Received: July 14, 2021. Revised: June 6, 2021.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Data Stream Clustering Using Fuzzy-based Evolving Cauchy Algorithm

Hussein A. A. Al-Khamees[1]* Nabeel Al-A'araji[1] Eman S. Al-Shamery[1]

[1] *Software department, Information Technology College, Babylon University, Iraq*
* Corresponding author's Email: hussein.alkhamees7@gmail.com

Abstract: Many different applications in the real world can generate huge amount of data, that has unconventional features including massive size, fast access, besides the evolving in its nature; this is data stream. Data stream clustering algorithms began to grow at breakneck speed. evolving Cauchy (eCauchy) is a significant algorithm of density-based data stream clustering. The major limitation of eCauchy is the high number of clusters generated in dynamic environments. This paper presents an evolving model for data stream by optimizing e-Cauchy algorithm to decrease the number of clusters and reach to an ideal number by implementing evolving mechanisms (adding, merging, splitting clusters) based on a specific membership function. Model is tested by two real datasets NSL-KDD99 and keystroke. Proposed model outperforms two other algorithms, e-Cauchy and FEAC-Stream. Model constructs five and four clusters with less time to implement 1.30 and 2.30 minutes respectively for each dataset.

Keywords: Data stream clustering, Density-based method, E-Cauchy algorithm, Evolving model.

1. Introduction

Recently, the data stream appears as a modern type of data that is a huge amount of data, online arriving with a high speed and ordered sequence, not static but evolving, concept drift appearance due to the change of data distribution besides, the dimension of the data stream, in general, is a high [1]. Data stream are generated by many applications implement in diverse systems such as Intrusion Detection Systems, retail stores, monitoring systems, social media analysis systems, sensor network and others [2].

According to the data stream characteristics, it needs processing in real-time. Thereupon as the traditional methods of clustering's task can't handle the data stream [1]. Usually, data stream mining aims to extract knowledge from these non-stop data [3].

Generally, the clustering task refers to group similar data samples in one cluster, at the same time it dissimilar to data samples in other clusters [4].

Nearly all data stream clustering algorithms produce clusters over the entire streaming data set. In addition, data stream clustering algorithms can classify into five methods that are, hierarchical method, partition method, density-based method, grid-based method, model-based method [5].

Evolving systems can change the general structure of the model designed to describe the data stream by updating the data at every time. This change was done by implemented several mechanisms [6].

The evolving fuzzy algorithms are significant type of the evolving system. Simply, through these algorithms the design model able to interact the given data [7].

The fuzzy systems are specific mathematical models which build upon the concept of fuzzy logic, in fuzzy logic the truth values are 'fuzzified' as assigned a value in the range [0,1]. This means that fuzzy logic is able to represent vague statements, expressing uncertainties and/or incomplete knowledge of humans [8].

The proposed model based on optimizing the evolving fuzzy algorithm (e-Cauchy) to decrease the number of clusters that generated from it and reach to an ideal number of clusters by implementing evolving mechanisms (adding, merging, splitting clusters) based on a specific membership function.

The proposed evolving model is tested by two real streaming datasets NSL-KDD99 and keystroke. The results of the proposed model are more accurate than e-Cauchy results, where these results showed that the final number of clusters are five clusters to NSL-KDD99 dataset and four clusters to the keystroke dataset respectively. Moreover, the time required to implement the model is (1.30) and (2.30) minutes for each dataset respectively. The proposed model is more efficient than Fast Evolutionary Algorithm for Clustering data streams (FEAC-Stream), where the number of generated clusters were compared for the normal class of KDD dataset.

The rest sections of this paper are organized as follows: section 2 introduces the related works, section 3 presents the clustering data stream, the evolving models is discusses in section 4, section 5 devotes to Cauchy clustering algorithm, section 6 explains the methodology, section 7 offers the implementation of Cauchy algorithm, section 8 displays the evolving mechanisms, the distribution of test data illustrates in section 9, section 10 discusses the membership function, section 11 dedicates to data set description, section 12 to evaluate the proposed model, finally, section 13 expounds to the experimental results and discussion.

2. Related works

The evolving models have attracted the attention of many researchers over the past years. This section aims to present some of these models that are most related to our proposed model.

Hartert et al. [9] proposed the Dynamic Fuzzy K-Nearest Neighbour model (DFKNN), for monitoring the online evolving systems. It consists of three steps, detection, adaptation, and validation. Mainly, the model implemented three mechanisms including add clusters depended on the distance from a data sample to the centre of the cluster, merging clusters when two clusters are very close to each other (by determining a merging threshold) and finally the removing clusters that implemented when the cluster exceeds a predefined threshold or exceeds a period of time. The model is high dependence on several parameters and their initial values.

PRECUP et al. [10] presented a model to describe the dynamics of human fingers for characterizing the person's hand. The testing step was implemented by comparing the synthetic data set with the movements of only three fingers and the model outputs represented by the finger angles of a person. The model implemented the adding clusters depending on the data sample potential. The data sample can be the center of the cluster when the distance to the nearest center is larger than a predefined threshold. The model lacks to the implementation of other evolving mechanisms.

An evolving Fuzzy Model [11] has been suggested to monitor the system of waste-water treatment plants and detect its fault. This model carried out the mechanisms of adding, merging, splitting and removing clusters. For adding cluster, the normalized Mahalanobis distance from the current data sample to cluster center is computed. Actually, if it is larger than 1, a new cluster is added. Also, in merging clusters, the same idea and type of measurement are used. Whereas the splitting mechanism depended on the model error rate. Finally, the removing clusters relied on the age of the cluster, if a cluster didn't receive any data sample during a period of time, it can be removed. The model depended on two conditions that should fulfill for adding a new cluster that makes the model tend to be somewhat complicated.

Pratama et al. [12] proposed an evolving model able to self-evolving in data stream environment which involves the drifts and shifts phenomena's that is PArsimonious Network based on Fuzzy Inference System (PANFIS). It applied the adding, merging and removing clusters. A cluster is added if the model error percentage of a new data sample is high, whereas the merging implemented for two memberships function to reduce the fuzzy sets depended on the similarity of both width and center of for each of the two fuzzy sets. Removal is done when the cluster is classified as unimportant and this occurs when the cluster is not effective in the model output. The model capability in real-time is weak.

3. Clustering data stream

The simplest method to understand the core of the clustering is that it assorts the similar data samples (that have similar attributes) into a cluster i.e. those samples have a high degree of similarity. Meanwhile, these samples (in a cluster) differ from those in other clusters [13].

The environment and behaviour of the data stream differs from the traditional data. Therefore, the data stream clustering algorithm should be carefully chosen to be an appropriate method, otherwise it will be a worthless method [14].

The density-based idea sprang up from employing different density functions in dense regions. If the space of data is measured by the density concept, the clusters that have different degrees of density are normal results [13].

The core idea of density-based algorithms is to

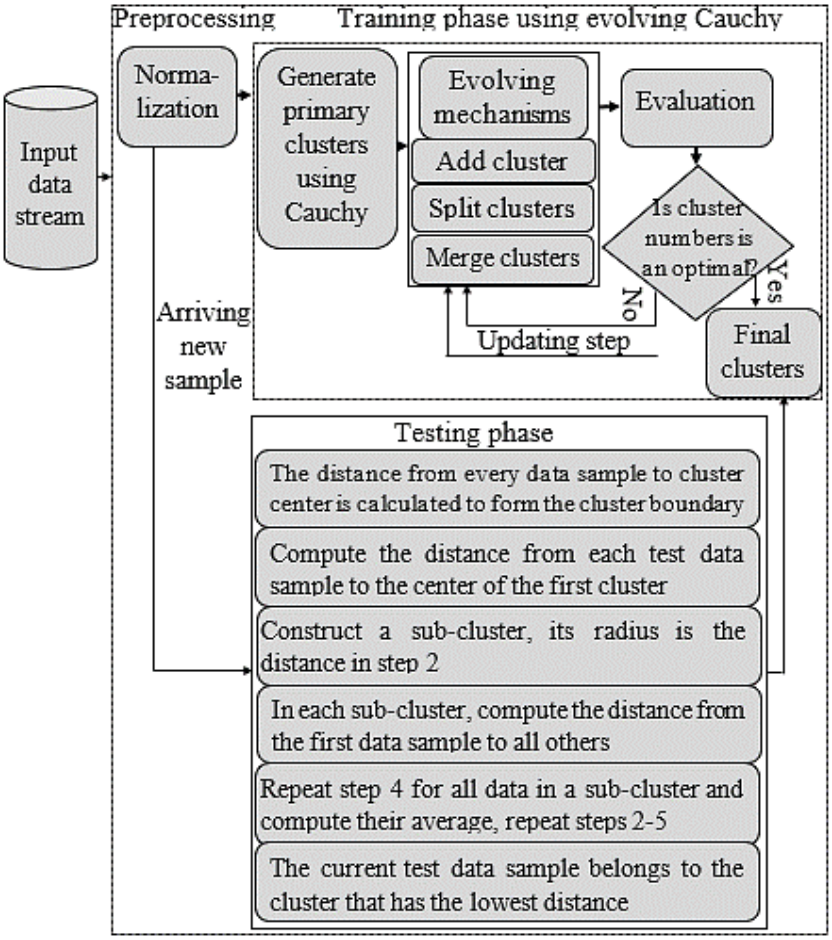

Figure. 1 Methodology of the proposed model

detect core data samples in the dense region then start the expansion [15].

There are many algorithms of the density-based clustering for data stream such as, HDenStream, rDenStream, MR-Stream, HDDStream, OPClustream, MuDi-Stream, SVStream, FEAC-Stream, Evolving-Cauchy (eCauchy) and others [2, 16-17].

4. The evolving models

Generally, the models can be divided into four categories [8].

1- First principle models (analytical models).

2- Knowledge-based models.

3- Data-driven models.

4- The hybrid models.

Indeed, the evolving models can be classified as data-driven models, that are:

1- Automatically adapted and extended.

2- Dynamically evolved based on the new data samples.

This means, evolving models have the ability for supporting many scenarios of streaming data. It takes into consideration the evolving of the structure and parameters when it needs or may be based on the properties of the samples of the data stream that arrived [18].

5. Cauchy clustering algorithm

The section aims to give clear and sufficient concepts of the eCauchy algorithm. The key idea of this algorithm is depended on the computation of the density for each data sample, and then this determines to which cluster must be appended.

Generally, e-Cauchy algorithm consists of three steps, calculating the density for every arriving sample, then comparing the density with thresholds and finally, clustering the samples [19]. e-Cauchy algorithm requires a small number of initial parameters [20].

The development step that was made to the basic algorithm includes the addition of an effective step using a specific strategy called trial and error that mainly depends on presenting candidate solutions and monitoring their results. If a solution has been found that is valid, the task can be considered complete. The goal of this strategy is to provide a single solution to the problem [21].

Accordingly, the model tests many values for the max density threshold to reach an optimal value.

6. Methodology

The designed model mainly consists of training and testing phases. In the training phase by using evolving Cauchy, firstly, implement the Cauchy clustering algorithm to cluster the training data samples and because this algorithm computes the density for every sample and the possibility of data samples convergence to each other, thus model builds many clusters that form the primary number of clusters, but is still a large number. Therefore, the evolving mechanisms can be implemented to get fewer clusters. These mechanisms contain add cluster, split clusters and merge clusters. In order to obtain an optimal results of clusters number, the model is evaluated after applied these mechanisms. Furthermore, the update step was included and it is a clear hint to re-implement a mechanism as long as the main goal is not met. The result of this phase is the formation of the final number of clusters. Fig. 1 is depicted this methodology.

While in the testing phase, a new method is presented to construct cluster boundaries and implement it to the result of training phase (final number of clusters) to determine the most appropriate cluster of the current test data sample and attach it to that cluster.

7. Implementation of cauchy algorithm

This section displays the pseudocode of the developed version of Cauchy algorithm.

1: **Input**: Dataset(d)
2: **Output**: clusters
3: procedure Cauchy clustering (d)
4: Set initial parameters M_j, C, $\mu^j = 1$, $S = 0$
5: **Repeat**
6: **Set** Γ_{max}
7: **For** each $p_i \in d$, i in 1,2,… size of (d)
8: calculates Y_i as

$$\frac{1}{1+\frac{1}{\sigma_t^2}(z(k)-\mu^j)^T(\sum^j)^{-1}(z(k)-\mu^j)+\frac{1}{\sigma_t^2}\frac{(M-1)}{\mu^j}q}$$

9: **If** $Y_i \leq \Gamma_{max}$
10: generate a new cluster and set the initial parameters of cluster I by M_j, $\mu^j = 1$, $S = 0$
11: increase the number of clusters by 1, $C = C+1$
12: **Else**
13: modify parameters of cluster e^j, $j \in c$
$M_j = M_j + 1$
$e_j(k) = z(k) - M_j$
$\mu^j_{Mj+1} = \mu^j_{Mj} + (1/M^j+1)\, e^j_{Mj}(k)$
$S^j_{M^j+1} = S^j_{M^j} + e^j_{M^j}(k)\,(z(k) - \mu^j_{M^j+1})^T$
$\sum^j_{M^j+1} = (1/_{M^j})\, S^j_{M^j}+1$
14: **End For**
15: **Until** get an optimal value of Γ_{max}

Cauchy clustering algorithm pseudocode

Cauchy algorithm contains two thresholds which are sigma and max_density thresholds. The most effective very influential threshold is the max_density Γ_{max}. Depending on its value, the number of resulting clusters varies for each executing as well as the execution time varies.

After assigning a specific value to Γ_{max} (line 6), the algorithm starts working. When a new sample arrives z(k), the procedure that computed the sample density is implemented (lines 7and 8), and it is compared with Γ_{max} (line 9) to determine the suitable cluster that must be appended to it. Therefore, the algorithm includes recursively the computation of the samples density. If the sample density is less than or equal to Γ_{max} (lines 10 and 11), then create a new cluster and add this sample to it, and set some required parameters. Otherwise, add the sample to an old cluster that has a minimum density, then modify cluster parameters (lines 12 and 13).

After several attempts by trial and error (lines 5 and 15), we set Γ_{max} to 0.0038 in order to generate a reasonable number of clusters during a small period of time. Furthermore, if these two thresholds are not set correctly, then the resulting cluster will not be good. In other words, these thresholds consider difficult to set in order to give reasonable results.

7.1 The parameters in cauchy algorithm

- **The number of elements per cluster:**

The number of elements in each cluster in the algorithm denoted by M_i. Initially, it is set to 1, and then the model is added 1 on each sample reception.

- **The number of clusters:**

This parameter is represented by C. Initially, it is set to 1 while the first sample arrives because it constructs the first cluster, and then is incremented by 1 after each cluster build.

- **The difference:**

It denoted by $e_j(k)$. it computes by subtracting the current sample z(k) from the current mean M_j.

$$e_j(k) = z(k) - M_j \quad (1)$$

- **The cluster center**

The cauchy clustering algorithm considers an efficient algorithm because it computes the center of each cluster recursively, that denoted by μj, and it computed by Eq. (2):

$$\mu^j_{Mj+1} = \mu^j_{Mj} + (1/M^j+1)\, e^j_{Mj}(k) \quad (2)$$

- **The covariance matrix**

Firstly, calculate S by Eq. (3), where it is a special parameter to calculate the covariance matrix.

$$S^j_{M}{}^j_{+1} = S^j_{M}{}^j + e^j_{M}{}^j(k)\, (z(k) - \mu^j{}_{M}{}^j_{+1})^{T} \quad (3)$$

Then the covariance matrix $\sum^j{}_M{}^j$ is calculate as:

$$\sum^j_{M}{}^j_{+1} = (1/{}_{M}{}^j)\, S^j_{M}{}^j + 1 \quad (4)$$

7.2 Density computation

The computation of density represents the core of the Cauchy algorithm. If the data set sample is denoted by z(k), then the data density can be defined simply as the sum of the distances resulted from the current sample z(k) and all beforehand samples which belong to a specific cluster.

Earlier, we compute the covariance matrix, the inverse of the covariance matrix denoted by $(\sum^j)^{-1}$. If the internal matrix is tantamount to the inverse covariance of the identical data set, of the sample μ^j, in this case the distance can be known as the Mahalanobis distance. The density of sample (i) represented by Y_i, can be calculated by Eq. (5):

$$\frac{1}{1+\frac{1}{\sigma^2_t}(z(k)-\mu^j)^T(\sum^j)^{-1}(z(k)-\mu^j)+\frac{1}{\sigma^2_t}\frac{(M-1)}{\mu^j}q} \quad (5)$$

Where σ^2_t refers to the square root of the first threshold.

8. The evolving mechanisms

In this model, three mechanisms are adopted:

8.1 Adding cluster

The first cluster was created based on the first data sample, then the density Y_i for each new data sample (newly arriving) is calculated to determine whether if attach it to on old cluster or create a new cluster for it based on comparison with Γ_{max}. Add a new cluster when Eq. (6) fulfilled:

$$Y_i \leq \Gamma_{max} \quad (6)$$

```
1: procedure adding cluster
2: arriving a data sample
3: compute its density Yi
4:      Repeat
5:          If this sample is the first data
6:              create the first cluster and make
                the sample as a center
7:          Else
8:              If Yi ≤ Γmax
9:                  generate a new cluster
10:                 set the initial cluster parameters
11:             Else
12:                 compute the density for all clusters
13:                 attach this sample to the cluster
                    has the minimum density
14:                 modify initial cluster parameters
        Until all data samples are examined
```

8.2 Splitting clusters

The model will check cluster by cluster in this mechanism. The concept of homogeneity of data samples within a cluster is used in the splitting mechanism. Simply, homogeneity means the degree of closeness of all data samples inside a cluster. Certainly, the model aims to achieve a high degree of homogeneity for each cluster. Therefore, the model searches carefully for clusters that have low convergence. If it is found, the model will implement the splitting mechanism in order to get clusters with high homogeneity samples.

Firstly, we determine a threshold for homogeneity which is (σ). The cluster homogeneity compares with a homogeneity threshold to determine whether the current cluster needs the splitting step or not. This mechanism is repeated in the model until the exemplary homogeneity is achieved. As a result, the mechanism outputs will be clusters have a good homogeneity.

```
1: procedure splitting clusters
2: set homogeneity threshold (σ)
3:      Repeat
4:          For each cluster
5:              compute the cluster homogeneity
6:                  If the cluster homogeneity ≤ σ
7:                      do splitting step
8:                  Else
9:                      go to step 4
10:         End For
11:     Until all clusters are checked
```

8.3 Merging clusters

Convergence is used to check the data samples in this mechanism. It means that the distance between a data sample of a cluster and its center must be less than the distance of this sample to other clusters' centers.

```
1: procedure merging clusters
2:     Repeat for each cluster
3:         For each data sample (x) in a cluster
4:             compute the distance from a sample
               to cluster center d_xc1
5:             compute the distance from a sample
               to all other cluster center d_xc1,2,..,n
6:               If d_xc2,3,..n ≤ d_xc1
7:                   merge a sample (x) with a
                     cluster_n
8:               Else
9:                   go to step 3
10:        End For
11:    Until all clusters are checked
```

This step is performed to check if there are data samples in a cluster may be closer to the center of another cluster than to the current one.

8.4 The evolving of clusters

The homogeneity of the data for a cluster is no less important than the final number of those clusters.

Since the process of evolving is the primary step in the proposed model, so the evolving will also be carried out on the number of clusters. More obviously, as long as the data within a single cluster are data samples of little homogeneity, the evolving will be repeated until all clusters have a high homogeneous.

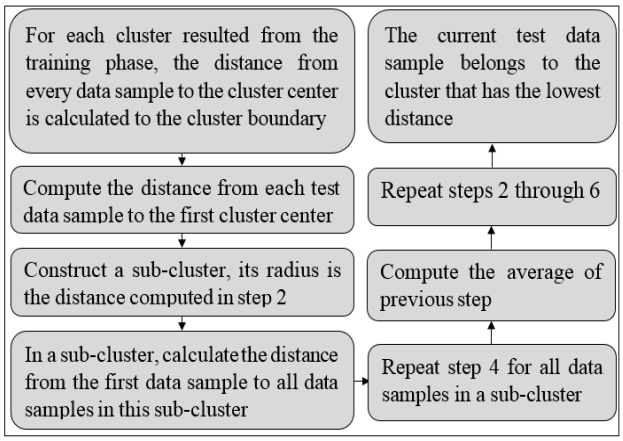

Figure. 2 The steps of distributing test data samples

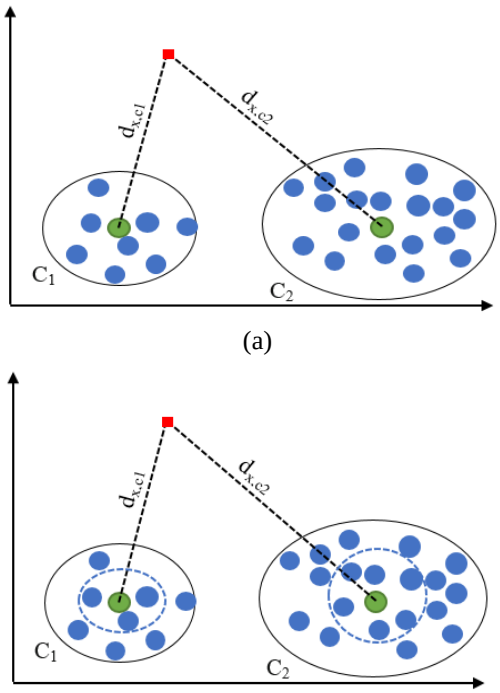

Figure. 3: (a) a new sample is arriving (b) construct a cluster boundary

9. The distribution of test data

The reconstructing clusters boundaries (in Fig. 1) or the distribution of test data consists of several steps, Fig. 2 illustrates these steps.

Briefly, the testing phase is the second step after the training. However, suppose we have two clusters C_1 and C_2.

Each one contains many data samples and every center is represented by data sample in green colour. When a new data sample (x) is arriving (red colour), the distance from it to all clusters is computed and then construct a sub-cluster (blue dashed lines). Fig. 3 (a) shows when a new sample arrived and (b) to construct a cluster boundary

10. The membership function

The membership function indicates the true degree of fuzzy logic. From a mathematics point of view, a membership function is a technique that implements for solving practical problems through experience instead of knowledge. Several membership functions are used by researchers such as Gaussian, trapezoidal, triangular and others [22]. Furthermore, the membership functions can be coupled with a conjunction operator, this situation is known as t-norms [23].

In order to complete the implementation of designed model perfectly, any previous membership function don't use. Alternatively, a new membership function is designed that proved to work well in the model. This membership function defined by Eq. (7).

$$F_{xcj} = \sum_{i=1}^{n} d\,(x, p_i)/n \quad (7)$$

where $p_i < d_{cj}$ and $F_{xcj=} \{f_{xc1}, f_{xc2},..,f_{xcj}\}$, j=1,2,…,n

The membership function that is specially designed for our model will determine and select the lowest (minimum) distance among all available values of (f_{xcj}). Thereupon, the cluster that has this minimum distance is selected by the model to append the test data sample (x) to it. After the repetition of the steps in section (9), now there is a value to every cluster, for example for the first cluster we have fxc_1 and for the second cluster f_{xc2} and so for the rest of the clusters. As a result, we now have a lot of different values for (n) clusters.

11. Data set description

The model is tested by two real data sets. The first data set is the NSL-KDD dataset, which was proposed in 2009 [24]. The main shortcoming in the KDDCUP'99 dataset has a massive number of redundant records, that led for many algorithms to biased towards those frequent records. Hence it will affect the final results of the model. Accordingly, and to overcome the imperfections that are described above, an enhancement version of the original KDD'99 dataset was presented, known as the NSL-KDD dataset. In the new dataset, each redundant record is omitted besides, all the records are re-balanced. As a result, the NSL-KDD dataset becomes more realistic and practical for evaluating algorithms. The connection types in KDD dataset classified into two classes normal and an attack. However, this dataset can be downloaded free.

The second dataset is the keystroke, which has been collected from 4 users who typed the same password during a period of time. It contains 1600 records and the main feature of this dataset is its ability to evolve according to the behaviour of its users [25]. Also, it can be downloaded free.

12. The evaluation

The final results contain many clusters with different properties. The final step in proposed model is the evaluation. In other words, these resulting clusters need to measure their quality [26]. Mainly the evaluation measurements can be classified into two types:

1- The internal measures, also known as unsupervised measures. It is also divided into two kinds, cluster cohesion and cluster separation.
2- The external measures, in contrast to the first type known as supervised measures.

Silhouette coefficient is one of the main measures of the clustering evaluation which belongs to the internal measures. The general idea of the silhouette coefficient is to compute the mean distance. The silhouette coefficient merges the cohesion and separation [27].

Mathematically, suppose we have [28]:

- K is a cluster which contains many data sample $x(i)$.
- $a_x(i)$ indicates the average distance from $x(i)$ to each data sample in the same cluster K.
- $b_x(i)$ represents the lowest average distance between $x(i)$ and each data sample in other clusters that isn't K.

Afterwards, the silhouette of the data sample x(i) can compute by Eq. (8):

$$S_{x(i)} = (b_{x(i)} - a_{x(i)})/\, max\,(a_{x(i)}, b_{x(i)}) \quad (8)$$

13. Experimental results and discussion

This section is devoted to discussing the implementation of the proposed model and the results that have been achieved. The core idea of the model is the evolving mechanisms therefore, the Silhouette Coefficient (SC) is computed before and after both of training and testing phases.

The max density threshold has influence in eCauchy algorithm, thus effects in the model result. Accordingly, many values of this threshold are tested during the model implementation.

For the first dataset (NSL-KDD), after several attempts, each of them has a certain threshold, Table 1 illustrates the results when the threshold is (0.0037). When analysing these results, it was noticed that there is a rise, then a decrease, and then a rise in (SC) of training progress in addition, (SC) of the test progress increased slightly.

Table 1. The threshold is (0.0037) to NSL-KDD data set

	Train progress	Test progress	Clusters number
Before evolving	-0.55	0.32	354
Epoch1	0.44		10
Epoch2	0.40		7
Epoch3	0.44		5
After evolving	0.44	0.35	5

Table 2 explains the results when the threshold is (0.0039). There is an instability in (SC) of training progress and also very simple increasing (SC) of test progress. Now threshold value sets to (0.0038), its results as follow, (SC) for the training phase before implementing the evolving is (–0.55), after the first epoch increases to (0.31) and after the second epoch achieves (0.46). In other words, the improvement in gradual values of the training phase is a stable and clear. Fig. 4 indicates these results.

While (SC) to the testing phase before implementing the evolving is (0.32) and after the implementation of the evolving increases to (0.43). Fig. 5 depicts these results. Initially, the number of generating clusters is 355 clusters, after the first epoch it reduces to 7 clusters and after the second epoch it reduces to only 5 clusters. Fig. 6 shows this change in the cluster numbers.

Finally, the model needed for (1.30) minutes to complete the execution to this dataset.

Table 2. The threshold is (0.0039) to NSL-KDD data set

	Train progress	Test progress	Clusters number
Before evolving	-0.54	0.03	560
Epoch1	0.10		4
Epoch2	0.20		3
After evolving	0.20	0.35	3

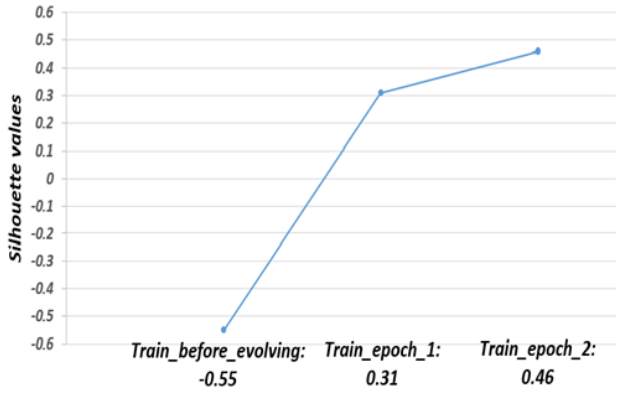

Figure. 4 SC for the training phase

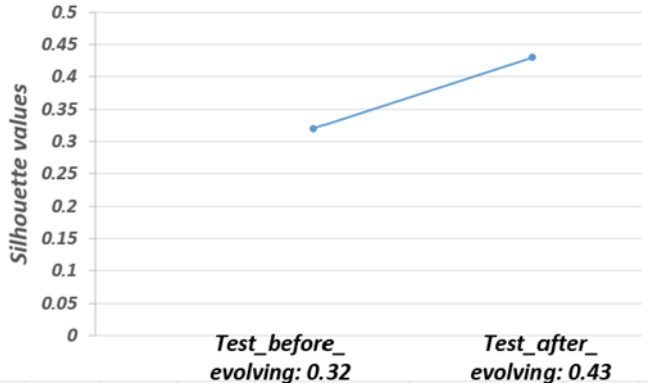

Figure. 5 SC for the testing phase

For the keystroke dataset, when the max_density threshold is set to (0.054) and (0.056) respectively, model gives the same results that shown in Table 3.

But when the max_density threshold sets to (0.055), then its results were as follows, (SC) to training phase before implementing the evolving is (–0.367), after the first epoch increases to (0.017), after the second epoch achieves (0.162) and at the third epoch it is (0.295). Fig. 7 indicates these results. While (SC) for the testing phase before implementing the evolving is (0.012) and after the implementation of the evolving increases to (0.189). Fig. 8 depicts these results.

Moreover, the number of generating clusters is 592 clusters, after the first epoch it reduces to 39 clusters, after the second epoch it decreases to only 8 clusters and it will be 4 clusters after the fourth epoch.

Table 3. The threshold sit to (0.054) and (0.056) to the keystroke data set

	Train progress	Test progress	Clusters number
Before evolving	-0.55	0.34	355
Epoch1	0.33		11
Epoch2	0.38		7
Epoch3	0.37		5
After evolving	0.37	0.35	5

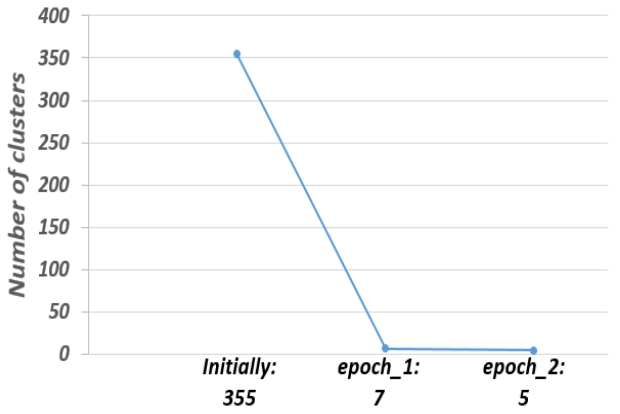

Figure. 6 The decrease in the number of clusters

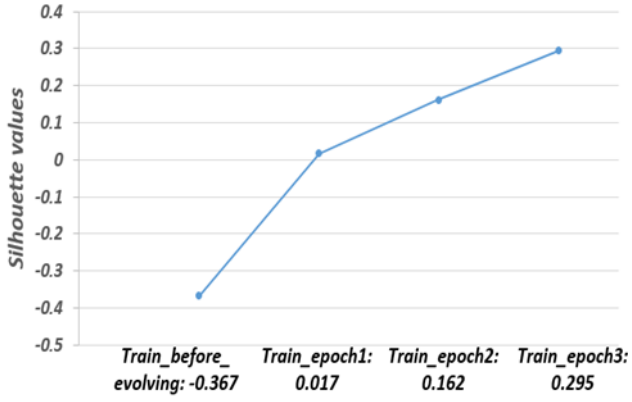

Figure. 7 SC for the training phase

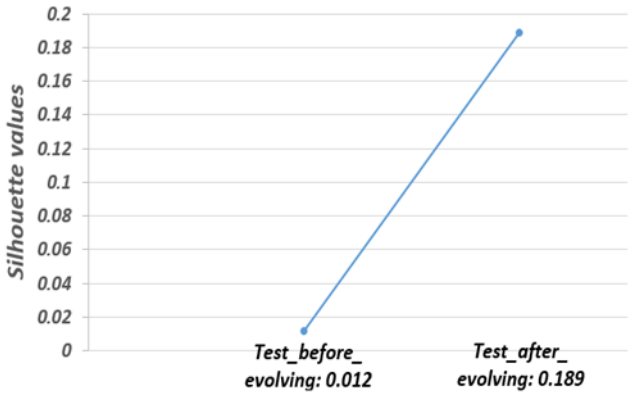

Figure. 8 SC for the testing phase

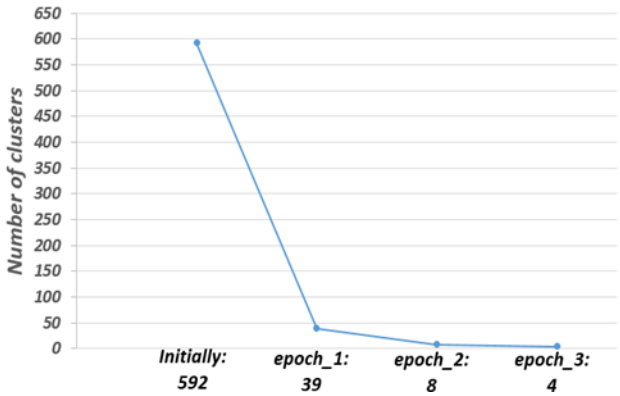

Figure. 9 The decrease in the number of clusters

Table 4. The difference in term of number of clusters

	Number of clusters	
	Before evolving	After evolving
e-Cauchy	1212	n/a
Proposed model	355	5

Fig. 9 illustrated these changes. Lastly, the model needed for (2.30) minutes to complete the execution to this dataset. The proposed model outperforms the e-Cauchy algorithm in term of the number of clusters during the test phase of KDD dataset, as shown in Table 4.

The proposed model is more efficient compared to FEAC-Stream algorithm, it built two clusters for the normal class in KDD dataset, whereas FEAC-Stream built 2-13 clusters for the same class.

14. Conclusion

The deficiency of methods that adopted evolving mechanisms, especially in the data stream environment, has become evident. The main shortcomings in some algorithm can be addressed by the evolving mechanisms model. The eCauchy algorithm belongs to the density-based method for data stream clustering, indeed it is considered one of the most recent algorithms of this method.

The large numbers of resulting clusters have remained a major obstacle in this algorithm. The proposed model was presented to overcome this limitation which based on optimizing e-Cauchy algorithm to reduce these numbers by implementing evolving mechanisms (adding, merging, splitting clusters) based on the fuzzy concept by applied a specific membership function.

The proposed model is tested using NSL-KDD99 and keystroke streaming datasets and is confirmed to be able for achieving an optimal number of clusters within short periods of time. The results proved that the proposed model outperforms two other algorithms that are e-Cauchy and FEAC-Stream, as it constructs 5 clusters for NSL-KDD99 dataset and four clusters for keystroke dataset in 1.30 and 2.30 minutes, respectively.

Conflicts of interest

The authors declare no conflict of interest.

Author contributions

The paper conceptualization, methodology, software, validation, formal analysis, investigation, resources, data curation, writing—original draft preparation have been done by 1st author. The writing—review and editing and visualization have been done by 2nd author. The supervision, project administration and funding acquisition have been done by 3rd author.

Acknowledgments

We would like to thank the University of Babylon and Babylon municipalities for providing adequate support and sponsorship for this work.

References

[1] H. L. Nguyen, Y. K. Woon, and W. K. Ng, “A survey on data stream clustering and classification”, *Knowledge Information Systems*, Vol. 45, No. 3, pp. 535–569, 2015.

[2] J. D. Andrade, 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.

[3] G. Kathiresan, K. Mohanta, and K. V. Asari, “Analyzing continuous data streams using improved stratified sampling and ensemble classification”, *International Journal of*

Intelligent Engineering and Systems, Vol. 11, No. 5, pp. 215–225, 2018.

[4] P. Kumar and A. Kanavalli, "A Similarity based K-Means Clustering Technique for Categorical Data in Data Mining Application", *International Journal of Intelligent Engineering and Systems*, Vol. 14, No. 2, pp. 43–51, 2021.

[5] U. Kokate, A. Deshpande, P. Mahalle, and P. Patil, "Data stream clustering techniques, applications, and models: Comparative analysis and discussion", *Big Data and Cognitive Computing*, Vol. 2, No. 4, pp. 1–30, 2018.

[6] A. O. C. Ayres and F. J. V. Zuben, "Multitask learning applied to evolving fuzzy-rule-based predictors", *Evolving Systems*, Vol. 12, No. 2, pp. 1-16, 2019.

[7] N. Kasabov and D. Filev, "Evolving Intelligent Systems: Methods, Learning, & Applications", In: *Proc. of International Conf. On Evolving Fuzzy Systems*, Ambelside, UK, pp. 8–18, 2006.

[8] E. Lughofer, *Evolving Fuzzy Systems – Methodologies, Advanced Concepts and Applications*, Vol. 266, Springer, Berlin, 2011.

[9] L. Hartert, M. S. Mouchaweh, and P. Billaudel, "A semi-supervised dynamic version of Fuzzy K-Nearest Neighbours to monitor evolving systems", *Evolving Systems*, Vol. 1, No. 1, pp. 3–15, 2010.

[10] R. E. Precup, T. A. Teban, A. Albu, A. I. S. Stinean, and C. A. B. Dragos, "Experiments in incremental online identification of fuzzy models of finger dynamics", *ROMANIAN JOURNAL OF INFORMATION SCIENCE AND TECHNOLOGY*, Vol. 21, No. 4, pp. 358–376, 2018.

[11] D. Dovžan, V. Logar, and I. Škrjanc, "Implementation of an Evolving Fuzzy Model (eFuMo) in a Monitoring System for a Waste-Water Treatment Process", *IEEE Transactions on Fuzzy Systems*, Vol. 23, No. 5, pp. 1761–1776, 2015.

[12] M. Pratama, S. G. Anavatti, P. P. Angelov, and E. Lughofer, "PANFIS: A novel incremental learning machine", *IEEE Transactions on Neural Netwoeks and Learning Systems*, Vol. 25, No. 1, pp. 55–67, 2014.

[13] N. A. Supardi, S. J. Abdulkadir, and N. Aziz, "An Evolutionary Stream Clustering Technique for Outlier Detection", In: *Proc. of International Conf. On Computational Intelligence (ICCI)*, Bandar Seri Iskandar, Malaysia, pp. 299-304, 2020.

[14] J. Gama, *Knowledge Discovery from Data Streams*, Chapman & Hall CRC Press, Atlanta, USA, 2010.

[15] A. Abdullatif, F. Masulli, and S. Rovetta, "Clustering of nonstationary data streams: A survey of fuzzy partitional methods", *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, Vol. 8, No. 4, pp. 1-18, 2018.

[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.

[17] I. Skrjanc, S. Ozawa, T. Ban, and D. Dovˇzan, "Large-scale cyber attacks monitoring using Evolving Cauchy Possibilistic Clustering", *Applied Soft Computing*,Vol. 62, pp. 592–601, 2018.

[18] X. Gu and P. P. Angelov, "Self - Boosting First - Order Autonomous Learning Neuro - Fuzzy Systems", *Applied Soft Computing*, Vol. 77, pp. 118-134, 2019.

[19] G. Wang and Q. Song, "Automatic clustering via outward statistical testing on density metrics", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 28, No. 8, pp. 1971–1985, 2016.

[20] D. Leite, I. Škrjanc, and F. Gomide, "An overview on evolving systems and learning from stream data", *Evolving Systems*, Vol. 11, No. 2, pp. 181–198, 2020.

[21] X. Bei, N. Chen, and S. Zhang, "On the complexity of trial and error", In: *Proc. of International Conf. On Theory of Computing (STOC'13)*, Palo Alto, California, USA, pp. 31-40, 2013.

[22] B. Belhadj and F. Kaabi, "New membership function for poverty measure", *Metroeconomica*, Vol. 71, No. 4, pp. 676–688, 2020.

[23] B. I. J. Lamarca and S. C. Ambat, "The development of a performance appraisal system using Decision Tree analysis and Fuzzy Logic", *International Journal of Intelligent Engineering and Systems*, Vol. 11, No. 4, pp. 11–19, 2018.

[24] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set", In: *Proc. of International Conf. On Computational Intelligence in Security and Defense Applications (CISDA)*, Ottawa, Canada, pp. 1-6, 2009.

[25] V. M. A. Souza, D. M. D. Reis, A. G. Maletzke, and G. E. A. P. A. Batista, "Challenges in benchmarking stream learning algorithms with real-world data", *Data Mining and Knowledge Discovery*, Vol. 34, No. 6. pp. 1805-1858, 2020.

[26] N. Nidheesh, K. A. A. Nazeer, and P. M. Ameer, "A Hierarchical Clustering algorithm based on

Silhouette Index for cancer subtype discovery from genomic data", *Neural Computing and Applications*, Vol. 32, No. 15, pp. 11459–11476, 2019.

[27] E. Rendón, I. M. Abundez, C. Gutierrez, S. D. Zagal, A. Arizmendi, E. M. Quiroz, and E. Arzate, "A comparison of internal and external cluster validation indexes", In: *Proc. of International Conf. On Applications of Mathematics and Computer Engineering (5th WSEAS)*, San Francisco, CA, USA, pp. 158–163, 2011.

[28] N. Kaoungku, K. Suksut, R. Chanklan, K. Kerdprasop, and N. Kerdprasop, "The silhouette width criterion for clustering and association mining to select image features", *International Journal of Machine Learning and Computing*, Vol. 8, No. 1, pp. 69–73, 2018.