Enhancing Data Streaming Clustering Algorithms for AutoML in Cloud Environments: A Novel Design Approach

Madhuri H. Parekh
Department of Computer Engineering, Marwadi University, Rajkot-Gujarat, India
madhuri.parekh109023@marwadiuniversity.ac (corresponding author)

Madhu Shambhu Shukla
Department of Computer Engineering-AI & Big Data, Marwadi University, Rajkot-Gujarat, India
madhu.shukla@marwadieducation.edu.in

Received: 23 August 2024 | Revised: 9 November 2024 | Accepted: 20 November 2024

 DOI: https://doi.org/10.48084/etasr.8806

ABSTRACT

The objective of this revision is to enhance existing AutoCloud clustering technology, which demonstrates optimal performance when dealing with clusters of specific dimensions and arrangements. AutoCloud uses the TEDA framework to break down the clustering challenge into two smaller problems, called micro cluster and macro cluster. AutoCloud is an innovative method that eliminates the requirement for any pre-existing understanding of datasets, where clusters can develop and combine when new information and explanations are presented. This study proposes an experimental configuration to generate microclusters and data clouds without imposing a certain topology on static datasets. MLAutoCloud uses a modified distance-based technique, utilizing the big data framework and incorporating the adjusted random index value with the TEDA framework for streaming data. The MLAutoCloud technique yielded optimal cluster numbers and achieved excellent data collection results, as seen in the test results on different datasets. Estimating thickness despite changes in the underlying assumptions is a process that could modify the variables used to provide data. The MLAutoCloud method is an effective way to generate a cloud clustering algorithm in the data streaming section.

Keywords-AutoCloud; clustering; data streams; big data; machine learning; spark; eccentricity; outliers; anomaly

I. INTRODUCTION

Clustering is a vital data mining technique employed in information partitioning and attribute engineering, which are extensively employed in circular foundation-purpose neural networks [1-3]. The AutoCloud method needs to be entitled to handle issues such as concept drift and evolution that arise during data stream processing [4]. In the AutoCloud system, the study of anomalies, outliers, or instances that are part of a cluster is of significant importance and is currently a popular trend in data mining [5]. Therefore, when faced with the task of clustering dynamic data stream system information, it is important to approach it systematically [6]. One way to do this is by gradually adding a unique node at an interval to the cluster, organizing the group, and storing the necessary information. Currently, online collection has become a tool used in several applications, including information innovation, development error detection, inconsistency detection, and recommendation systems [4, 5, 7]. An innovative evolving clustering approach founded on the involved knowledge can be used to continuously inform the instruction basis of the arrangement, which is a developing Takagi-Sugeno (eTS) useful imprecise prototype [8]. Additionally, based on the distance technique, consumers with a distance measure often anticipate the cluster emerging from a comparable outline [9].

This study identifies the data stream's sequential logic as an AutoCloud, denoting an autonomous operation in the cluster without a predetermined value. A logical, systematic method called TEDA (Typicality and Eccentricity Data Analytic) is used to find variance that is incrementally parametric supply based on the proximity linking model [9]. The goal of AutoCloud is to implement a technique that is both efficient and resourceful for constructing the new cluster and merging it with historical data based on the expansion of data across instances [6]. A precise n-dimensional information example's conformity with the principles of its previous reading is connected to the feeling of typicality. Eccentricities refer to the degree of dissimilarity between data samples and their allocation. This suggests that significant eccentricities in the data sample are typically considered anomalous [10]. TEDA was developed focusing on the concept of increasing communication. Consider the information contribution

$x_k \in Rn$ in the stream. This contribution entails classifying a sample of n dimensional data x_1, x_2,..., x_k which is a subset of X, where k stands as the specific interval at which the model was acquired. The cumulative closeness $\pi(.)$ of x_k is estimated by approximating the value of each existing data sample [5]. The aggregate of the gaps between a specific example x and every of the k essentials in the dataset X is represented by $\pi k(x)$.

The interval between two points is the range of data samples A and B and k represents the specific moment in time when data point x is taken as a sample.

$$\pi k(x) = \sum_{i=1}^{k}(dk, di) \quad (1)$$

Initiating the calculation of $\pi k(x)$ and determining the eccentricity $\xi k(x)$ serves as a means to differentiate the data sample x_k based on its value in relation to all other existing data points up until time k.

$$\xi k(x) = \frac{2\pi k(x)}{\sum_1^k \pi k(xi)}, \sum_{i=1}^{k} \pi k(xi) > 0, \ \ k > 2 \quad (2)$$

The oddity can be continuously provided for the Euclidean distance store value as follows:

$$\xi(x_k) = \frac{1}{K} + \frac{(\mu_k - x_k)^{\tau}(\mu_k - x_k)}{k\sigma_k^2} \quad (3)$$

In this case, $\xi(X_K)$ represents the oddity of the example x_k with respect to each previous sample in the sets of statistics [9].

$$\mu_K = \frac{k-1}{k}\mu_{K-1} + \frac{X_k}{k}, \ k \geq 1, \ \mu_1 = x_1 \quad (4)$$

$$\sigma_k^2 = \frac{k-1}{k}\sigma_{k-1}^2 + \frac{1}{k-1}||x_k - \mu_k||^2, \sigma_1^2 = 0 \quad (5)$$

According to eccentricity, to find the typicality, $\tau(x_k)$ is equal to twice $\xi(x_k)$. This denotes the value difference between a random data point x_k and every other data point up until the time stamp at k [9].

$$\tau(x_k) = 1 - \xi(x_k), \ k \geq 2 \quad (6)$$

The ideas of the TEDA framework involve a sample point A that is farther from or on top of the statistics set than sample point B. This means that sample point A is the most unusual and less typical compared to B [5]. The standardized oddity $\xi(x_k)$ and typicality $\tau(x_k)$ can be evaluated as follows:

$$\xi(x_k) = \frac{\xi(x_k)}{2} \quad (7)$$

$$\tau(x_k) = \frac{\tau(x_k)}{k-2} \quad (8)$$

The Chebyshev inequality can recycle to find an endpoint equal built proceeding the outlier records and the standardized deviation (x_k). If the data sample (x_k) is different from the mean, it must be called an odd sample. An outlier is a sample (x_k) that is not close to the mean. The equation for an outlier describes this. The m number tells how far the records are from the mean. Many times, the meaning of m is diverse after the meaning of 3.

$$\xi(x_k) > \frac{m^2+1}{2k}, \ m > 0 \quad (9)$$

Here, m corresponds to the numeral of standard deviation. Suppose a hypothetical situation that assumes the value of m as 3. The saturation level for organizing records is (x_k) as an anomaly is $\xi(x_k) > \frac{5}{k}$. This is called a data stream [10], where data comes in very quickly. The primary phase is mining the data stream cluster. This cluster has a pattern that can be used to separate the data chunks into clusters for review to ensure that all data points are the same [11-13]. As data streams, there are always more and better tools for finding patterns [14]. A review of the literature shows that there are grouping methods for constant datasets. However, they cannot be used on streaming data because the amount and speed of data coming in are not always the same and change over time [15-19]. In the context of this irregularity, a significant data streaming grouping procedure aims to complete the minimum amount of work with minimal latency and to identify the fundamental and growing structure of a record by an undetermined amount of gatherings [20, 21]. All this can be accomplished by properly splitting and reuniting clusters [22]. Furthermore, many studies have crucial information on generating a randomly produced collection that occurs in a wide range of data stream submissions, including geography, satellites, empirical, and sensor readings [23, 24]. Instead of finding actual outliers, the stream optics algorithm is subjected to the pruning idea, which reduces memory usage and speeds up the process of discovering possible clusters [25]. Big data analysis makes it possible to investigate the connections between a collection of separate data, exposing numerous facts that help to predict the best course of action and facilitate the achievement of the intended objectives [26].

II. WORKING ON THE AUTOCLOUD

In AutoCloud, Euclidean distance is used to figure out how close samples and clouds are to each other. For a data sample x that arrived at time k, it can determine the mean and average variance using eccentricity and typicality. A sample x of the provided data contributes to or generates additional clouds. Microclustering occurs when these steps are repeated over and over again. The given process satisfies the idea of drift problems [8]. In this way, AutoCloud can set up a growing collection for streaming data online. Microclustering is completed in a very short amount of time. As this study aims to focus on an entirely online account, two-phase algorithms are hard to judge with the given move [27].

A. Advantages of AutoCloud Design

Developing excellent information clouds for the ideas of drift and changing data is helped by the fact that existing nodes can be combined or a new cluster can be made with changed parameters. The second benefit shows that it can run batch processing jobs or run without storing past data samples in memory, so it can be used in real-time applications that do recursive calculations. The third benefit that proves this is that it is totally unsupervised and does not require previous knowledge, so it can be used in most real-time systems [28].

B. Weaknesses of AutoCloud Algorithm

The first issue with AutoCloud is that there is no support in terms of analysis data in which there is no temporal

dependence between data samples. There are no exact outlines or shapes in the data clouds, and they are represented graphically by an ellipse. The usage of Euclidean distance, which requires that data clusters be represented as ellipses, is a second issue. In addition, AutoCloud does not consider a specific approach for an adaptive value of m, affecting its performance [27]. The flowchart of the AutoCloud algorithm can be seen in [27].

III. METHOD AND EXPERIMENTS ON THE AUTOCLOUD ALGORITHM

This approach takes data samples and uses Euclidean distance to compare their distance to clouds. For each data sample recorded, a choice must be made about which of the real clouds it belongs to. Because of this situation, any real clouds that are affected change to show that this new sample is added. Otherwise, if the sample differs significantly from every other cloud, a new cloud is created. Finally, each pair of clouds is examined to see if they need to be merged. The program creates the first cloud, which is then filled out with the first two samples. The third sample is where the process of creating new data clouds begins. This is because the space between the points is based on the quality levels. It is impossible to tell if two places are close or far apart if the range is not known ahead of time, which is the case here. This is the reason for requiring a third point before the search can figure out how important the space between any two points is. This process is explained in algorithmic steps [28]. The proposed method aims to fix the problems with data stream groups. Static datasets were used as example data values. Also, some automatically generated data were added to examine how AutoCloud works. The data cluster used datasets from [28], containing both real and synthetic datasets that are frequently used in data clustering. Each sample is a member of a group and falls in the midpoint of each group in certain circumstances where information is known.

IV. EXPERIMENTS WITH FIXED DATASETS IN AUTO CLOUD

Before adjusting for concept drift, an information stream cluster model was examined on its ability to cluster immobile structures [29]. The AutoCloud experiment selected two preset data samples for clustering. In [27], the permanent dataset1 was presented to examine the Chameleon cluster model. FD-D1 is the first fixed dataset, and FD-D2 is the second fixed dataset, including 3031 data samples arranged into 9 clusters and 2551 data samples arranged into 8 clusters, respectively. The datasets came from [30]. Some of the authentic and actual data sets in this collection are commonly utilized for data clustering tasks.

TABLE I. PRACTICAL SETTING IN THE DATASETS

Dataset	Fixed dataset 1	Fixed dataset 2
Samples	3031	2551
Clusters	9	8

Each analysis used a random sampling of fixed datasets, which were all assessed without the use of window partitions. The following chart displays the results. In practical evaluation, the operation was achieved 30 times for each sequence of methods and each dataset.

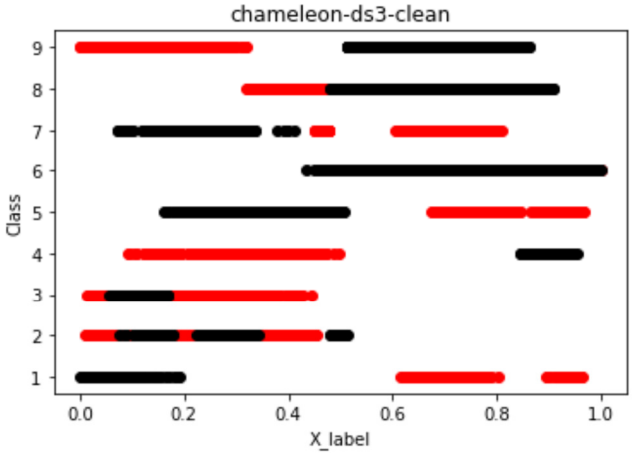

Fig. 1. Fixed dataset 1 output.

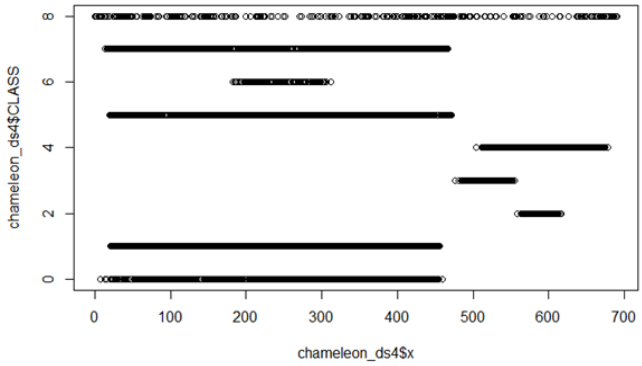

Fig. 2. Fixed dataset 2 output.

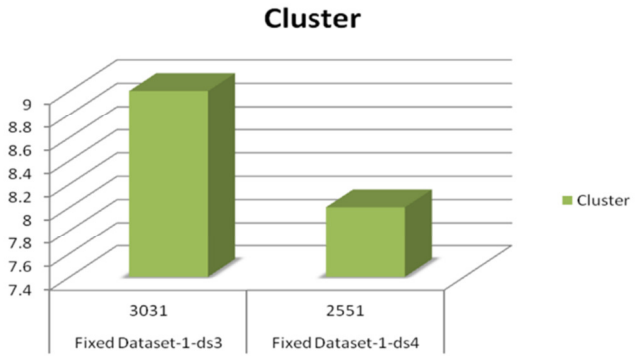

Fig. 3. Results on fixed datasets.

V. PROPOSED METHOD FOR MLAUTOCLOUD

The first step in MLAutoCloud, which uses both TEDA and Spark architecture, is to join together as a group to progress toward coarse data structures, identify data clouds, and identify restrictions such as conservative data in the group that is tied to a certain cluster. Here, using the TEDA and Spark frameworks, the dataset is combined and partitioned based on the updated distance. MLAutoCloud uses microclusters on actual datasets to calculate the precise number of clusters using the Mahalanobis distance. The flowchart of the MLAutoCloud algorithm can be seen in [29].

VI. MLAUTOCLOUD EXPERIMENTS ON REAL DATASETS

Data streaming algorithms such as DenStream, CluStream, and StreamKM are associated with the MLAutoCloud algorithm. An adjusted rand measure was employed as the loss function, measuring the degree of similarity between two data clusterings. The modified random index value's score was 1.0 when there was an equal separation. The index score of the modified random index value was 0.0 when an arbitrary node was added individually to the cluster and data samples at that point. An apparent assessment of the predicted and observed labels may be obtained in this case using the ARI loss function. Every detected cluster is given a unique name. When a is used in this scenario, the clusters in the X or M subset and the Y or N subset add up to the identical pair of items inside S. In this case, b refers to the distinct group in the Y subset and the distinct cluster in the X subset that vary from S in several fundamental pairings.

TABLE II. EVENT TABLE

	N1	**N2**	**Ns**	**Sums**
M1	Q11	Q12	Q1s	a1
M2	Q21	Q22	Q2s	a2
Ms	Qr1	Qr2	Qrs	ar
Sums	b1	b2	Br	

The rand index value is calculated by:

$$R = \frac{(a+b)}{\frac{n}{2}} \quad (9)$$

$$ARI = \frac{\Sigma_{ij}\binom{nij}{2} - \frac{[\Sigma_i\binom{ai}{2}\Sigma_j\binom{bi}{2}]}{\binom{n}{2}}}{\frac{1}{2}[\Sigma_i\binom{ai}{2}\Sigma_j\binom{bi}{2}] - \frac{[\Sigma_i\binom{ai}{2}\Sigma_j\binom{bi}{2}]}{\binom{n}{2}}} \quad (10)$$

Most hyperparameters from each process can be utilized by a grid investigation to determine the precise number of cluster datasets. The modification parameter was applied to the samples that the first training in the event table (Table II) recommended. The cluster graphs of the actual and fixed datasets are shown in Figures 5 and 6. Table III shows the accuracy table for both actual and fixed datasets.

TABLE III. PRACTICAL SETTING IN SUPPORT OF THE INFORMATION SETS

Data Set	**Fixed dataset 1**	**Fixed dataset 2**	**Real dataset**
Example	3031	2551	2219803
Group	9	8	54

TABLE IV. EXACTNESS DIFFERENT STREAM ALGORITHM TABLE

Dataset	**DenStrem**	**CluStream**	**StreamKM**	**MLAutoCloud**
Fixed dataset 1	0.21	0.55	0.62	0.38
Fixed dataset 2	0.22	0.36	0.37	0.41
Real dataset	-	-	-	0.97

The experimental results indicate that cluster information flow from arbitrary shapes is more comfortable with incremental drift with low magnitude on the AutoCloud step-by-step logic.

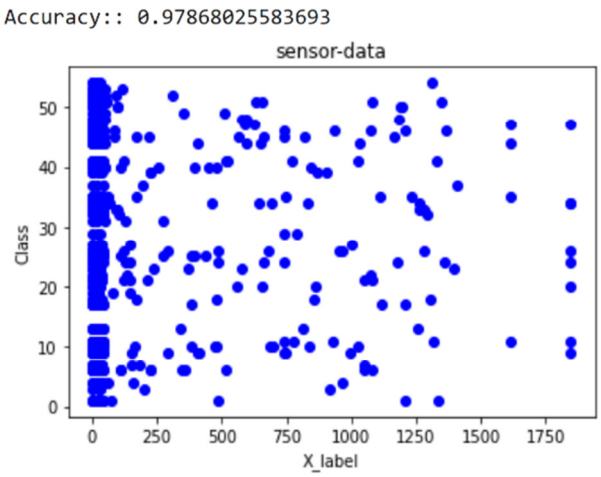

Fig. 4. Sensor real datasets.

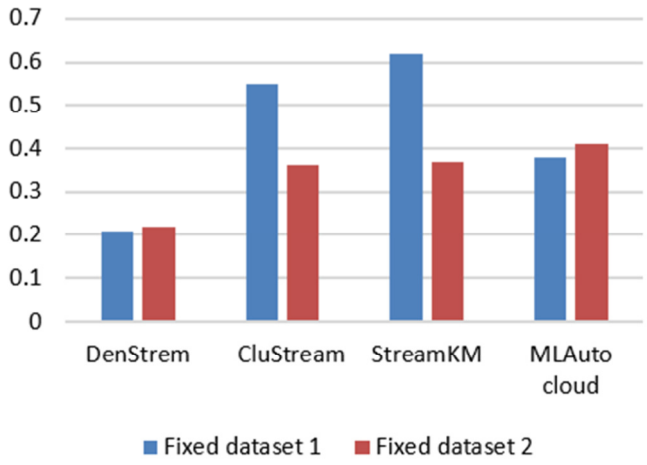

Fig. 5. Accuracy graph with fixed datasets.

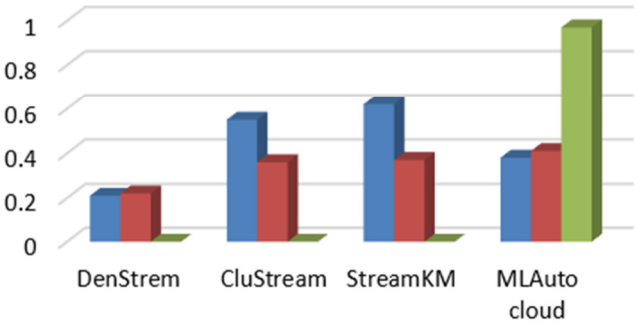

Fig. 6. Accuracy graph with fixed and actual datasets.

These results show that since outliers are hidden in the data stream, anomalies and noise that are not needed must be efficiently identified by deploying clustering techniques [25, 26, 29, 31].

VII. CONCLUSION

The current AutoCloud method lacks precise edges or precise forms. Since AutoCloud uses Euclidean distance, data clusters must be represented as ellipses. To address the limitations of AutoCloud, machine learning is used to determine the exact number of clusters with specified forms or outlines. In particular, MLAutoCloud employs microclusters on real datasets. Any form of data cluster is formed using the Mahalanobis distance. AutoCloud created nine clusters using fixed dataset 1 with 3031 data samples and eight clusters using fixed dataset 2 with 2551 data samples. Different stream methods were used for both fixed datasets, with accuracies of 0.21, 0.55, and 0.62 and 0.22, 0.36, and 0.37 for DenStream, CluStream, and StreamKM, respectively. The accuracy of MLAutoCloud on fixed datasets 1 and 2 was 0.38 and 0.41, respectively. 54 clusters were created with an accuracy of 0.97 using a real dataset of 2219803 samples. The MLAutoCloud method employs a wide range of uninformed cluster figures and precisely determines the right amount of clusters. Future studies should investigate an algorithm using the PySpark architecture to effectively remove anomalies and enable effective streaming in the direction of the cluster, achieving better results for multiple clusters. The main focus should be on addressing the difficulties brought forth by the enormous volume of data.

REFERENCES

[1] Q. Song, J. Ni, and G. Wang, "A Fast Clustering-Based Feature Subset Selection Algorithm for High-Dimensional Data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 1, pp. 1–14, Jan. 2013, https://doi.org/10.1109/TKDE.2011.181.

[2] I. Czarnowski and P. Jędrzejowicz, "Kernel-Based Fuzzy C-Means Clustering Algorithm for RBF Network Initialization," in *Intelligent Decision Technologies 2016*, 2016, pp. 337–347, https://doi.org/10.1007/978-3-319-39630-9_28.

[3] J. S. R. Jang, C. T. Sun, and E. Mizutani, "Neuro-Fuzzy and Soft Computing-A Computational Approach to Learning and Machine Intelligence [Book Review]," *IEEE Transactions on Automatic Control*, vol. 42, no. 10, pp. 1482–1484, Oct. 1997, https://doi.org/10.1109/TAC.1997.633847.

[4] C. G. Bezerra, B. S. J. Costa, L. A. Guedes, and P. P. Angelov, "An evolving approach to data streams clustering based on typicality and eccentricity data analytics," *Information Sciences*, vol. 518, pp. 13–28, May 2020, https://doi.org/10.1016/j.ins.2019.12.022.

[5] J. Maia *et al.*, "Evolving clustering algorithm based on mixture of typicalities for stream data mining," *Future Generation Computer Systems*, vol. 106, pp. 672–684, May 2020, https://doi.org/10.1016/j.future.2020.01.017.

[6] B. S. J. Costa, P. P. Angelov, and L. A. Guedes, "Fully unsupervised fault detection and identification based on recursive density estimation and self-evolving cloud-based classifier," *Neurocomputing*, vol. 150, pp. 289–303, Feb. 2015, https://doi.org/10.1016/j.neucom.2014.05.086.

[7] J. Gama, P. P. Rodrigues, E. Spinosa, and A. Carvalho, "Knowledge Discovery from Data Streams," in *Web Intelligence and Security*, IOS Press, 2010, pp. 125–138.

[8] A. Lemos, W. Caminhas, and F. Gomide, "Multivariable Gaussian Evolving Fuzzy Modeling System," *IEEE Transactions on Fuzzy Systems*, vol. 19, no. 1, pp. 91–104, Feb. 2011, https://doi.org/10.1109/TFUZZ.2010.2087381.

[9] P. Angelov, "Anomaly detection based on eccentricity analysis," in *2014 IEEE Symposium on Evolving and Autonomous Learning Systems (EALS)*, Orlando, FL, USA, Dec. 2014, pp. 1–8, https://doi.org/10.1109/EALS.2014.7009497.

[10] N. Su, J. Liu, C. Yan, T. Liu, and X. An, "An arbitrary shape clustering algorithm over variable density data streams," *Journal of Algorithms & Computational Technology*, vol. 11, no. 1, pp. 93–99, Mar. 2017, https://doi.org/10.1177/1748301816670163.

[11] J. Jacques and C. Preda, "Functional data clustering: a survey," *Advances in Data Analysis and Classification*, vol. 8, no. 3, pp. 231–255, Sep. 2014, https://doi.org/10.1007/s11634-013-0158-y.

[12] A. Lemos, W. Caminhas, and F. Gomide, "Multivariable Gaussian Evolving Fuzzy Modeling System," *IEEE Transactions on Fuzzy Systems*, vol. 19, no. 1, pp. 91–104, Oct. 2011, https://doi.org/10.1109/TFUZZ.2010.2087381.

[13] H. L. Nguyen, Y. K. Woon, and W. K. Ng, "A survey on data stream clustering and classification," *Knowledge and Information Systems*, vol. 45, no. 3, pp. 535–569, Dec. 2015, https://doi.org/10.1007/s10115-014-0808-1.

[14] D. Puschmann, P. Barnaghi, and R. Tafazolli, "Adaptive Clustering for Dynamic IoT Data Streams," *IEEE Internet of Things Journal*, vol. 4, no. 1, pp. 64–74, Oct. 2017, https://doi.org/10.1109/JIOT.2016.2618909.

[15] M. Chenaghlou, M. Moshtaghi, C. Leckie, and M. Salehi, "Online Clustering for Evolving Data Streams with Online Anomaly Detection," in *Advances in Knowledge Discovery and Data Mining*, Melbourne, Australia, 2018, pp. 508–521, https://doi.org/10.1007/978-3-319-93037-4_40.

[16] S. Mansalis, E. Ntoutsi, N. Pelekis, and Y. Theodoridis, "An evaluation of data stream clustering algorithms," *Statistical Analysis and Data Mining: The ASA Data Science Journal*, vol. 11, no. 4, pp. 167–187, 2018, https://doi.org/10.1002/sam.11380.

[17] M. Sayed-Mouchaweh and E. Lughofer, *Learning in Non-Stationary Environments: Methods and Applications*. Springer Science & Business Media, 2012.

[18] J. A. Silva, E. R. Faria, R. C. Barros, E. R. Hruschka, A. C. P. L. F. de Carvalho, and J. Gama, "Data stream clustering: A survey," *ACM Computing Surveys*, vol. 46, no. 1, Apr. 2013, https://doi.org/10.1145/2522968.2522981.

[19] A. Amini, T. Y. Wah, and H. Saboohi, "On Density-Based Data Streams Clustering Algorithms: A Survey," *Journal of Computer Science and Technology*, vol. 29, no. 1, pp. 116–141, Jan. 2014, https://doi.org/10.1007/s11390-014-1416-y.

[20] J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques*, 3rd ed. Morgan Kaufmann, 2011.

[21] E. Lughofer and M. Sayed-Mouchaweh, "Autonomous data stream clustering implementing split-and-merge concepts – Towards a plug-and-play approach," *Information Sciences*, vol. 304, pp. 54–79, May 2015, https://doi.org/10.1016/j.ins.2015.01.010.

[22] S. Ding, F. Wu, J. Qian, H. Jia, and F. Jin, "Research on data stream clustering algorithms," *Artificial Intelligence Review*, vol. 43, no. 4, pp. 593–600, Apr. 2015, https://doi.org/10.1007/s10462-013-9398-7.

[23] K. Partington and J. A. Cardille, "Uncovering Dominant Land-Cover Patterns of Quebec: Representative Landscapes, Spatial Clusters, and Fences," *Land*, vol. 2, no. 4, pp. 756–773, Dec. 2013, https://doi.org/10.3390/land2040756.

[24] P. Angelov, Autonomous Learning Systems: From Data Streams to Knowledge in Real-time. John Wiley & Sons, 2012.

[25] M. Shukla, Y. P. Kosta, and M. Jayswal, "A Modified Approach of OPTICS Algorithm for Data Streams," *Engineering, Technology & Applied Science Research*, vol. 7, no. 2, pp. 1478–1481, Apr. 2017, https://doi.org/10.48084/etasr.963.

[26] A. I. Abueid, "Big Data and Cloud Computing Opportunities and Application Areas," *Engineering, Technology & Applied Science Research*, vol. 14, no. 3, pp. 14509–14516, Jun. 2024, https://doi.org/10.48084/etasr.7339.

[27] M. Parekh and M. Shukla, "Survey of Streaming Clustering Algorithms in Machine Learning on Big Data Architecture," in *Information and Communication Technology for Competitive Strategies (ICTCS 2021)*, Jaipur, India, 2023, pp. 503–514, https://doi.org/10.1007/978-981-19-0095-2_48.

[28] P. Chauhan and M. Shukla, "A review on outlier detection techniques on data stream by using different approaches of K-Means algorithm," in

2015 International Conference on Advances in Computer Engineering and Applications, Ghaziabad, India, Mar. 2015, pp. 580–585, https://doi.org/10.1109/ICACEA.2015.7164758.

[29] J. Tamboli and M. Shukla, "A survey of outlier detection algorithms for data streams," in *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, New Delhi, India, Mar. 2016, Art. no. 3535–3540.

[30] P. Fränti and S. Sieranoja, "K-means properties on six clustering benchmark datasets," *Applied Intelligence*, vol. 48, no. 12, pp. 4743–4759, Dec. 2018, https://doi.org/10.1007/s10489-018-1238-7.

[31] M. Shukla, Y. P. Kosta, and P. Chauhan, "Analysis and evaluation of outlier detection algorithms in data streams," in *2015 International Conference on Computer, Communication and Control (IC4)*, Indore, India, Sep. 2015, pp. 1–8, https://doi.org/10.1109/IC4.2015.7375696.