International Journal of Electrical and Computer Engineering (IJECE)
Vol. 9, No. 4, August 2019, pp. 2659~2667
ISSN: 2088-8708, DOI: 10.11591/ijece.v9i4.pp2659-2667

An improved differential evolution algorithm for data stream clustering

Bhaskar Adepu[1], Jayadev Gyani[2], G. Narsimha[3]
[1]Department of Information Technology, Kakatiya Institute of Technology & Science, India
[2]Department of CS, College of Computer & Information Sciences, Majmaah University, Saudi Arabia
[3]Department of CSE, JNTUH College of Engineering, India

Article Info

Article history:

Received Jun 27, 2018
Revised Jan 9, 2019
Accepted Jan 11, 2019

Keywords:

Concept drift
Datastream clustering
Differential evolution
Encoding scheme
Entropy theory

ABSTRACT

A Few algorithms were actualized by the analysts for performing clustering of data streams. Most of these algorithms require that the number of clusters (*K*) has to be fixed by the customer based on input data and it can be kept settled all through the clustering process. Stream clustering has faced few difficulties in picking up *K*. In this paper, we propose an efficient approach for data stream clustering by embracing an Improved Differential Evolution (IDE) algorithm. The IDE algorithm is one of the quick, powerful and productive global optimization approach for programmed clustering. In our proposed approach, we additionally apply an entropy based method for distinguishing the concept drift in the data stream and in this way updating the clustering procedure online. We demonstrated that our proposed method is contrasted with Genetic Algorithm and identified as proficient optimization algorithm. The performance of our proposed technique is assessed and cr eates the accuracy of 92.29%, the precision is 86.96%, recall is 90.30% and F-measure estimate is 88.60%.

Copyright © 2019 Institute of Advanced Engineering and Science.
All rights reserved.

Corresponding Author:

Bhaskar Adepu,
Kakatiya Institute of Technology & Science,
Affiliated to Kakatiya University,
Warangal, 506015-India.
Email: bhaskar_adepu@yahoo.com

1. INTRODUCTION

Nowadays, datastreams becomes an important source of data. In recent years, multiple organizations generate huge amounts of data. Data stream application domains includes information analysis in network data flow monitoring, Internet of Things (IoT) applications regularly sending sensors data, web page access and web click information, weather forecasting information and the economic information produced by finance and securities companies and so on [1-5]. Conventional data mining methods mostly focused on mining static and memory resident data repositories. However, with the emergence of data streams and technological developments changed the way people store, process and communicate the data [6]. Data streams are temporarily ordered, fast changing, infinite and massively potential. It may be not possible to store the entire data stream into memory [7]. Stream mining has to deal with rapid and dynamic data with real time processing and aiming at extracting useful and interesting patterns. The biggest challenge is finding valuable information in a single scanning of massive data streams [7, 8].

Various algorithms and procedures proposed for mining data streams can be grouped into two groups of techniques. One group ofalgorithms can achieve desired clustering results, but insufficiency of data storage capacity which leads us to process data dynamically in extracting knowledge. Another group of algorithms refers to streaming of data and applies mining techniques [9]. These two groups of techniques express some difficulties in clusterin data streams.Some of the difficulties includes: visiting of data once

during the processing of data stream, performance of processing stream is crucial and detecting a change or a concept drift during the time whether gradual or abrupt in the evolutionary data stream is difficult [10]. In the area of data stream clustering, multiple techniques have been proposed. They are K-means clustering algorithm [11-13], Fast Evolutionary Algorithm for Clustering (FEAC) [14], A Support Vector Clustering (SVC) based algorithm [15, 16], Multiclass Novelty Detection (MND) algorithm [17, 18], Fuzzy C-Mean algorithm [1, 19], BIRCH algorithm [20], KNN (K-Nearest Neighbor) algorithm [8].

For clustering data in many real world problems, K-means algorithm is popularly used because of its simplicity and scalabilityof that algorithm for most of the real world applications. K-means algorithm has a limitation that we have to specify the number of clusters i.e. k-value as an input parameter to the algorithm [11-13]. For automatically estimating the number of clusters (i.e.k-value) from the input data, there is an algorithm known as Fast Evolutionary Algorithm for Clustering (FEAC) [14] hasshown to be efficient. The limitation of this algorithm was that it is not applied for data stream analysis. Support Vector Clustering (SVC) algorithm is an efficient and effective data stream clustering algorithm. The disadvantage of this algorithm isthat it cannot find arbitary shape clusters because most of these algorithms are based on K-means algorithms [15, 16]. Multiclass Novelty Detection algorithm can be used in various applications like intrusion, fault and fraud detections, spam filters and in text mining [17, 18]. BIRCH is a hierarchical clustering algorithm which is based on calculating the distance between the new data point and the remaining known datapoints. After that, compare these distances with a threshold to determine the category of new data point. The limitation of this algorithm is that it does not function effectively for the data with arbitrary shape [20]. KNN algorithm is a skewed approach and is widely used method forsolving classification and pattern recognition problems in machine learning.KNN algorithm is used to avoid the high computational complexity even though we could not get a satisfactory performance in many applications [19, 21-24].

In this paper, we propose an Improved Differential Evolution algorithm (IDE) for the data stream clustering. At first, from the input data streams, the nearest cluster center is assessed for every incoming object and the clusters are updated. Around then any concept drift occurs, the approaching objects are put in the buffer till a settled time period. From that point onwards, use the IDE-based optimization for finding the optimal *K* value. On the off chance that any concept drift occurs, the underlying advances are rehashed.

The rest of this paper is portrayed in the segment underneath. The proposed method is delineated in section 2. The overwiew of the IDE algorithm is explained in section 2.2. Our proposed IDE Stream algorithm is delineated in section 2.7. Results and the conclusion were depicted in sections 3 and 4.

2. PROPOSED METHOD

Consider the data stream which consists of N number of objects and each object is an *l*-dimensional feature vector $\mathbf{x}_i=[x_i^j]$ where j=1 to *l* and 1<= *i*<=N. Initially the number of clusters $k \in \{2, \cdots k_{max}\}$ and the centroids are randomly selected from the given datastream. The normal distance between closest cluster center and the underlying objects are evaluated and updated. The evaluated cluster centers are updated and this procedure is rehashed until the point when some ceasing rule is met. At the mean time, if any concept drift occurs, it utilizes the entropy theory which is presented in the section 2.1.

Our proposed IDE algorithm doesn't require the streaming module to store the data for outline which utilizes the clustering module or module for estimation of k-value for apportioning the data. Here, the updated apportioning data are kept up by utilizing a single component; it is done in the online mode. Distinctive quantities of trail arrangements are utilized for our approach, which has diverse cluster centers and its coordinates. The best answer for the updated clustering data is chosen and the most noticeably awful arrangements are disposed of in the light of the IDE algorithm. In the data stream clustering process, just a single object is touching base at once. At first, the quantity of clusters is assessed by IDE Stream and the each evaluated cluster measure it ought to be equivalent to the underlying size objects from the stream. At that point, the assessed cluster is kept up in an online manner, subsequent to building up the underlying cluster.

The procedure of IDE Stream algorithm has appeared in Figure 1. At the underlying phase of the data stream processing, the number of clusters is randomly selected from the information of data stream. The normal distance between closest cluster center and the underlying objects are evaluated and updated. With specific goal to recognize changes in the data partition, the clusters are administered by the entropy theory. At the point when the entropy test triggers an alarm, that is the point at which the real clusters being updated don't mirror the adjustments in the data streams and these clustered objects are put in the buffer for some time until any concept drift or change occurs. At that point, the IDE scan is begun for optimization and it is dealt in the encoding scheme.

2.1. Entropy theory for detecting concept drift

At the time of partitioning the data, the data distribution may change over time in unforeseen ways, this problem is known as concept drift. Theconcept drift is going on, while concept evolution appearance to vanishing of clusters. To recognize the concept drift, the accompanying speculations are considered for the entropy theory. Here, just a single idea is holed by the data stream. Because of this reason, the concept drift isn't happening in light of the fact that the data stream is steady. On the off chance that the data stream does not have any concept drift, the time series of the entropy strategy is about zero and stays stationary. In the event that the data stream has any concept drift, the entropy is can't expand on the last idea, it just coordinates with the underlying one. That implies, the precision judgment of the back to back data is can't control the framework, if the updating is not finished. Because of this reason, the discovery of concept drift is critical. Thus, the entropy estimation is established in the light of the membership esteems and it is critical for identifying the concept drift.

In our method shannon's entropy strategy is utilized for the entropy estimation. Here, the discrete random variables are considered as X and the conceivable esteems are considered as: $\left\{X_{i}^{l}, X_{i+1}^{l}, X_{i+2}^{l}, \cdots, X_{i+d}^{l}\right\}$ and the probability mass function is $P(X)$. The entropy $E(X)$ is calculated for every random variable X using the following equation.

$$E(X) = -\sum_{i=1}^{n} P_i(X) \log(P_i(X)) \quad (1)$$

The non-consistency of the given cluster is assessed in view of the entropy measure. For the all-out data clustering, the entropy theory is a proficient technique. On the off chance that the entropy esteems are high, the vulnerability of the IDE is bigger. The time series of the classification entropy points is almost zero if the data stream does not contain any concept drift, generally, the esteem will turn out to be extensive.

2.2. Overview of IDE Stream algorithm

Figure 1 shown the procedure of IDE Stream algorithm.

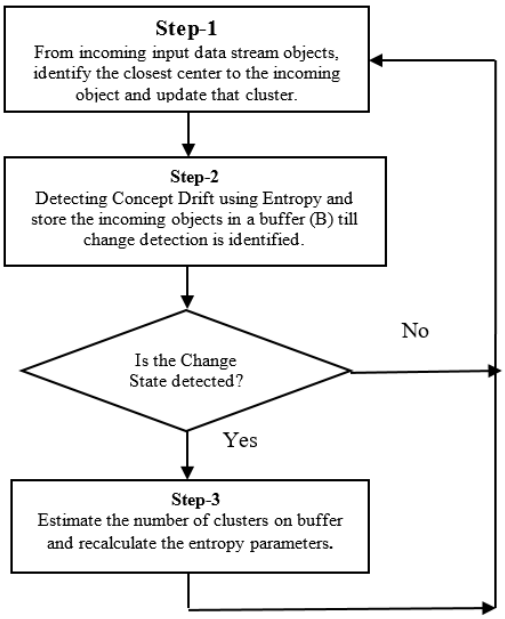

Figure 1. Process of IDE Stream algorithm

2.3. Buffer for IDE Stream

In the wake of recognizing the concept drift, the genuine clusters are updated, and these outdated clusters are put away in buffer for some time to run the IDE algorithm. Here, the base size of the buffer is 10%×initial size. For the computational assets, just the base size of the buffer is accessible. To characterize the warning and the alarm states, there are two threshold values, specifically ϕ_w and ϕ_a are utilized

(i.e. $\phi_a > \phi_w$). Right when the alarm threshold value ϕ_a isn't as much as the estimation of the contrast between the two variables, the stationary state is triggered. Starting now and into the foreseeable future, to address the data partition and recognize the optimized clusters, the encoding scheme is executed to run the IDE algorithm. For the efficient and robust optimization process, the IDE algorithm is an efficient and well known algorithm which is a population based algorithm. The floating point representations are used in this algorithm.

2.4. Encoding scheme

For the clustering problems, the candidate solutions (individuals) are described by the encoding scheme of [14] is adopted by the IDE-stream algorithm. A data stream X is a significant arrangement of illustrations and it is given as follow:

$$X_i^l(t) = \{x_i^l, x_{i+1}^l, x_{i+2}^l, \cdots, x_{i,d}^l(t)\} i.e., X = \{x_i^j\}_{j=1}^{l} \quad (2)$$

where $i \Longrightarrow$ Incoming object $l \Longrightarrow$ Number of attributes in each object.

The above equation is potentially unbounded $(N \to \infty)$. Each case is depicted by a n-dimensional attributes vector $X_i = [x_i^l]_{l=1}^{n}$. At that point, the input data points (D) are partitioned into k number of non-overlapping clusters $C = \{C_1, C_2, \cdots, C_k\}$ such that satisfying the equation

$$C_i \neq \phi \; ; \; C_i \cap C_j = \phi, \quad i, j = 1,2,\cdots,k \;\; i \neq j \; ; \; \bigcup_{i=1}^{k} C_i = D \quad (3)$$

The above expression clarified, that the objects in the similar clusters are like each other, and the objects in the diverse clusters are disparate. In the data set, the closeness and disparity between the objects are found by evaluating the Euclidean distance between the points $X_i = [x_i^l]_{l=1}^{n}$. The above partitioned data C is changed into the integer string of N positions by using encoded process. Here, the position of every string and the numerical orders for the objects of the datasets are almost similar. The example encoding scheme of the dataset is shown in Figure 2.

1	***1***	***1***	***1***	***3***	***1***	***2***	***3***	***3***	***2***	***3***
O_1	O_2	O_3	O_4	O_5	O_6	O_7	O_8	O_9	O_{10}	***K***

Figure 2. Encoding scheme

In the above example, the dataset consists of ten objects *i.e.* $O_i, i = \{1,\cdots,10\}$ and the encoding clusters are three $(k = 3)$. Here, the i[th] object of the dataset is stored in the i[th] position and the k-value is stored in the last position.

Here, the feature vectors are used to describe the every cluster and the number of objects N, data object's linear sum called S_1 and their squared sum called S_2 and time 't' of the most recent objects that the cluster received are the four genuine quantities of measurement. The centroid of the cluster is evaluated by using the initial three components and the importance of the cluster is weighted by the rest of the component. At time 'T', the cluster weight 'W' is evaluated as follows:

$$W = e^{-\frac{T-t}{v}} \quad (4)$$

where W is represented as the weight of the cluster, v is represented as the user defined parameter used to control the fading factor. When the weighting value of the cluster is less than 0.1 (i.e. W<0.1), that cluster is removed from the data partition.

2.5. Validity indices of the clusters

The validity index is utilized to evaluate the number of clusters. The evolutionary algorithm optimizes the k-value based on the fitness function and it is estimated by using the validity index. The well knownvalidity indexused in our method is a Simplified Silhouette (SS) index which is used in [5, 14]. The compactness and the partition of the clusters are assessed by utilizing the Silhouette width. The given arrangement of the data point is given as takes after,

$$X_i^l = x_i^1, x_i^2, x_i^3, \cdots, x_{i,j}^l \quad (5)$$

where in the above equation, $i = 1, 2, 3 \cdots N$ the number of objects in the partition and $j = 1, 2, 3 \cdots K$ the number of cluster's range variations.The above considered data points X_i^l comprises the cluster C_a. In C_a, the alternate objectives are distinguished in view of the X_i^l difference capacity and it is meant by $a(X_i^l)$. In cluster C_a, the normal divergence capacity of every single other object is indicated as $d(X_i^l, C_b)$. The minimum dissimilarity function of the data points is selected, if the cluster $C_b \neq C_a$, which is given beneath,

$$a(X_i^l) = dis\left\{X_i^l, centroid(j=1)\right\} \quad (6)$$

$$b(X_i^l) = \min\left\{dis\ X_i^l,\ centroid(j \neq i)\right\} \quad (7)\backslash$$

After computing the dissimilarity function, the silhouette estimation $S(X_i^l)$ is given as follows,

$$S(X_i^l) = \frac{b(X_i^l) - a(X_i^l)}{\max\left\{a(X_i^l), b(X_i^l)\right\}} \quad (8)$$

The silhouette values are only ranges between the interval [0, 1], the closest cluster values are accessed according to the equation (11). At that point, when the silhouette value is nearer to 1, it represents X_i^l is clustered precisely, generally the data points are wrongly clustered. The overall silhouette index of the portioning cluster $C = \left\{C_1, C_2, \cdots, C_k\right\}$, is given as:

$$SS = \frac{1}{N}\sum_{i=1}^{N} S(X_i^l) \quad (9)$$

In the above equation the maximum value of SS $(SS_{\max})$ is known as the fitness function of the object in the cluster, which is utilized to determine the quality of the partitioning data. Here, which cluster has the maximum SS value, that is considered as the best clustering and then the evolutionary search is started.

2.6. Innovative evolutionary search

In our method, the DE algorithm is modified based on the adjustment of population X, mutation and crossover CR estimations, because the best optimization results are obtained through the crossover and the mutation rate.

In our IDE implementation, initially the candidate solutions $\{x_i^l, x_{i+1}^l, x_{i+2}^l, \cdots, x_{i,d}^l(t)\}$ are generated from the initial population X and it is shown in section 2.4 as the encoding scheme. Then the silhouette index $SS_{\max}$ is evaluated for each individual to know the fitness and this estimation is shown in section 2.5. After that, the other objects $x_i^l(t),\ x_j^l(t)$ and $x_p^l(t)$ are randomly generated from the initial generated candidate solution. Then the difference between $x_i^l(t)$ and $x_j^l(t)$ are estimated and the estimated difference values are scaled by scalar S, it is represented as $S \in [0, 1]$. Here, the scaled value of the two

random vector's weighted difference values $O(X^{l}_{i,u}(t) - X^{l}_{j,u}(t))$ is added to the third vector $x^{l}_{p}(t)$ to generate the new vector $O^{l}_{i}(t+1)$, it is known as the mutation and it is given as follows:

$$O_{k,u}(t+1) = \begin{cases} if \quad rand_u[0,1] < CR, \\ X_{p,u}(t) + O(X^{l}_{i,u}(t) - X^{l}_{j,u}(t)) \\ otherwise, \quad X_{k,u}(t) \end{cases} \tag{10}$$

Then the other predetermined vector is mixed with the mutated vector, it is known as the crossover rate (CR). It is the scalar parameter of the algorithm and in between 0 and 1, i.e., $CR \in [0,1]$. Atlast, assess the new candidate solution with the $SS_{max}$and then the initial candidate solution $X^{l}_{i}(t)$ is substituted in the new candidate solution $X^{l}_{i}(t+1)$. Here, if the newly generated candidate solution yields the maximum objective function, these solutions are considered as the best solution otherwise the solution is retained in the population; it is depicted in the equation below,

$$X^{l}_{i}(t+1) = \begin{cases} if \quad S(O_i(t+1)) > S(X^{l}_{i}(t)), \quad O_i(t+1) \\ if \quad (O_i(t+1)) \le S(X^{l}_{i}(t)), \quad X_i(t) \end{cases} \tag{11}$$

In the above equation, S(.) is the objective equation. To enhance our IDE calculation, we have enhanced the properties of IDE in different ways. To scale the weighted contrast vector, $X^{l}_{i,u}(t) - X^{l}_{j,u}(t)$, the scaling factor S is using and it extends in the range of 0.5 and 1.

2.7. Pseudo code of IDE Stream algorithm

Step 1 : Generate the candidate solutions from the initial population(X) i.e.the individual points $x^{l}_{1}, x^{l}_{2} \cdots, x^{l}_{n}$ are randomly generated from the data sets.

Step 2 : CalcualteSS_{max}foreach individual. [Use eq.9]

Step 3 : Randomly choose three objects $X^{l}_{i}(t)$, $X^{l}_{j}(t)$, $X^{l}_{p}(t)$ from the initially generated candidate solutions.

Step 4 : Estimate the difference between any two objects and scale it in the range [0, 1] and add this value to third object to generate a new object. [Use eq.10]

Step 5 : Perform crossover by mixing this mutated object resulted from step4 with predefined object such that crossover rate is in the range [0,1].

Step 6 : Assess the newly generated candidate solution and output the best result. [Use eq.11]

Repeat step 2 to step 6 until stopping criteria is met.i.e. K<=SS_{max}

3. RESULTS AND ANALYSIS

The performance of the IDE algorithm is experimentally assessed by contrasting and the use of late created optimization algorithm, to be specific genetic algorithm [25]. The primary objective of this calculation is, to optimize the candidate solution to produce the best outcome. The candidate solutions are picked relies upon the fitness function, the nature of the candidate solutions is assessed as for the optimization issue. The principle favorable circumstances of this algorithm are, it can deal with a few competitor arrangements all the while. All things considered, in numerous handy applications, the rough assurance of the data set is unthinkable. In our technique, if any progressions are happening in the season of DE based clustering, which is contrasted and the IDE algorithm for utilizing a similar objective portrayal and the silhouette index of the IDE.

3.1. Datasets description

We used three datasets and they are KDD CUP'99 [4],[5], forest cover type [5],[26], electric power consumption dataset [26]. We compared the accuracy, precision, recall and F-Measures on these datasets.

(i) KDD CUP'99 Dataset: KDD CUP'99 is the first real data, we use 10% version of detection set, and this version is more challenging and more concentrated than the full version. 34 continuous attributes are used in this data set which contains 23 different classes of data and 494,020 connection records. Each object is either a normal connection or an intrusion. Popular minimum-maximum procedure is used for normalizing the dataset attribute values within [0, 1] before running the experiment.

(ii) Forest Cover Type Dataset: Forest cover type is the second real dataset, which contain 7 forest cover type of 581,012 geospatial descriptions. Totally 54 attributes are described in objectsout of which 10 are quantitative attributes and 44 are binary attributes. These 10 quantitative attributes are widely used in many experiments as reported in the literature. Similar to the KDD CUP'99, Popular minimum-maximum procedure is used for normalizing the dataset attribute values within [0, 1] before running the experiment. By taking the data input, the data set is converted to a data stream.

(iii) Electric Power Consumption Dataset: Electricity dataset is the third real dataset, which contains 2,075,259 number of instances and 9 different attributes. The main characteristic of this dataset is multivariate and time series. The attribute values are normalized within [0, 1]. This datasetgave better results for each activation function and configuration. It performed better than the other datasets.

3.2. Evaluation measures

The accurate k-values are predicted by using clustering oriented approach. In our method, we use four important evaluation metrics namely precision, accuracy, recall and F-measure for our experiments. In a cluster group, the accurate grouped data percentage is measured by the cluster purity. The online components generate the micro clusters and the quality of this cluster is evaluated based on this cluster purity. The accuracy evaluation is utilized to detect the correctly assigned class, based on this accuracy, the cluster purity is evaluated. For stream clustering, the precision, recall, accuracy and F-measure are used as evaluation measures. The ratio between the number of retrieved relevant samples and the total number of grouped samples is known as the precision, which is utilized to determine the accurate clustering results of the clustering method. The ratio between the number of retrieved relevant samples and the total number of samples is known as accuracy. The contribution of the precision and recall is known as an F-measure. The accuracy, precision, recall and F-measure are given as follows:

$$\text{Accuracy} = \frac{\text{TP}}{\text{TP+FP+FN}} \tag{12}$$

$$\text{Precision (P)} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{13}$$

$$\text{Recall (R)} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{14}$$

$$\text{F} - \text{measure} = 2\left(\frac{\text{P.R}}{\text{P} + \text{R}}\right) \tag{15}$$

In the above equations, TP is the true positive, TN is the true negative, FP is the false positive and FN is the false negative samples. Table 1 shows the results of evaluation mesures computed on three different data sets. This shows that our proposed IDE Stream algorithm performs better compared with Genetic Algorithm (GA). Following graphs represents comparison of our proposed IDE Stream algorithm with genetic algorithm in four evaluation measures on three datasets as shown in Figure 3.

Table 1. Evaluation measures on three datasets

Data Sets	Evaluation Measures							
	Precision		Recall		Accuracy		F_Measure	
	GA	Proposed IDE	GA	Proposed IDE	GA	Proposed IDE	GA	Proposed IDE
1. KDD CUP'99	0.7815	0.8631	0.8134	0.9023	0.8634	0.9149	0.7972	0.8823
2. Foreset Cover Type	0.8117	0.8657	0.8430	0.9056	0.8452	0.9314	0.8271	0.8852
3. Electric Power Consumption	0.8174	0.8802	0.8475	0.9012	0.8819	0.9225	0.8321	0.8906
Average value	0.8035	0.8697	0.8346	0.9030	0.8635	0.9229	0.8188	0.8860

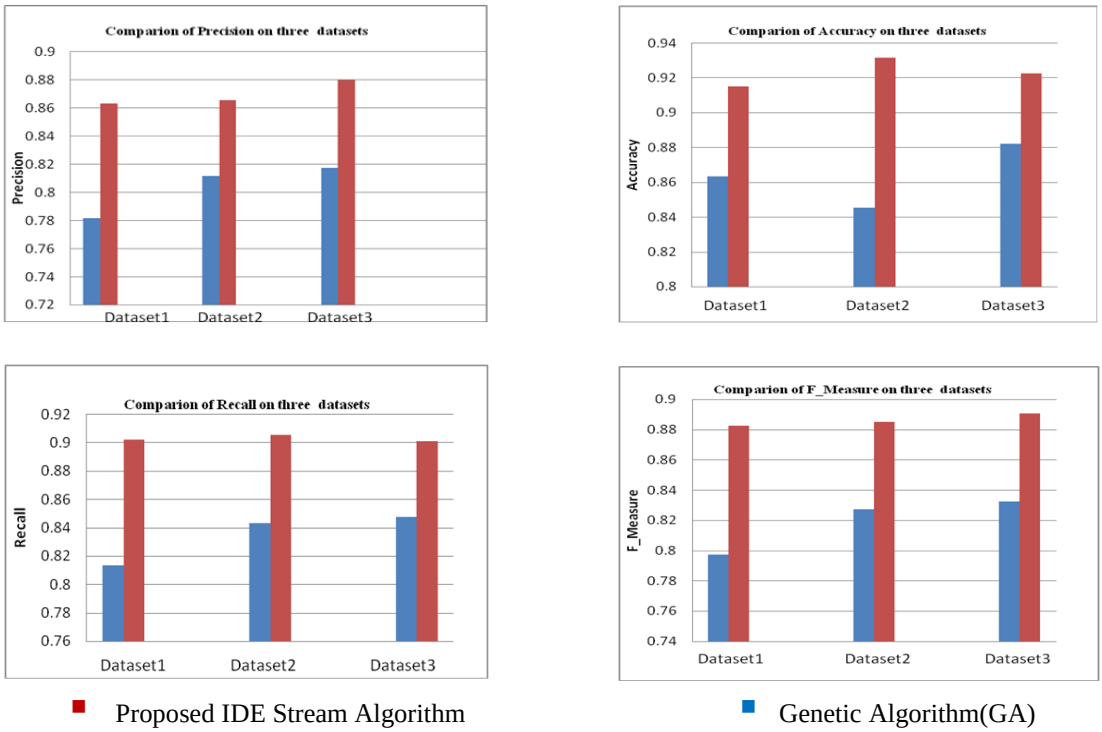

Figure 3. The evaluation of comparison between proposed IDEStream algorithm and genetic algorithm

4. CONCLUSION

In this paper we presented an IDE Stream algorithm for automatic clustering of stream data. Primary contributions of this paper are to detect the optimal number of clusters automatically for all data sets. Our testing data sets are taken from the KDD CUP'99, forest cover types, and electric power consumption datasets. Our proposed IDE algorithm is more efficient than the existing well known clustering algorithms such as Genetic algorithm. In our proposed approach, we also apply an entropy based technique for detecting the concept drift thereby updating the clustering process. Our experimental results show that the proposed method is simple, practical and impactful. The performance of our method is estimated based on the accuracy, precision, recall and F-measure values. In the future work, we could study how to enhance the system to improve the precision. In future we will implement this IDE algorithm for different types of attributes in the clustering domain.

REFERENCES

[1] B. Zhang, et al., "Data stream clustering based on Fuzzy C-Mean algorithm and entropy theory," *Signal Processing*, pp. 111-116, 2015.
[2] J. Barddal, et al., "SNCStream+: Extending a high quality true anytime data stream clustering algorithm," *Journal of Information Systems*, vol. 62, pp. 60-73, 2016.
[3] M. Hosseini, et al., "An ensemble of cluster-based classifiers for semi-supervised classification of non-stationary data streams," *Knowledge and Information Systems*, vol/issue: 46(3), pp. 567-597, 2015.
[4] M. Khalilian, et al., "Data stream clustering by divide and conquer approach based on vector model," *Journal of Big Data*, vol/issue: 3(1), 2016.
[5] J. A. Silva, et al., "An evolutionary algorithm for clustering data streams with a variable number of clusters," *Expert Systems with Applications*, vol. 67, pp. 228-238, 2017.
[6] E. de Faria, et al., "Evaluation of Multiclass Novelty Detection Algorithms for Data Streams,' *IEEE Transactions on Knowledge and Data Engineering*, vol/issue: 27(11), pp. 2961-2973, 2015.
[7] C. Wang, et al., "SVStream: A Support Vector-Based Algorithm for Clustering Data Streams," *IEEE Transactions on Knowledge and Data Engineering*, vol/issue: 25(6), pp. 1410-1424, 2013.
[8] D. Adeniyi, et al., "Automated web usage data mining and recommendation system using K-Nearest Neighbor (KNN) classification method," *Applied Computing and Informatics*, vol/issue: 12(1), pp. 90-108, 2016.
[9] J. Zhang, et al., "Distributed data stream clustering algorithm based on affinity propagation," *Journal of Computer Applications*, vol/issue: 33(9), pp. 2477-2481, 2013.

[10] K. Guo and Q. Zhang, "Fast Clustering-Based Anonymization Algorithm for Data Streams," *Journal of Software Engineering*, vol/issue: 24(8), pp. 1852-1867, 2014.
[11] R. Hyde, et al., "Fully online clustering of evolving data streams into arbitrarily shaped clusters," *Journal of Information Sciences*, vol. 382-383, pp. 96-114, 2017.
[12] N. Chaturvedi, et al., "An Improvement in K-mean Clustering Algorithm Using Better Time and Accuracy," *International Journal of Programming Languages and Applications*, vol/issue: 3(4), pp. 13-19, 2013.
[13] M. Naldi and R. Campello, "Evolutionary k-means for distributed data sets," *Neuro Computing*, vol. 127, pp. 30-42, 2014.
[14] V. S. Alves, et al., "Towards a Fast Evolutionary Algorithm for Clustering," *IEEE congress on evolutionary computation*, IEEE Press, pp. 1776-1783, 2006.
[15] Y. Ping, et al., "Fast and scalable support vector clustering for large-scale data analysis," *Knowledge and Information Systems*, vol/issue: 43(2), pp. 281-310, 2014.
[16] E. de Faria, et al., "MINAS: multiclass learning algorithm for novelty detection in data streams," *Data Mining and Knowledge Discovery*, vol/issue: 30(3), pp. 640-680, 2015.
[17] P. D. and A. Dixit, "Multi Novel Class Classification of Feature Evolving Data Streams with J48," *International Journal of Computer Applications*, vol/issue: 124(11), pp. 31-36, 2015.
[18] Y. Han, "Improved BIRCH Clustering Algorithm and Human Resource Management Efficiency: An Organizational Learning Perspective," *International Journal of Security and Its Applications*, vol/issue: 10(8), pp. 385-394, 2016.
[19] Y. Liu, "Fuzzy-Clustering Web based on Mining," *Journal of Multimedia*, vol/issue: 9(1), 2014.
[20] H. Lee, et al., "A MapReduce-based kNN Join Query Processing Algorithm for Analyzing Large-scale Data," *Journal of KIISE*, vol/issue: 42(4), pp. 504-511, 2015.
[21] Z. Miller, et al., "Twitter spammer detection using data stream clustering," *Information Sciences*, vol. 260, pp. 64-73, 2014.
[22] T. Velmurugan, "Performance based analysis between k-Means and Fuzzy C-Means clustering algorithms for connection oriented telecommunication data," *Applied Soft Computing*, vol. 19, pp. 134-146, 2014.
[23] R. Fok, et al., "Mining Evolving Data Streams with Particle Filters," *Computational Intelligence*, vol/issue: 33(2), pp. 147-180, 2015.
[24] V. Bhatnagar, et al., "Clustering data streams using grid-based synopsis," *Knowledge and Information Systems*, vol/issue: 41(1), pp. 127-152, 2013.
[25] X. Yuan, et al., "A Genetic Algorithm-Based, Dynamic Clustering Method towards Improved WSN Longevity," *Journal of Network and Systems Management*, vol/issue: 25(1), pp. 21-46, 2016.
[26] D. Marrón, et al., "Data stream classification using random feature functions and novel method combinations," *Journal of Systems and Software*, vol. 127, pp. 195-204, 2017.

BIOGRAPHIES OF AUTHORS

Bhaskar Adepu is an Associate Professor at Kakatiya Institute of Technology & Science, Warangal and Affiliated to Kakatiya University, India. He is pursuing his Ph.D. from Jawaharlal Nehru Technological University (JNTU), Hyderabad. He received M.Tech. (CSE) from JNTU-Hyderabad in 2010. His research interests include Data Mining, Image Processing and Artificial Intelligence. He delivered guest lectures in the field of data mining and artificial intelligence at various platforms. He is a Member of IEEE and a member of ISTE.

Jayadev Gyani is an Assistant Professor at the College of Computer and Information Sciences, Majmaah University, Al Majmaah 15341, Saudi Arabia. He received his Ph.D. from the University of Hyderabad in 2009. He has published more than 80 refereed journals and conference articles in the area of software engineering, data mining, web information systems and digital image processing. Dr.Gyani was program Co-chair and Vice Chair for few international conferences. He is a Member of IEEE Computer Society and a Member of ACM.

Narsimha Gugulotu is a Professor and Head at JNTUH College of Engineering, Sultanpur, Jawaharlal Nehru Technological University, Hyderabad, India. He received his Ph.D. from the Osmania University, Hyderabad in 2009. He has published more than 60 refereed journals and conference articles in the area of Data mining, Mobile Computing, Computer Networks, Cloud Computing and Big data analytics. Dr.Narsimha is catering various administrative and academic responsibilities at various capacities. He is also reviewer for few international conferences. He is a Member of IEEE Computer Society and a Member of ISTE.