Towards Real-Time Feature Tracking Technique using Adaptive Micro-Clusters

Mahmood Shakir Hammoodi, Frederic Stahl, Mark Tennant, Atta Badii

Abstract

Data Streams are unbounded, sequential data instances that are generated with high velocity. Classifying sequential data instances is a very challenging problem in machine learning with applications in network intrusion detection, financial markets and sensor networks. Data Stream classification is concerned with the automatic labelling of unseen instances from the stream in real-time. For this the classifier needs to adapt to concept drifts and can only have one pass through the data if the stream is fast. This research paper presents our work on a real-time pre-processing technique, in particular feature tracking technique that take concept drift into consideration. The feature tracking technique is aimed to improve Data Stream Mining (DSM) classification algorithms through enabling real-time feature selection. The technique is based on adaptive summaries of the data and class distributions, known as Micro-Clusters. Currently the technique is able to detect concept drift and identifies which features have been involved.

Key words: Data Stream Classification, High Velocity data Streams, Concept Drift, Feature Selection

Mahmood Shakir Hammoodi, Frederic Stahl, Mark Tennant, Atta Badii
Department of Computer Science, University of Reading, PO Box 225, Whiteknights, Reading, RG6 6AY, UK,e-mail: m.s.h.hammoodi@pgr.reading.ac.uk, pre.mahmood.shakir@uobabylon.edu.iq, F.T.Stahl@reading.ac.uk, m.tennant@pgr.reading.ac.uk, atta.badii@reading.ac.uk

1 Introduction

Velocity in Big Data Analytics [1] refers to data that is generated at a high speed in real-time and challenges our computational capabilities in terms of storing and processing the data [2]. DSM requires techniques that are incremental, computationally efficient and can adapt to concept drift for applications such as real-time analytics of chemical plant data in the chemical process industry [3], intrusion detection in telecommunications [4], etc. A concept drift occurs if the pattern encoded in the data stream changes. DSM has developed various real-time versions of established predictive data mining algorithms that adapt to concept drift and keep the model accurate over time, such as CVFDT [5] and G-eRules [6]. The benefit of classifier independent concept drift detection methods is that it gives information about the dynamics of the data generation [7]. Common drift detection methods are for example ADaptive sliding WINdow (ADWIN) [8], Drift Detection Method (DDM) by [9] and the Early Drift Detection Method (EDDM)[10]. To the best of our knowledge, no drift detection method provides insights into which features are involved in the concept drift, which is potentially valuable information. For example, if a feature is contributing to a concept drift it can be assumed that the feature may have become either more or less relevant for the current concept. In his paper we used feature contribution information for the development of an efficient real-time feature tracking method. A classification algorithm combined with our method would not require examining the entire feature space for feature selection as our approach would feed forward feature contribution information. This paper proposes a concept drift detection method for data stream classification that also feeds forward information about the involvement of individual features in the drift for feature selection purposes. The proposed method could be used with any learning algorithms either as a real-time wrapper for a batch classifier or realised inside a real-time adaptive classifier such as [11, 6].

This paper is organised as follows: Section 2 introduces related works, Section 4 introduces the proposed concept drift and feature selection method, Section 5 evaluates the methodology as a proof of concept and Section 6 provides concluding remarks.

2 Related Works

Many researchers proposed approaches for clustering and classification which are the most common DSM tasks [12, 13] as well as drifts detection methods with feature selection as discussed below.

2.1 Classification

Several methods have been proposed for predictive analytics on data streams. The most notable data stream classifier is probably the Hoeffding Tree family of algorithms. Hoeffding Tree algorithm by [11] induces a decision tree incrementally in real-time. Hoeffding Tree is improved in terms of speed and accuracy by proposing Very Fast Decision Tree (VFDT) [5]. Although, high accuracy using a small sample is the main advantage of these algorithms, concept drifting cannot be handled as a created subtree can only expand from the child nodes onwards. Therefore, further improvements have been done by the development of adaptive trees that can alter entire subtrees using a sliding window model. The new version VFDT was termed CVFTD where C stands for Concept Drift [5]. Loosely speaking in CVFDT alternative subtrees are induced over time and if an alternative subtree outperform (i.e. in terms of accuracy) the current active subtree, then the current subtree is replaced with the alternative one. Further data stream classification algorithms have been proposed, such as Accelerated Particle Swarm Optimisation (APSO) with Swarm Search [14], a Similarity Search Structure Called the Rank Cover Tree (RCT) [15], Online Data Stream Classification with Limited Labels [16], Prototype-based Classification Model [17], On Demand Classification [18], Adaptive Nearest Neighbour Classification for Data Streams [19], Ensemble based Classification [20], VFDR [21], and G-eRules [6]. Although, some of these works have focused on drifts detection, none of these algorithms take online feature selection into consideration. If feature selection is applied at all, then it is mostly at the beginning of the data stream and it is assumed that the contribution of each feature to the concept stays the same over time. However, this is an unrealistic assumption. The application the relevance of features for the concept may change over time and thus an online feature selection strategy could very well improve the classifier's predictive accuracy. Moreover, if features that have become irrelevant over time are taken out from the adaptation process, then the classifier could potentially increase in speed as well.

2.2 Clustering

Also clustering algorithms have the requirement to only use a single pass over the training data in order to form their clusters, as a fast processing of the data is essential in order to keep the model as accurate as possible over time (i.e. quick adaptation to concept drifts). Several algorithms have been proposed for clustering data streams such as CluStream [22], HPStream [23], E-Stream [24], POD-Clus [25], ClusTree [26], HUE-Stream [27], and MC-NN [28]. MC-NN is a development by some of the authors of this paper. MC-NN is actually a classification algorithm, however, the underlying data structure on

which the classification approach is based are Micro-Clusters. Micro-Clusters are statistical summaries which have been extended for predictive analytics from the aforementioned CluStream algorithm. The feature tracking technique presented in this paper is also based on an adaptation of MC-NN and hence MC-NN will be highlighted in more detail in Section 3.

2.3 Concept Drifts Detection Methods and Feature Selection

The aforementioned classifiers and cluster analysis methods in Sections 2.1 and 2.2 are capable of detecting concept drift on their own. Whereas there also exist stand alone concept drift detection techniques that can be used in combination with batch learning algorithms and a sliding window approach. For example some of these techniques are ADaptive sliding WINdow (ADWIN)[8], Drift Detection Method (DDM) [9], Early Drift Detection Method (EDDM) [10], etc. However, we are not aware of any drift detection method that also provides information into the causes of the drift, i.e. which features were involved. It is possible that features may become more or less relevant for a data mining model, i.e. a decision tree. In this paper we aim to use the knowledge about a feature's involvement in the concept drift in order to develop a real-time feature selection technique that can be used by a range of classification approaches as it feeds forward information about the involvement if individual features in the drift.

3 Micro-Cluster Nearest Neighbour (MC-NN)

This section summarises the MC-NN approach developed by some of the authors of this paper. MC-NN has been originally developed for predictive data stream analytics. However, the underlying Micro-Cluster structure has been adapted in order to develop a feature tracker for online feature selection purposes. Micro-Clusters in MC-NN provide a statistical summary of feature values retrieved from the stream over time stamps. In [28], the structure of Micro-Clusters is: $< CF2^x, CF1^x, CF1^t, n, CL, \epsilon, \Theta, \alpha, \Omega >$. Details about the Micro-Cluster structure components are listed in Table 1.

A Micro-Cluster's centroid is calculated by $\frac{CF1^x}{n}$. Micro-Clusters are updated by adding a new instance to its nearest Micro-Cluster. The error ϵ is decremented by 1 if a new instance matches the CL. Otherwise, if the nearest Micro-Cluster does not match the CL of the new data instance, then the new instance is added to the closest Micro-Cluster matching the instance's CL, however, the error epsilon is incremented by 1 in both involved Micro-

Table 1 The Structure of MC-NN Micro-Clusters

The Structure of MC-NN Micro-Clusters	
$CF2^x$	a vector with the sum of squares of the features,
$CF1^x$	a vector with the sum of feature values,
$CF1^t$	a vector with the sum of time stamps,
n	the number of data instances in the cluster,
CL	the cluster's majority class label,
ϵ	the error count,
Θ	the error threshold for splitting the Micro-Cluster,
α	the initial time stamp, and
Ω	a threshold for the Micro-Cluster's performance.

Clusters. MC-NN splits a Micro-Cluster once the error count reaches Θ into two new Micro-Clusters while the original Micro-Cluster is removed in order to fit the data stream better. The new Micro-Clusters are placed about the original Micro-Cluster's feature of greatest $Variance$ for a feature x which can be calculated using equation 1.

$$Variance[x] = \sqrt{\left(\frac{CF2^x}{n}\right) - \left(\frac{CF1^x}{n}\right)^2} \tag{1}$$

The maximum variance of a feature x means that this feature may contribute to miss-classification. The attribute with the maximum variance is either adding or subtracting the amount of variance (adding in one Micro-Cluster, subtracting in the other), as shown in Figure 1. New centroid values of the two newly generated Micro-Clusters are calculated.

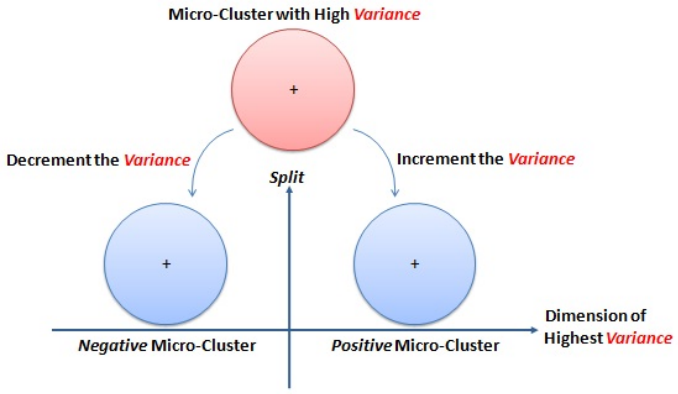

Fig. 1 Splitting of a Micro-Cluster

MC-NN removes a Micro-Cluster if it has not participated recently in classifications, which can be calculated from $CF1^t$ by measuring the Triangle Number (Equation 2). We call this Micro-Cluster Death.

$$Triangle\ Number\ \Delta(T) = ((T^2 + T)/2) \tag{2}$$

The Triangle Number gives more importance to recent instances than older ones, as shown in Figure 2.

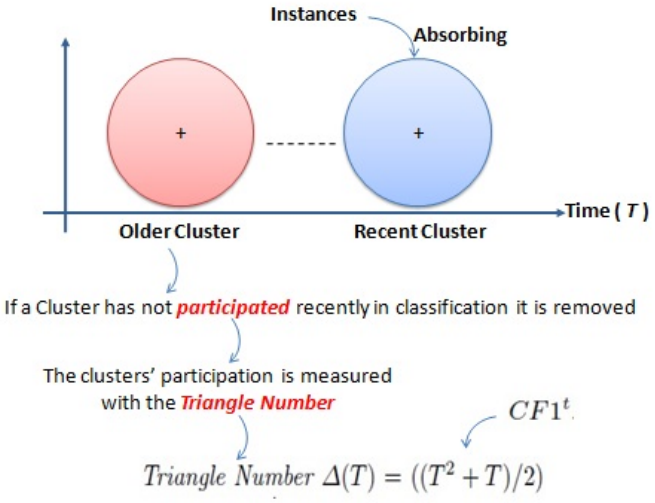

Fig. 2 Death and removal of a Micro-Cluster

Whereas MC-NN has been originally developed for classification purposed, we propose to adapt MC-NN's Micro-Cluster structure to detect concept drift and to enable the tracking of each feature's involvement during concept drift.

4 Real-Time Feature Tracking using Adaptive Micro-Clusters

This section describes our technique for concept drift detection with feature tracking. The technique uses a sliding window approach and consists of four main components which are the normalisation of each feature of a new training instance, a Low Pass Filter to filter each feature a Micro-Cluster in which the new instance has been absorbed, Micro-Cluster Feature Tracker with Drift Detection (MCFT-DD) and Feature Analysis to identify irrelevant features, as shown in Figure 3. MCFT-DD is the central component of this

techniques which his based on the MC-NN approach highlighted in Section 3.

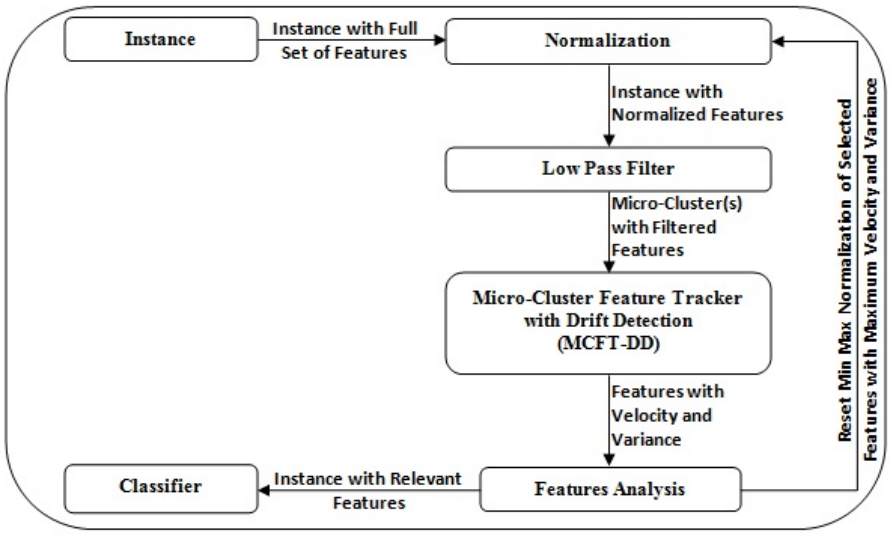

Fig. 3 Real-Time Feature Selection Technique using Adaptive Micro-Clusters

4.1 Normalisation

Normalisation is applied to fit the data (i.e., each feature of a new training instance) to be almost normally distributed in a pre-defined boundary such as [0,100]. We are using Min-Max Normalisation technique which can be applied in real-time processing as minimum and maximum values of a feature x can be reinitialised when new instances arrives. Normalisation is used to avoid bias of features which can potentially lead to miss-classifications as the relevant relations between target class labels and features are considered by the classifier to be more or less important than they actually are. Equation 3 shows how to normalise a new data instance.

$$x = \left(\frac{current\ value - \min x}{\max x - \min x} \right) * (\max range - \min range) + \min range \quad (3)$$

Where x is a feature value of the new data instance, *max range* is the maximum range (default 100) of feature x and *min range* is the minimum range (default 0) of feature x. This equation is applied for every new data

instance. Other normalisation techniques, such as the Z-Score have been considered, however the alternative techniques rely on standard deviation and mean which require buffering of data before normalisation can be applied, which is undesirable in real-time data analytics. However, the normalisation process described here can be updated incrementally instance by instance. Next the Micro-Cluster in which the normalised data instance needs to be absorbed is determined using the same approach as described in Section 3 and a Low Pass Filter is applied on the Micro-Cluster.

4.2 Low Pass Filter (LPF)

LPF is a filter that passes signals with a frequency lower than a certain cutoff frequency and attenuates signals with frequencies higher than the cutoff frequency. In this work LPF is used to avoid both outliers and noise in order to improve concept drift detection and feature tracking. Therefore, LPF is used to filter each normalised feature of a new training instance together with its nearest Micro-Cluster using Equation 4.

$$new\ filter = \alpha * new\ value + (1 - \alpha) * old\ filter \tag{4}$$

The α threshold is set for 0.5 by default. The centroid of each filtered feature of a Micro-Cluster is calculated. Micro-Clusters with filtered features are then passed to MCFT-DD.

4.3 Micro-Clusters Feature Tracker with Drifts Detection (MCFT-DD)

MCFT-DD the component that of our technique that detects concept drift and tracks the relevance of features. Loosely speaking the MCFT-DD enables this through monitoring the split and death rate of the Micro-Clusters in order to detect drifts; and by monitoring $Velocity$ and $Variance$ of the individual Micro-Clusters in order to track which features are involved in a concept drift. The rates of Micro-Cluster splits and removals (Figures 1 and 2) are calculated concurrently for each time window upon the data stream in order to detect if a concept drift has happened. This is illustrated in Figure 4.

If the percentage of the split and death rates differs from the mean (of the statistical windows) by 50% (default value), then this is considered to be a concept drift. Then a closer look can be taken into the individual features through examining their change in velocity. This is tracked through an extension of MC-NN's Micro-Cluster structure by: $< CF1^{hx}, n_h >$. Where the

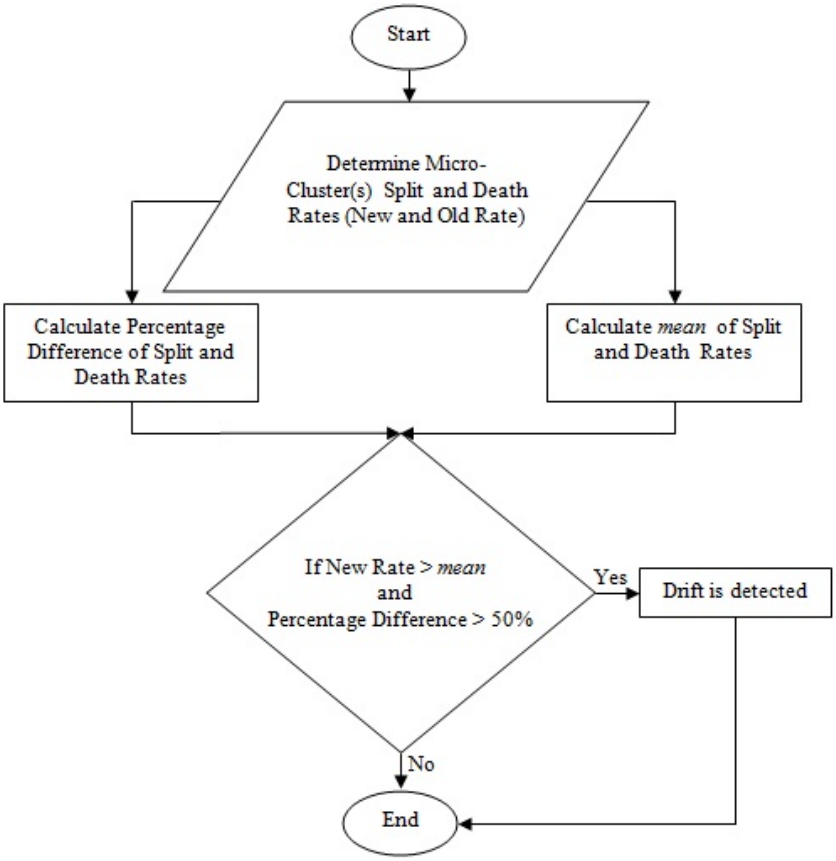

Fig. 4 Flowchart of MCFT-DD.

components of the structure above are equivalent to $CF1^x$ and n. However, the h denotes that these components are *historical* summaries (taken from the statistical windows); the value of h is a fixed user defined parameter and denotes how many time stamps the historical summaries are behind the recent ones. The *Velocity* of a feature x can then be calculated by equation 5

$$Velocity[x] = \mid \frac{CF1^x}{n} - \frac{CF1^{hx}}{n_h} \mid \tag{5}$$

For the purpose of feature analysis, which will be discussed later in the Section, a counter is kept for each feature of a Micro-Cluster. The counter is incremented by 1 if the particular feature was the feature with the highest $Variance$ in the current time window. A high $Velocity$ combined with maximum $Variance$ during a concept drift indicates that the feature changed.

The assumption here is that this particular feature may have changed its contribution towards the classification technique. Thus feature selection can be limited to examining only features that have changed their velocity when there is a concept drift detected. Section 5 evaluates this approach as a principal proof of concept.

4.4 Feature Analysis

Feature analysis is facilitated using historical data (i.e. a statistical window between $time - 1$ and $time$ where $time$ refers to a drift point). For each time window, $Velocity$ rate and history of maximum $Variance$ are calculated separately. Once a drift is detected, features can be analysed. If a feature x with maximum $Velocity$ appears in history of maximum $Variance$ then the Max and Min value of normalisation of the feature needs to be reset. In addition, this feature can be selected as irrelevant feature for a classifier, as shown in algorithm 1.

Algorithm 1 Feature Analysis

Input: $Velocity$[features] and history of maximum $Variance$[features] of a statistical window between $time - 1$ and $time$ of a drift point
Output: Relevant features and reset Min and Max value of normalisation of selected features

for each drift detection **do**
  Apply *median* filter to $Velocity$[Features]
  for each feature with $Velocity$ greater than *median* **do**
    if feature appears with $Variance$ which is greater than 0 value **then**
      $Normalization \Leftarrow reset\ min\ and\ max\ value$
      $instance \Leftarrow delete\ irrelevant\ feature$
    end if
  end for
end for
$Classifier \Leftarrow instance\ with\ relevant\ feature(s)$

5 Experimental Evaluation

This section evaluates the proposed technique in terms of adaptivity to concept drift with feature analysis. The implementation of our experiments were realised in the Java based Massive Online Analysis (MOA) framework [29].

5.1 Experimental Setup

For the experiments we used the following datasets. **Artificial Datasets**,***SEA Generator***, this artificial dataset was introduced in [30], it generates an artificial data stream with continuous attributes. The third attribute is irrelevant for distinguishing between the class labels. ***HyperPlane Generator***, this artificial dataset creates a linearly separable model. It slowly rotates in 'D' dimensions continuously changing the linear decision boundary of the stream. This constant concept change makes it very difficult for data stream classifiers to keep a good classification accuracy and computational efficiency. We generated a stream with continuous attributes. The 1^{st} attribute is irrelevant for distinguishing between the class labels (i.e., it is generated randomly). ***Random Tree Generator*** was introduced in [11] and generates a stream based on a randomly generated tree. New examples are generated by assigning uniformly distributed random values to features, which then determine the class label using the randomly generated tree. In each generated stream, 2 features are swapped making one of them irrelevant for the classification task as shown in table 2, which shows an overview of generated streams including setting of our method and which features have been swapped.

Table 2 Setup of the data streams

Generator	Instances	Features	CL	Swapped Features	Time	Θ	Ω	Window Size
SEA	10,000	3	2	2 with 3	After 5000	3	50	1000
HyperPlane	10,000	3	2	1 with 2	After 5000	300	50	1000
RandomTree	100,000	3	2	2 with 3	After 50,000	5000	50	10,000

5.2 Results

The evaluation is focused on drifts detection with feature analysis to identify relevant features for a classifier. Micro-Clusters' Split and Death rates are used for detecting drifts. Whereas, $Velocity$ and $Variance$ are used for analysing features.

For the experiments the default parameters stated in Section 5.1 of the method were used unless stated otherwise. In Figure 5 it can be seen that the split and death rates at the time of concept drift increase, indicating that the current set of Micro-Clusters does not fit the concept encoded in

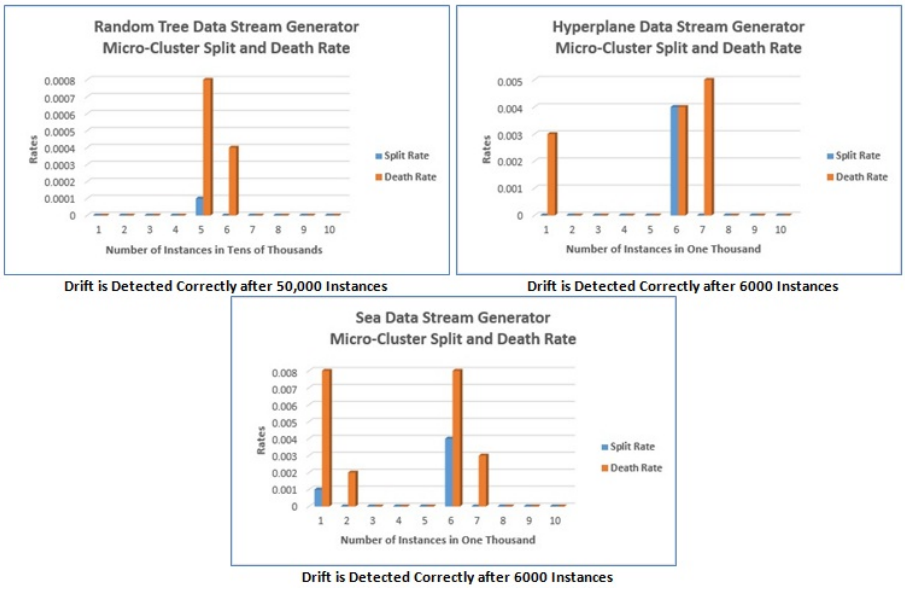

Fig. 5 The Micro-Cluster Split and Death Rates

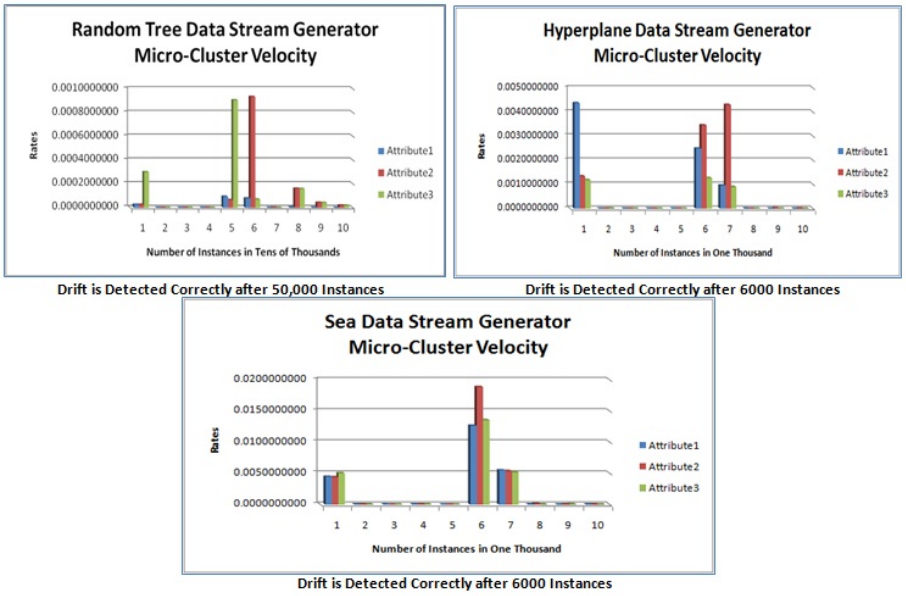

Fig. 6 The Micro-Cluster *Velocity* Rate

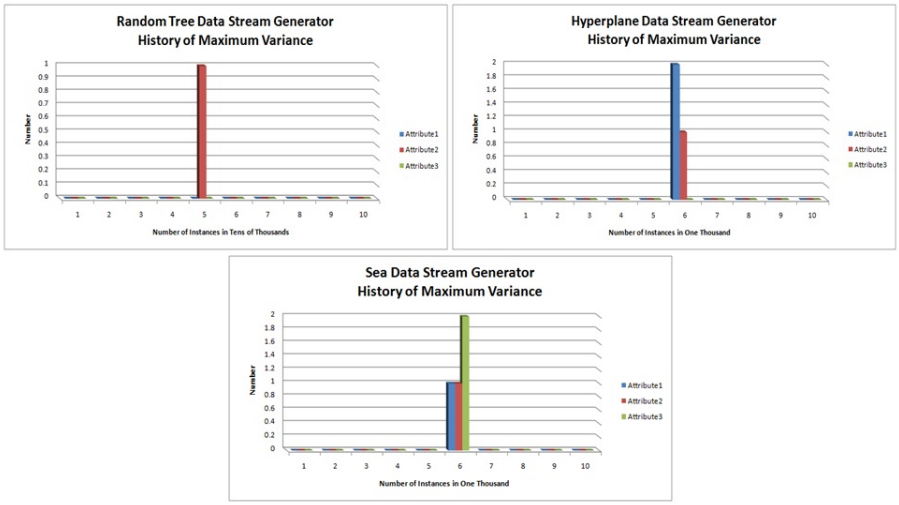

Fig. 7 History of Maximum $Variance$

the data anymore. Figures 6 and 7 show the velocities of the features in the Micro-Clusters and history of maximum $Variance$. In this case we know that the concept appeared due to swapping of features, hence we would expect a higher $Velocity$ and $Variance$ of those swapped features as it can be seen in Figures 6 and 7. Thus the method is capable of detecting a concept drift but also delivers an indication which features are involved. This information can then be used by a classifier to perform real-time feature selection.

6 Conclusion

This paper introduced a novel Micro-Cluster based methodology for drift detection in data streams. Different compared with existing drift detection techniques, the proposed method is also capable to detect which features have been involved in the drift through the velocity and variance of Micro-Clusters in different dimensions; and thus can be used to implement real-time feature selection techniques. The experimental proof of concept shows that the methods can successfully detect concept drifts and identify drifting features. Ongoing and future work comprises an in depth evaluation of the method in combination with feature selection and a classification algorithm.

References

1. Ebbers, M., Abdel-Gayed, A., Budhi, V.B., Dolot, F., Kamat, V., Picone, R., Trevelin, J., et al.: Addressing Data Volume, Velocity, and Variety with IBM InfoSphere Streams V3. 0. IBM Redbooks (2013)
2. Babcock, B., Babu, S., Datar, M., Motwani, R., Widom, J.: Models and issues in data stream systems. In: Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, ACM (2002) 1–16
3. Kadlec, P., Gabrys, B., Strandt, S.: Data-driven soft sensors in the process industry. Computers & Chemical Engineering **33**(4) (2009) 795–814
4. Jadhav, A., Jadhav, P., Kulkarni, P.: A novel approach for the design of network intrusion detection system (nids). In: Sensor Network Security Technology and Privacy Communication System (SNS & PCS), 2013 International Conference on, IEEE (2013) 22–27
5. Hulten, G., Spencer, L., Domingos, P.: Mining time-changing data streams. In: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, ACM (2001) 97–106
6. Le, T., Stahl, F., Gomes, J.B., Gaber, M.M., Di Fatta, G.: Computationally efficient rule-based classification for continuous streaming data. In: Research and Development in Intelligent Systems XXXI. Springer (2014) 21–34
7. Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., Bouchachia, A.: A survey on concept drift adaptation. ACM Computing Surveys (CSUR) **46**(4) (2014) 44
8. Bifet, A., Gavalda, R.: Learning from time-changing data with adaptive windowing. In: SDM. Volume 7., SIAM (2007) 2007
9. Gama, J., Medas, P., Castillo, G., Rodrigues, P.: Learning with drift detection. In: Advances in artificial intelligence–SBIA 2004. Springer (2004) 286–295
10. Baena-Garcıa, M., del Campo-Ávila, J., Fidalgo, R., Bifet, A., Gavalda, R., Morales-Bueno, R.: Early drift detection method. In: Fourth international workshop on knowledge discovery from data streams. Volume 6. (2006) 77–86
11. Domingos, P., Hulten, G.: Mining high-speed data streams. In: Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM (2000) 71–80
12. Aggarwal, C.C.: Data streams: models and algorithms. Volume 31. Springer Science & Business Media (2007)
13. Gama, J.: Knowledge discovery from data streams. CRC Press (2010)
14. Fong, S., Wong, R., Vasilakos, A.: Accelerated pso swarm search feature selection for data stream mining big data. (2015)
15. Houle, M.E., Nett, M.: Rank-based similarity search: Reducing the dimensional dependence. Pattern Analysis and Machine Intelligence, IEEE Transactions on **37**(1) (2015) 136–150
16. Loo, H., Marsono, M.: Online data stream classification with incremental semi-supervised learning. In: Proceedings of the Second ACM IKDD Conference on Data Sciences, ACM (2015) 132–133
17. Shao, J., Ahmadi, Z., Kramer, S.: Prototype-based learning on concept-drifting data streams. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM (2014) 412–421
18. Aggarwal, C.C., Han, J., Wang, J., Yu, P.S.: On demand classification of data streams. In: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM (2004) 503–508
19. Law, Y.N., Zaniolo, C.: An adaptive nearest neighbor classification algorithm for data streams. In: Knowledge Discovery in Databases: PKDD 2005. Springer (2005) 108–120

20. Wang, H., Fan, W., Yu, P.S., Han, J.: Mining concept-drifting data streams using ensemble classifiers. In: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM (2003) 226–235
21. Gama, J., Kosina, P., et al.: Learning decision rules from data streams. In: IJCAI Proceedings-International Joint Conference on Artificial Intelligence. Volume 22. (2011) 1255
22. Aggarwal, C.C., Han, J., Wang, J., Yu, P.S.: A framework for clustering evolving data streams. In: Proceedings of the 29th international conference on Very large data bases-Volume 29, VLDB Endowment (2003) 81–92
23. Aggarwal, C.C., Han, J., Wang, J., Yu, P.S.: A framework for projected clustering of high dimensional data streams. In: Proceedings of the Thirtieth international conference on Very large data bases-Volume 30, VLDB Endowment (2004) 852–863
24. Udommanetanakit, K., Rakthanmanon, T., Waiyamai, K.: E-stream: Evolution-based technique for stream clustering. In: Advanced Data Mining and Applications. Springer (2007) 605–615
25. Rodrigues, P.P., Gama, J., Pedroso, J.P.: Hierarchical clustering of time-series data streams. Knowledge and Data Engineering, IEEE Transactions on **20**(5) (2008) 615–627
26. Kranen, P., Assent, I., Baldauf, C., Seidl, T.: Self-adaptive anytime stream clustering. In: Data Mining, 2009. ICDM'09. Ninth IEEE International Conference on, IEEE (2009) 249–258
27. Meesuksabai, W., Kangkachit, T., Waiyamai, K.: Hue-stream: Evolution-based clustering technique for heterogeneous data streams with uncertainty. In: Advanced Data Mining and Applications. Springer (2011) 27–40
28. Tennant, M., Stahl, F., Gomes, J.B.: Fast adaptive real-time classification for data streams with concept drift. In: Internet and Distributed Computing Systems. Springer (2015) 265–272
29. Bifet, A., Holmes, G., Kirkby, R., Pfahringer, B.: Moa: Massive online analysis. Journal of Machine Learning Research **11**(May) (2010) 1601–1604
30. Street, W.N., Kim, Y.: A streaming ensemble algorithm (sea) for large-scale classification. In: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, ACM (2001) 377–382