A Complex Network-Based Anytime Data Stream Clustering Algorithm

Conference Paper · November 2015

DOI: 10.1007/978-3-319-26532-2_68

3 authors:

Jean Paul Barddal
Pontifícia Universidade Católica do Paraná (PUC-PR)
57 PUBLICATIONS **880** CITATIONS
SEE PROFILE

Heitor Murilo Gomes
The University of Waikato
58 PUBLICATIONS **968** CITATIONS
SEE PROFILE

Fabrício Enembreck
Pontifícia Universidade Católica do Paraná (PUC-PR)
143 PUBLICATIONS **1,418** CITATIONS
SEE PROFILE

A Complex Network-based Anytime Data Stream Clustering Algorithm

Jean Paul Barddal, Heitor Murilo Gomes and Fabrício Enembreck

Graduate Program in Informatics (PPGIa), Pontifícia Universidade Católica do Paraná, R. Imaculada Conceição, 1155

Abstract. Data stream mining is an active area of research that poses challenging research problems. In the latter years, a variety of data stream clustering algorithms have been proposed to perform unsupervised learning using a two-step framework. Additionally, dealing with non-stationary, unbounded data streams requires the development of algorithms capable of performing fast and incremental clustering addressing time and memory limitations without jeopardizing clustering quality. In this paper we present CNDenStream, a one-step data stream clustering algorithm capable of finding non-hyper-spherical clusters which, in opposition to other data stream clustering algorithms, is able to maintain updated clusters after the arrival of each instance by using a complex network construction and evolution model based on homophily. Empirical studies show that CNDenStream is able to surpass other algorithms in clustering quality and requires a feasible amount of resources when compared to other algorithms presented in the literature.

1 Introduction

Data stream clustering can be described as the act of grouping streaming data in meaningful classes [4]. Data stream clustering is subject to acting within limited time, memory and treating data incrementally with single pass processing. Apart from the time and memory space constraints, two requirements are long-awaited for data stream clustering algorithms. Data stream clustering algorithms must not make assumptions about the number of clusters, since it is not often known in advance and due to the temporal aspect, the number of ground-truth clusters may change regularly [13] and they must be capable of discovering clusters with arbitrary shapes, since most of data streams are not Gaussian distributed.

In this paper we present an extension of the DenStream algorithm: CNDenStream. CNDenStream, in opposition to other data stream clustering algorithms, performs a single step processing to find clusters. CNDenStream does so by using a complex network construction and evolution model based on homophily. Moreover, CNDenStream is not bounded to the k-means algorithm [11] to find clusters, therefore it is able to discover clusters with arbitrary shapes and not only hyper-spherical clusters.

The remainder of this work is organized as follows: Sec. 2 surveys related work for data stream clustering. Sec. 3 introduces basic concepts of complex

networks. In Sec. 4 we present our proposal: CNDenStream. In Sec. 5 we present a performance study and discuss about parametrization sensitivity. Finally, Sec. 6 concludes this paper and presents future work.

2 Related Work

A variety of data stream clustering algorithms were developed throughout the last decade. Generally, these algorithms are divided in online and offline steps.

During the online step, algorithms incrementally update specific data structures aiming at dealing with the evolving nature of data streams and time-space constraints. One widely used data structure is the feature vector, a triplet $CF = \langle LS, SS, N \rangle$, where LS stands for the sum of the objects $\boldsymbol{x}_i$ summarized, SS is the squared sum of these objects and N is the amount of objects [13]. Feature vectors are able to represent hyper-spherical clusters incrementally due to its incremental and additive properties. Basically, an instance $\boldsymbol{x}_i$ can increment an feature vector CF_j as follows: $LS_j \leftarrow LS_j + \boldsymbol{x}_i$, $SS_j \leftarrow SS_j + (\boldsymbol{x}_i)^2$ and $N_j \leftarrow N_j + 1$. As for the additive property, two feature vectors CF_i and CF_j can be merged into a third CF_l as follows: $LS_l \leftarrow LS_i + LS_j$, $SS_l \leftarrow SS_i + SS_j$, and $N_l \leftarrow N_i + N_j$. Also, in order to assign more importance to recently retrieved instances in clustering, various window models featuring sliding, damped and landmark were developed [13]. During the offline step, conventional batch clustering algorithms are used to form final clusters using the CFs.

In the following sections we describe other data stream clustering algorithms.

2.1 CluStream

CluStream adopts the landmark windowing technique, treating the stream based on data chunks of size $\mathcal{H}$ [1]. CluStream assumes a number q of CFs that are maintained at any instant of the stream. Initial CFs are computed with an amount of instances $\mathcal{N}$, also determined by the user. CluStream computes an Euclidean distance for each instance $\boldsymbol{x}_i$ to each CF, then, determines whether the distance to the closest CF_j is less or equal to its radius. Positively, $\boldsymbol{x}_i$ is merged within CF_j. Conversely, $\boldsymbol{x}_i$ starts a new CF_k. If the amount of CFs is above q, the two closest CFs are merged. When $\mathcal{H}$ is reached, all q CFs are recomputed with the next N instances obtained from the stream.

On the offline step, CluStream uses a modification of the k-means or DBSCAN algorithms to obtain clusters based on the q CFs computed during the online step. In this paper, we compare the DBSCAN version, since k-means is highly dependent of the user-given parameter of ground-truth clusters K.

2.2 ClusTree

ClusTree [9] maintains CFs in a hierarchy with different granularity levels. Depending on how much time is available to process each instance, ClusTree performs a search in the R-Tree in order to find the most similar CF. Accordingly

to user-given thresholds, it is determined whether this instance should or not be merged. In the negative case, a new CF is then created and added to the R-Tree. ClusTree also copes with noisy data by using outlier-buffers.

In order to assign more importance to recent data, ClusTree assigns a exponentially decaying weight for all CFs' components.

On the offline step, algorithms such as k-means and DBSCAN are used in order to obtain clusters, where CFs centers are treated as centroids.

2.3 DenStream

DenStream is based on the DBSCAN algorithm, which guarantees the union of the ϵ-neighborhood of clusters which covers all dense areas of the attribute space. A core object is an object which ϵ-neighborhood has at least ψ neighbors and a dense area is the union of all ϵ-neighborhoods of all core objects. DenStream defines the concept of a core-micro-cluster in a time instant t, which is a temporal extension to a CF, as $CMC(w, c, r)$ to a group of near instances $\boldsymbol{x}_i, \boldsymbol{x}_{i+1}, \ldots, \boldsymbol{x}_n$ where w is its weight, c its center and r its radius. DenStream assumes two types of micro-clusters: potential and outlier micro-clusters. A micro-cluster is said potential or outlier based on its weight restriction, where $w \geq \beta\psi$ implies in a potential micro-cluster and outlier otherwise and $0 \leq \beta \leq 1$.

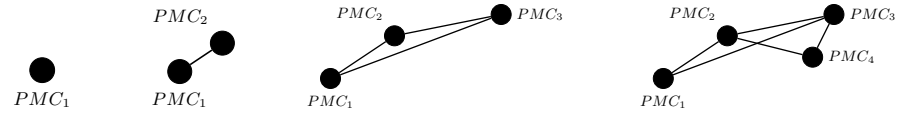
The online step of DenStream has the objective of maintaining a group of potential and outlier micro-clusters. At the arrival of each instance $\boldsymbol{x}_i$, DenStream tries to aggregate $\boldsymbol{x}_i$ to the closest potential micro-cluster accordingly to the weight restrictions. In the negative case, the same occurs for outlier micro-clusters. If $\boldsymbol{x}_i$ was aggregated in an outlier micro-cluster, the weight restriction is checked to determine whether this micro-cluster should be promoted to a potential micro-cluster. Conversely, if $\boldsymbol{x}_i$ was not merged with any micro-cluster at all, it starts a new outlier micro-cluster.

The offline step of DenStream uses the DBSCAN algorithm to find clusters based on current potential micro-clusters.

3 Complex Networks

Complex Network Theory has been applied in many research fields, from computer science to sociology, mainly due to its formal description of structural variables. Although complex networks analysis are mixed with social network analysis for subjective topics, such as an individual behavior in society, both of its building blocks can be represented computationally as a graph.

Different complex network models were developed over the years, aiming at representing the evolutionary aspects of real networks [3,7,12,14]. The first complex network model is denominated random [7]. This model is based on the hypothesis that the existence of a connection between any pair of nodes is given by a probability p. The Small-world model, based on the studies of "Small-World" conducted in [12], incorporates attributes of both random and regular (lattices) networks and presents high clustering coefficient and a small average



(a) PMC_1 (b) PMC_2 (c) PMC_3 (d) PMC_4

Fig. 1. Insertion example using $k = 2$.

path length. Finally, the Scale–free model aims on modeling networks presented in real–world situations with higher accuracy than both random and small–world networks by performing nodes additions and edges rewirings throughout time. In this paper we adapt the rewiring component for the clustering task so they occur accordingly to homophily. Homophily is a characteristic of real networks where nodes tend to eradicate connections with dissimilar nodes and replace these by new connections with more similar ones.

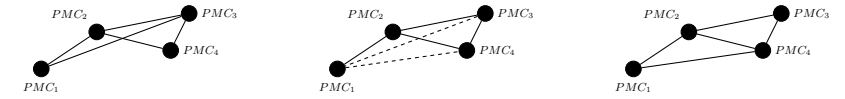
4 CNDenStream

Complex Network-based DenStream (CNDenStream) is based on the hypothesis that intra-cluster data are related due to high similarity and inter-cluster data are not related, due to high dissimilarity. CNDenStream generates a complex network $G = (V, E, W)$, where the set of nodes V are micro-clusters, edges E represent connections between these nodes, W is a set of weights (Euclidian distances) w_i associated to each edge $e_i \in E$ where subgroups in this network represent clusters and an outlier micro-cluster buffer $\mathcal{B}$. In order to keep track of clusters during the stream without the need of batch processing during the offline step, CNDenStream uses an homophily-based insertion and rewiring procedures inspired in complex networks theory.

Initially, CNDenStream stores the first N instances retrieved from $\mathcal{S}$ in a buffer to an initial DBSCAN run, thus finding initial potential and outlier micro-clusters. While outlier micro-clusters are stored in an outlier buffer $\mathcal{B}$, potential micro-clusters PMC_i are added to the network G, where each potential micro-cluster establishes with the k closest possible neighbors (considering Euclidian distances) currently in V.

Afterwards, PMC_i is added to V and edges and corresponding weights are added to its correspondent sets E and W. Fig. 1 presents the insertion of 4 potential micro-clusters, namely PMC_1 to PMC_4. The insertion procedure is able to connect the last added node to the k-most similar nodes in G, nevertheless, the same can not be said for the other nodes currently in G. In Fig. 1(d) one can see that after the addition of PMC_4, PMC_1 should be connected to PMC_4 instead of PMC_3, since $d(PMC_1, PMC_4) < d(PMC_1, PMC_3)$.

After the addition of each PMC_i obtained from the DBSCAN initial run to the network, all nodes $PMC_i \in V$ perform rewirings based on homophily, such that each PMC_i replaces its edges with higher dissimilarities w by edges



(a) Node PMC_1 rewiring. (b) Euclidean distance comparison. (c) Node PMC_1 rewired.

Fig. 2. Example of node PMC_1 rewiring.

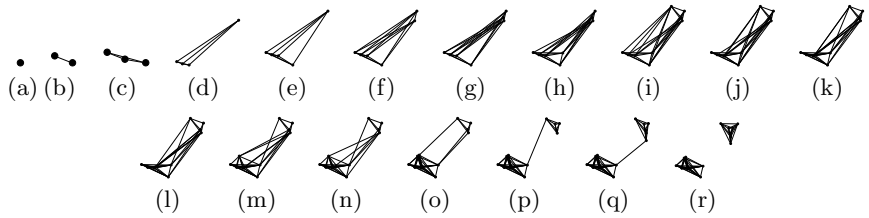


Fig. 3. Insertion of potential micro-clusters obtained from the insertion of micro-clusters during the RBF_2 experiment.

to its closest neighbors, i.e., edges with lower dissimilarity. For every PMC_i the Euclidean distances for all of its 2-hop neighbors are then computed. A 2-hop neighborhood is assumed since potential closest nodes are likely to be neighbors (2-hop) of the current neighbors (1-hop). This 2-hop neighborhood is an approximation in order to prevent distance computation between all nodes, which would be computationally costly. With the results of these Euclidean distances, PMC_i replaces edges by the most dissimilar instances with some similar ones, yet, maintaining its degree d_i.

In order to present how the rewiring procedure works, we refer back to the addition of nodes presented in Fig. 1, where one could see that PMC_1 is capable of connecting itself with higher similar nodes. Therefore, Fig. 2 presents the rewiring of node PMC_1. Firstly, Euclidean distances between PMC_1 and its 2-hop neighborhood are computed, and compared to its current neighbors (1-hop). In Fig. 2(b) one can see that $d(PMC_1, PMC_4) < d(PMC_1, PMC_3)$. Consequently, PMC_1, in order to maintain its degree $d_1 = 2$, must eliminate the its current most dissimilar edge to replace it with a similar one. Fig. 2(c) the edge between PMC_1 and PMC_3 is removed from G and a new one connecting PMC_1 and PMC_4 and its corresponding weight $d(\cdot, \cdot)$ are added to E and W.

Due to the rewiring process, communities of potential micro-clusters tend to appear naturally since the amount of intra-clusters edges between similar micro-clusters grows, while those of dissimilar micro-clusters shrinks. Fig. 3 presents the evolution of a network as instances arrive, where one can see that the rewiring procedure enlarges the amount of intra-cluster edges and diminishes the amount of inter-clusters connections. This procedure is repeated until, in Fig. 3(r), two clusters emerge.

After the DBSCAN execution and the initial network is build, all arriving instances $\boldsymbol{x}_i$ are processed according to an adaptation of the DenStream algorithm. Firstly, CNDenStream finds the potential micro-cluster in V which minimizes the dissimilarity with $\boldsymbol{x}_i$: PMC_i. Afterwards, CNDenStream verifies whether the addition of $\boldsymbol{x}_i$ with PMC_i results in a micro-cluster with a radius below ϵ, if true, then $\boldsymbol{x}_i$ is added to PMC_i. Otherwise, this process is repeated within the outlier micro-clusters: the most similar outlier micro-cluster OMC_j to $\boldsymbol{x}_i$ is found and if the addition of $\boldsymbol{x}_i$ results in a micro-cluster with radius below ϵ, $\boldsymbol{x}_i$ is then added to OMC_j.

When an outlier micro-cluster OMC_j is promoted to a potential micro-cluster, i.e., $w(OMC_j) \geq \beta\psi$, it is removed from the outlier buffer $\mathcal{B}$, and thus it is inserted in the network G.

As in DenStream, micro-clusters weights' decay exponentially with time. When the weight $w(PMC_i)$ of an micro-cluster PMC_i is below $\beta\psi$, it is removed from the network, or from $\mathcal{B}$. In the first case, all neighbors PMC_j of PMC_i are allowed to rewire in order to maintain their degree d_j after the PMC_i's removal. When the micro-cluster is an outlier, it is simply removed from $\mathcal{B}$.

CNDenStream's power resides in the rewiring process which aims to enable each micro-cluster to establish connections with the most similar micro-clusters. As presented in Fig. 1, the rewiring procedure finds clusters without using any batch clustering algorithm at the offline step such as k-means or DBSCAN.

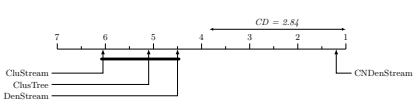
One could argue about the effects of the parameter k on the construction and evolution of the network, therefore, in Sec. 5, we discuss about the parameter sensitivity and show that $k = 4$ is a good choice for many data streams domains.

5 Experimental Evaluation

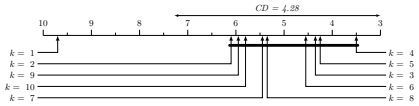
Our proposal is evaluated in several experiments with different types of data domains. Synthetic data streams were generated using the Radial Basis Function (RBF) generator, which creates a user-given number of drifting centroids, each defined by a class label, position, weight and standard deviation accordingly to a Gaussian distribution. In our experiments, the RBF generator is used for modeling concept drifts every 500 instances. Three data streams using the RBF generator were created changing the dimension of the instances $d = \{2, 5, 10\}$.

Additionally, we evaluated algorithms in two massive datasets, namely Forest Covertype [8] and KDD'99 [2], where clusters are non-hyper-spherical.

Algorithms parameters were set accordingly to its original papers. CluStream parameters are: a horizon $\mathcal{H} = 1000$ and $q = 1000$ [1]. ClusTree parameters are: a horizon $\mathcal{H} = 1000$ and a maximum tree height = 8 [9]. DenStream parameters are: $\psi = 1$, $N = 1000$, $\lambda = 0.25$, $\epsilon = 0.02$, $\beta = 0.2$ and an offline step multiplier $\eta = 2$ [6]. Finally, CNDenStream parameters are: $\psi = 1$, $N = 1000$, $\lambda = 0.25$, $\epsilon = 0.02$ and $\beta = 0.2$ and $k = 4$. All experiments were performed on a Intel Xeon CPU E5649 @ 2.53GHz $\times 8$ based computer running CentOS with 16GB of memory at MOA framework [5].



(a) CMM comparison with other algorithms.



(b) CMM comparison when varying the parameter k.

Fig. 4. Critical distances CMM comparison for results obtained in experiments.

5.1 Discussion

In order to evaluate algorithms in terms of clustering quality, we have adopted the Cluster Mapping Measure (CMM). CMM is an external clustering evaluation metric that accounts for non-associated and misassociated instances and noisy data inclusion [10]. Also, CMM considers recently retrieved instances with more weight than older ones by using an exponential decay function inside evaluation windows. In Fig. 4(a) we summarize the results obtained by algorithms after applying Friedman's and Nemenyi's tests, where one can see that CNDenStream is superior when compared to others with a 95% confidence level.

Besides CMM, we evaluated both CPU Time and RAM-Hours, however, Friedman test pointed out that there is no significant difference between algorithms in these two dimensions.

5.2 Parameter Sensitivity

In opposition to pure density-based algorithms, CNDenStream relies on the amount of connections k established at the arrival of each instance parameter to find and keep track of clusters. Therefore, to determine whether different values of k affect results directly, we ran all experiments varying it in the $[1; 10]$ interval.

In Fig. 4(b) we summarize the results obtained by applying Friedman and Nemenyi's tests, where one can see that $k \in [2; 10] \succ k = 1$. Also, one can see that $k = 4$ presents the best averaged rank, therefore, this value is adopted as a default value for CNDenStream.

6 Conclusion

In this paper CNDenStream algorithm was presented. CNDenStream is a one-step incremental complex network-based data stream clustering algorithm. It was empirically evaluated in both real and synthetic datasets where one can see that it achieves significant superior CMM when compared to others algorithms, while demanding similar resources (CPU Time and RAM-Hours). Additionally, CNDenStream does not make assumptions about the number of ground-truth clusters. This characteristic also allows the algorithm to naturally cope with concept evolutions.

In future works we expect to use archive programming techniques to optimize distance computation and develop a specific graph implementation to reduce memory usage. Besides, we envision experiments with other evaluation metrics, algorithms and datasets.

References

1. Charu C. Aggarwal. A framework for diagnosing changes in evolving data streams. In *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data*, SIGMOD '03, pages 575–586, New York, NY, USA, 2003. ACM.
2. Charu C. Aggarwal, Jiawei Han, Jianyong Wang, and Philip S. Yu. A framework for clustering evolving data streams. In *Proceedings of the 29th International Conference on Very Large Data Bases - Volume 29*, VLDB '03, pages 81–92. VLDB Endowment, 2003.
3. R. Albert and A. L. Barabási. Statistical mechanics of complex networks. In *Reviews of Modern Physics*, pages 139–148. The American Physical Society, January 2002.
4. Amineh Amini and Teh Ying Wah. On density-based data streams clustering algorithms: A survey. *Journal of Computer Science and Technology*, 29(1):116–141, 2014.
5. Albert Bifet, Geoff Holmes, Richard Kirkby, and Bernhard Pfahringer. Moa: Massive online analysis. *The Journal of Machine Learning Research*, 11:1601–1604, 2010.
6. Feng Cao, Martin Ester, Weining Qian, and Aoying Zhou. Density-based clustering over an evolving data stream with noise. In *SDM*, pages 328–339, 2006.
7. P. Erdos and A. Rényi. On the evolution of random graphs. In *Publication of the Mathematical Institute of the Hungarian Academy of Sciences*, pages 17–61, 1960.
8. Petr Kosina and João Gama. Very fast decision rules for multi-class problems. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, SAC '12, pages 795–800, New York, NY, USA, 2012. ACM.
9. Philipp Kranen, Ira Assent, Corinna Baldauf, and Thomas Seidl. The clustree: Indexing micro-clusters for anytime stream mining. *Knowl. Inf. Syst.*, 29(2):249–272, November 2011.
10. Hardy Kremer, Philipp Kranen, Timm Jansen, Thomas Seidl, Albert Bifet, Geoff Holmes, and Bernhard Pfahringer. An effective evaluation measure for clustering on evolving data streams. In *Proc. of the 17th ACM Conference on Knowledge Discovery and Data Mining (SIGKDD 2011), San Diego, CA, USA*, pages 868–876, New York, NY, USA, 2011. ACM.
11. S. Lloyd. Least squares quantization in pcm. *IEEE Trans. Inf. Theor.*, 28(2):129–137, September 1982.
12. S. Milgram. The small world problem. *Psychology Today*, 1(1):61–67, May 1967.
13. Jonathan A. Silva, Elaine R. Faria, Rodrigo C. Barros, Eduardo R. Hruschka, André C. P. L. F. de Carvalho, and João Gama. Data stream clustering: A survey. *ACM Comput. Surv.*, 46(1):13:1–13:31, July 2013.
14. D. J. Watts and S. H. Strogatz. Collective dynamics of small-world networks. *Nature*, 393(6684):440–442, June 1998.