelligence Group, "Data stream repository," Department of Computing, Federal University of São Carlos, São Carlos, Brazil, 2017. [Online]. Available: http://github.com/CIG-UFSCar/DS_Datasets

[23] E. R. de Faria, I. R. Goncalves, J. Gama, A. C. P. de Leon Ferreira *et al.*, "Evaluation of multiclass novelty detection algorithms for data streams," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 11, pp. 2961–2973, 2015.