Online Clustering for Novelty Detection and Concept Drift in Data Streams

Kemilly Dearo Garcia[1,2](✉), Mannes Poel[1], Joost N. Kok[1], and André C. P. L. F. de Carvalho[2]

[1] University of Twente, Enschede, The Netherlands
{k.dearogarcia,m.poel,j.n.kok}@utwente.nl
[2] ICMC, University of São Paulo, São Paulo, Brazil
andre@icmc.usp.br

Abstract. Data streams are related to large amounts of data that can continuously arrive with a probability distribution that may change over time. Depending on the changes in the data distribution, different phenomena can occur, like new classes can appear or concept drift can occur in existing classes. Machine Learning algorithms have been often used to model this data. New classes are patterns that were not seen during the training of the current classification model, but appear after some time. Concept drift occurs when the concepts associated with a dataset change as new data arrive. This paper proposes a new algorithm based on kNN that uses micro-clusters as prototypes and incrementally updates the micro-clusters or creates new micro-clusters when novelties are detected. In the online phase, each instance close to a micro-cluster is considered an extension of the micro-cluster, being used to adapt the model to concept drift. The proposed algorithm is experimentally compared with a state-of-the-art classifier from the data stream literature and one baseline. According to the experimental results, the proposed algorithm increases the predictive performance over time by incrementally learning changes in the data distribution.

Keywords: Data stream · Concept drift · Novelty detection · Online learning

1 Introduction

Data streams are known as data that can continuously arrive in streams, with a probability distribution that can change over time [14]. As new data arrive, models previously induced can become outdated [6]. In addition, due to the great amount of data generated, it is not feasible to store all incoming data in the main memory, requiring the removal of previous outdated data and online processing of incoming data [12,16].

Depending on the changes in the data distribution, different phenomena can occur, like concept drifts [14,25] and novelties [12,20]. Concept drift refers to changes in the concept definitions of a normal class [15]. Novelties concepts are are patterns that were not present during the training of the classification model, but appear later on in the data stream [12]. In these situations, it is important to adapt the classification model to the current data distribution, otherwise its predictive performance can decrease along the time.

In this work, normal concepts are a set of *normal classes* used to train the classification model and novelties concepts are the *new classes* that emerge in a data stream along the time [12].

Novelty detection is a Machine Learning (ML) task based on the identification of novelties in the data [11]. In data streams, the novelty detection can be divided in two phases: *offline* and *online phase*. In the offline phase, a classification model is trained using an initial, static, labeled dataset. In the online phase, the model is updated using unlabeled data arriving in streams. The update occurs when the predictive performance of the model decreases, usually because of change in the data distribution. Thus, the model can be continuously updated [15].

One of the strategies to deal with novelty detection and concept drift is by explicitly detecting changes in parts of the stream, comparing the current concept with previous concepts from time to time [19]. An example of this strategy is to continuously calculate the model classification error. This strategy assumes that the data arriving in the stream are labeled.

Another strategy is to store in a buffer the potential novelty classes instances. However, the use of a buffer with fixed size may ignore instances with relevant information about persistent concepts. Furthermore, the size of the buffer affects its efficient use when the degree and speed of changes vary in the data stream. Another deficiency of updating the model using a buffer with fixed size is the possibility to forget old, but relevant, information.

There are two problems in assuming that the arriving data is labeled. First, the process of labeling an instance usually has a cost, which increases with the complexity and need of domain expertise. Second, if the data arrives in high speed, there will not be enough time to label them. Thus, in this paper we assume that the instances in a data stream come unlabeled.

Due to the lack of labeled data in a data streams, the update of the model can rely only on the predictive attribute values. To deal with this limitation it is possible to use clustering algorithms. Clusters can summarize the main data profiles present in a data stream and be updated to incorporate changes in class profiles and detect the appearance of novelties [2]. When clustering algorithms are applied to data streams, micro-clusters can be used as a strategy to summarize data present in different periods of time [3]. Each micro-cluster can be structured as a temporal extension of a CF (Cluster Feature Vector) [24], which is a compact statistical representation of a set of instances.

In this paper we propose *Higia*, a novelty detection algorithm based on kNN (k-Nearest Neighbor) that uses *micro-clusters* [3] as prototypes and incrementally updates the micro-clusters or creates new micro-clusters when a novelty is detected. *Higia* training is divided into offline learning and online learning. During the offline learning phase, we assume that there is data from one or more normal classes. The instances from each normal class are summarized into a set of micro-clusters. Each micro-cluster has instances from the same normal class label. In the online learning phase, each instance close to a micro-cluster is considered an extension of the micro-cluster, a concept drift. This instance is then used to adapt the predictive model to this concept drift. However, if a set of new instances are close together in a dense region, they are considered representative of new classes, named novelties.

This paper is structured as follows. Section 2, presents previous related works for novelty and concept drift detection in data streams. Data stream, novelty, concept drift and micro-clusters are introduced in Section 3. The proposed algorithm, Higia, is described in Sect. 4. Section 5 presents the experimental setup and analyses the results obtained. Finally, Sect. 6 has the main conclusions from this study and points out future work directions.

2 Related Work

This section briefly presents previous works using ML-based approaches for novelty and concept drift detection in data streams. Most of these studies use supervised ML algorithms, classification algorithms, to induce classifiers.

Most of the classification algorithms proposed for data stream mining are based on online learning [9,12,15,19]. Some of them continuously update the classification model using true labeled data [1,4,21]. However, as previously mentioned, true labels are not always available at feasible time, delaying the updating of the classification model. Assuming the absence of labels in the online phase, others apply clustering algorithms in the arriving data. Thus, the clusters are representatives of normal and new classes [5,12,17,23].

One of the first algorithms to use clusters for novelty detection in data streams is OLINDDA (OnLIne Novelty and Drift Detection Algorithm) [23], [22]. During the offline phase a single model is build by a set of clusters with data from the normal classes. After, in the online phase, whenever a new instance arrives, it is calculated the distance between it and the closest cluster from the normal model. When the distance is large, according to a threshold value, the instance is stored in a buffer, where it can later be defined as a novelty after a clustering step.

ECSMiner [21] (Enhanced Classifier for Data Streams with novel class Miner) is an ensemble of models, each model is represented by a set of clusters created using the clustering algorithm k-means. ECSMiner also stores in a buffer the instances that are distant from the normal clusters. The ensemble is updated when the instances stored in the buffer receive their true label. Afterwards, the ensemble predictive accuracy is calculated. The model with the lowest accuracy is updated with the novelties found in the buffer. While waiting for labeled data, the model can wait for a long period of time to be updated, which could reduce the accuracy of the ensemble. Besides, it is not always guaranteed that all data will be labeled, since it may be application dependent.

Another novelty detection algorithm, MINAS (MultI-class learNing Algorithm for data Streams) [12], also uses an offline phase followed by an online phase. In its offline phase, the data is separated by labels in subsets. From each subset it is generated a set of micro-clusters representing each class. In the online phase, the incoming data is stored in a buffer if is not identified by the model. When the buffer reaches a certain size, a clustering algorithm is applied in the data stored in the buffer. Valid micro-clusters are marked as extension of a known class or as novelty, and are incorporated into the classification model.

3 Problem Formalization

A data stream is a sequence of instances, potentially of infinity size, that can be formally represented by [12,15,16]: $D_{tr} = \{(X_1, y_1), (X_2, y_2), ..., (X_{tr}, y_{tr})\}$, where X_{tr} is an instance arriving in time tr and y_{tr} is the target class of this instance. Due to finite resources, each instance must be processed only once.

Concept drift is a change in the distribution probability of a problem target classes [15]. Formally, the joint probability distribution $P_{tr+1}(X, y)$ over instances X and label y can change over time, defined by tr, so that $P_{tr}(X, y) \neq P_{tr+1}(X, y)$.

Assuming that in the offline phase a dataset has m classes, the set $Y^{Nor} = \{y_1, y_2, ..., y_m\}$ represents the set of Normal Classes. These classes are used to build the initial classification model. Afterwards, during the online phase, a novel class, y_{m+1}, has the following property: $y_{m+1}, \notin Y^{Nor}$ i.e., y_{m+1} was not used in the training of the classification model, but emerges during the online phase. Therefore, for any given new set of novel classes, $Y^{New} = \{y_{m+1}, y_{m+2}, ..., y_n\}$ any novelty detection approach must be able to fast detect novelties as they appear [12,16]. Considering the sets of normal classes and new classes, the total set of classes is simply defined as $Y = Y^{Nor} \cup Y^{New}$.

Micro-clustering [3] is a strategy commonly used to summarize data coming from a stream in different periods of time. Each micro-cluster $C_j = (n, \boldsymbol{LS}, \boldsymbol{SS}, t)$ stores four components: the number of its instances n, the linear sum of its instances $\boldsymbol{LS}$, the square sum of its instances $\boldsymbol{SS}$ and the timestamp, t, of when the last instance was incorporated in the micro-cluster. By using these values, it is possible to calculate the centroid ($c_j = \boldsymbol{LS}/n$) and the radius ($r_{C_j} = 2 \times (\boldsymbol{SS} \times n/n^2 + \boldsymbol{SS}/n^2)$) of a micro-cluster, which can be used to classify new instances.

4 Methodology

The proposed algorithm, Higia, is based on the assumption that in a data stream, an ideal classifier should be able to learn the current concept in feasible time without forgetting relevant past information [9].

Higia induces a classification model using unsupervised online learning. Its training also occurs by an offline phase followed by an online phase. In the offline phase, a predictive model, illustrated by Fig. 1, is induced from a batch containing labelled data. As in [12], the model is composed by micro-clusters created using the CluStream algorithm [3].

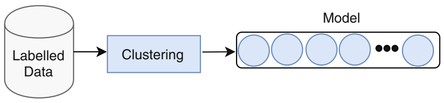


Fig. 1. Higia offline phase.

In the online phase, illustrated by Fig. 2, as new data arrives, the model classify each new instance as *normal*, *extension* or *unknown*. For the classification, the model calculates the euclidean distance between each instance and the centroids of the micro-clusters from the *normal* classes.

Algorithm 1. Higia: Online Phase

```
1: input: $X_{tr}$, $T$, $k$
2: Let $\psi_k$ be a list of the $k$ nearest micro-clusters to $X_{tr}$
3: if majority of $\psi_k$ have the same label then
4:     Let $C_j$ be the nearest micro-cluster to $X_{tr}$
5:     Let $c_j$ be the centroid of $C_j$
6:     Let $radius\,(C_j)$ be the radius of $C_j$
7:     $dist \leftarrow EuclidianDistance(X_{tr}, C_j)$
8:     if dist $\leq radius\,(C_j)$ then
9:         update $C_j$ with $X_{tr}$
10:        classify $X_{tr}$ with the same label of $C_j$
11:    else if $dist \leq (radius\,(C_j) \times T)$ then
12:        create extension of $C_j$ with centroid $X_{tr}$ and radius 0.5
13:        classify $X_{tr}$ with the same label of $C_j$
14: else
15:    add $X_{tr}$ to buffer
16:    classify $X_{tr}$ as unknown
```

Algorithm 1 describes how Higia works in the online learning phase when a new instance, X_{tr}, arrives. First, Higia finds the k micro-clusters closest to X_{tr}. If the majority of these micro-clusters have the same label and if the smallest distance is less than the radius of the nearest micro-cluster C_j, the instance is classified with the label of C_j. Besides, C_j is updated with X_{tr}. If the distance is larger than the radius of C_j, but is smaller than a given threshold T, the instance is added to the model as an extension. The threshold is multiplied by the radius of C_j and indicates the maximum drift of C_j.

If the distance is larger than the radius of C_j and T, then X_{tr} is labeled as *unknown* and stored in a buffer. The instances stored in the buffer are incrementally forming micro-clusters. When a micro-cluster has a given amount of instances, defined by a hyperparameter, a novelty is found and the new micro-cluster is added to the model as a new class.

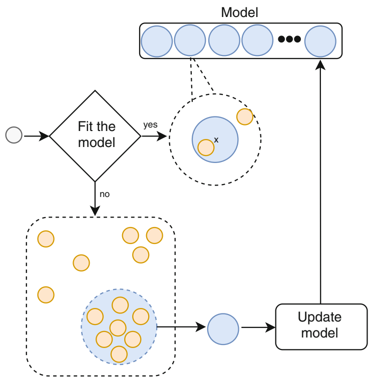


Fig. 2. Higia online phase.

5 Experimental Evaluation

In this section, we present the experiments carried out to assess the predictive performance of Higia. In this experiments, Higia was compared with two other algorithms, MINAS and the kNN algorithm. From the algorithms described in Sect. 2, MINAS is the only unsupervised algorithm for multiclass novelty detection. The kNN algorithm from the MOA framework [8] was used as a baseline. In the offline phase, all three algorithms are initialized with the same batch of labeled data representing 10% of the dataset. We assume that the instances in the training data are from the normal classes and that new classes can continuously appear during the online phase.

We adopted the accuracy measure to evaluate the predictive performance of the models over time. The accuracy is the amount of correctly classified data divided by the amount of total data in a window of 1000 instances. We also use the total accuracy measure to evaluate the performance for the whole dataset.

For a more detailed analyses we used the evaluation approach proposed in [21]. This evaluation approach has 3 measures: M_{New} = the percentage of instances from the new classes misclassified as existing class, F_{New} = the percentage of instances from the normal classes classified as novelties, and Err = average misclassification error.

The accuracy counts as mistake the instances classified as *unknown*. To analyse the algorithms in terms of misclassification of the total classes, (Y), without the influence of *unknown*, we used the Err measure.

5.1 Datasets

The experiments were performed on both synthetic and real datasets, commonly used in novelty detection studies [3,4,10,13,18,21]. The synthetic datasets are: MOA, SynD, CDT, UG and Gear. The real dataset used was the Forest Cover dataset.

The *MOA* dataset [12] has concept drift, appearance of new classes, recurrence and disappearance of existing classes. The real shape of the clusters in this dataset is normally distributed hyper-spheres. The *SynD* [4,13,21] has no novelty, but the concept associated with known classes changes over time. Finally, *CDT, UG* and *Gear* are non stationary datasets[1]. In these datasets, a single novelty occurs and concept drifts happens at every interval of 400 instances in CDT, 1000 instances in UG and 2000 instances in Gear.

Regarding the only real dataset, *Forest Cover* [7], it contains observations from 7 types of forest in the United States. It has 7 classes and 54 attributes. In the experiments, the training set is formed by observations from 3 normal classes.

Table 1 presents some basic statistics collected from these datasets: the number of normal and new classes, number of attributes, number of instances in the minority and majority classes.

Table 1. Statistical information for each dataset

Statistics	1CDT	MOA	Gear	UG	SynD	Forest Cover
Attributes	2	4	2	2	10	54
Classes	2	4	2	2	2	7
Normal classes	1	2	2	1	2	3
New classes	1	2	0	1	0	4
Instances MinCla	7199	9987	99935	44999	124660	587
Instances MajCla	7200	18180	100065	45000	125340	18350

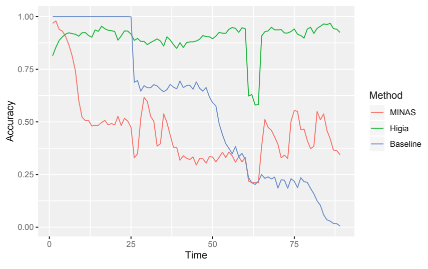
5.2 Results and Discussion

In these experiments we used the default parameters of MINAS. Because MINAS uses the label of the nearest micro-cluster to classify a new instance, we set $k = 1$ in the Higia algorithm. As for the other parameter of Higia, the *threshold*, we set it to $threshold = 1.1$ as in MINAS [12]. Also for Higia, we experimentally defined the parameter $radius = 0.5$. The kNN has the default parameters, $k = 1$ and window size ($w = 1000$), and there is no online training. It is used to understand how the model looses predictive accuracy over time without its update to the changes in the data stream.

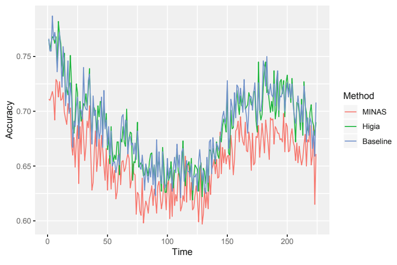
Next, we present the predictive accuracy over time of the 3 algorithms. As can be seen in Fig. 3, the predictive performance of Higia was better than those obtained by MINAS and the kNN baseline. Higia incrementally updates the normal and the new micro-clusters in the classification model. As consequence, model represents the current probability distribution of the data streams better

[1] https://www.sites.google.com/site/nonstationaryarchive/.

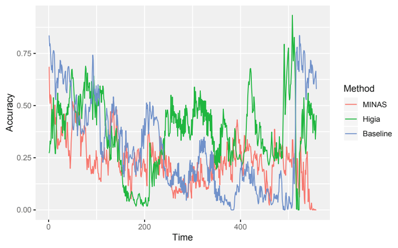
than MINAS. The model induced and updated by the MINAS algorithm is sensitive to the buffer size, i.e., the model needs to wait until the buffer is full of *unknown* instances to be updated. Thus, by not being able to quickly adapt to the changes in the data stream, the model loose accuracy along the time.



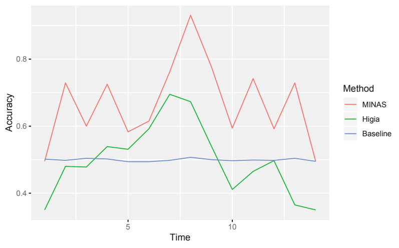
(a) MOA



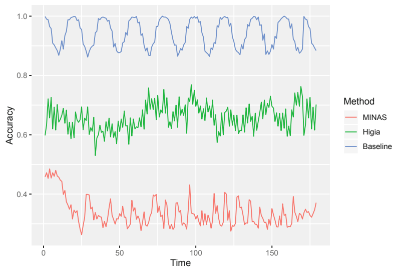
(b) SynD



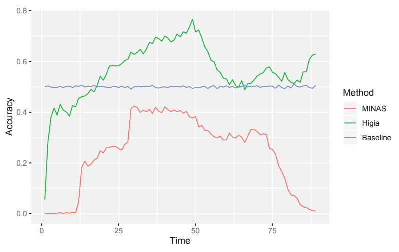
(c) Covertype



(d) CDT



(e) GEARS



(f) UG

Fig. 3. Predictive accuracy over time

The baseline, kNN, presented the worst predictive performance in most of the datasets. This is somewhat expected, since it does not learn along the time. However, we can see from Fig. 3e that, for one of the datasets, the baseline had the best predictive accuracy. A possible reason is that the training set of this dataset has data from all classes. Meanwhile, for this dataset, different from the baseline, the generalization of the models induced by MINAS and Higia was not able to capture the behavior of the data. This indicates the importance of the initial training even in data stream scenarios with concept drift.

Table 2 summarizes the predictive performance obtained by the Higia and MINAS algorithms in all datasets. According to these results, for the datasets *MOA, CDT and UG*, Higia had the lowest F_{New}, but the highest M_{New}. This shows that Higia models gave more relevance to the normal classes than to the novelties. The only exception is the *CoverType* dataset. However, these results did not seem to affect the accuracy, *Acc*, of Higia in most datasets. Higia presented the lowest *Err* values in almost all datasets, meaning that, for these datasets the model made less missclassifications and labeled less instances as *unknown* than MINAS.

Table 2. Performance metrics for each algorithm

	M_{New}		F_{New}		Err		Acc	
	Higia	MINAS	Higia	MINAS	Higia	MINAS	Higia	MINAS
MOA	0.11	0.00	0.00	0.46	**0.08**	0.48	**0.62**	0.46
SynD	0.00	0.00	0.00	0.00	**0.30**	0.34	**0.70**	0.66
CoverType	0.18	0.46	0.49	0.24	**0.49**	0.54	**0.37**	0.23
CDT	0.43	0.00	0.01	0.00	0.46	**0.03**	0.47	**0.68**
GEARS	0.00	0.00	0.00	0.00	**0.34**	0.66	**0.66**	0.34
UG	0.78	0.41	0.03	0.36	**0.60**	0.70	**0.68**	0.24

An exception occurred for the dataset *CDT*, which has a seasonal overlap between the normal class and the new class. For this dataset, Higia incremental learning mixed instances from different classes into the same micro-clusters, reducing *Acc* and increasing *Err*.

Additional experiments were performed to analyse the impact of the parameter k ($k = 1, 3, 5, 10, 20$) in Higia predictive accuracy. We used the *CDT* dataset because Higia had a lower performance with this dataset. Figure 4 shows that, for this dataset, the accuracy is more stable over time for higher k. We also see a reduction of *Err*, Table 3, with the increase of the parameter k.

Table 3. Higia ($k = 1, 3, 5, 10, 20$) performance metrics for dataset CDT

	M_{New}	F_{New}	Err	ACC
1NN	0.43	0.01	0.46	0.47
3NN	0.42	0.01	0.41	0.51
5NN	0.38	0.01	0.32	0.54
10NN	0.32	0.02	0.32	0.47
20NN	0.45	0.02	0.35	0.60

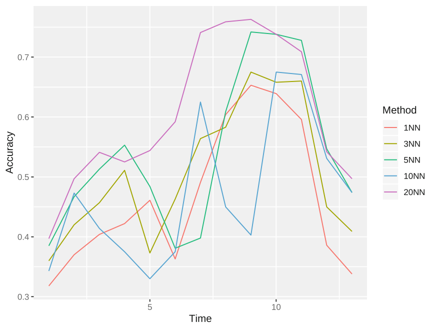


Fig. 4. Higia accuracy over time in CDT dataset varying k.

6 Conclusions and Future Work

Data stream mining has gained a great deal of attention in the last decades. Novelty detection algorithms have been successfully applied to many applications. However, most of the proposed algorithms assume that instances arriving in a stream are labeled, which is often not the case.

This paper presented Higia, a new unsupervised learning algorithm based on micro-clusters for novelty and concept drift detection in data streams. The micro-clusters are incrementally updated every time a new instance arrives from a data stream. When a novelty is detected, Higia creates new micro-clusters to represent the new class.

Considering several performance metrics, Higia was compared with MINAS, a state-of-the-art unsupervised novelty detection algorithm, and a k-NN-based baseline. According to the experimental results, Higia presented a better predictive performance than these other algorithms. Besides, Higia labeled less instances as *unknown* because it can faster adapt the model to the current concept than the compared algorithms.

As future work, the authors want to study alternatives to discard outdated information and the inclusion of an unsupervised concept drift tracker.

References

1. Abdallah, Z.S., Gaber, M.M., Srinivasan, B., Krishnaswamy, S.: Anynovel: detection of novel concepts in evolving data streams. Evol. Syst. **7**(2), 73–93 (2016). https://doi.org/10.1007/s12530-016-9147-7
2. Aggarwal, C.C.: Data Streams: Models and Algorithms, vol. 31. Springer, Heidelberg (2007). https://doi.org/10.1007/978-0-387-47534-9
3. Aggarwal, C.C., Han, J., Wang, J., Yu, P.S.: A framework for clustering evolving data streams. In: VLDB 2003, Proceedings of 29th International Conference on Very Large Data Bases, Berlin, Germany, 9–12 September 2003, pp. 81–92 (2003)

4. Al-Khateeb, T., Masud, M.M., Khan, L., Aggarwal, C.C., Han, J., Thuraisingham, B.M.: Stream classification with recurring and novel class detection using class-based ensemble. In: 12th IEEE International Conference on Data Mining, ICDM 2012, Brussels, Belgium, 10–13 December 2012, pp. 31–40 (2012). https://doi.org/10.1109/ICDM.2012.125
5. Amini, A., Teh, Y.W., Saboohi, H.: On density-based data streams clustering algorithms: a survey. J. Comput. Sci. Technol. **29**(1), 116–141 (2014). https://doi.org/10.1007/s11390-014-1416-y
6. de Andrade Silva, J., Faria, E.R., Barros, R.C., Hruschka, E.R., de Carvalho, A.C.P.L.F., Gama, J.: Data stream clustering: a survey. ACM Comput. Surv. **46**(1), 13:1–13:31 (2013). https://doi.org/10.1145/2522968.2522981
7. Asuncion, A., Newman, D.: UCI machine learning repository (2007). http://www.ics.uci.edu/~mlearn/MLRepository.html
8. Bifet, A., et al.: MOA: massive online analysis, a framework for stream classification and clustering. In: Proceedings of the First Workshop on Applications of Pattern Analysis, WAPA 2010, Cumberland Lodge, Windsor, UK, 1–3 September 2010, pp. 44–50 (2010). http://www.jmlr.org/proceedings/papers/v11/bifet10a.html
9. Bifet, A., Pfahringer, B., Read, J., Holmes, G.: Efficient data stream classification via probabilistic adaptive windows. In: Proceedings of the 28th Annual ACM Symposium on Applied Computing, SAC 2013, Coimbra, Portugal, 18–22 March 2013, pp. 801–806 (2013). https://doi.org/10.1145/2480362.2480516
10. Cao, F., Ester, M., Qian, W., Zhou, A.: Density-based clustering over an evolving data stream with noise. In: Proceedings of the Sixth SIAM International Conference on Data Mining, Bethesda, MD, USA, 20–22 April 2006, pp. 328–339 (2006). https://doi.org/10.1137/1.9781611972764.29
11. Ding, X., Li, Y., Belatreche, A., Maguire, L.P.: An experimental evaluation of novelty detection methods. Neurocomputing **135**, 313–327 (2014). https://doi.org/10.1016/j.neucom.2013.12.002
12. Faria, E.R., Gama, J., de Carvalho, A.C.P.L.F.: Novelty detection algorithm for data streams multi-class problems. In: Proceedings of the 28th Annual ACM Symposium on Applied Computing, SAC 2013, Coimbra, Portugal, 18–22 March 2013, pp. 795–800 (2013). https://doi.org/10.1145/2480362.2480515
13. Faria, E.R., Gonçalves, I.J.C.R., de Carvalho, A.C.P.L.F., Gama, J.: Novelty detection in data streams. Artif. Intell. Rev. **45**(2), 235–269 (2016). https://doi.org/10.1007/s10462-015-9444-8
14. Gama, J.: Knowledge Discovery from Data Streams. Chapman and Hall/CRC Data Mining and Knowledge Discovery Series. CRC Press, Boca Raton (2010)
15. Gama, J., Zliobaite, I., Bifet, A., Pechenizkiy, M., Bouchachia, A.: A survey on concept drift adaptation. ACM Comput. Surv. **46**(4), 44:1–44:37 (2014). https://doi.org/10.1145/2523813
16. Garcia, K.D., de Carvalho, A.C.P.L.F., Mendes-Moreira, J.: A cluster-based prototype reduction for online classification. In: Yin, H., Camacho, D., Novais, P., Tallón-Ballesteros, A.J. (eds.) IDEAL 2018. LNCS, vol. 11314, pp. 603–610. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-03493-1_63
17. Hayat, M.Z., Hashemi, M.R.: A DCT based approach for detecting novelty and concept drift in data streams. In: Second International Conference of Soft Computing and Pattern Recognition, SoCPaR 2010, Cergy Pontoise/Paris, France, 7–10 December 2010, pp. 373–378 (2010). https://doi.org/10.1109/SOCPAR.2010.5686734

18. Ienco, D., Zliobaite, I., Pfahringer, B.: High density-focused uncertainty sampling for active learning over evolving stream data. In: Proceedings of the 3rd International Workshop on Big Data, Streams and Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications, BigMine 2014, New York City, USA, 24 August 2014, pp. 133–148 (2014). http://jmlr.org/proceedings/papers/v36/ienco14.html
19. Losing, V., Hammer, B., Wersing, H.: KNN classifier with self adjusting memory for heterogeneous concept drift. In: IEEE 16th International Conference on Data Mining, ICDM 2016, Barcelona, Spain, 12–15 December 2016, pp. 291–300 (2016). https://doi.org/10.1109/ICDM.2016.0040
20. Markou, M., Singh, S.: Novelty detection: a review - part 1: statistical approaches. Sig. Process. **83**(12), 2481–2497 (2003). https://doi.org/10.1016/j.sigpro.2003.07.018
21. Masud, M.M., Gao, J., Khan, L., Han, J., Thuraisingham, B.M.: Classification and novel class detection in concept-drifting data streams under time constraints. IEEE Trans. Knowl. Data Eng. **23**(6), 859–874 (2011). https://doi.org/10.1109/TKDE.2010.61
22. Spinosa, E.J., de Carvalho, A.C.P.L.F., Gama, J.: Cluster-based novel concept detection in data streams applied to intrusion detection in computer networks. In: Proceedings of the 2008 ACM Symposium on Applied Computing (SAC), Fortaleza, Ceara, Brazil, 16–20 March 2008, pp. 976–980 (2008)
23. Spinosa, E.J., de Carvalho, A.C.P.L.F., Gama, J.: Novelty detection with application to data streams. Intell. Data Anal. **13**(3), 405–422 (2009). https://doi.org/10.3233/IDA-2009-0373
24. Zhang, T., Ramakrishnan, R., Livny, M.: BIRCH: an efficient data clustering method for very large databases. In: Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data, Montreal, Quebec, Canada, 4–6 June 1996, pp. 103–114 (1996). https://doi.org/10.1145/233269.233324
25. Zliobaite, I., Bifet, A., Pfahringer, B., Holmes, G.: Active learning with drifting streaming data. IEEE Trans. Neural Netw. Learn. Syst. **25**(1), 27–39 (2014). https://doi.org/10.1109/TNNLS.2012.2236570