**、Research Proposal**

**Note : not to exceed 6 pages, Text to be in Arial font 11 pt, single spacing**

1. Project Title:

An In-depth Evaluation of Stream Clustering Algorithms

1. Project Objectives:

The proposed research targets at addressing the research issues relating to the modularity design and benchmark study of *data stream clustering* algorithms. Specifically, we will propose the following key mechanisms with algorithmic and technical innovations. **First**, we provide a taxonomy of existing stream clustering algorithms by carefully summarizing their key design aspects including *data structure*, *window model*, *offline refinement*, *outlier detection*, *summarization*, and *concept drift detection and adaption*. For each design aspect, we also discuss the state-of-the-art variants and reveals any potential improvements. **Second**, we assemble and implement about 20 state-of-the-art stream clustering algorithms following a modularity design, which separates key design aspects. Ultimately, each algorithm can be constructed by composing software components implemented for different design choices of each design aspect. This allows us to better comprehend the different trade-offs and performance behaviours among algorithms in a much finer granularity. **Third**, we design a comprehensive benchmark suite for benchmarking stream clustering algorithms. The benchmark must include diversity of data and application scenarios, which is a prerequisite for any meaningful benchmark for evaluating stream clustering algorithms. **Fourth**, with the benchmark, we conduct a detailed profiling study on the different design aspects of stream clustering algorithms. We explore and further reveal how various key design aspects of the algorithm affect clustering performance respectively and the relationship between design components. Besides, the findings and optimizations proposed from this project can be applicable to future studies in terms of new algorithms and/or new hardware architectures. For example, the final evaluation results can be used as a reference in extensions of the stream clustering key design aspects in current and future stream clustering methods. We hope our work can bring the entire community to the common consciousness and move forward. This is also a necessary and fundamental step to develop more efficient massive online AI-driven applications, where data stream clustering operation is often used as a preliminary and important building block.

1. Area of Research **(including innovative claims***):*

*• Overview of the proposal.*

Advanced analysis of big data streams from sensors and devices is becoming a key area of data mining research as the number of applications requiring such processing increases rapidly. Data stream clustering is one of the fundamental stream analysis operations and has been studied widely over the last decades. As streams consist of unbounded, non-stationary data objects that continuously arrive at rapid rates, their intrinsic nature requires stream clustering algorithms capable of processing data and extracting knowledge under the challenges of limited time, memory, and requires only single pass (i.e., allow nearly no revisit of the input data). In the last decades, enormous stream clustering algorithms have been proposed to satisfy these technical requirements. However, to the best of our knowledge, there is still a lack of in-depth comparison of the different approaches that have been performed, and it remains unclear of key weaknesses and strengths of each of the existing stream clustering algorithms. On the one hand, many of existing algorithms are designed with implicit assumptions such as their specifically targeted application domains including network intrusion detection[6], sensor network monitoring and stock market analysis. Furthermore, many of them are evaluated based on different performance metrics (e.g., varying ways of determining clustering accuracy). As a result, there is no one “standard” implementation and researchers and practitioners are often left alone to decide which algorithm they should adopt in their applications. We observe that there are several design choices that have different trade-offs and performance behaviours, but there is no comprehensive experiment study to evaluate them on modern operating environment. On the other hand, many libraries and frameworks of data stream mining have been proposed such as MOA[8], SAMOA, StreamDM[16], and FlinkML. However, there are multiple limitations of existing libraries and frameworks. First, the current frameworks or packages do not cover comprehensive state-of-the-art stream clustering algorithms, partially because most of the state-of-the-art stream clustering algorithms are not open-sourced. As a result, most prior empirical evaluation works are biased especially if they rely on existing frameworks with a narrowed subset of algorithms[21]. Second, their implementations of algorithms are fragmented, ad-hoc, and sometimes even based on different programming languages and compilers. This also leads to unfair comparisons due to the differences caused by programming languages and compilers. Moreover, they tend to use a complex language environment with dependence on others, which are hardly extensible.

To respond to this situation, the proposed research targets at addressing the research issues relating to the modularity design and benchmark study of data stream clustering algorithms. Specifically, we have the following inter-related goals:

1. **A taxonomy of stream clustering algorithms:** enormous stream clustering algorithms have been proposed and many of them are designed with implicit assumptions such as their specifically targeted application domains. This naturally leads to an open question: which one is better? Rather than comparing algorithms entirely, we provide a taxonomy of 20 state-of-the-art stream clustering algorithms by summarizing their key design aspects. For each design aspect, we discuss the state-of-the-art variants and reveals any potential improvements. This is critical to identify the entire solution space for stream clustering algorithms rather than proposing incremental improvements over existing work.
2. **A modularity design of stream clustering algorithms:** We implement stream clustering algorithms following a modularity design, which separates key design aspects. This allows us to better comprehend their different trade-offs and performance behaviours and is only made possible by our careful analysis of existing algorithms and on our taxonomy of stream clustering algorithms. Different from prior works, we implement all existing algorithms from the ground up inside the same code-base to eliminate the differences among existing implementations caused by programming languages and compilers.
3. **A** sophisticate **benchmark for stream clustering:** The benchmark must include diversity of data and workloads, which is a prerequisite for any meaningful benchmark for evaluating stream clustering algorithms. In particular, there are two important aspects to consider in the designing of our benchmark: 1) it must cover a wide range of representative application domains; 2) it must be scalable to suit different scale of input workloads; 3) it must cover a wide variety of evaluation data to test the adaptability of stream clustering algorithms towards different data types.
4. **A detailed profiling study of benchmarking stream clustering algorithms:** We conduct a detailed profiling study on the different design aspects of stream clustering algorithms. The findings and optimizations can be applicable to future studies in terms of new algorithms and/or new hardware architectures. We systematically examine the profiling results of stream clustering algorithms running the benchmark, with careful considerations on the following dimensions: 1) accuracy: there are different ways to measure the accuracy of stream clustering results and we need to consider all of them comprehensively; 2) Performance analysis: verify overall throughput and the specific latency of each key design module in the stream clustering algorithms; 3) processor architectures such as stalls in pipelines and cache/memory systems; 4) applications with different resource demands and service-level-agreements;

*• Novelty of the proposal viz-a-viz current approaches in this field.*

This proposal is timely, and we are inspired by the ubiquity of data streams in today’s emerging applications such as network intrusion detection [25], Infection clusters detection of epidemics, obstacle recognition of autopilot [26], spammer detection in social networks[27], news recommendation [22], etc. For example, the broad deployment of 5G networks provides higher data rates and speeds, which stimulated a tremendous increase in data traffic and brought potential vulnerabilities, threats, and attacks in the exploited system. Therefore, ensure how to apply intrusion detection, spam filters, and fraud detection in the rapidly-arriving data streams generated by network traffic or host processes will become a vital concern. Stream clustering algorithms can help to identify suspicious activities. Those interesting AI-driven applications use stream clustering as the building block. However, to the best of our knowledge, there is still a lack of in-depth comparison of the different algorithms has been performed, and it remains unclear of key weaknesses and strengths of each of the existing stream clustering algorithms.

First, many existing algorithms are designed with implicit assumptions such as their specifically targeted application domains including network intrusion detection, sensor network monitoring and stock market analysis. Subsequently, they implicitly assume different internal data structures, outlier detection mechanisms and many of them are evaluated based on different performance metrics (e.g., varying ways of determining clustering accuracy). As a result, there is no one “standard” implementation and researchers and practitioners are often left alone to decide which algorithm they should adopt in their applications. We observe that there are several design choices that have different trade-offs and performance behaviours, but there is no comprehensive experiment study to evaluate them on modern operating environment leaving an unpleasant literature gap.

Second, we acknowledge that many open-sourced libraries and frameworks of data stream mining have already been proposed, such as MOA[8], SAMOA, StreamDM[16], and FlinkML. However, there are multiple limitations of existing libraries and frameworks. First of all, the current frameworks or packages do not cover comprehensive state-of-the-art stream clustering algorithms, partially because most of the state-of-the-art stream clustering algorithms are not open-sourced. As a result, most prior empirical evaluation studies are biased especially if they rely on existing frameworks with a narrowed subset of algorithms. In addition, their implementations of algorithms are fragmented, ad-hoc, and sometimes even based on different programming languages and compilers. This also leads to unfair comparisons due to the differences caused by programming languages and compilers. Moreover, they tend to use a complex language environment with dependence on others, which are hardly extensible.

Third, although general-purpose SPEs that can provide fast stream computations do exist in both industry and academia (e.g., Apache Spark Streaming and Apache Flink), they are not natively designed nor optimized for stream mining tasks. In reality, data scientists usually prefer to first load the entire datasets into tools like Weka, and then apply mining algorithms. Those tools are generally lacking the capability of stream processing, elastic scaling, fault tolerance mechanisms, as well as the absence of unified API specifications and common data abstraction limits data scientists’ ability to efficiently program and maintain large machine learning pipelines. Additionally, as the pipeline gets longer and more complex, the combined workflow’s performance deteriorates due to expensive data movement across the functions and a lack of cross-operation optimizations.

In summary, to the best of our knowledge, the proposed project is the first comprehensive study in benchmarking and optimizing modern stream clustering algorithms. The ultimate result of the proposed project is a highly customizable, extendable, scalable benchmark framework equipping with state-of-the-art stream clustering algorithms, a comprehensive set of application workloads, and performance evaluation tools.

1. Potential Impact of Proposed Research:

The impact of the proposed research is multi-folds.

**First**, our proposed benchmark includes the state-of-the-art stream clustering algorithms and evaluation mechanisms. It can guide users to apply the suitable algorithm in their applications appropriately including but not limited to customer click streams, telephone records, large sets of Web pages, multimedia data, financial transactions, and observational science data. For example, event detection in social networks aims to detect events that burst into attention at a certain time and space[27]. The events can be represented as clusters of entities, while the entities are clustered according to the similarities of their belonging contexts with a stream clustering algorithm. The outcome of this research is also timely with the popularity of big-data applications and monitoring services such as sensor networks and the Web. We believe that the success of this research project is able to bring economic and societal impact to not just the computing domain but also other business sectors to enable real-time data stream analysis.

**Second**, our proposed benchmark is modularized according to key design components of the prominent stream clustering algorithms. It can offer insight into the merits and limitations of existing stream clustering algorithms and future research can rely on our benchmark to further explore new design options and hence new stream clustering algorithms. The benchmark can also serve as a reference to examine the superiority of future stream clustering approaches and hopefully bring the entire community to the common conscious and move forward.

**Third**, although data streams are often distributed, many stream clustering applications like event detection tend to use centralized solutions, introducing delays in the whole process. Meanwhile, the existing research on stream clustering algorithms does not have a focus on parallelization. Companied with the increasing number of connected devices and the daily scaling up data generated by them with accelerating rates under the deployment of 5G, it is more and more important to provide a continuous performance improvement for existing algorithms both with the help of parallel programming and modern computing architectures. Our modularized framework can offer a thorough performance analysis of existing algorithms varying with different design components and be further utilized to develop distributed versions for parallelizing practice and theoretical analysis.

**Fourth**, driven by the widely deployed 5G and IoT technology, we envision that massive stream mining will soon become a mandatory task for every AI-driven application, where data stream clustering operation will be often used as a preliminary and important building block. 5G can provide higher device density and bandwidth with low latency, which creates new opportunities in computing and increases security risks like attacks, spammers, and frauds. This leads to AI applications playing a vital role in major computing areas involved with data management like network verification, public cloud security and privacy, edge computing and IoT [31]. Data stream clustering can automated identify evolving patterns from highly dynamic data sets and discover unusual data points in real-time. Therefore, it is widely used in many use cases. For instance, apart from the extensive use of intrusion detection in 5G network security, modern connected automobiles can use stream clustering to identify obstacles and corners to improve safety; it can also provide forest type detection based on real-time data collected by drones. Considering the huge amount of data storage and transmission overhead, we simply cannot collect all the data sources before processing. To this end, the proposed research is a necessary and fundamental step before we can proceed to design a more efficient stream data mining algorithms and systems in follow up works.

* Real-life applications

Our work will cover a wide range of novel real-world applications compared with prior works to guide users to apply the suitable algorithm in their applications appropriately. In this part, We will describe several representative applications in our framework in detail, of which corresponding benchmark data will involve different data types.

1. *Network intrusion detection (Network security).* Since the data produced by network devices and end-user terminals go beyond the memory constraints of monitoring equipment, stream clustering is gaining popularity in the network security community; it can help identify unusual events and then generate alerts for the systems. Its applications are not limited to malicious content detection from incoming/outgoing traffic analysis, fully automated clustering from extensive traffic data patterns, threats and loopholes detections, etc. One of the major challenging tasks is network intrusion detection, which identifies specific types of network attacks among the data gathered from various sources on hosts and networks in intrusion detection systems(IDS). We will include actual collected network traffic data in our benchmark suite for network intrusion detection applications.
2. *Simultaneous Localization And Mapping (Connected auto vehicles/ 5G IoT).* Modern autonomous vehicles may collect more than 1TB of data per day. With the help of 5G, lower latency, more bandwidth, and more reliable connectivity will provide more opportunities for more bi-directional communication. The data is potentially generated by and shared with other nearby vehicles in an area, extending the reach of any vehicle’s built-in sensors, allowing them to ‘see’ farther and identify through obstacles and around corners for more excellent safety. The light detection and ranging (LiDAR) technology allows sensing surrounding objects with fine-grained resolution in large areas. Their data, which is generated continuously at very high rates, can provide information to support automated functionality in many SLAM (Simultaneous Localization And Mapping) problems such as obstacle recognition, 3D localization, and indoor scene segmentation. Clustering point clouds is a key problem in extracting this type of information from the disordered point data collected from the LiDAR devices. This proposal also focuses on testing some clustering applications in SLAM problems while designing our benchmark suite.
3. *Infection clusters evolution (COVID-19 epidemics analysis).* Understanding the trajectories and evolutions of COVID-19 infection clusters of cases and death rates helps governments discover better opportunities for effectively responding to the impact of the pandemic as rapid responses to health epidemics are crucial for saving lives. Previous work focuses on spatial-temporal analysis [32] to detect evolving infection clusters of COVID-19 to understand better and mitigate the spread of the disease and recognize the outbreak pattern. However, the algorithms they use are almost offline and lack real-time analysis. As stream clustering algorithms can handle concept drift with massive stream data, we will treat daily generated infection cases data as a workload in our framework to detect the evolution of evolving infection clusters of COVID-19.
4. *Event detection (Social network analysis).* The rapid growth of social networks makes them become the primary medium for discussing what is happening around real-world events. The information that is generated on social platforms like Twitter can produce rich data streams for insights into ongoing topics or conversations around them[28]. We regard these ongoing activities as a data stream, model events a list of clusters of trending entities over time. In this way, we can use stream clustering algorithms to produce an updated set of events that evolves over time. Our framework will also treat event detection in Twitter as a workload.
5. *News recommendation (Communication Media).*
6. Technical Description:
   * Methodology & Approach

We propose to research technical design, implementation, and benchmark study of data stream clustering algorithms. The major thrusts in this proposal are to: 1) design a taxonomy of stream clustering algorithm; 2) A modularity design of stream clustering algorithms; 3) A sophisticate benchmark for stream clustering; and 4) A detailed profiling study of benchmarking stream clustering algorithms. They are grouped into four task packages accordingly. We will describe the details of those thrusts one by one.

**First, a taxonomy of stream clustering algorithm:** We provide a taxonomy of around 20 stream clustering algorithms by summarizing their key design aspects. For each design aspect, we discuss the state-of-the-art variants and reveals any potential improvements. This is critical to identify the entire solution space for stream clustering algorithms rather than proposing incremental improvements over existing work. As a start, we plan to include a list of widely used streaming clustering algorithms into our benchmark. Among them, the source code of *StreamKM++[11]*, *CluStream[3]*, *ClusTree[9]*, *DenStream[7]* and *D-Stream[4]*, *BICO[16]* and *COBWEB[1]*, *DBSTREAM[21]*, *BIRCH[2]*, *PreDeConStream[13]*, and *HDDStream[12]* can be extracted from the existing frameworks including *MOA[8]* and *subspaceMOA[15]*. Some newly proposed but not open-sourced algorithms such as *EDMStream[22]* and *TSF-DBSCAN[29]* will be implemented based on the corresponding papers.

**Second, a modularity design of stream clustering algorithms:** Different from prior works, we implement all existing algorithms from ground up inside the same code-base to eliminate the differences among existing implementations caused by programming languages and compilers. Instead of implementing each algorithm separately, our framework will be modularized according to the a few key design aspects summarized in the following table.

|  |  |
| --- | --- |
| **Design Aspects** | **Design Choices** |
| Data structure for statistical summary | Feature vectors, prototype arrays, coreset trees, and data grids |
| Window model | Landmark model, sliding-window model, and damped model |
| Offline refinement strategy | With (and which) or without an offline clustering algorithm to refine the online clustering results. |
| Outlier detection mechanism | Statistics-based, proximity-based /distance based, density-based, clustering-based methods |
| Summarization methods | Sampling methods, histograms, wavelets, sketches, micro cluster, grid. |
| Concept drift detection mechanism | Error-based, distribution-based, multiple hypothesis test based |
| Concept drift adaption mechanism | Retrain, ensemble, adjust |

**Third, a benchmark for stream clustering:** The benchmark must include a diversity of data and workloads, which is a prerequisite for any meaningful benchmark for evaluating stream clustering algorithms. Here we design our benchmark according to the variety of stream clustering workloads discussed in this proposal and cover different types of stream data.

1. **Network Security-Monitoring [25]**. We include the [MAWILab](mailto:http:/www.fukuda-lab.org/mawilab/documentation.html" \l "what), a well-known dataset for network monitoring application in our benchmark. MAWILab is a public collection of 15-minute network traffic traces captured daily on a backbone link between Japan and the US since 2001. Building on this repository, the MAWILab project uses a combination of four traditional anomaly detectors (PCA, KL, Hough, and Gamma) to label the collected traffic partially, and we could use these labels to cluster the traces.
2. **Social Network Services(SNS) Analysis[27].** Social network services are an online platform that people use to build social networks or social relationships with other people who share similar personal or career interests, activities, backgrounds, or real-life connections. Here we use [Stream Tweets Data](mailto:https:/developer.twitter.com/en/docs/tutorials/stream-tweets-in-real-time), offered by Twitter, one of the most famous online platforms providing microblogging and social networking service on which users post and interact with messages known as "tweets"
3. **Mass Communication Media[22].** Here, we use [News Aggregator Data Set](mailto:http:/archive.ics.uci.edu/ml/datasets/News+Aggregator), the famous dataset for news categoryclustering in mass communication media. This dataset contains headlines, URLs, and categories for 422,937 news stories collected by a web aggregator between March 10th, 2014 and August 10th, 2014. News categories included in this dataset include business, science and technology; entertainment; and health. Different news articles that refer to the same news item (e.g., several articles about recently released employment statistics) are also categorized together.
4. **Electronic Health Records.** To stay on top of the current hottest topic, we will use [Coronavirus Source Data](mailto:https:/github.com/owid/covid-19-data/tree/master/public/data) in the field of EHR. This data set is a collection of the COVID-19 data updated daily and includes data on confirmed cases, deaths, and testing from Johns Hopkins University (JHU).
5. **SLAM(Simultaneous Localization And Mapping).** We utilize [Ford Campus Vision and Lidar Data set](mailto:http:/robots.engin.umich.edu/SoftwareData/Ford) for testing the proposed algorithms on three-dimensional data clustering tasks in the field of SLAM. This data set is collected by an autonomous ground vehicle testbed with a Velodyne HDL-64E LiDAR scanner [18]. The vehicle path trajectory in this dataset contains several large and small objects (e.g. buildings, vehicles, pedestrians, vegetation, etc.), which is beneficial for testing the clustering performance of the proposed algorithms.

**Forth, a detailed profiling study of benchmarking stream clustering algorithms:** We conduct a detailed profiling study on the different design aspects of stream clustering algorithms. The findings and optimizations can be applicable to future studies in terms of new algorithms and/or new hardware architectures. We systematically examine the profiling results of stream clustering algorithms running the benchmark, with careful considerations on various aspects. As a start, we determine the performance metrics for stream clustering applications based on four aspects: effectiveness, efficiency, adaptability, and scalability. i) *Cluster quality* is an essential performance metric for all clustering algorithms to detect whether the cluster is effective. We use both internal and external measures for cluster quality. Specifically, internal measures consider the inner structure and properties of the clusters, such as the compactness of clusters or the distance between them. External measures are external validity criteria that compare the clusters against a ground truth. In these measures, the existing true groups of input data are compared with the data partition obtained by algorithms. Data stream clustering must process data objects quickly and incrementally within a limited time and memory to provide timely results and rapidly detect outliers with corresponding actions; ii) To measure the efficiency of algorithms, *throughput* (defined by the number of input data objects processed by algorithm per unit of time) is chosen as another performance metric; iii) As one of the requirements for stream clustering is the adaptability for handling with concept drift of data streams and data distribution changes. We *use Clusters and outlier evolution tracking* as a metric in our evaluation to represent algorithms' adaptability; iv) Lastly, we measure the scalability to data dimensions and continuously arriving objects by response time as our metric varying different dimensions and data stream arriving rates to represent respectively. Optimizing algorithms for requirements in every aspect is not possible. Performance metrics can be conflicting with each other sometimes. Thus, existing stream clustering algorithms are designed to optimize only a few performance metrics and sometimes even sacrifice their performance in other metrics. Besides those algorithmic aspects, we also compare the algorithms in terms of hardware utilizations such as CPU and memory bandwidth utilizations. We also take specific service-level-agreement requirements from real-world applications into consideration while comparing different stream clustering algorithms.

Challenges, risks & proposed mitigation plans

There are mainly three key challenges raised from the designs and implementations of our chosen algorithms, performance metrics and benchmark suite.

First, our research aims to include the most comprehensive state-of-the-art algorithms. Ther are two obstacles to achieve the goal: 1) enormous stream clustering algorithms have been proposed and can be applied in our benchmark; 2) many newly proposed stream clustering algorithms are not open sourced. We propose to first summarize design aspects of different stream clustering algorithms into a clear taxonomy based on the existing surveys and corresponding research papers. Then, we only need to implement a subset of representative algorithms that can cover different design choices in our benchmark.

Second, it is challenging to compare the results of clustering as it is commonly considered as a subjective task. Unless with ground truth, prior works are often biased towards a subset of evaluation criteria to evaluate the structure and properties of resulting clusters, e.g., the sum of squared errors (SSE) (evaluating compactness of clusters), sum squared distances SSQ, silhouette coefficient, etc. Meanwhile, validating partitions generated with non-stationary data evolved with time needs to consider the temporal criterion to evaluate the quality of the partition and their behaviour over time. To give a comprehensive comparison, we propose to implement all those evaluation metrics inside the same benchmark, and include workloads with ground truth, e.g., purity, f-measure, homogeneity, etc. In order to further capture the temporal criterion, we propose to generate the performance metric as a function of time.

Find some recent works on the similar topics and list down their name, organization, country.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Main Author names | Organization | Country | Research title | Main contributions |
| 1 | Jonathan de Andrade Silva/  (JO˜AO GAMA) | University of S˜ao Paulo  /(University of Porto) | Brazil/  (Portugal | Data stream clustering: A survey[14] |  |
| 2 | Albert Bifet | University of Waikato | New Zealand | MOA: Massive Online Analysis, a Framework for Stream Classification and Clustering[10] | a java-based unified stream data mining framework encompassing a multitude of clustering algorithms, including StreamKM++[11], CluStream[3], ClusTree[9], DenStream[7] and D-Stream[4] , BICO [16], and COBWEB[1] |
| 3 | Seidl, Thomas/  Marwan Hassani | RWTH Aachen University | Germany | Subspace MOA: Subspace Stream Clustering Evaluation Using the MOA Framework. | a framework built on the MOA focusing on clustering high-dimensional stream data by implementing HDDStream[12] and PreDeConStream[13] with a provided R-package[18]. |
| 4 | Matthias Carnein | University of Münster | Germany | An Empirical Comparison of Stream Clustering Algorithms.[23] | evaluated ten stream clustering algorithms among existing stream data mining frameworks (stream package[19],streamMOA[17]) |
| 5 | Matthias Carnein | University of Münster | Germany | Optimizing Data Stream Representation: An Extensive Survey on Stream Clustering Algorithms[28] |  |
|  |  |  |  |  |  |

@Wangxin, Zhenyu

Reference

1. D. H. Fisher. Knowledge acquisition via incremental conceptual clustering. Machine Learning, 2(2):139–172, 1987.
2. T. Zhang, R. Ramakrishnan, and M. Livny. Birch: An efficient data clus-tering method for very large databases. InProceedings of the 1996 ACM SIGMOD International Conference on Management of Data, SIGMOD ’96, page 103–114, New York, NY, USA, 1996. Association for Computing Machinery.
3. C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu. A framework for clustering evolving data streams. InProceedings of the 29th International Conference on Very Large Data Bases - Volume 29, VLDB ’03, page 81–92. VLDB Endowment, 2003.
4. Y. Chen and L. Tu. Density-based clustering for real-time stream data. In Proceedings of the 13th ACM SIGKDD International Conference on Knowl-edge Discovery and Data Mining, KDD ’07, page 133–142, New York, NY,USA, 2007. Association for Computing Machinery.
5. M. Kontaki, A. Papadopoulos, and Y. Manolopoulos. Continuous trend-based clustering in data streams. In Data Warehousing and Knowledge Discovery, 2008.
6. M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani. A detailed analysisof the kdd cup 99 data set. In 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, pages 1–6, 2009.
7. L. Wan, W. K. Ng, X. H. Dang, P. S. Yu, and K. Zhang. Density-based clustering of data streams at multiple resolutions.ACM Trans. Knowl.Discov. Data, 3(3), July 2009.
8. A. Bifet, G. Holmes, R. Kirkby, and B. Pfahringer. Moa: Massive onlineanalysis. Journal of Machine Learning Research, 11:1601–1604, 2010
9. P. Kranen, I. Assent, C. Baldauf, and T. Seidl. The clustree: indexing micro-clusters for anytime stream mining.Knowledge and Information Systems, 29(2):249–272, 2011.
10. A. da Silva, R. Chiky, and G. H ́ebrail. A clustering approach for sampling data streams in sensor networks. Knowledge and Information Systems,32:1–23, 2012.
11. M. R. Ackermann, M. M ̈artens, C. Raupach, K. Swierkot, C. Lammersen,and C. Sohler. Streamkm++: A clustering algorithm for data streams. ACM J. Exp. Algorithmics, 17, May 2012.
12. Ntoutsi, Irene & Zimek, Arthur & Palpanas, Themis & Kröger, Peer & Kriegel, Hans-Peter. Density-based Projected Clustering over High Dimensional Data Streams. Proceedings of the 12th SIAM International Conference on Data Mining, SDM 2012.
13. Hassani M., Spaus P., Gaber M.M., Seidl T. Density-Based Projected Clustering of Data Streams. In: Hüllermeier E., Link S., Fober T., Seeger B. (eds) Scalable Uncertainty Management. SUM 2012. Lecture Notes in Computer Science, vol 7520. Springer, Berlin, Heidelberg.
14. J. A. Silva, E. R. Faria, R. C. Barros, E. R. Hruschka, A. C. P. L. F. d. Carvalho, and J. a. Gama. Data stream clustering: A survey. ACM Computer Survey, 46(1), 2013.
15. H. M., K. Y., and S. T. Subspace moa: Subspace stream clustering evaluation using the moa framework.Database Systems for Advanced Applications, 2013.
16. H. Fichtenberger, M. Gill ́e, M. Schmidt, C. Schwiegelshohn, and C. Sohler. Bico: Birch meets coresets for k-means clustering. In Algorithms – ESA 2013, pages 481–492, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.
17. Hahsler M, Bolanos M, Forrest J (2015) streamMOA: interface for MOA stream clustering algorithms. <https://cran.r-project.org/web/packages/streamMOA/> . Accessed 16 Jun 2021
18. Huawei Noah’s Ark Lab (2015). streamDM. <http://huawei-noah.github.io/streamDM/> . Accessed 16 Jun 2021
19. Hahsler M, Bolanos M, Forrest J, Carnein M, Assenmacher D (2018) stream: infrastructure for data stream mining. <https://cran.r-project.org/web/packages/stream/> .Accessed 16 Jun 2021
20. Hassani M, Hansen M, Kim Y, Seidl T (2016) subspaceMOA: interface to ’subspaceMOA’. <https://cran.r-project.org/web/packages/subspaceMOA/> . Accessed 16 Jun 2021
21. Morales GDF, Bifet A. SAMOA: scalable advanced massive online analysis. J Mach Learn Res. 2015;16:149–53.
22. M. Hahsler and M. Bola ̃nos. Clustering data streams based on shared density between micro-clusters. IEEE Transactions on Knowledge and Data Engineering, 28(6):1449–1461, 2016.
23. S. Gong, Y. Zhang, and G. Yu. Clustering stream data by exploring the evolution of density mountain. Proc. VLDB Endow., 11(4):393–405, Dec.2017.
24. M. Carnein, D. Assenmacher, and H. Trautmann. An empirical comparison of stream clustering algorithms. In Proceedings of the Computing Frontiers Conference.(CF’ 17), 17:361–366, 2017.
25. M. Carnein and H. Trautmann. evostream – evolutionary stream clustering utilizing idle times. Big Data Research, 14:101–111, 2018.
26. Pavol Mulinka and Pedro Casas. 2018. Stream-based Machine Learning for Network Security and Anomaly Detection. In Proceedings of the 2018 Workshop on Big Data Analytics and Machine Learning for Data Communication Networks (Big-DAMA '18). Association for Computing Machinery, New York, NY, USA, 1–7.
27. H. Najdataei, Y. Nikolakopoulos, V. Gulisano and M. Papatriantafilou, "Continuous and Parallel LiDAR Point-Cloud Clustering," 2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS), 2018, pp. 671-684
28. Mateusz Fedoryszak, Brent Frederick, Vijay Rajaram, and Changtao Zhong. 2019. Real-time Event Detection on Social Data Streams. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '19). Association for Computing Machinery, New York, NY, USA, 2774–2782.
29. Carnein, M., Trautmann, H. Optimizing Data Stream Representation: An Extensive Survey on Stream Clustering Algorithms. Bus Inf Syst Eng 61, 277–297 (2019).
30. A. Bechini, F. Marcelloni, and A. Renda. Tsf-dbscan: a novel fuzzy density-based approach for clustering unbounded data streams.IEEE Transactionson Fuzzy Systems, 2020.
31. Loghin, Dumitrel & Ta, Quang-Trung & Wang, Wei & Xiao, Xiaokui & Yang, Yang & Zhang, Meihui & Zhang, Zhonghua & Cai, Shaofeng & Chen, Gang & Dinh, Anh & Fan, Feiyi & Lin, Qian & Ng, Janice & Ooi, Beng & Sun, Xutao. (2020). The Disruptions of 5G on Data-Driven Technologies and Applications. IEEE Transactions on Knowledge and Data Engineering.
32. Adam, D.C., Wu, P., Wong, J.Y. et al. Clustering and superspreading potential of SARS-CoV-2 infections in Hong Kong. Nat Med 26, 1714–1719 (2020).