Efficient Active Novel Class Detection for Data Stream Classification

Mohamed-Rafik Bouguelia, Yolande Belaïd, Abdel Belaïd

2014 22nd International Conference on Pattern Recognition

Efficient Active Novel Class Detection for Data Stream Classification

Mohamed-Rafik Bouguelia, Yolande Belaïd and Abdel Belaïd
Université de Lorraine - LORIA, UMR 7503
Vandoeuvre-les-Nancy, F-**54506**, France
Email: {mohamed.bouguelia, yolande.belaid, abdel.belaid}@loria.fr

***Abstract*—One substantial aspect of data stream classification is the possible appearance of novel unseen classes which must be identified in order to avoid confusion with existing classes. Detecting such new classes is omitted by most existing techniques and rarely addressed in the literature. We address this issue and propose an efficient method to identify novel class emergence in a multi-class data stream. The proposed method incrementally maintains a covered feature space of existing (known) classes. An incoming data point is designated as "insider" or "outsider" depending on whether it lies inside or outside the covered space area. An insider represents a possible instance of an existing class, while an outsider may be an instance of a possible novel class. The proposed method is able to iteratively select those insiders (resp. outsiders) that are more likely to be members of a novel (resp. an existing) class, and eventually distinguish the actual novel and existing classes accurately. We show how to actively query the labels of the identified novel class instances that are most uncertain. The method also allows us to balance between the rapidity of the novelty detection and its efficiency. Experiments using real world data prove the effectiveness of our approach for both the novel class detection and classification accuracy.**

I. Introduction

In usual classification methods, a classification model is built by performing several passes over a static dataset. This is not possible in the case of data streams where data is massively and continuously arriving from an infinite-length stream. Several methods for data stream classification have been proposed [1], [2], [3]. Most of these methods assume a fixed number of classes in the stream. However, in real world scenarios this assumption is often not satisfied. Indeed, it is difficult to obtain labelled instances from all possible classes. In addition, due to the evolving nature of the stream, novel (unknown) classes may appear at any time. If such novel classes are not detected, all of their instances will be inevitably misclassified in some existing (known) classes with which the model has already been trained. This major problem of data stream classification is generally referred to as "concept-evolution"[1]

A naive classification approach (e.g. SVM) is able to discriminate between classes that it has been trained with, but fails when it comes to classify undetected novel class instances. Novel class identification is thus very important since it allows us to considerably reduce the human labour that would be required to correct the misclassifications due to an undetected novel class. Furthermore, it may discover new patterns that we didn't even intend to look for.

[1]This is different from the so-called "concept-drift" [3] where changes concern the relation between existing data and their associated class labels.

Active learning methods [4] are convenient for data stream classification because they reduce the manual labelling cost, by querying from a human labeller only the class labels of data which are informative for learning (usually uncertain instances). However, usual stream-based active learning methods [4], [5], [6] implicitly assume that the data used for their initialization, covers all possible classes, and they query labels of instances whose informativeness is determined according to these known classes. Therefore, they fail at detecting novel classes. Novel class detection is not trivial for those algorithms because they should query labels in different unexplored regions of the feature space in order to detect possible novel classes. This represents an additional difficulty for active learning to reduce the label complexity [7].

Traditional one-class novelty and outlier detection methods [8], [9], [10] assume only one class of regular data and are unable to determine if an irregular data is a novel class instance or just an outlier. Some methods like [11] are based on clustering for novel class detection in data streams, but also assume that there is only one regular known class. Some active learning methods like [12], [13] provide strategies to discover unknown classes, but they need the whole dataset to be available beforehand which is not convenient for data streams. [14] proposes a data stream classification method with novel class detection. This method builds a decision boundary of known classes by applying K-means on large data stream chunks, which reduces the importance of its streaming nature. This method was also considered with active learning in [15]. However their uncertainty strategy is defined according to the known classes. Therefore, uncertain instances of known classes are presented together with all instances that are identified as novel, for manual labelling. The method we propose is different. It incrementally maintains a multi-class model which is able to identify the emergence of a novel class whose instances are self-similar. The method is able to select the instances for which we are most uncertain about their novelty, in order to query their true class labels.

We already proposed in [5] a stream-based semi-supervised active learning algorithm called A2ING. In this paper, we extend this algorithm to automatically detect the novel classes. The proposed novel class detection method incrementally maintains a dynamically evolving topology of nodes (data-representatives). Nodes are centers of hyperspheres covering regions of the feature space where data of known classes has been observed. A data point lying outside the covered area is possibly a member of a novel class. By collecting more outsiders, our algorithm is able to accurately distinguish the

1051-4651/14 $31.00 © 2014 IEEE
DOI 10.1109/ICPR.2014.487

outsiders that are self-similar and more likely to be members of a novel class, from those which are more likely to belong to some existing classes that are expanding. The algorithm queries the true class labels of only the instances that allow to best discriminate between the identified novel classes and the existing classes.

This paper is organized as follows. In Section II we briefly describe our previous algorithm A2ING. In Section III we introduce our method for novel class detection in multi-class data streams. In Section IV we present the experimental evaluation. Finally, we conclude and give limitations and future work in Section V.

II. Overview of A2ING

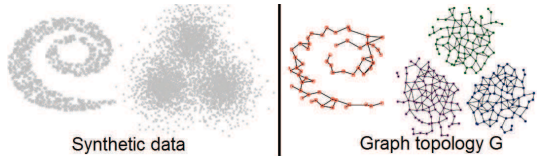

Fig. 1. A2ING incrementally learns the topology of data. Left: synthetic 2 dimensional data. Right: graph topology G of labelled nodes

A2ING [5] maintains a graph topology G (Fig. 1) of labelled nodes which is modified at each new data point arrival. Each node $n \in G$ is a continuously updated feature-vector. Let $x \in \mathbb{R}^p$ be a new data point from the stream. The algorithm operates in two steps:

Querying the class label of x: A2ING predicts y the class label of x, or queries it from a human labeller if it is uncertain. Let $P(y|x)$ be the probability that x belongs to a class y. $P(y|x)$ is proportional to the number of nodes that are labelled with y, among the K nearest nodes to x in G. Let P_1 and P_2 be respectively the first and the second highest probabilities, such that $P_1 \geq P_2$. The predicted class for x is the class with the highest probability (P_1). Let $\Delta_x = P_1 - P_2$. The data point x is more uncertain when Δ_x is smaller (closer to 0), because the probability that x belongs to its most probable class is close to the probability of belonging to its second most probable class. According to this measure, A2ING only queries the true class labels of uncertain data from the labeller.

Updating the graph topology G: at this step, we have a classified data point (x, y) where y is either the predicted or the queried class label of x. The graph G is updated using (x, y). Let n_1 and n_2 be the nearest and the second nearest nodes from x respectively. Given a distance threshold r, the algorithm incrementally updates the dynamic graph G, as follows:

- **if** $r < \text{dist}(x, n_1)$ **then** (1^{st} *case, Fig. 2.a*)
  - Add a new node n_{new} based on x and labelled with y
- **if** $\text{dist}(x, n_1) \leq r < \text{dist}(x, n_2)$ **then** (2^{nd} *case, Fig. 2.b*)
  - Add a new node n_{new} based on x and labelled with y
  - Link n_{new} to n_1 by a new edge
- **if** $\text{dist}(x, n_1) < \text{dist}(x, n_2) \leq r$ **then** (3^{rd} *case, Fig. 2.c*)
  - Link n_1 to n_2 by a new edge (if not linked)
  - Update n_1 to be closer to x if it is labelled with y, and farther otherwise

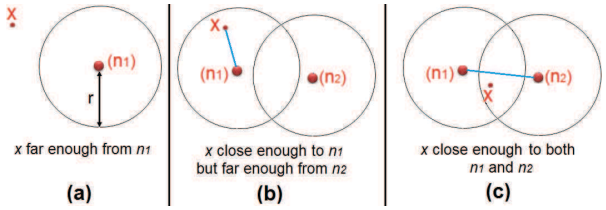

Fig. 2. A2ING's three cases for generating and adapting nodes

  - Update the neighbouring nodes of n_1 (linked to n_1 by an edge) to be closer to x if they are labelled with y, and farther otherwise.

For more details about querying uncertain instances and updating nodes, we can refer to [5].

A2ING focuses on querying labels of uncertain data whose uncertainty is determined according to known classes (i.e. Δ_x measures the ambiguity between the two most probable *existing* classes). Like the usual active learning methods, A2ING does not explore regions of the feature space where novel classes may appear. Therefore, it fails at detecting novel classes, except if -by chance- some novel class instances lies within its uncertainty region.

III. Proposed Novel Class Detection Method

In this section we extend A2ING by introducing a new method for novel class detection. We first present in Section III-A an adaptive strategy for outsiders detection. We explain in Section III-B how to accurately distinguish novel and existing classes. In Section III-C we show how to actively query labels of uncertain novel class instances.

A. Covered feature space and adaptive outsiders detection

Each node $n \in G$ constitutes a center of a hypersphere defined by the radius r. The covered feature space area is the union of all hyperspheres. An instance x is outside (resp. inside) the covered area if the distance to its nearest node is higher (resp. smaller) than r. This is straightforward since all hyperspheres has the same radius r.

$$x \stackrel{\text{def}}{=} outsider \iff \min_{n \in G} \text{dist}(x, n) > r$$

$$x \stackrel{\text{def}}{=} insider \iff \min_{n \in G} \text{dist}(x, n) \leq r$$

Given a fixed value of r, we want to figure out how much outsiders (resp. insiders) are effectively members of a novel (resp. existing) class. We perform a test using *optdigits* dataset[2] where we use 7 classes in the stream, and we keep 3 unknown classes to introduce them at time t together with the remaining existing class instances. Using different fixed values of r we determine the rate of (i) outsiders that are truly members of novel classes (novel precision), (ii) insiders that are truly members of existing classes (existing precision), (iii) novel classes instances that are detected as outsiders (novel recall), (iv) existing classes instances that are detected as insiders (existing recall), (v) F-scores for novel and existing

[2]http://archive.ics.uci.edu/ml/

classes (harmonic mean of precision and recall), and (vi) the overall accuracy of correct outsiders/insiders detection. Fig. 3 shows the correct distinction between novel and existing class instances according to different values of r. When r is too small, many data points are detected as outsiders and most of them are wrongly considered as novel class instances. Similarly, when r is too big, many data points are detected as insiders and most of them are wrongly considered as existing class instances. In Fig. 3, an optimal value of r for the considered dataset seems to be around 23. Actually, it is very difficult to manually initialize r with a convenient value, since it highly depends on the data. Therefore, since our algorithm is active and semi-supervised, we use the label information to automatically adjust the radius r.

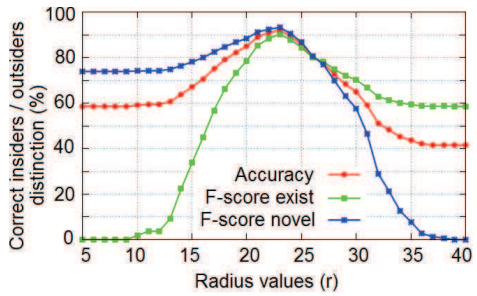

Fig. 3. The ability of correctly distinguishing novel and existing classes according to r values, on the optdigits dataset. An optimal value is around 23

Let x be a data point whose class label is predicted using A2ING or queried (if x is uncertain or outsider). Let n_1 be the nearest node from x, and ϵ (such that $0 < \epsilon \ll 1$) a small but constant learning rate. r is incrementally adapted as follows:

1) Assume x is an outsider; if $label(x) = label(n_1)$ then x is considered as a false-outsider with respect to the current value of r. In this case, we slightly increase r as:

$$r \leftarrow r + \epsilon \times |\operatorname{dist}(x, n_1) - r|$$

2) Assume x is an insider; if $label(x) \neq label(n_1)$ then x is considered as a false-insider with respect to the current value of r. In this case, we slightly decrease r as:

$$r \leftarrow r - \epsilon \times |\operatorname{dist}(x, n_1) - r|$$

We show in the experiments (Fig. 5) that for any initial value of r, it always converges to a value which is close to the optimal one (i.e. its initial value does not considerably affect the results). Therefore, r can be always initialized to the smallest possible initial value $r = 0$, since it is adaptive. The convergence of r can be analysed using *regret analysis*, though it is not done in this paper due to space limitation.

B. Accurate distinction of novel and existing classes

According to the previous subsection, we decide whether a new data point belongs to a novel or an existing class as soon as we receive it, depending on whether it is an outsider or an insider. Nonetheless, by collecting more data points, we are able to distinguish more accurately the novel and existing class instances, by determining the outsiders that are self-similar. This is possible when a waiting time is allowed until some more data are collected from the stream, or when a mini-batch of data is received at once from the stream instead of a single data point.

Let B be a mini-batch (buffer) of new unlabelled instances collected from the stream. These instances are initially separated into two sets: N (set of outsiders) and E (set of insiders) as follows:

$$N = \{x \in B : \min_{n \in G} \operatorname{dist}(x, n) > r\}$$

$$E = \{x \in B : \min_{n \in G} \operatorname{dist}(x, n) \leq r\}$$

Remember that A2ING maintains the graph topology G of labelled nodes summarizing the existing class instances (before integrating points of B). Let $\tilde{G}$ be the graph topology of unlabelled nodes obtained from N using Algorithm 1.

Algorithm 1 getUnlabelledGraph (N, r)

1: Initialize $\tilde{G}$ with two nodes chosen randomly from N
2: **for all** $x \in N$ **do**
3: Let n_1, n_2 be the two nearest nodes to x in $\tilde{G}$
4: **if** $\operatorname{dist}(x, n_1) > r$ **then**
5: Add a new node n_{new} to $\tilde{G}$
6: **if** $\operatorname{dist}(x, n_1) \leq r < \operatorname{dist}(x, n_2)$ **then**
7: Add a new node n_{new} to $\tilde{G}$
8: Link n_{new} to n_1 by a new edge
9: **if** $\operatorname{dist}(x, n_1) < \operatorname{dist}(x, n_2) \leq r$ **then**
10: Link n_1 to n_2 by a new edge
11: Update n_1 and its neighbouring nodes closer to x
12: **end for**
13: **return** $\tilde{G}$

Given a data point $x \in B$, let d_x^E and d_x^N denote the mean distance from x to its k nearest nodes from G and $\tilde{G}$ respectively:

$$d_x^E = \frac{\sum_{n \in S_x} \operatorname{dist}(x, n)}{k} \qquad d_x^N = \frac{\sum_{n \in \tilde{S}_x} \operatorname{dist}(x, n)}{k}$$

where S_x (resp. $\tilde{S}_x$) is the set of the k nearest nodes to x from G (resp. from $\tilde{G}$).

The probability $P(N|x)$ that x belongs to a novel class is proportional to the distance d_x^E (inversely proportional to d_x^N), and the probability $P(E|x)$ that x belongs to an existing class is proportional to the distance d_x^N. They are thus determined as follows:

$$P(N|x) = \frac{d_x^E}{d_x^E + d_x^N} \qquad P(E|x) = \frac{d_x^N}{d_x^E + d_x^N}$$

Algorithm 2 shows how to accurately distinguish the existing and novel class instances. Algorithm 1 is called at line 4 in order to get a graph of unlabelled nodes from N, which is used to determine the probabilities of belonging to novel or existing classes. At line 5, the set N' contains the false-novel instances which are identified as those in N that are more likely to belong to existing classes (having $P(N|x) < P(E|x)$). Similarly, at line 6 the set E' contains the false-existing instances. At lines 7 and 8, the two sets E and N are updated by moving the selected false-novel instances

Algorithm 2 distinguishExistingNovel (G, B, r)

$E \coloneqq \{x \in B \mid \min_{n \in G} \text{dist}(x, n) \leq r\}$ // insiders
$N \coloneqq \{x \in B \mid \min_{n \in G} \text{dist}(x, n) > r\}$ // outsiders
repeat
  $\tilde{G} = \text{getUnlabelledGraph}(N, r)$ // call Algorithm 1
  $N' \coloneqq \{x \in N \mid P(N|x) < P(E|x)\}$
  $E' \coloneqq \{x \in E \mid P(N|x) > P(E|x)\}$
  $N \coloneqq \{x \in N \mid x \notin N'\} \cup E'$
  $E \coloneqq \{x \in E \mid x \notin E'\} \cup N'$
until $N' = E' = \emptyset$ // no more points moves
return N, E

from N to E, and vice versa. This task is repeated until no more instance moves between the two sets N and E (or until a desired number of iterations is reached). Finally, the algorithm returns the sets N and E of instances which are identified as members of novel classes and existing classes respectively.

Fig. 4 shows the accuracy of distinguishing novel and existing classes in B, according to different iterations. The algorithm terminates only after 5 iterations. When no iteration is performed, the instances in B are just separated into outsiders and insiders, and the obtained accuracy of distinction among novel and existing classes is 88.15%. After just one iteration, the accuracy rise to 96.66%.

Depending on the application, it is possible to balance between the rapidity of the novelty detection and its efficiency by controlling the buffer's size $|B|$. For example, if a given application operates in real time and do not tolerate any waiting time, the buffer size can be set to 1, which turns out to be the same scenario as in Section III-A (i.e. an outsider will be immediately considered as a novel class instance). The influence of the size of B on the final results will be discussed in the experiments (Fig. 7).

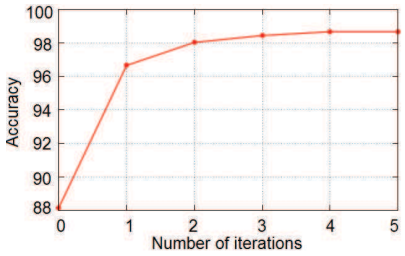

Fig. 4. Accuracy of distinguishing novel and existing classes at different iterations, on the optdigits dataset, in a buffer of size $|B| = 200$

C. Querying labels of the uncertain novel class instances

After distinguishing instances that are probably members of a novel class, the model should be updated with some labelled instances of that class. Querying the class labels of all the instances that are detected as novel is not convenient since labelling is costly and time consuming. It is possible to randomly select some instances from N for manual labelling. It is also possible to query the labels of the nodes obtained from N by applying Algorithm 1. Nonetheless, a more convenient strategy would be to query the class labels of instances in N for which the method is most uncertain about their novelty.

Uncertain novel class instances typically lie in the overlapping regions of novel classes with existing classes. An uncertain instance x would then have a low quantity $Q_x = |P(N|x) - P(E|x)|$, because the probability $P(N|x)$ that x belongs to a novel class would be close to the probability $P(E|x)$ that x belongs to an existing class. Therefore, instead of randomly selecting m instances from N for manual labelling, we select the m instances having the lowest Q_x values. Such instances are informative because knowing their true class labels would be useful to better discriminate between the novel and existing classes.

IV. Experimental Evaluation

We use for our experiments different public datasets obtained from the UCI machine learning repository[3]. We also consider a real administrative documents dataset provided by the ITESOFT[4] company. Each document was processed by an OCR and represented as a bag-of-words which is a sparse feature-vector containing the occurrence counts of words in the document.

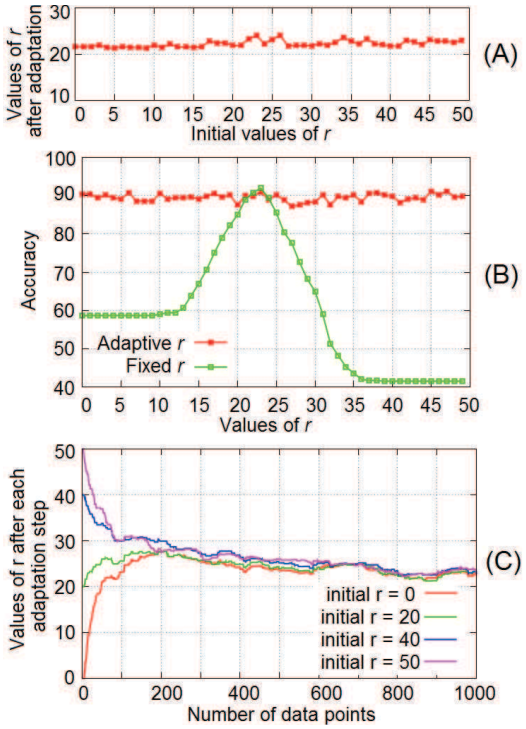

Fig. 5. Ability of detecting outsiders/insiders using an adaptive radius r, on the optdigits dataset

A. Ability of outsiders detection

The first set of experiments allows to figure out the ability of the proposed method to correctly detect outsiders and insiders using the adaptive radius r described in Section III-A. We consider that a data point is correctly detected as outsider (resp. insider) if it belongs to a novel (resp. existing) class. We mainly show that given a dataset and any initial value for r:

[3] http://archive.ics.uci.edu/ml/
[4] http://www.itesoft.com/

(i) it converges close to the same value, (ii) this value is close to the optimal one, and (iii) it quickly converges to this value (in terms of the number of data required).

Fig. 5 (A) shows the values of r obtained after adaptation, according to the initial values of r. We can see that whatever is the initial value of r, it always converges to values which are close to each other. Fig. 5 (B) show the accuracy of distinction between outsiders and insiders according to different values of r. We can see that when using a fixed value of r (i.e. manually set and not adaptive) we determine an optimal value around $r = 23$ for this dataset. Whereas, when using an adaptive r, the obtained accuracy is always close to the optimal whatever is the initial r value. Fig. 5 (C) shows the current value of r according to the number of data points seen from the stream, at different initializations of r. We can observe again that whatever is the initial value of r, it converges quickly without requiring many data points for its adaptation. This proves that the method is insensitive to initial values of r, thus, it can always be initialized to 0.

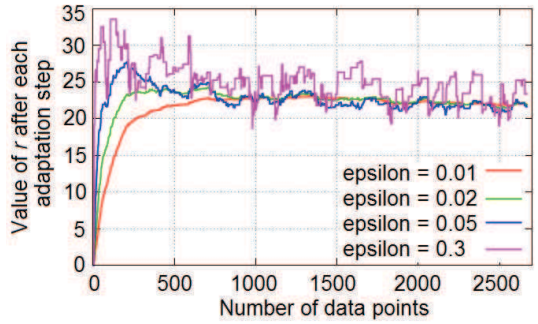

Fig. 6. The effect of ϵ on the adaptation of r, on the optdigits dataset

An intuitive parameter involved in the adaptation of r, is the learning rate ϵ. In order to show the effect of ϵ on r, Fig. 6 shows the current value of r according to the number of data points seen from the stream, using different values of ϵ. The learning rate is typically much less than 1 and slightly higher than 0 (i.e. $0 < \epsilon \ll 1$). For all values of ϵ, r eventually stabilizes, but this happens with different velocities according to the ϵ value. Indeed, when ϵ is low (e.g. 0.01), r changes slowly at each adaptation step. On the other hand, a high ϵ value (e.g. 0.3), makes r change more significantly at each adaptation step. For the reminder of the experiments we set ϵ to a constant value $\epsilon = 0.02$ which represents a convenient value for almost all datasets.

B. Ability to accurately distinguish existing and novel classes

The second set of experiments allows to figure out the ability of the proposed method to accurately detect the novel class instances and distinguish them from existing class instances as described in Section III-B.

Fig. 7 (A) shows the average time complexity required for the novel class detection according to different values of the buffer (mini-batch) size $|B|$. Fig. 7 (B) shows the corresponding accuracy of distinction between novel and existing class instances. The more instances are collected from the stream (i.e. higher buffer size), the higher the accuracy, but from

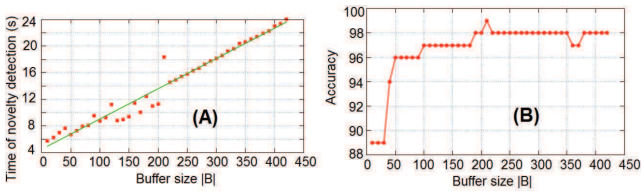

Fig. 7. Time and Accuracy according to the buffer size, on the optdigits dataset

a certain buffer size the accuracy becomes almost constant. This is due to the fact that collecting sufficient number of data points allows to more accurately detect the outsiders that are self-similar and confirms the presence of a novel class formed by those outsiders. As expected, Fig. 7 (A) shows that a higher buffer size implies a higher processing time, and that the required time increases linearly with the buffer size.

C. Comparison

We compare our proposed novel-class detection method to a well known one-class novelty detection method based on SVM [8] (we use the python implementation available on scikit-learn [16]). Both methods are used with A2ING as a base classifier in order to compare the final classification results. Each dataset is organized as a stream where novel classes are introduced progressively. Table I shows the obtained results in terms of the F-scores for novel and existing classes (F_N% and F_E%), the accuracy of distinguishing novel and existing classes correctly (Acc_1 %), and the final classification accuracy (Acc_2%). The F-score F_N is the harmonic mean of precision P_N and recall R_N. Similarly, F_E is the harmonic mean of precision P_E and recall R_E. They are expressed as:

$$F_N = \frac{2\times P_N \times R_N}{P_N + R_N} \quad \text{with} \begin{cases} P_N = \frac{\text{true\_novel}}{\text{true\_novel} + \text{false\_novel}} \\ R_N = \frac{\text{true\_novel}}{\text{true\_novel} + \text{false\_exist}} \end{cases}$$

$$F_E = \frac{2\times P_E \times R_E}{P_E + R_E} \quad \text{with} \begin{cases} P_E = \frac{\text{true\_exist}}{\text{true\_exist} + \text{false\_exist}} \\ R_E = \frac{\text{true\_exist}}{\text{true\_exist} + \text{false\_novel}} \end{cases}$$

where true_novel is the number of instances that are correctly identified as members of novel classes; false_novel is the the number of existing class instances that are wrongly identified as members of novel classes; true_exist is the number of instances that are correctly identified as members of existing classes; and false_exist is the number of novel class instances that are wrongly identified as members of existing classes. Acc_1 is expressed as

$$Acc_1 = \frac{\text{true\_novel} + \text{true\_exist}}{\text{true\_novel} + \text{false\_novel} + \text{true\_exist} + \text{false\_exist}}$$

The final classification accuracy Acc_2 represents the number of instances that are correctly classified in their corresponding classes on the total number of instances.

The SVM-based novelty detection method is considered in two cases: in the 1^{st} case, the known classes are represented by one SVM model (SVM Novelty V_1). In the 2^{nd} case, the known classes are represented by the union of several SVM models, each of them trained on one known class (SVM Novelty V_2). Our method is also considered in two cases: with

and without a buffer, i.e., $|B| = 200$ (mini-batch case) and $|B| = 1$ (online case) respectively.

Concerning the correct distinction between novel and existing classes, we can see from Table I (F_N, F_E, and Acc_1) that: (i) the SVM Novelty detection method performs better in the case where several SVM models are used (SVM Novelty V_2), and (ii) the proposed method always performs better than the SVM method, and much better in the mini-batch case. Regarding the final classification accuracy (Acc_2), Table I shows that it is generally proportional to the accurate distinction between novel and existing classes (Acc_1). However, in some cases like the "optdigit" and the "documents" datasets, the final classification accuracy is slightly better when "SVM Novelty V_2" is used, compared to the proposed method in the online case (without buffer). This is due to the fact that the final classification accuracy also depends on the informativeness of the missed (undetected) novel class instances.

TABLE I. COMPARATIVE RESULTS

Method	F_N **%**	F_E **%**	Acc_1 **%**	Acc_2 **%**
optdigits dataset				
SVM Novelty V_1	71.70	46.53	63.0	95.13
SVM Novelty V_2	86.19	79.80	83.6	*98.25*
Proposed (without buffer)	*91.32*	*86.04*	*89.3*	97.33
Proposed (with buffer)	**98.26**	**97.63**	**98.0**	**98.46**
pendigits dataset				
SVM Novelty V_1	83.38	62.66	77.0	95.01
SVM Novelty V_2	88.98	85.28	87.4	97.44
Proposed (without buffer)	*89.97*	*86.51*	*88.5*	*97.61*
Proposed (with buffer)	**94.73**	**93.54**	**94.2**	**98.30**
letters-recognition dataset				
SVM Novelty V_1	18.69	77.86	65.2	82.33
SVM Novelty V_2	55.98	*80.31*	72.8	82.42
Proposed (without buffer)	*74.60*	79.96	*77.6*	*84.67*
Proposed (with buffer)	**87.21**	**90.53**	**86.8**	**85.40**
documents dataset				
SVM Novelty V_1	29.52	89.38	81.0	88.47
SVM Novelty V_2	24.69	*88.24*	79.66	*88.91*
Proposed (without buffer)	*59.25*	88.17	*81.66*	88.69
Proposed (with buffer)	**90.84**	**91.14**	**91.0**	**90.08**
Average results over all datasets				
SVM Novelty V_1	50.82	69.10	71.55	90.23
SVM Novelty V_2	63.96	83.40	80.86	91.75
Proposed (without buffer)	*78.78*	*85.17*	*84.26*	*92.07*
Proposed (with buffer)	**92.76**	**93.21**	**92.5**	**93.06**

V. CONCLUSION AND FUTURE WORK

In this paper we addressed the novel class detection problem for data stream classification. Our method differs from the existing novel class detection methods in many aspects. First, it maintains a multi-class model of existing classes while being able to detect the emergence of novel classes in an infinite-length data stream. Second, it actively queries only the true class labels of the informative novel and existing class instances. Third, it does not assume that the data is generated from any specific distribution. Finally, it starts from scratch and it is adaptive. Our experimental evaluation on real datasets shows a very satisfying results achieved by our method. Nonetheless, one limitation of the proposed method is that its performance is negatively affected by the *rare and small* novel classes which highly *overlaps* with existing classes. The few existing methods for detecting such classes (like [12]) make assumptions which do not hold for data streams. This remains an open research challenge which has not been resolved yet for data streams. Moreover, since the proposed method depends on the distance function, it may suffer from the curse of dimensionality (in highly dimensional data) and from the presence of noisy features. However this can be tackled by using the shared-nearest-neighbour distance proposed in [17] which has been shown to improve results when there is a high number of irrelevant features. Another metric for high dimensional data has been proposed in [18] and can also be used as an alternative to the distance.

One future work that we also want to investigate in, is dealing with label noise. We are not always so lucky into obtaining a perfectly reliable label when querying it from a human labeller. Indeed, mislabelling may occur because the labeller may not be a domain expert. Therefore, detecting such labelling errors is an important issue. Theoretical analysis of the algorithm properties (like the eventual convergence) constitute another future work.

REFERENCES

[1] M.M. Gaber, A. Zaslavsky, and S. Krishnaswamy, *Survey of classification methods in data streams*. In Data Streams, Springer, pp. 39-59, 2007

[2] C.C. Aggarwal, *Data streams: models and algorithms*. Springer, Vol. 31, 2007

[3] I. Zliobaite, A. Bifet, B. Pfahringer, G. Holmes, *Active learning with evolving streaming data*. In ECML PKDD, pp. 597-612, 2011

[4] B. Settles, *Active learning*. In Synthesis Lectures on Artificial Intelligence and Machine Learning, pp. 1-114, 2012

[5] M.R. Bouguelia, Y. Belaïd and A. Belaïd, *A stream-based semi-supervised active learning approach for document classification*. In ICDAR, pp. 611-615, 2013

[6] A.B. Goldberg, et al. *OASIS: Online Active Semi-Supervised Learning*. In AAAI, 2011

[7] S. Dasgupta, *Coarse sample complexity bounds for active learning*. In NIPS, pp. 235-242, 2005

[8] B. Scholkopf, J. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson. *Estimating the support of a high-dimensional distribution*. Neural Computation, 13, pp. 1443-1471, 2001

[9] D. Agarwal, *An Empirical Bayes Approach to Detect Anomalies in Dynamic Multidimensional Arrays*, In ICDM, pp. 1-8, 2005

[10] S. Subramaniam, T. Palpanas, D. Papadopoulos, V. Kalogeraki, and D. Gunopulos, *Online Outlier Detection in Sensor Data Using Non-Parametric Models*, In VLDB, pp. 187-198, 2006.

[11] E.J. Spinosa, A.P. de Leon, F. de Carvalho, and J. Gama, *Cluster-Based Novel Concept Detection in Data Streams Applied to Intrusion Detection in Computer Networks*, In SAC, pp. 976-980, 2008.

[12] J. He, and J. Carbonell. *Nearest-neighbor-based active learning for rare category detection*, In NIPS, pp. 633-640, 2007.

[13] N.F. Escudeiro, and A.M. Jorge, *D-Confidence: an active learning strategy to reduce label disclosure complexity in the presence of imbalanced class distributions*. In JBCS, 18(4), pp. 311-330, 2012

[14] M.M. Masud, J. Gao, L. Khan, J. Han, B. Thuraisingham, *Classification and novel class detection in concept-drifting data streams under time constraints*. In TKDE, 23(6), 859-874, 2011

[15] M.M. Masud, J. Gao, L. Khan, J. Han, B. Thuraisingham, *Classification and novel class detection in data streams with active mining*. In Advances in Knowledge Discovery and Data Mining, Springer Berlin Heidelberg, pp. 311-324, 2010

[16] F. Pedregosa, et al., *scikit-learn, Scikit-learn: Machine Learning in Python*. In Journal of Machine Learning Research, pp. 2825-2830, 2011

[17] M.E. Houle, H.P. Kriegel, P. Krger, E. Schubert, A. Zimek, *Can shared-neighbor distances defeat the curse of dimensionality?*. In SSDBM, Springer Berlin Heidelberg, pp. 482-500, 2010

[18] J.C. Lamirel, *Enhancing classification accuracy with the help of feature maximization metric*. In ICTAI, pp. 1082-3409, 2013