Journal of Theoretical and Applied Information Technology
15th May 2021. Vol.99. No 9

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

AN EVOLVING APPROACH TO DATA STREAMS CLUSTERING BASED ON CHEBYCHEV WITH FALSE MERGING

MUSTAFA TAREQ[1], ELANKOVAN A. SUNDARARAJAN[2]

[1,2]Center for Software Technology and Management, Faculty of Information Science and Technology
Universiti Kebangsaan Malaysia (UKM), Bangi 43600, Malaysia
E-mail: [1]P89127@siswa.ukm.edu.my, [2]elan@ukm.edu.my

ABSTRACT

Data stream clustering is not a recent subject in data analysis and statistics. However, with the rapid deployment of the Internet of Things (IoT), this field faces new practical challenges that researchers intend to solve. These challenges can be summarized under four main categories: high-dimensionality, fast speed of data flow, real-time constraints, and evolving nature. Although numerous density-based algorithms have recently been established for data stream clustering, these algorithms are not without their problems. The quality of these existing clustering algorithms is dramatically reduced when the distance function is used. Another problem is false merging which happens when two or more clusters overlap on top of each other. This paper introduces a new online approach, which is called "Clustering of Evolving data streams based on Chebychev distance with false Merging" (CEC-Merge). The primary goal of this method is to improve the efficiency of the clusters. CEC-Merge is completely online and contains two major steps. The initial step generates a Core Micro-Clusters (CMCs) and the second step merge these CMCs to constructs Macro-clusters. The Chebychev distance function is applied to measure the distance from the new point with the existing CMCs centre. CEC-Merge uses two important parameters to avoid false merging such as "minimum links" and "time intervals". The proposed technique was validated using various consistency measurements on real and artificial evolving data streams and compared with well-known methods. The CEC-Merge provides an effective solution for improving the efficiency of the cluster.

Keyword: *Clustering, Evolving data stream, Chebychev, Core Micro-Cluster, Density based-clustering.*

1. INTRODUCTION

Numerous applications currently depend on massive amounts of data that may require processing in real-time. IoT is one of the critical applications of the data stream [1]. A recent domain emerged from the increase of devices that are connected to other devices or the Internet. Such associated devices incorporate smartphones, drones, wearable electronics (for example, clothes and watches), sensors, home appliances, adhoc network devices and other devices [2-4]. The IoT additionally has a huge potential in various industries, for example, transportation, retail, and health or resource consumption [5-7]. Those devices are generally equipped with an assortment of sensors that can be gathering data continuously in real-time or on multiple occasions per minute. Those associated devices create an extreme amount of data stream with a strong spatiotemporal component.

Several studies attempted to provide a concise description of the style of clustering data streams. For instance, some researchers approached the clustering activity as a task of the broader concept of the data mining techniques [8-13] used for clustering large datasets to make data points in a group similar to each other and differentiate them from points of other clusters [14-17]. The clustering of data streams raises new problems, including noisy data, limited memory, evolving data, limited time, high-dimensional data, and single-pass clustering [18-26]. Data streaming requires real-time processing to manage the large arrival rates of data, and interpreted results are expected within a short timeframe [27, 28]. Maintaining the whole data in dynamic memory is often impossible caused by the unlimited quantity of data being transmitted [29, 30]. Moreover, the data stream passes only once, making multiple active scans impossible to perform [9-13, 31, 32].

Clustering plays a crucial role in assessing data stream mining [15, 33-37]. Several density-based clustering techniques for data streams have been developed over the last few years such as Cauchy [38], i-CODAS [39], CEDAS [34], CODAS [40], and BOCEDS [41]. These algorithms using the Euclidean distance function to determine the distance between an incoming data point and the existing CMCs centre, and the distance between a new or modified CMC centre and another CMC centre. The commonly utilised Euclidian distance function in the literature does not capture the clustering efficiency, rendering it an inadequate metric to be applied to optimise both the homogeneity between various clusters and the heterogeneity inside each cluster. In addition, the dynamical metrics used by the existing clustering algorithms ignore handling the problem of the cluster false merging that happens when two or more clusters overlap on the top of each other. This leads to false prediction in the clustering analysis caused by the evolving natures of the clusters and therefore reduce the quality of the clusters.

This paper introduces a new online approach, which is called “Clustering of Evolving data streams based on Chebychev distance with false Merging” (CEC-Merge). The primary goal of this method is to avoid false merging for evolving clusters and improve the quality of the clusters. CEC-Merge is completely online and contains two major steps. The initial step generates a Core Micro-Clusters (CMCs) and the second step merge these CMCs to constructs Macro-clusters. The proposed algorithm used the Chebychev distance function to measure the distance from the object with the existing CMCs centre. CEC-Merge uses two important parameters to avoid false merging such as “minimum links” and “time intervals”. We depend on different sample speeds to evaluate the performance of the CEC-Merge.

The rest of this paper is arranged as follows. Section 2 presents the state-of-the-art of previous studies. Section 3 explains the methodology of CEC-Merge. Section 4 presents the use of datasets to determine the performance of the CEC-Merge, and the conclusions taken from this analysis are given in Section 5.

2. RELATED WORK

The clustering data stream has drawn various researchers' interest. A substantial issue with clustering data stream is how to analyze this rapidly evolving stream data or how to preserve the overwhelming volume of data for later analysis. Numerous approaches have been presented in the literature that involve data stream clustering.

“Clustering Online Data-streams into Arbitrary Shapes” (CODAS) has been suggested in the field of density-based data stream clustering as an online clustering technique for evolving data streams [40]. It is a data-driven technique that creates micro-clusters by encapsulate the information and construct a highly adaptable cluster that can be scaled to n-dimensional information. Although, the created groups don't develop in this technique. Thus, “Clustering of Evolving Data streams into Arbitrary Shapes” (CEDAS), is another online technique proposed by [34] to improve the CODAS technique by incorporating a linear aging process to cope with the features of data sources. CEDAS contains two major stages. In the initial stage, micro-clusters are generated, or the objects are applied to the existing micro-clusters or outliers. Whereas in the second stage the algorithm intersecting the existing micro-cluster to generate Macro-cluster. Each Macro-cluster in CEDAS contains a graph indicating overlapping every micro-cluster. This technique can cope with noise data and dealing with the evolving data stream. Like other density-based techniques, CEDAS has low clustering quality when dealing with evolving data streams.

Recently CEDAS algorithm was improved by an “online Clustering of the Evolving data stream based on adaptive Chebychev distance” (CEC) [42]. The key objective of the CEC algorithm is to deliver a high cluster quality. This algorithm generated the CMCs by summarizing the information of data points. Unlike the CEDAS algorithm, CEC algorithm used the Chebychev distance function to determine the distance from incoming data and the CMCs or outliers centre, and the distances between the modified CMC centre with another. The CEC technique is capable of detecting outliers and extracting arbitrary shaped clusters. This technique is accessible to handle n-dimensional data and produce high clusters quality. However, this algorithm is not efficient when dealing with evolving data streams.

In [43] the authors have suggested a new “Density-based method for Clustering Data stream using Genetic Algorithm” (DCDGA). It is entirely online using the Chebychev distance function and Genetic Algorithm (GA) to design a consistent and precise function to solve the clustering problem. In order to provide high-quality clusters among data streams, the DCDGA algorithm uses a GA to optimise the radius of the CMCs and density threshold parameters. Whereas, the Chebyshev function is used to calculate the maximum distance between the objects and the centre of CMCs and the maximum distance between the new or modified CMC centre and others. The DCDGA algorithm has good cluster quality and efficiency over various time and data sample speed.

Another new fully online clustering technique is "Clustering of Evolving Data streams via a density Grid-based Method" (CEDGM) proposed by [42]. This algorithm involves two major stages. In the first stage, the grid-based process is implemented to produce CMCs in the area that is unclustered when new objects arrived. At this point, the data region is divided into sub-segments called a grid. All clustering operations are performed on the structure of the grid. Where each object in the data stream falls into an empty space of the grid, the cluster is formed based on the grids density regions. The algorithms determine the outliers and CMCs when a new object arrives and check the arriving object belong to any existing outliers or CMCs. After checking the distance between incoming objects and the existing outliers or CMCs centre less than the radius value using the Euclidean distance function. The algorithm will update the outliers and CMCs and determine the grid coordinate. Otherwise, a new outlier is formed. The fade is used in this technique to reduce the energy of the CMCs and removes them if no information is gotten for a while and their energy bellow zero. In the second stage, the algorithm incorporates any overlapping CMCs into Macro-clusters. Each Macro-cluster is a CMC intersecting graph in which the adjacency relationships for each CMC are stored as a property of that CMC. The advantage of this graph arrangement is that calculations required to distinguish clusters are minimised if one of the CMC dies. The CEDGM algorithm has the capability to distinguish arbitrarily shaped clusters, outliers and also has less computational time.

3. METHODOLOGY

In this study, the CEC-Merge technique aims to provide high-efficiency clusters. It is a fully online clustering technique and it applies Chebychev distance function to measure the distance from an arriving object and the outliers or CMCs centre, and also the distances between new or modified CMC centre with others. In addition, this technique considers two important dynamical matrics to avoid clusters false merging when using evolving data streams, namely, the minimum links and time intervals. In this case, if a new CMC arrives, it must be allocated to the cluster of the closest CMC shell region and creates new links on the neighbours of CMCs and triggers a timer for each created link. The algorithm will merge clusters that have a number of links above a certain level of minimum links with the time that has passed on each one more than the time interval. Thereafter, it returns the updated cluster. The process of merging two clusters is depicted in Figure 1.

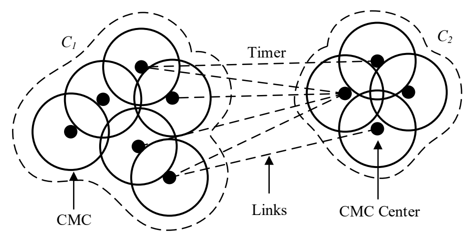


Figure 1: The process of merging two clusters

Each CMC in CEC-Merge algorithm contains a kernel region $r<=r_{0/2}$ and shell reagin r_0. The structure of a CMC is shown in Figure 2. The intersecting process between the kernel regions of CMCs with the shell regions of other CMCs produces Macro-clusters. The cluster can be considered Macro-clusters if the density of the CMC exceeds the density threshold.

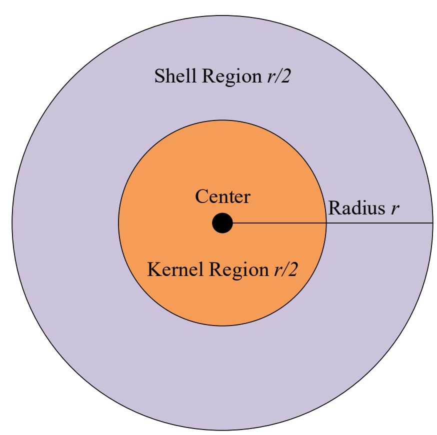


Figure 2: The structure of CMC

In this technique, the objects can fall into three districts as depicted in Figure 3:

1. If the object falls in the unoccupied space, the algorithm will produce a new outlier.
2. If the object falls in the CMC shell area, the algorithm will assign the data point to the cluster and the centre of CMC will be updated C_u.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

3. If the object falls in the CMC kernel area, the object assigned to the CMC and the cluster count is respectively updated C_u.

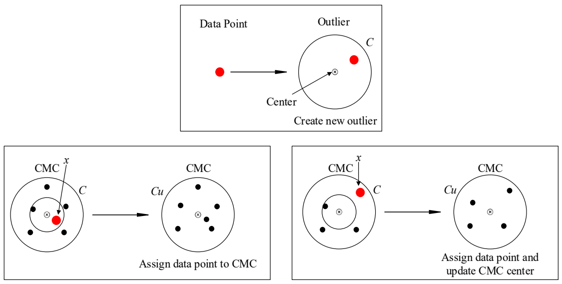


Figure 3: The new object fall into three regions

3.1 Distance Measures

Clustering approaches are based on determining the similarity of the data vector by measuring the distance between pairs. There is no standard distance metric that can better match all clustering applications. The significant features of distance measurement are illustrated in the following points:

a. The distance from point *a* to itself is always zero.
b. Distance is always positive.
c. There is always the same distance between *a* and *b* as between *b* and *a*.
d. The distance between point *a* to point *b* cannot exceed the total of the distance *a* and another point *c* and the distance between *c* and *b*.

The measurement of Chebychev distance is based on the maximum difference in attributes. It is also known as the distance of the chessboard, or $L\infty$ norm, or the norm of Chebychev. It calculates the absolute size of the differences between the data point and the centre of CMC. The distance between vector x and centroid c of Chebyshev is determined by:

$$d(x,c) = max|x_1 - x_2|, |y_1 - y_2| \qquad (1)$$

3.2 Description of The Proposed CEC-Merge Algorithm

The CEC-Merge algorithm contains four essential steps. Figure 4 depicted the flowchart of the CEC-Merge process. For every data point the CEC-Merge algorithm implements these steps if needed:

(i) Parameter Selection.
(ii) Assign the CMC.
(iii) Kill Weak CMC.
(iv) Update Cluster Graph.

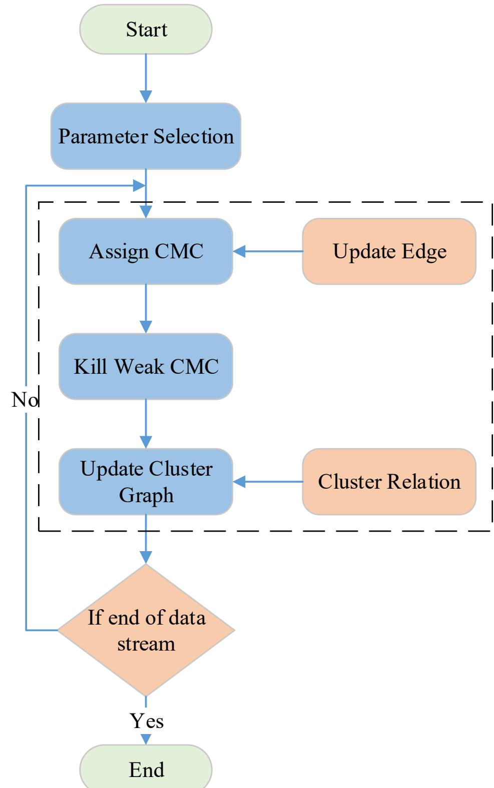


Figure 4: The flowchart of CEC-Merge

i. Parameter selection

Before the CEC-Merge algorithm has been implemented. Several internal parameters are defined on the basis of an application comparable to another algorithm. The CEC-Merge algorithm requires a variety of internal parameters to be performed. The parametric qualities rely upon the application as follows:

a) Radius: is the minimum distance allowed for an object from the CMC centre to belong to it.

b) Decay: is a significant boundary used to signifies the number of last samples we consider for processing at time *t*. For example, if the decay parameter is set to *D* and we are at a time *t*, then we consider the time *t*, *t-1*,*t-2*, ...,*t-D+1* sample.

c) Minimum Threshold: is the minimum number of objects that are needed to form the CMC or promote the current outlier to CMC.

d) Sample Speed: this parameter indicates the speed of the data to be processed.

e) Minimum links: represents the minimum number of links between two clusters to merge

Journal of Theoretical and Applied Information Technology
15th May 2021. Vol.99. No 9

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

then into one cluster while avoiding false merging.

f) **Time intervals**: this parameter indicates the minimum time interval of the intersection between two clusters to merge them into one while avoiding false merging.

ii. Assign the CMC

This section of the approach uses a density Core Micro-clustering method. Each object $x_1, x_2, \ldots, x_n$ in the data stream falls into the empty data space, and the cluster is formed depending on the density of data. When the data arrives, this method identifies the outliers and CMCs and their neighbours. Next step, the algorithm check if the newly arriving data point belongs to any existing outliers or CMCs. Then, the algorithm will use the Chebychev function to calculate the d distance from the data and the nearest CMCs centre. If the distance $d(x_n, c)<R$, where x_n denotes the arriving data at a time t, c is the centre of CMCs or outliers, R is the maximum radius. In this case, the data point belongs to the CMC or outliers; otherwise, a new outlier is created. Further confirmation is directed to choose whether the update unit is an outlier, and if the number of data points is higher than the density threshold, then the algorithm will promote the outlier to CMC. A new or modified CMC must be assigned to the cluster of the nearest CMC shell region by determining edge.

In the update edges function, the algorithm creates new links on the neighbours of CMCs and triggers a timer for each created link.

Algorithm 1: Assign the CMC

Input: x, radius, Density Threshold
Output: CMCs, Outliers, Clusters Relations, Number of Clusters
Start:
Calculate the distance between data point and CMCs or Outliers center using Chebychev
If $D(x_i, c) > radius$ **then**
  Create new outlier
Else
  Update the CMC
  Determine the life of CMC //go to Algorithm 2
Else If (*the data point x is nearer to an outlier*) **then**
    Update the outlier
   If (*density(outlier) > Density Threshold*) **then**
    Update the Outlier to be a new CMC
    CMC = CMC + 1
    Determine the life of CMC//go to Algorithm 2
   End
End
If (*new CMC was arrived*) **then**
  Edges = determine Edges (CMC)
  Add the new link to the neighbors CMCs and trigger its timer
  Update cluster graph (Cluster Relation)
//go to Algorithm 3
End

iii. Kill Weak CMC

This section of the CEC-Merge technique reduces and excludes the energy of CMCs if their energy is below zero. The algorithm using the fading window to reduce the life of CMCs. When removing the CMC, also the quantity of CMCs will be reduced, and the edge will be updated.

Algorithm 2: Remove weak CMC

Input: CMCs, Fade
Decrease the life of CMCs using the $decay = 1/Fade$
If (*CMC. life <= 0*) **then**
  Remove the dead CMCs and all references
  Remove all edges containing the CMCs
  Reduce the number of CMCs
End

iv. Update Cluster Graph

Clustering graph is used to form Macro-clusters by interconnecting shell region of CMC with kernel region of other CMC while avoiding false merging. some change is made by the clustering graph if either:

Case1. A CMC centre location has been shifted.

Case2. A CMC is moved or when a new CMC arrives.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Case3. A CMC has died and been removed.

The goal of cluster relation is to merge clusters that have a number of links above a certain level of the minimum links with the time that has passed on each one more than the time interval. Thereafter, it returns the updated cluster.

Algorithm 3: Update Clusters Graph

```
Input: CMCs, Clusters Relations, Minimum Links, Time Interval
Output: Clusters Relations, Number of Clusters
Start:
    For each relation in Clusters Relations do
        If (relation.timer > Time Interval) then
            Links = determine Links (relation)
            If Links > Minimum Links then
                Merge Clusters
            End
        Else
        Remove(relation)
        End
    End
End
```

4. RESULTS AND DISCUSSION

The efficiency comparison of the CEC-Merge algorithm with that of the CEC [41] and CEDAS[33] algorithms are discussed in this section. The CEC-Merge algorithm is implemented using MATLAB R18a, and their performances are evaluated on a PC with an Intel i5 CPU with 16GB RAM. The proposed and other algorithms parameters are set as *Decay = 1000, Radius = 0.05, Density Threshold = 4 data points, Time Intervals = 150 s, Minimum Links = 25*. The minimum links is set to ensure all CMCs are connected in the same cluster or between two clusters in the same relation. Whereas, the time intervals are set on each created link to avoid false merging when two clusters are getting close to each other or overlapping on top of each other.

Real-world and artificial datasets were used to evaluate the proposed CEC-Merge algorithm, such as evolving network intrusion detection (KDDCUP'99) [25, 34, 35, 41, 44-47], evolving Spiral datasets. The KDDCUP'99 containing two weeks of TCP connection logs of LAN traffic. It contains 4,999,000 objects depicted by 42 attributes with 34 dimensions. We used 10% of the data for this experiment. Besides, 2d-dimension synthetic datasets are used, i.e., Spiral contains 6.012 records [42, 43, 47]. The purpose of the experiment is to verify the average accuracy, average purity, and average Normalized Mutual Information (NMI).

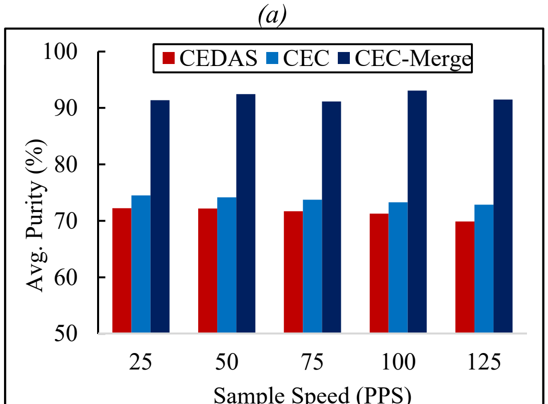
4.1 Cluster Purity

Purity is applied for evaluating the CEC-Merge algorithm and can be described as the number of data points in each cluster divided by the number of ground truth [34, 39-43, 47-49]:

$$Purity = \frac{\sum_{i=1}^{N} n_i^d}{n_i} \tag{2}$$

where n_i^d is the dominant class sample, n_i is the number of samples present in a cluster and N is the number of clusters. The comparison between the CEC-Merge, CEC and CEDAS algorithms on the KDDCUP'99 and Spiral datasets in terms of average purity with varies sample speeds are depicted in Figure 5. When the speed of the sample differs from 25 to 125 pits per second (PPS), the proposed CEC-Merge algorithm has better average purity compared to other algorithms. For example, at sample speed 100 PPS, the CEC-Merge average purity in the KDDCUP'99 achieves 94.31% compared with the values of the CEC of 92.47%, respectively; the value of CEDAS of 87.39%, respectively. While the average purity of CEC-Merge in the Spiral dataset is above 90% at all sample speed. We conclude that the CEC-Merge algorithm is capable of achieving higher average purity than the CEC and CEDAS algorithms, the reason is that the proposed algorithm has the ability to detect false merging resulted from evolving data streams and avoid it by using the time interval and the minimum links, thereby enhancing the clustering purity.

(a)



Sample Speed (PPS)

(b)

Figure 5: Performance of the CEC-Merge, CEC and CEDAS using KDDCUP'99 and Spiral datasets in terms of purity

Journal of Theoretical and Applied Information Technology
15th May 2021. Vol.99. No 9

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

4.2 Cluster Accuracy

Accuracy is used for evaluating the clustering of data streams and can be described as a number of samples in a cluster that belong to that cluster and do not belong to any other cluster [34, 39-43, 47, 48]:

$$Accuracy = \frac{\sum_{i=1}^{N} n_i^d}{\sum_{i=1}^{N} n_i} \quad (3)$$

where n_i^d is the dominant class sample, n_i is the number of samples present in a cluster and N is the number of clusters. The comparison of the CEC-Merge algorithm with CEC and CEDAS algorithms in terms of average accuracy depicted in Figure 6. The evolving KDDCUP'99 and spiral datasets were used to evaluate the proposed algorithm. When the speed of the sample differs from 25 to 125 PPS, the CEC-Merge algorithm always achieves high average accuracy compared to CEC and CEDAS algorithms. For example, at a speed of 125 PPS, the CECC-Merge in the KDDCUP'99 achieves 97.20%, CEC achieves 94.5% and CEDAS achieves 90.53%. Regarding the average accuracy with Spiral, the CEC-Merge reaches 85.85%, whereas CEC reaches 72.77% and CEDAS reaches 70.83%. The outcome shows that the CEC-Merge algorithm can obtain high average accuracy than the CEC and CEDAS algorithms. This result is interpreted by the awareness of false merging when the clusters overlap due to evolving; their overlapping is temporary and does not mean unifying the clusters.

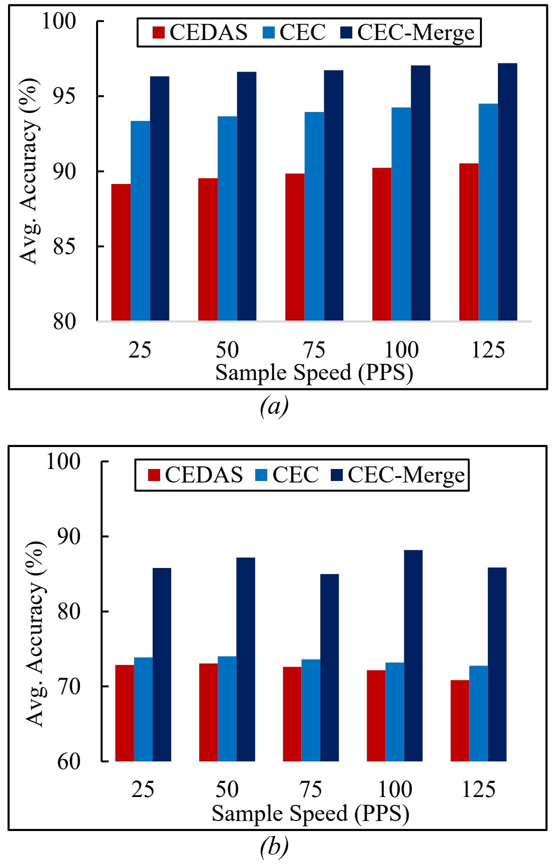


Figure 6: Performance of the CEC-Merge, CEC and CEDAS using KDDCUP'99 and Spiral datasets in terms of accuracy

4.3 Normalized Mutual Information (NMI)

NMI is used to assess the clustering of data streams and show how similar two clusterings are [47, 50-52]:

$$NMI\ (P^*, L^t) = \frac{\sum_{i=1}^{c}\sum_{j=1}^{c} n_{ij}\log_2\left(\frac{n n_{ij}}{n_i^* n_j^t}\right)}{\sqrt{\sum_{i=1}^{c} n_i^*\log_2\left(\frac{n_i^*}{n}\right)\times\sum_{i=1}^{c} n_j^t\log_2\left(\frac{n_j^t}{n}\right)}} \quad (4)$$

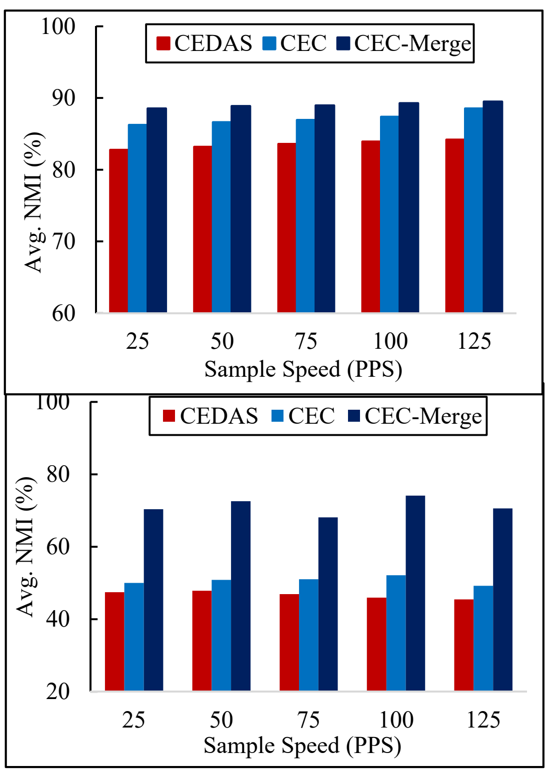
where P^* is the consensus partition, L^t is the ground-truth of the dataset, n is the total number of data points in the given dataset X, n_{ij} is the number of data points in the intersection of the i^{th} cluster of $P^*$and the j^{th} cluster of L^t, n_i^* and n_j^t which are the number of data points in the i^{th} cluster in $P^*$and the number of data points in the j^{th} cluster in L^t, respectively. The comparison between the proposed CEC-Merge, CEC and CEDAS in terms of the

Journal of Theoretical and Applied Information Technology
15th May 2021. Vol.99. No 9
© 2021 Little Lion Scientific

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

average normalized mutual information is illustrated in Figure 7. We test them on the same evolving KDDCup'99 and Spiral dataset. The evaluation shows that the CEC-Merge algorithm has superior normalized mutual information than the CEC and CEDAS algorithms. For example, when the speed is 25 PPS, the CEC-Merge algorithm with KDDCUP'99 reaches 88.55%, whereas CEC reaches 86.26% and CEDAS reaches 82.79%, respectively. Regarding the average normalized mutual information with Spiral, the CEC-Merge reaches 70.39%, whereas CEC reaches 50.03% and CEDAS reaches 47.48%. We conclude that the CEC-Merge can achieve superior average normalized mutual information than the CEC and CEDAS algorithms. This evaluation shows the similarity between two clusters and the reason to obtain better NMI in our proposed algorithm because the clustering solution perfectly matches a given class labelling of the data.

(a)



(b)

Figure 7: Performance of the CEC-Merge, CEC and CEDAS using KDDCUP'99 and Spiral datasets in terms of NMI.

5. CONCLUSIONS

This paper introduced an improved approach for evolving data streams clustering using a density core micro-cluster with Chebychev distance function and false merging called CEC-Merge. The proposed algorithm considered two important metrics such as the minimum links and time intervals to enhance the clustering quality. This algorithm can handle the evolving nature of data streams in an online manner. This algorithm is compared with the CEC and CEDAS algorithms in terms of average purity, average accuracy, and average normalized mutual information.

The CEC-Merge algorithm is tested by the evolving data streams such as KDDCUP'99 and Spiral. It further demonstrates its ability to produce high cluster efficiency. Evolving data streams using different quality metrics show that the clusters generated in the CEC-Merge algorithm are pure and more accurate than those of similar existing clustering algorithms. This is due to its ability to detect false merging resulting from evolving and avoiding it by using the time interval and the minimum links, and does not mean unifying the clusters. However, one of the limitations of the CEC-Merge algorithm requires high number of distance function calls to calculate the distance between the data point and the CMC or outlier centres, especially with high dimensional data. Our future studies will focus on using false merging with the density grid-based method.

ACKNOWLEDGMENTS

This work is partly supported by the Ministry of Higher Education Malaysia, grant number "FRGS/1/2018/ICT04/UKM/02/1".

REFERENCES

[1] M. Ge, H. Bangui, and B. Buhnova, "Big data for internet of things: a survey," *Future Generation Computer Systems,* vol. 87, pp. 601-614, 2018.

[2] W. H. Hassan, "Current research on Internet of Things (IoT) security: A survey," *Computer Networks,* vol. 148, pp. 283-294, 2019.

[3] M. Mohammadi, A. Al-Fuqaha, S. Sorour, and M. Guizani, "Deep learning for IoT big data and streaming analytics: A survey," *IEEE Communications Surveys & Tutorials,* vol. 20, no. 4, pp. 2923-2960, 2018.

[4] M. A. Azzawi, R. Hassan, and K. A. A. Bakar, "A review on Internet of Things (IoT) in healthcare," *International Journal of Applied*

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Engineering Research, vol. 11, no. 20, pp. 10216-10221, 2016.

[5] Y. N. Malek *et al.*, "On the use of IoT and big data technologies for real-time monitoring and data processing," *Procedia computer science,* vol. 113, pp. 429-434, 2017.

[6] E. Ahmed *et al.*, "The role of big data analytics in Internet of Things," *Computer Networks,* vol. 129, pp. 459-471, 2017.

[7] D. V. Dimitrov, "Medical internet of things and big data in healthcare," *Healthcare informatics research,* vol. 22, no. 3, pp. 156-163, 2016.

[8] C. C. Aggarwal, *Data streams: models and algorithms*. Springer Science & Business Media, 2007.

[9] L. O'callaghan, N. Mishra, A. Meyerson, S. Guha, and R. Motwani, "Streaming-data algorithms for high-quality clustering," in *Proceedings 18th International Conference on Data Engineering*, 2002, pp. 685-694: IEEE.

[10] D. Barbará, "Requirements for clustering data streams," *ACM sIGKDD Explorations Newsletter,* vol. 3, no. 2, pp. 23-27, 2002.

[11] S. Guha, A. Meyerson, N. Mishra, R. Motwani, and L. O'Callaghan, "Clustering data streams: Theory and practice," *IEEE transactions on knowledge and data engineering,* vol. 15, no. 3, pp. 515-528, 2003.

[12] C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu, "A framework for clustering evolving data streams," in *Proceedings of the 29th international conference on Very large data bases-Volume 29*, 2003, pp. 81-92: VLDB Endowment.

[13] A. Zubaroğlu and V. Atalay, "Data Stream Clustering: A Review," *arXiv preprint arXiv:2007.10781,* 2020.

[14] M. Ghesmoune, M. Lebbah, and H. Azzag, "State-of-the-art on clustering data streams," *Big Data Analytics,* vol. 1, no. 1, p. 13, 2016.

[15] B. Aubaidan, M. Mohd, and M. Albared, "Comparative study of k-means and k-means++ clustering algorithms on crime domain," *Journal of Computer Science,* vol. 10, no. 7, pp. 1197-1206, 2014.

[16] M. Mohd *et al.*, "Optimal initial centroid in k-means for crime topic," *Journal of Theoretical and Applied Information Technology,* vol. 45, no. 1, pp. 19-26, 2012.

[17] M. T. Abd, M. Mohd, and M. T. Abd, "Investigation of Data Representation Methods with Machine Learning Algorithms for Biomedical Named Enttity Recognition," in *2018 Fourth International Conference on Information Retrieval and Knowledge Management (CAMP)*, 2018, pp. 1-6: IEEE.

[18] J. de Andrade Silva, E. R. Hruschka, and J. Gama, "An evolutionary algorithm for clustering data streams with a variable number of clusters," *Expert Systems with Applications,* vol. 67, pp. 228-238, 2017.

[19] M. D. de Assuncao, A. da Silva Veith, and R. Buyya, "Distributed data stream processing and edge computing: A survey on resource elasticity and future directions," *Journal of Network and Computer Applications,* vol. 103, pp. 1-17, 2018.

[20] M. Khalilian and N. Mustapha, "Data stream clustering: Challenges and issues," *arXiv preprint arXiv:1006.5261,* 2010.

[21] G. Krempl *et al.*, "Open challenges for data stream mining research," *ACM SIGKDD explorations newsletter,* vol. 16, no. 1, pp. 1-10, 2014.

[22] 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, 2015.

[23] J. A. Silva, E. R. Faria, R. C. Barros, E. R. Hruschka, A. C. De Carvalho, and J. Gama, "Data stream clustering: A survey," *ACM Computing Surveys (CSUR),* vol. 46, no. 1, p. 13, 2013.

[24] M. H. ur Rehman, C. S. Liew, T. Y. Wah, and M. K. Khan, "Towards next-generation heterogeneous mobile data stream mining applications: Opportunities, challenges, and future research directions," *Journal of Network and Computer Applications,* vol. 79, pp. 1-24, 2017.

[25] D. Brown and Y. Shi, "A Distributed Density-Grid Clustering Algorithm for Multi-Dimensional Data," in *2020 10th Annual Computing and Communication Workshop and Conference (CCWC)*, 2020, pp. 0001-0008: IEEE.

[26] R. Ahmed, G. Dalkılıç, and Y. Erten, "DGStream: High quality and efficiency stream clustering algorithm," *Expert Systems with Applications,* vol. 141, p. 112947, 2020.

[27] M. Mousavi, H. Khotanlou, A. A. Bakar, and M. Vakilian, "Varying density method for data stream clustering," *Applied Soft Computing,* vol. 97, p. 106797, 2020.

[28] M. Aljibawi, M. Z. A. Nazri, and Z. Othman, "A survey on clustering density based data stream algorithms," *Int. J. Eng. Technol,* vol. 7, no. 36, pp. 147-153, 2018.

[29] M. Mousavi, A. A. Bakar, and M. Vakilian, "Data stream clustering algorithms: A review," *Int J Adv Soft Comput Appl,* vol. 7, no. 3, p. 13, 2015.

[30] M. Mousavi and A. A. Bakar, "Improved density based algorithm for data stream clustering," *Jurnal Teknologi,* vol. 77, no. 18, 2015.

[31] M. R. Ackermann, M. Märtens, C. Raupach, K. Swierkot, C. Lammersen, and C. Sohler, "StreamKM++: A clustering algorithm for data streams," *Journal of Experimental Algorithmics (JEA),* vol. 17, p. 2.4, 2012.

[32] M. Aljibawi, M. Z. A. Nazri, and Z. Othman, "A survey on clustering density based data stream algorithms," *International Journal of Engineering and Technology (UAE),* vol. 7, no. 4.36 Special Issue 36, pp. 147-153, 2018.

[33] T. Li and C. Ding, "Data clustering: algorithms and applications," ed: Boca Raton: CRC Press, 2013.

[34] R. Hyde, P. Angelov, and A. R. MacKenzie, "Fully online clustering of evolving data streams into arbitrarily shaped clusters," *Information Sciences,* vol. 382, pp. 96-114, 2017.

[35] A. Amini, H. Saboohi, T. Herawan, and T. Y. Wah, "MuDi-Stream: A multi density clustering algorithm for evolving data stream," *Journal of Network and Computer Applications,* vol. 59, pp. 370-385, 2016.

[36] C. C. Aggarwal and C. K. Reddy, "Data Clustering: Algorithms and Applications. vol. 2," ed: Chapman & Hall/CRC, 2013.

[37] J. Youn, J. Shim, and S.-G. Lee, "Efficient data stream clustering with sliding windows based on locality-sensitive hashing," *IEEE Access,* vol. 6, pp. 63757-63776, 2018.

[38] I. Škrjanc, S. Ozawa, T. Ban, and D. Dovžan, "Large-scale cyber attacks monitoring using evolving cauchy possibilistic clustering," *Applied Soft Computing,* vol. 62, pp. 592-601, 2018.

[39] M. K. Islam, M. M. Ahmed, and K. Z. Zamli, "i-CODAS: An Improved Online Data Stream Clustering in Arbitrary Shaped Clusters," *Engineering Letters,* vol. 27, no. 4, 2019.

[40] R. Hyde and P. Angelov, "A new online clustering approach for data in arbitrary shaped clusters," in *2015 IEEE 2nd International Conference on Cybernetics (CYBCONF)*, 2015, pp. 228-233: IEEE.

[41] M. K. Islam, M. M. Ahmed, and K. Z. Zamli, "A buffer-based online clustering for evolving data stream," *Information Sciences,* vol. 489, pp. 113-135, 2019.

[42] M. Tareq, E. A. Sundararajan, and M. Mohd, "Online clustering of evolving data stream based on adaptive Chebychev distance," in *Proc. 281st Int. Conf. IIER*, 2020, pp. 41-46.

[43] M. Tareq and E. A. Sundararajan, "A New Density-Based Method for Clustering Data Stream Using Genetic Algorithm," *Technology Reports of Kansai University,* vol. 62, no. 11, p. 16, 2020.

[44] N. Ohadi, A. Kamandi, M. Shabankhah, S. M. Fatemi, S. M. Hosseini, and A. Mahmoudi, "SW-DBSCAN: A Grid-based DBSCAN Algorithm for Large Datasets," in *2020 6th International Conference on Web Research (ICWR)*, 2020, pp. 139-145: IEEE.

[45] M. D. Parmar *et al.*, "FREDPC: A feasible residual error-based density peak clustering algorithm with the fragment merging strategy," *IEEE Access,* vol. 7, pp. 89789-89804, 2019.

[46] L. Wang and H. Li, "Clustering algorithm based on grid and density for data stream," in *AIP Conference Proceedings*, 2017, vol. 1839, no. 1, p. 020202: AIP Publishing LLC.

[47] M. Tareq, E. A. Sundararajan, M. Mohd, and N. S. Sani, "Online Clustering of Evolving Data Streams Using a Density Grid-Based Method," *IEEE Access,* vol. 8, pp. 166472-166490, 2020.

[48] C. Fahy, S. Yang, and M. Gongora, "Ant colony stream clustering: A fast density clustering algorithm for dynamic data

streams," *IEEE transactions on cybernetics,* vol. 49, no. 6, pp. 2215-2228, 2018.

[49] Y. Chen and L. Tu, "Density-based clustering for real-time stream data," in *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2007, pp. 133-142.

[50] M. Mojarad, S. Nejatian, H. Parvin, and M. Mohammadpoor, "A fuzzy clustering ensemble based on cluster clustering and iterative fusion of base clusters," *Applied Intelligence,* vol. 49, no. 7, pp. 2567-2581, 2019.

[51] M. Mojarad, H. Parvin, S. Nejatian, and V. Rezaie, "Consensus function based on clusters clustering and iterative fusion of base clusters," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems,* vol. 27, no. 01, pp. 97-120, 2019.

[52] H. Alizadeh, M. Yousefnezhad, and B. M. Bidgoli, "Wisdom of crowds cluster ensemble," *Intelligent Data Analysis,* vol. 19, no. 3, pp. 485-503, 2015.