Visual and Dynamic Change Detection for Data Streams

Lydia Boudjeloud-Assala[1](✉), Philippe Pinheiro[2], Alexandre Blansché[1], Thomas Tamisier[2], and Benoît Otjaques[2]

[1] Université de Lorraine, Laboratoire d'Informatique Théorique et Appliquée - EA 3097, Ile du Saulcy, 57045 Metz Cedex 1, France
lydia.boudjeloud-assala@univ-lorraine.fr

[2] LIST - Luxembourg Institute of Science and Technology, Esch-sur-Alzette, Luxembourg

Abstract. We propose in this paper a new approach to detect and visualize the change in a streaming clustering. This approach can be used to explore visually the data streams. We assume that the data stream structure can be different during the time. Our objective is to alert the user on the structure change during the time period. A common approach to deal with data streams is to observe and process it in a window. The principle of the proposed approach is to apply a data exploration method on each window. We then propose to visualize the change between all windows for each extracted cluster. The user can investigate more precisely the change between the two windows through a visual projection for each extracted cluster.

Keywords: Visual exploration · Dynamic clustering · Change detection · Data stream

1 Introduction

A data stream is an ordered sequence of data items arriving continuously, generally time-stamped, in a database [13]. In recent years, data streams have received considerable attention in various applications, such as Internet traffic [3], financial tickers [20] and scientific literature analysis [21]. When dealing with data streams, one important task is the indication of the stream change. Change detection is the process of identifying differences in the state of an object or phenomenon by observing it at different times or different locations in space. In the streaming context, it is the process of segmenting a data stream into different segments by identifying the points where the stream dynamics changes [18]. The design of general, scalable and statistically relevant change detection methods is a great challenge. Change detection consists in comparing the underlying distribution of tuples observed at different times. Two temporal windows are defined: the *reference* and the *current*. An overview of the main change detection approaches is given by Dries [11]. Change detection in the distribution of tuples can be considered as a statistical hypothesis test which involves two samples of multidimensional tuples. Such problems are studied in the statistical literature. Approaches based on nearest-neighbor analyses [14] or distance between density estimates [4] have been developed. To detect change in a data stream, the algorithm proposed by Kifer et al. [15] compares the distributional distance between two sliding windows with a given threshold. Their approach requires no prior assumptions on the nature of the data distribution. They can compare two sliding windows of different sizes because the distance used by their algorithm is the Kolmogorov-Smirnov statistical distance. In Bondu and Boullé [8] the change detection problem is turned into a supervised learning task. Authors chose to exploit the supervised discretization method of a continuous variable and estimate the conditional distribution of classes owing to a piecewise constant estimator. We assume in our work, that the two consecutive windows of data are given. Many studies have been devoted to developing strategies of choosing, sampling, splitting, growing, and shrinking the windows for optimal change detection [1,5,12]. In terms of visualization, the only relevant work, to our knowledge, is the work of Kranen et al. [16]. The visualization component allows to visualize the stream as well as the clustering results (the CluStream algorithm [2]), and compare experiments with different settings in parallel (F1-measure, Precision, Recall, or other evaluation measures [6]).

This paper proposes a new approach to explore visually the data stream. A data stream evolves over time and a clustering result and visualization are desired at each time step. We want to alert the user on the structure change during the time period, assuming that the data stream structure can be different in time. Since a data stream is infinite in nature, a common approach is observing and processing it in a window. The principle of the proposed approach is to apply a clustering method on each window. The clustering algorithm is not applying to find the clustering results but is applied to explore the data and find optimal structure in the current time. We then study a data structure in each window in a static way, and a stream structure can change between two adjacent (consecutive) windows. We propose to visualize the change between all windows as a stream. If the data structure changes, it is indicated through a visual streaming change and then the user can investigate more precisely the change between the two windows through a visual projection. As a multidimensional data visualization methods, we use a two dimensional reduction method Stochastic Neighbor Embedding (t-SNE) [19]. The approach for the data exploration in different windows, uses a cluster extraction method based on a cluster limit detection method for clusters defined by their center. To extract clusters we use a deterministic approach, which simplifies the change detection. Our objective is not to search or to find the real classes in the data, even if our method finds them with a certain accuracy rate [9]. Our objective is to use a clustering deterministic approach to explore a data stream through different windows, and alert the user if change is detected between windows. The clusters are extracted according to a specific order and this order is fix as long as the data structure does not change. We can then evaluate the stability between two clustering structures. The first originality of our approach is the possibility to detect the change for a data stream, through a deterministic clustering approach, which is impossible to achieve with other clustering algorithms such as k-means, because the different algorithms are generally stochastic and the solutions are generally instable. The second originality is to allow this change detection through a visual heat map of the streaming change and the window data 2D projection.

The remainder of this paper is organized as follows: Sect. 2 gives the details of the proposed deterministic clustering algorithm for data exploration with evaluations. Section 3 describes the change detection approach and evaluation comments. Section 4 presents the proposed tools for visual streaming change detection and comments of the experimental results. Finally, we draw the conclusions and future work.

2 Data Stream Exploration

We propose a generic iterative approach that extract clusters one after the other, based on an optimization method. The method extracts the best cluster at each iteration according to the evaluation criterion. The extracted cluster represents a subset of the most homogeneous and separate data from other objects in each iteration. The iterative process is performed by a deterministic optimization method finding at each iteration the best cluster based on a particular evaluation criterion. For our application on data stream exploration and change detection, we fix the number of iterations and in the same time the total number of extracted clusters at the beginning. These extracted clusters summarize the information of the data structure. We assume that the clusters are defined by their center. In order to obtain the set of objects contained in the cluster, we select the ones close enough to the center. We choose the threshold by computing the distance between all the objects and the cluster center and ordering them from the closest object to the farthest. We then try to find an abrupt increasing of distance that will indicate the limit of the cluster [9]. Most of the evaluation criteria for clustering evaluate the complete clustering and not each cluster separately. Only two criteria exist, the Wemmert-Gançarski compactness criterion [10] gives an evaluation for each cluster, but this evaluation is based on the position of other cluster centers. As we only define a cluster by its center and its limit, we cannot use such a criterion. The intra-class inertia also gives an evaluation for each cluster, but is biased by the cluster size and the variance on the dimensions. We propose two evaluation criteria for single clusters. The first one is the inertia ratio IR, the ratio of the intra-class inertia by the total inertia of the data, normalized by the number of objects in the extracted cluster and the whole data set:

$$IR\left(C_k\right) = \frac{Card(D) \sum_{o \in C_k} d\left(o, c_k\right)^2}{Card(C_k) \sum_{o \in D} d\left(o, g\right)^2} \tag{1}$$

The second proposed evaluation criterion, the cluster limit ratio CLR, the ratio between the distance of the last object of the extracted cluster and the first object outside the cluster.

$$CLR\left(C_k\right) = \frac{\max\limits_{o \in C_k}\left(d\left(o, c_k\right)\right)}{\min\limits_{o \notin C_k}\left(d\left(o, c_k\right)\right)} \tag{2}$$

Each cluster is extracted independently. The two proposed criteria provide an evaluation of the compactness of one cluster. However, as each cluster is extracted independently, we need to make sure that the extracted clusters are different (different subset of objects). To ensure that, we propose to add a penalty for overlapping cluster OP, defined as follows:

$$OP\left(C_k\right) = \lambda \frac{\sum\limits_{o \in C_k} nb_o}{Card(C_k)} \tag{3}$$

where nb_o is the number of clusters to which the object o has been previously assigned, and $\lambda \geq 0$ is a weight chosen for the penalty depending on the significance one gives to overlapping clusters. This penalty is simply added to the criterion IR or CLR.

To extract the cluster, we propose an optimization method. For our data stream application through windows, we use an exhaustive search, which tests each data objects as center, in different windows. The extracted clusters have then a deterministic order. For each cluster we obtain a center ID and a threshold (radius, cluster limit) according to the two criterion IR and CLR.

3 Change Detection Method

A common approach to deal with data streams is a window processing. Based on the window width, a window can be a fixed-size window or a variable-size window, it can be also a time-driven model or tuple-driven model. For our application we choose a sliding fixed-size and tuple-driven window model to deal with a data stream. Sliding window model is useful when we need to make decisions based on the recent observations such as stock data stream or sensor data streams. Let window $w[t_c, t_c + N - 1]$ be a sliding window, where t_c is the current time. The window starts at the time t_c and ends at the current time $t_c + N - 1$, the window size is N (in terms of data objects). An object is expired if its time stamp t is less than the time t_c. An active object is defined as an item whose time stamp t lies in the range $[t_c, t_c + N - 1]$. Data objects are inserted into the sliding window as they arrive, and the oldest objects are removed from it. We consider two possibilities of sliding windows:

- sliding windows with overlapping windows, with the overlapping size set to $N/2$ between two adjacent windows,
- sliding windows without overlapping.

We study data in each window off-line, and a stream structure can change between two adjacent (consecutive) windows. We assume that the data stream structure can be different during the time. Our objective is to alert the user on the structure change during the time period. We describe in the following the proposed method for the visual change detection.

The key idea underlying the window-based change detection methods is to compare two samples extracted from two windows: the reference window and the current window. Let w_1 and w_2 denote two basic windows of size N (in term of data objects). The problem of change detection in data streams is to define a change. Given a distance function which measures the dissimilarity of two sliding windows $d(w_1, w_2)$, and ω a distance-based threshold used to decide whether a change occurs. A change occurs if the dissimilarity measure between two windows exceeds a given threshold. Window-based methods for detecting changes may reduce the memory and time required to execute a change detector. For our approach, we firstly propose to compute a distance between all adjacent windows ($d(w_1, w_2)$, $d(w_2, w_3)$,..., $d(w_{t_c-N+1}, w_{t_c})$), and to compute the distance between a referenced w_{ref} window with others ($d(w_{ref}, w_1)$, $d(w_{ref}, w_2)$,..., $d(w_{ref}, w_{t_c-N+1}$, $d(w_{ref}, w_{t_c})$). As we summarize the information of the data structure in the extracted clusters according to different windows, we do not apply a change detection method on two extracted samples, but we apply the distance function on the extracted centers that summarise the data structure information. We then obtain a distance vector ($d(w_{t_c-N+1}, w_{t_c})$) associated to each cluster. A new cell value is added to this vector at each studied new window. This distance vector contains the streaming change information of each cluster.

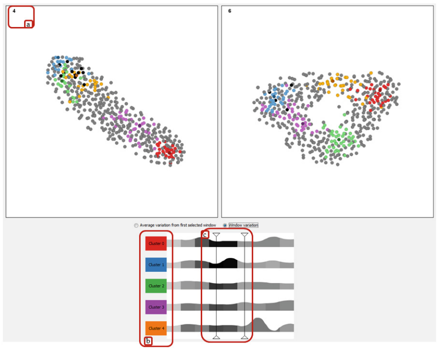

Fig. 1. The tool interface: (a) a window number, (b) cluster identification, (c) change detection - user can chose the windows (between window 4 and window 6 as seen in the t-SNE projection) (Color figure online)

We propose to visualize the change for each cluster between all windows as a stream. If the data structure changes, it is indicated through a visual streaming change and then the user can investigate more precisely the change between the two windows through a visual projection. The approach for the exploration of data in different windows uses a cluster extraction method. This method is based on a cluster limit detection for clusters defined by their center. For the change detection method, we summarize streaming change information of each cluster through a distance vector. We then use a time series with variation color for each cluster to project the streaming change of the vector distance change, where the color represent the distance measure between two all adjacent windows or with a referenced window. The ω threshold, is then automatically determined from a variation of the normalized vector distance (in the range $[0, 1]$). As we can see in the Fig. 1, the change detection between two consecutive or with referenced windows are indicated through a heat map information where the smaller values ($\simeq 0$) are indicated in light color, the larger ones ($\simeq 1$) are indicated in dark color. Each cluster is represented separately one after other, we can see an example with 5 clusters in the Fig. 1b. The color variation represents the difference between two adjacent windows ($\varDelta = d(w_{t_c-N+1}, w_{t_c})$), which represents the distance between the cluster centers at each step of the stream. If the cluster center change is not important, it is indicated in light color, otherwise, is indicated in dark color. The variations in height of the time series indicate the importance of the extracted cluster in term of elements.

4 Experimental Results

In order to evaluate the change detection method through different windows on stream data set, we provide some results on real data stream. We firstly demonstrate the effectiveness of the proposed method with WaveForm data set from University of California at Irvine (UCI) machine learning benchmark repository [7]. WaveFrom data set is composed on 5000 objects with 21 dimensions (attributes), all of which include noise. WaveFrom data set is composed by 3 classes of waves. Each class is generated from a combination of 2 or 3 "base" waves and each instance is generated with added noise (mean 0, variance 1) in each dimension. The projected form of this data set is a triangle form. We apply our tools on unsupervised exploration process, so the classes are not visualized. We decompose the data set on 10 windows, in each window we have 500 objects sampled from the whole data set, and they have also a triangle form in the 2D projection. We insert between window 3 and 5, a sample data set composed by 500 objects, taken from one axis of the whole data set, that we call *the intrusive set* (Fig. 1). The objective is to find as good as possible the change that occurs by the window 4. The data stream analysis proceeds up to the window 9, the Figs. 2 and 3 present two possibilities for the user to see the variations. From the user selected windows (Fig. 3), here, the user select with the cursor the 4th window and the indicated windows are projected (Fig. 1). The variation for all two consecutive windows (Fig. 2), here, the cursor indicate the projected windows (Fig. 1).

We also test our method on Sea Concepts [17] data set from Ubiquitous Knowledge Discovery and Data Analysis repository. Sea Concepts Data set is composed by 60000 objects, 3 dimensions and 3 classes. Dimensions are numeric

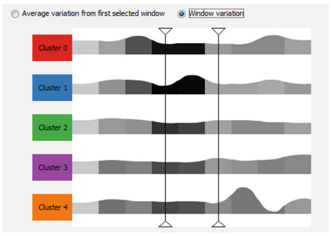

Fig. 2. Visual change detection with two consecutive windows variation for WaveForm data set.

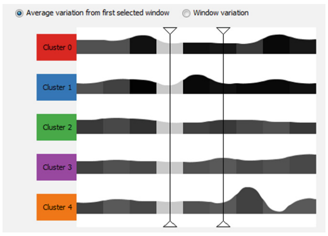

Fig. 3. Visual change detection with average variation from the user selected windows for WaveForm data set.

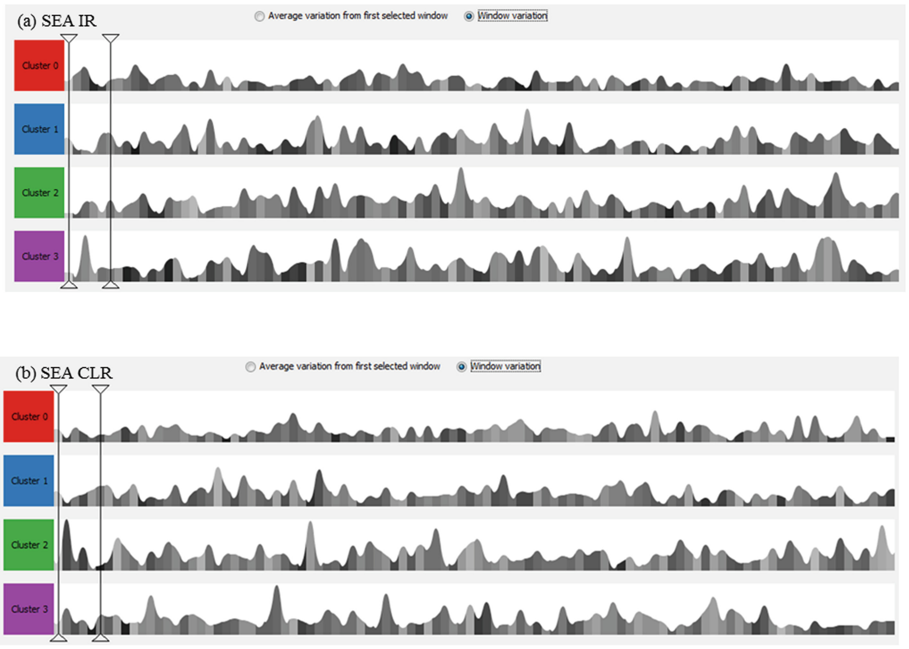

Fig. 4. Visual change detection with two consecutive windows variation for Sea Concepts data set: (a) With IR cluster evaluation, (b) With CLR cluster evaluation (Color figure online).

between 0 and 10. Data set has about 10 % of noise. We decompose the data sets on 100 windows respectively, in each window we fix the size of 600 as a number of tuple entering in the window. This data set is generally used for data stream mining problems. In the Fig. 4, we can see the obtained visual time series with IR and CLR cluster evaluation. The Sea data set has 3 clusters but we iterate our approach at 4 iteration, because the first extracted cluster has very small size compared with other clusters (the amplitude of the time series). In term of color change, the first and the third clusters are rather stable (Fig. 4), the second and the forth are less. In the last third clusters we can see 4 dark zone (Fig. 4a), that indicate change in the stream. The user can then, use this tools to explore and interpret more easily the obtained results. The number of the clusters and the number of projected windows are fixed by the user. The different experiments and visualizations on synthetic and real data sets show the effectiveness of the proposed approach.

5 Conclusions

We propose in this paper a new approach to explore visually data streams. We assume that the data stream structure can be different during the time. We want to alert the user on the structure change during the time period. The first originality of our approach is the possibility to detect the change for a data stream, through a clustering deterministic approach. The second originality is to allow this change detection through a visual projection of the streaming change and the 2D projection of the windows data. Experiments and visualizations on data sets show the effectiveness of the proposed approach. We can visually detect the change and explore the visual data set projections to explain and interpret this change. In a visual and interactive point of view, there are certainly more things to do, such as improving the interaction. In fact, it may be interesting to see the change for a specific cluster trough the stream. It is also, interesting to test all other possibilities of visual interaction, to thereby improve the visual data stream mining.

References

1. Adams, R., MacKay, D.: Bayesian online changepoint detection. Technical report, University of Cambridge (2007)
2. Aggarwal, C.C., Watson, T.J., Ctr, R., Han, J., Wang, J., Yu, P.S.: A framework for clustering evolving data streams. In: VLDB, pp. 81–92 (2003)
3. Aizen, J., Huttenlocher, D., Kleinberg, J., Novak, A.: Traffic-based feedback on the web. Proc. Natl. Acad.Sci. **101**(Suppl. 1), 5254–5260 (2004)
4. Anderson, N.H., Hall, P., Titterington, D.M.: Two-sample test statistics for measuring discrepancies between two multivariate probability density functions using kernel-based density estimates. J. Multivar. Anal. **50**(1), 41–54 (1994)
5. Bifet, A., Gavalda, R.: Learning from time-changing data with adaptive windowing. In: Seventh SIAM International Conference Data Mining, pp. 443–448 (2007)
6. Bifet, A., Holmes, G., Pfahringer, B., Kranen, P., Kremer, H., Jansen, T., Seidl, T.: Moa: massive online analysis, a framework for stream classification and clustering. In: WAPA, pp. 44–50 (2010)
7. Blake, C., Merz, C.: UCI repository of machine learning databses. Technical report, University of California, Irvine, Department of Information and Computer Sciences (1998). http://archive.ics.uci.edu/ml/datasets.html. Accessed January 2014
8. Bondu, A., Boulle, M.: A supervised approach for change detection in data streams. In: The 2011 International Joint Conference on Neural Networks (IJCNN), pp. 519–526 (2011)

9. Boudjeloud-Assala, L., Blansché, A.: Iterative evolutionary subspace clustering. In: International Conference on Neural Information Processing, vol. 1, pp. 424–431 (2012)
10. Wemmert, C., Gançarski, P., Korczak, J.: A collaborative approach to combine multiple learning methods. Int. J. Artif. Intell. Tools **9**(1), 59–78 (2000)
11. Dries, A., Rückert, U.: Adaptive concept drift detection. Stat. Anal. Data Min. **2**(5–6), 311–327 (2009)
12. Gama, J., Medas, P., Castillo, G., Rodrigues, P.: Learning with drift detection. In: Bazzan, A.L.C., Labidi, S. (eds.) SBIA 2004. LNCS (LNAI), vol. 3171, pp. 286–295. Springer, Heidelberg (2004)
13. Golab, L., Özsu, M.T.: Issues in data stream management. SIGMOD Rec. **32**(2), 5–14 (2003)
14. Hall, P.: Permutation tests for equality of distributions in high-dimensional settings. Biometrika **89**(2), 359–374 (2002)
15. Kifer, D., Ben-David, S., Gehrke, J.: Detecting change in data streams. In: Endowment, V. (ed.) Proceedings of the Thirtieth International Conference on Very Large Data Bases, vol. 30, p. 191 (2004)
16. Kranen, P., Kremer, H., Jansen, T., Seidl, T., Bifet, A., Holmes, G., Pfahringer, B.: Clustering performance on evolving data streams: assessing algorithms and evaluation measures within MOA. In: ICDM Workshops, pp. 1400–1403 (2010)
17. Street, W.N., Kim, Y.: A streaming ensemble algorithm SEA for large-scale classification. ACM Press, pp. 377–382 (2001). http://www.liaad.up.pt/kdus/products/datasets-for-concept-drift Accessed April 2014
18. Tran, D.H.: Change detection in streaming mining. Ph.D. thesis, Faculty of computer science and automation, Ilmenau University of Technology (2013)
19. van der Maaten, L., Hinton, G.: Visualizing high-dimensional data using t-SNE. J. Mach. Learn. Res. **9**, 2579–2605 (2008). Nov
20. Vlachos, M., Wu, K.-L., Chen, S.-K., Yu, P.S.: Fast burst correlation of financial data. In: Jorge, A.M., Torgo, L., Brazdil, P.B., Camacho, R., Gama, J. (eds.) PKDD 2005. LNCS (LNAI), vol. 3721, pp. 368–379. Springer, Heidelberg (2005)
21. Wei, C.-P., Chang, Y.-H.: Discovering event evolution patterns from document sequences. IEEE Trans. Syst. Man Cybern. A Syst. Hum. **37**(2), 273–283 (2007)