Tetkik: Akan Veri Kümeleme Algoritmalarını Çalıştırma ve Karşılaştırma

Rowanda D. Ahmed[1], Gökhan Dalkılıç[2], Murat Erten[3]
Izmir Institute of Technology[1,3], Dokuz Eylül University[2]
Computer Engineering Department, İzmir, Turkey
rowandaahmed@iyte.edu.tr, dalkilic@cs.deu.edu.tr, muraterten@iyte.edu.tr

Özet: Son zamanlarda, veri akışlarını kümelemek uygulamalar daha fazla durdurulamaz veri akışı üretirken bilgi keşfi için inanılmaz derecede önemli bir araştırma alanı haline gelmiştir.Bu makalede, kümeleme, akışlar ve veri akışlarını kümeleme algoritmalarını en önemli akım kümeleme algoritmalarının irdelenmesini yapılarını da göz önünde bulundurarak tanıtıyoruz. Çalışmamızın ek bir katkısı ve akış kümeleme alanında yapılmış tetkit ve gözden geçirme makalelerinden farklı olarak en bilinen akış kümeleme algoritmalarının Pratik kısmını, yani: (i) CluStream; (ii) DenStream; (iii) D-Stream; and (iv) ClusTree, MOA Java çerçevesine bağlı olarak, her biri için bazı performans metriklerinin hesaplanmasıyla birlikte deney sonuçlarını göstererek sunuyoruz.
Anahtar Kelimeler: Veri Madenciliği; Kümeleme; Veri Akışı Kümeleme; Yoğunluk-Tabanlı Algoritmalar.

Survey: Running and Comparing Stream Clustering Algorithms

Rowanda D. Ahmed[1], Gökhan Dalkılıç[2], Murat Erten[3]
Izmir Institute of Technology[1,3], Dokuz Eylül University[2]
Computer Engineering Department, İzmir, Turkey
rowandaahmed@iyte.edu.tr, dalkilic@cs.deu.edu.tr, muraterten@iyte.edu.tr

Abstract. Recently, clustering data streams have become an incredibly important research area for knowledge discovery as applications produce more and more unstoppable streaming data. In this paper we introduce clustering, streams and data streaming clustering algorithms, as well as discussions of the most important stream clustering algorithms, considering their structure. As an additional contribution of our work and differently from review and survey papers in stream clustering, we offer the practical part of the most known stream clustering algorithms, namely: (i) CluStream; (ii) DenStream; (iii) D-Stream; and (iv) ClusTree, showing their experimental results along with some performance metrics computation of for each, depending on MOA framework.
Keywords: Data Mining; Clustering; Data Stream Clustering; Density-base Algorithms.

1 Introduction

Clustering process is the most main and crucial part in the data mining field. And it passes through many levels of developments [1] especially in recent years. Every moment the wide, applicability of streaming data leads to an unlimited rate growing large amounts of data. So, we need non-stopped strategies for analyzing and clustering the non-stopped online coming stream objects in which there is always an up-to-date view of the objects seen and clustered so far. Stream data clustering faces additional challenges beyond those of the static data clustering faces. Some challenges streams bring are as follows: **Single-pass processing** the endless amounts of data streams, as well as maintaining all the necessary summary information which will be used in the clustering process. **Limited time is** needed for processing each record and it must be small. This ensures maintaining always such a current clustering model and keeping up with the stream speed. **Limited memory** that processing in the stream must be done without storing, buffering or revisiting data. **Varying time allowances,** that is the usual case for stream data is in in a bursty state. The continuously **Evolving data** over time [2] is so ubiquitous in today's applications. Therefore, the clusters in some point of time may no longer still relevant in the future [3], which leads to record many interesting changes and characteristics in an evolving data stream for many business applications effective usage [4]. **Concept drift**, **novelty**, **number and size of clusters** and **outlier detection** should be considered as well. In addition to the **scalability** issue which considered the most important issue in stream clustering when the data sets are very large [5]. Data stream applications may need results at **any time**, so the model must be available at any time.

Performance metrics. determine how good the obtained clustering reflect the data. We can classify the Cluster evaluation methods into two kinds: extrinsic methods and intrinsic methods.

Extrinsic methods (supervised). when the ground truths are available, so we can assign like score to the clustering. In Extrinsic methods, the ground truth compared to a clustering results. Purity, Precision, and recall metrics are examples of extrinsic methods. **Precision-Recall** is used to measure the success of the prediction, especially in the very imbalanced classes case. In information retrieval, precision measures the output relevancy, but recall deals with the returned results and measures how many truly relevant from them. **F1-Score** is the harmonic mean or weighted average of recall and precision to evaluate an algorithm. The F1 score reaches the best score at 1 and worst score at 0. **Purity**: is a measure of how extent clusters contain a single class [6]. The higher the purity is the better. A purity score of 1 is possible by putting each data point in its own class. **SSQ**: The sum of squared distances over all data points to their corresponding cluster centers. The smaller the SSQ is the better. **Homogeneity**: Each cluster contains only members of a single class. **Completeness**: we suppose that all members of some class are assigned to the one cluster. The Completeness and Homogeneity bounds are from 0 to 1. The higher is the better for both.

Intrinsic methods (unsupervised). when the ground truths are unavailable. In Intrinsic methods, how compact the clusters are, how well the clusters are separated are evaluated, i.e. Intrinsic methods are all about measuring the clustering goodness. **Silhouette coefficient** is an example of intrinsic methods. It is a metric used to

compare how the point will fit in some cluster with the assigned one by computing both values. Silhouette Coefficient is a combination of Cohesion and Separation measures and its bounds are between 0 and 1. The higher the Silhouette Coefficient, the better. This survey research paper is organized as follows: in section 2, we will study in general variety of stream clustering methods. In section 3, 4, 5 and 6 we will study each of the following stream clustering methods; CluStream, DenStream, D-Stream and ClusTree respectively. In section 7 we will compare by running between the above-mentioned algorithms. Finally, the conclusion will be in section 8.

2 Stream Clustering Algorithms

The old unsophisticated data stream algorithms suffer from many problems. One naive suggested solution refers to stream buffering for later handling, but this is impossible because the stream is endless. Some approaches lost a valuable information due to dropping some data to keep up with the stream speed. Other approaches are suggested to solve the stream speed adaptation, these approaches try to scale only for the fastest stream speed, which resulted in a poor-quality clustering. Clearly, these solutions mentioned above do not make the best use of the information that is contained in the stream and of the time available. The current stream clustering algorithms each try to solve some of the above-mentioned problems.

Stream clustering can be categorized into three main categories; prototype methods, density methods, and model-based methods [7]. Example on prototype-based algorithms K-Means [8], it aims to partition the objects of the dataset into k clusters or groups so that each object belongs to the cluster corresponds to the closest mean, and then refine these clusters iteratively. The density-based algorithms like DBSCAN [9], it groups the points that look close together, considering dense concentrated data points areas, and then these dense areas form the final clusters. Then the low-density areas are marks as outliers. Model-based algorithms, Expectation Maximization (EM) [10] is an example. In [11], there are many clustering methods have been spoken about that are designed for static not stream data. There are different clustering paradigms studied the data stream clustering. Earlier data stream clustering paradigms like [12] deal with data stream clustering as if static clustering but in a continuous version, they are single phase divide and conquer schemes based. These data stream algorithms divide the data streams into segments and based on a k-means algorithm to discover clusters in data streams. Such approaches don't consider the evolving data due to that they treat the recent and the outdated data the same. To solve the evolving data problem, moving window techniques are proposed [13]. CluStream [14] uses a two-phase strategy. In the online phase, it analyzes the stream coming data and stores its summary statistics using micro-clusters. While in the offline phase, it uses these statistics along with other parameters to generate final clusters. Wang et al. [15], Clustering data streams on the Two-tier structure, is an online-offline framework. It based on CluStream, but in an improved offline phase. DenStream [16] presents the structures as core and outlier micro clusters to maintain and summarize clusters and distinguish the potential clusters and outliers. CobWeb [17] is an incremental system for hierarchical

conceptual clustering data. It uses a classification tree in organizing observations. Density-based clustering method like D-Stream [18], is a natural and attractive basic data streams clustering strategy because it can find arbitrarily and interwoven shaped clusters and can detect and handle noises. BIRCH [19], maintained a hierarchical index for faster access in the very large databases. A mapping to a frequent item set for multidimensional streams is represented in [20]. Kernels based approach [21] is proposed to discover arbitrary shape clusters in stream data. Alternative graphs based is proposed in [22], and fractal dimensions grid based is proposed in [23], and [16], [24] proposed a density-based approach. None from the above approaches can adapt the clustering model size to keep up with the online stream speed nor capable of delivering such a result or can interrupt the process at any time moment. The anytime idea is a very active research field in data mining, there are variant algorithms in that field. ClusTree [25] is a free parameter any-time algorithm, it adapts itself to the stream speed automatically, in addition to its capability of detecting novelty, outliers, and concept drift in the stream. To maintain stream summaries, it uses a self-adaptive and compact index structure.

3 CluStream Algorithm

CluStream is an online-offline Algorithm adopts the idea of streaming over many time windows, that by streaming over many time windows gains more understanding about what is going on in clustering process and the clusters behaviors and evolving. CluStream algorithm stores data summary statistics in its online component, while uses these statistics along with other parameters in the offline component to achieve the final clustering results. CluStream algorithm uses the micro-clusters concept as well as the pyramidal time frame.

3.1 CluStream Clustering Framework

To understand the CluStream framework, it is important to clarify some points. Firstly, summary statistics in the online component are incrementally updated to cope with the offline clustering. Secondly, it is important to determine the time interval between the snapshots for storing the summary statistics. Thirdly, addressing how CluStream uses the online information benefits in gaining hints for the user in setting the time horizon and how to deal with the evolution. CluStream store the online statistics in a form of micro-clusters, for that it uses the feature vector [19] due to its natural properties of additive, subtraction, and multiplications. Algorithm stores these micro-clusters in periods of times following pyramidal pattern which provides such good and reasonable tradeoff between the ability to get the summary statistics stored in the micro-clusters from various time horizons and the storage requirements. Definition 1 in [14] facilitates understanding of the above-mentioned concepts.

3.2 Online phase (Micro-Cluster Maintenance)

CluStream using many points initially to produce the initial micro-clusters uses k-means in its offline phase. Now and after the creation of these initial micro-clusters, CluStream starts the online process with every arriving data point to update the stored micro-clusters created previously. The new coming data point either it absorbs in the nearest one or it may create its new micro-cluster as own. When a new point arrives, the algorithm computes the distance between the new point and the nearest micro-cluster center. If the choice is to merge the arriving data point, the algorithm merges this data point with the closest micro cluster. Otherwise, if the new point doesn't be founded to be within the range of any micro-cluster regarding its radius parameter, the new data point may categorize as an outlier or as a new micro-cluster seed.

3.3 Offline phase - Macro-Cluster Creating

Micro-clusters are efficiently maintained to be as intermediate statistical representations using the stored micro-clusters summary statistics instead of the very large volume data stream. This process enables the user from flexibility exploring stream clusters over different horizons. The user provides two parameters; the first is the time-horizon h and this benefits in history determination in which to create higher level clusters. The second parameter is the higher-level clusters number k, and this to determine if we can find extra detailed clusters. Note that at each step of the algorithm, the current set of micro-clusters depend on the all the stream processing entire history since the very beginning of it. To find the clusters over a specific time horizon, we need to find the corresponding micro-clusters making use from the additive and subtractive properties which extended from [19].

4 DenStream Algorithm

In the data stream clustering, maintaining all the data is impossible. Therefore, it is important to discover algorithms with a single pass clustering, unknown parameters and evolving the changed data. Micro-cluster technique in stream clustering saves the necessary information of the data objects the stream, which compresses the data effectively. The following subsections will be about four well-known density-based data stream algorithms which extend micro-clustering technique, these are DenStream, C-DenStream, rDenStream, and SDStream. We will start illustrating DenStream algorithm with more details because the others based on it.

4.1 Density Micro-Clustering Algorithms

Adopting the density concept benefits in discovering the arbitrarily shaped clusters by distinguishing those dense areas from the scattered sparse areas. Due to memory

limits, micro-cluster is a well-known strategy special for this purpose. Micro-clusters are the optimal representation for the summary statistics stored in the online component and to be used afterward for the offline component clustering. Fig.1 shows the framework of micro-clusters in density-based clustering. The p-micro-clusters and o-micro-clusters [16] are the potential and outlier clusters, which are mainly differ in their weights. In this section, we will show in brief the density-based micro-clustering [19] four stream algorithms, DenStream, SDStream, rDenstream and C-DenStream with their cons and pros. So, the user can choose the suitable algorithm according to his own preferences and what he is interested in. Like the speed, higher accuracy and so on.

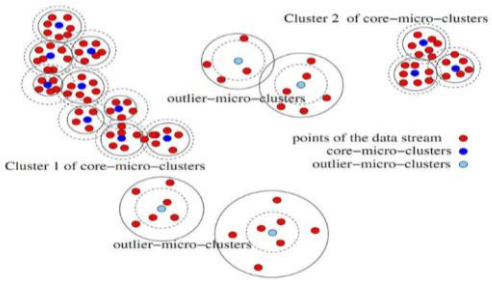

Fig. 1. a) Density based micro-clusters framework [26].

The **DenStream** algorithm puts a weight for each data point. It is a fast algorithm due to its ability to delete those outdated data points and the outliers before merging. And because it doesn't merge data points into a micro-cluster and that facilitate exploring the outliers and save time. **SDStream** algorithm uses only the recent data stream objects, while other algorithms consider the data stream points whole. It stores the micro-clusters in the EHCF data structure form. So, SDStream save memory and it can process only with the most recent data points over the sliding window length, which means better addressing the clusters changes and evolutions. This algorithm is suitable when the applications interested more in the recent data streams. **rDenStream** algorithm based mainly on DenStream; however, rDenStrean emphasize on handling outliers, such that it chose not to remove the data in the outlier buffer and relearn from it which result in a higher accuracy compared to DenStream algorithm. But on the other hand, it is slower than DenStream and a memory usage as well. **C-DenStream** is a real applicable algorithm. It adopts the constraint concept on micro-clusters. In addition, to guide the clustering process, it uses a background knowledge. In this algorithm, the clusters can't have formed unless they conform with application semantics like geographical natural borders and objects.

5 D-Stream

D-Stream is a density-based framework for clustering stream data. It is an online-offline algorithm. The online one continuously reads input data records and maps it into a density grid and the offline one computes the grid density, detect and removes sporadic grids from the grid-list and adjust the clustering. D-Stream algorithm exploits the complicated relationships between the decay factor, data density, and cluster structure [18]. In addition, it captures the dynamic changes by adopting a density decaying technique to a data stream to generate clusters of arbitrary shapes effectively and efficiently.

5.1 D-Stream Components

Grid inspection and gap determination. D-Stream algorithm progressively does adjust clustering process every gap time. But what is the optimal time length of the grid inspection? Note that if we choose this time interval to be too long, data streams dynamical changes will not be recognized well. On the other hand, that if we choose this time interval to be too small, then it will result in too much computation, which makes the work so heavy. D-Stream adopts the idea of setting the time interval to be the smallest of those two times intervals [18].

Sporadic grids removing. The very high number of grids is a critical big challenge D-Stream algorithm faces. But the most grids are either empty or don't receive over and over data records. So, in processing and storing, only those not empty grids can be regarded, and the others are neglected. The sporadic grids who receive so little data can be deleted from the data space along with their characteristic vectors and reset their density to zero. It is experimentally proven that deleting this kind of grids doesn't affect the clustering quality. While the sporadic grids with many data records but their densities became less than the threshold by the effect of decay factor have a hope to upgrade and to become dense or transitional grids, so it is wrong to delete these grids.

D-Stream clustering algorithm. The algorithm continues reading from the flow stream of data records and computes all grids densities. Initial clustering process generates the initial clusters. After the first gap then adjust-clustering method is executed repeatedly every time equal gap. Initial-clustering and adjust-clustering methods are described in detail supported by pseudo codes in [27].

6 ClusTree

ClusTree is a parameter-free and any-time data stream algorithm, it finds solutions for a lot of challenges like limited memory and time, maintaining results in any point of time and adapting the clustering process according to the steam speed. In addition,

ClusTree uses novel descent strategies for handling the slow streams, and it uses aggregation mechanisms for handling the fast streams. Also, ClusTree uses an exponential decay function to give more importance to the more recent data points. It is a self-adaptive data stream clustering algorithm as well as it is scalable to give any-time clustering results. ClusTree' efficiency and effectivity are experimentally approved.

6.1 Self adaptive and anytime data stream clustering

ClusTree is a structure with an index stream clustering algorithm, it can store and maintain a compact view for any time the user seeks it. It is the first stream clustering algorithm for the anytime merit. ClusTree is a self-adaptive algorithm so that it can adapt to the coming data points stream speed automatically.

Micro-clusters and anytime insert. Periodically, ClusTree algorithm computes the mean and variance of its micro-clusters. As stream data points flow, the cluster feature is updated incrementally, it can be considered as the true representation of the micro-clusters. So that, stream data points can be mapped to the true or the most similar micro-cluster. ClusTree algorithm hierarchy builds micro-clusters with a granularity at different levels [25]. So that, it adopts hierarchical indexing structures from R-tree family [17], [28], [29] which render preserving cluster features as well as locate a right place efficiently, that's to insert any coming stream object into such a suitable micro-cluster. For the algorithm to determine leaf entry containing the most similar micro cluster to the new coming object, it descends the tree down until reach that leaf. If the similarity is enough, algorithm updates the micro-cluster tuple values. Otherwise, it creates a new micro-cluster and forms its cluster feature (CF) with its values. ClusTree has a buffer as a temporary place for storing either objects or aggregates that during insertion, due to the fast stream, can't reach the leaf level. So that the cluster feature is stored in a suitable entry buffer when the insertion is interrupted and so that there is not enough time to descend the tree down to reach the leaf.

Up-to-date clustering maintaining. ClusTree algorithm, to give the most recent objects more importance, it uses an exponential famous decay function that depends on time, $\omega(t) = \beta - \lambda * \Delta t$, with the λ is the decay rate to control the objects weighs in such a way makes the algorithm can control the forgotten or maintaining objects more by playing with the decay rate value. As we increase the value of the decay factor, the old objects are forgotten becoming faster and so on. Every object has its arrival and last update timestamps which needed in computing the decay function. While the insertion process and descend the tree down to the leaf, all entries in the node are updated to the arrival timestamp. So that the node entries all have the same timestamp always. Every inner node in the tree structure summarizes its subtree. So, contacting the last update timestamp field in every node, besides using weighting formula,

ensures capturing decay with time correctly. And to avoid splitting, the algorithm weighs with time. approved.

6.2 Cluster shapes and cluster transitions

There are several strategies suggested by details in [30], can process cluster transitions like outlier detection, concept drift detection, and novelty. ClusTree can apply any from these approaches to its output clusters. So, ClusTree algorithm can be used to detect arbitrary shape clusters, outlier detection, concept drift detection, novelty and time horizon.

7 Experimental Results

In experimental results, we adopt the Massive Online Analysis (MOA) framework [31] for online learning from data streams. MOA is so closely related to WEKA, implemented in Java and released under GPL. It includes a collection of online and offline components in addition to many tools for evaluating classification and clustering. It is easy to design and run experiments on MOA, and it is easy to extend it too. The experiments were done on Intel core(TM) i7-4510U CPU @ 2.60GHz, 6.00GB RAM Lenovo. Fig. 2 shows the results from running four stream clustering algorithms, CluStream, DenStream, D-Stream and ClusTree based on synthetically Radial Basis Function (RBF) based on generated data. Synthetic data is easier in reproducing its storage and transmission costs are little [32].

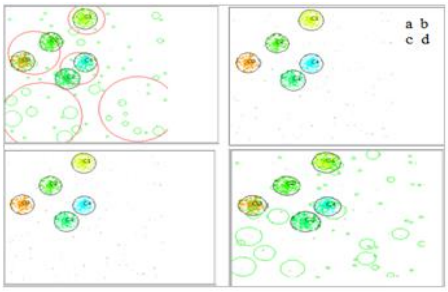

Fig. 2. a) CluStream b) DenStream c) D-Stream d) ClusTree with 50000 points (*Synthetic dataset*).

Fig. 3 shows the quality comparison between ClusTree with CluStream, D-Stream and DenStream algorithms using the sum of squared distance (SQQ).

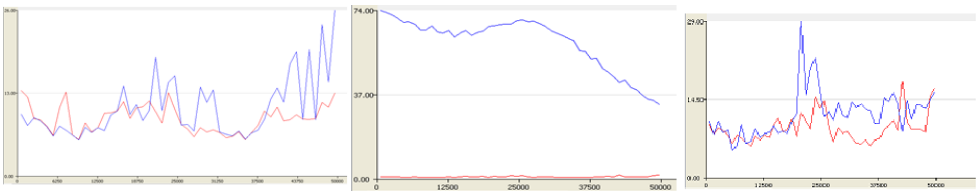

Fig. 3. SSQ for 50000 points. ClusTree in red, CluStream (*left*) and D-Stream (*middle*), DenStream (*right*) in blue.

From the Fig. 3, it is shown that ClusTree has a quality with a virtually higher than the D-Stream algorithm, and at the same range as CluStream. ClusTree is also a little bit better than DenStream in SSQ and it is so clear from the results listed in Table 1 below too.

The separation (BSS) is the between clusters sum of squared distances. The higher the BSS, the better clustering quality. And it is shown in Fig. 4 (*left*) how much CluStream is better than ClusTree. CluStream is from two to five times BSS better than ClusTree.

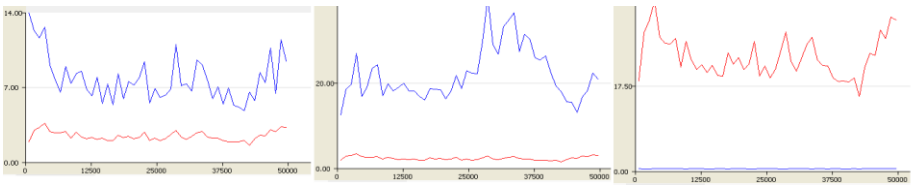

Fig. 4. BSS for 50000 points. ClusTree in red, CluStream (*left*) and DenStream (*middle*), D-Stream (*right*) in blue.

In Fig. 4 (*middle*), it is a comparison between ClusTree and DenStream. DenStream is much better than ClusTree and it is corresponding to order of magnitude.

In terms of BSS metric, ClusTree is only better than D-Stream and this is shown clearly in Fig. 4 (*right*) and from Table 1 too. The values of BSS in clustering 50000 stream data points are 72.62, 217.82, 6.49 and 22.93 for CluStream, DenStream, D-Stream and ClusTree respectively. Note that the D-Stream BSS value is the least one so that it is the worse algorithm in terms of BSS performance metric.

Table 1. Stream Clustering Algorithms metric performance (Clustering 50000 points).

	CluStream	DenStream	D-Stream	ClusTree
Separability	0.80	0.80	0.38	0.80
Noise	0.84	0.84	1.00	0.84
F1-P	0.72	0.60	0.33	0.74
Purity	0.85	0.88	0.64	0.86

Precision	1.00	1.00	1.00	1.00
Recall	0.88	0.67	0.99	0.89
Redundancy	0.06	0.00	0.00	0.04
SSQ	8.93	11.51	3117.54	8.94
BSS	72.62	217.82	6.49	22.93

8 Conclusion

In the introduction, it is indicated due to the unlimited rate growing large amounts of data how much the need for such non-stopped strategies for analyzing and clustering the non-stopped online coming stream objects especially in recent years becomes so important. So that a wide set of techniques and strategies have been proposed for analyzing and clustering stream. Also, in this paper additional challenges stream data clustering are mentioned. In addition to explaining the most important performance metrics used in comparing between clustering methods quality. Finally, experimental results of these streaming methods have also been displayed.

References

1. R. D. Ahmed, M. Alhanjouri,: New Density-Based Clustering Technique: GMDBSCAN-UR, Islamic university of Gaza, International Journal of Advanced Research in Computer Science, No. 1, Jan-Feb 2012.
2. Huachun Liu, Xiangning Hou and Zhong Yang,: Design of Intrusion Detection System Based on Improved K - means Algorithm[J], Computer Technology and Development, 2016(01): 101-105.
3. C. C. Aggarwal,: A Survey of Stream Clustering Algorithms, Chapter 10, IBM T. J. Watson Research Center, Yorktown Heights, NY 10598
4. C. C. Aggarwal.: A Framework for Diagnosing Changes in Evolving Data Streams. ACM SIG- MOD Conference, 2003.
5. F. Faranstorm, J. Lewis, and C. Elkan.: Scalability for Clustering Algorithms Revisited. ACM SIGKDD Explorations, 2(1): pp.51-57, 2000
6. Manning, Christopher D.; Raghavan, Prabhakar; Schütze, Hinrich.: Introduction to Information Retrieval. Cambridge University Press. ISBN 978-0-521-86571-5.
7. A. Al Abd Alazeez, S. Jassim, and H. Du,: EINCKM: An Enhanced Prototype-based Method for Clustering Evolving Data Streams in Big Data, Proc. 6th Int. Conf. Pattern Recognit. Appl. Methods, no. Icpram, pp. 173–183, 2017.
8. J. MacQueen,: Some Methods for classification and Analysis of Multivariate Observations," 5th Berkeley Symp. Math. Stat. Probab. 1967, vol. 1, no. 233, pp. 281–297, 1967.
9. M. Ester, H.-P. Kriegel, J. Sander, and X. Xu,: A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise, 2nd Int. Conf. Know- ledge Discov. Data Min., vol. 2, pp. 226–231, 1996.
10. A. P. Dempster, N. M. Laird, and D. B. Rubin,: Maximum Likelihood from Incomplete Data via the EM Algorithm, J. R. Stat. Soc. Ser. B (Methodological), vol. 39, no. 1, pp. 1–38, 1977.

11. Jain A, Zhang Z, Chang EY (2006).: Adaptive non-linear clustering in data streams, CIKM, pp 122–131
12. S. Guha, A. Meyerson, N. Mishra, R. Motwani, and L. O'Callaghan.: Clustering data streams: Theory and practice. Trans. Know. Eng., 15(3):515–528, 2003.
13. D. Barbar´a.: Requirements for clustering data streams. SIGKDD Explorations Newsletter, 3(2):23–27, 2002.
14. Aggarwal CC, Han J, Wang J, Yu PS (2003).: A framework for clustering evolving data streams. VLDB, Berlin, pp 81–92.
15. Z. Wang, B. Wang, C. Zhou, and X. Xu.: Clustering Data streams on the Two-tier structure. Advanced Web Technologies and Applications, pages 416–425, 2004.
16. F. Cao, M. Ester, W. Qian, and A. Zhou.: Density-based clustering over an evolving data stream with noise, in SIAM Conference on Data Mining, 2006, pp. 328–339.
17. V.Kanageswari, Dr.A.Pethalakshmi.: A Novel Approach of Clustering Using COBWEB Department of Computer Science, M.V.Muthiah Government Arts College for Women, Dindigul Tamil Nadu – India.
18. Y. Chen, and L. Tu.: Density-based clustering for real time stream data, ACM KDD Conference, 2007.
19. Zhang T, Ramakrishnan R, Livny M (1996).: BIRCH: an efficient data clustering method for very large databases. SIGMOD, NY, USA.
20. Assent I, Krieger R, Glavic B, Seidl T (2008).: Clustering multidimensional sequences in spatial and temporal databases. Knowl Inf Syst 16(1):29–51
21. Jain A, Zhang Z, Chang EY (2006).: Adaptive non-linear clustering in data streams, CIKM, pp 122–131
22. Lühr S, Lazarescu M (2009).: Incremental clustering of dynamic data streams using connectivity based representative points. Data Knowl Eng 68(1):1–27.
23. Barbará D, Chen P (2000).: Using the fractal dimension to cluster datasets, KDD, pp 260–264.
24. Chen Y, Tu L(2007).: Density-based clustering for real-time stream data, KDD, pp 133–142.
25. P. Kranen, C. Baldauf, T. Seidl, I. Assent, The ClusTree: indexing micro-clusters for anytime stream mining, RWTH Aachen University, Aachen, Germany, Aarhus University, Aarhus, 2010.
26. D. K. Tasoulis, G. Ross, and N. M. Adams, "Visualising the cluster structure of data streams," in Proceedings of the 7th international conference on Intelligent data analysis, ser. IDA'07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 81–92.
27. David Arthur, Sergei Vassilvitskii.: k-means++: The Advantages of Careful Seeding.
28. A. Zhou, F. Cao, W. Qian, and C. Jin.: Tracking clusters in evolving data streams over sliding windows, Knowledge and Information Systems, vol. 15, pp. 181–214, May 2008.
29. Guttman A (1984) R-trees: A Dynamic Index Structure for Spatial Searching. SIGMOD, Boston, pp 47–57
30. Spiliopoulou M, Ntoutsi I, Theodoridis Y, Schult R (2006) Monic: Modeling and Monitoring Cluster Transitions, KDD, pp 706–711.
31. A. Bifet, G. Holmes, R. Kirkby, B. Pfahringer, (2010), MOA: Massive Online Analysis; Journal of Machine Learning Research 11: 1601-1604
32. A. Bifet, G. Holmes, R. Kirkby, B. Pfahringer.: Data Stream Mining: A Practical Approach, May 2011.