Gaussian Mean Shift Ellipsoidal Clustering-Based R-Tree Indexing For Multidimensional Data Stream Analysis

Chitra. K, Maheswari. D

Abstract: Data Stream analysis is a process of extracting the valuable information from the continuous data records. A data stream is an ordered sequence of instances in many applications and it read-only once or a small period of time with lesser computing and storage capabilities. Most of the recent research work aimed to reduce the dimensionality of high dimensional data. Further, the space complexity remained unaddressed in the existing works. In order to overcome these limitations, a Gaussian Mean Shift Hyper-Ellipsoidal clustering-based R-Tree indexing (GMSHEC-RTI) technique is developed for reducing the dimensionality of high-dimensional data with lesser space complexity. In the GMSHEC-RTI technique, the sliding window model is applied for handling the incoming data stream to minimize the clustering time. Then the Gaussian Mean Shift Hyper-Ellipsoidal clustering technique is applied for partitioning the data into different ellipsoidal shape clusters that depends on Mahalanobis distance metric with higher clustering accuracy. Then, the GMSHEC-RTI technique constructs R-tree for indexing and storing the clustered high dimensional data for further processing which results in minimizing the space complexity. Experiment is conducted with different metrics such as clustering accuracy, false-positive rate, time complexity and space complexity with respect to a number of data.

Index Terms: Dimensionality reduction, Gaussian Mean Shift Hyper-Ellipsoidal clustering, Mahalanobis distance, Multidimensional Data Stream, R-tree indexing, Sliding window

1 INTRODUCTION

Analyzing the large volume of datasets and extracting useful pattern in many applications such as web stream is difficult since the large dataset includes the more data that are continuously added and queried. For managing and analyzing the large volume of data, the clustering process is used. Datastream clustering is the process of dividing the total dataset into different groups. However, there are several issues that are identified in the previous works on data stream clustering solutions. In order to addresses the existing issues, GMSHEC-RTI Technique is developed. The major contribution of the proposed technique is summarized as follows,To minimize the dimensionality of the dataset, the GMSHEC-RTI technique is introduced for multidimensional data stream analysis. The Gaussian Mean Shift Hyper-Ellipsoidal clustering technique is applied to divide the total dataset into different clusters based on the distance between the data and cluster mean. Then nearest cluster centers are identified and merged to form a new cluster. This helps to groups all the data arrived in the sliding window. As a result, the GMSHEC-RTI technique improves the clustering accuracy and minimizes the false positive rate.To minimize the time complexity of data clustering, the GMSHEC-RTI technique uses the sliding window concept for handling a large number of incoming data streams in the slides. To minimize the space complexity, the R-tree indexing is applied in the GMSHEC-RTI technique. After clustering, the group of data is stored in the leaf nodes of the tree with lesser memory consumption. Besides, R-tree efficiently performs the data insertion and deletion operation.

- *ManuscriptChitra. K*, Research Scholar, School of Computer Studies, Rathnavel Subramaniam College of Arts and Science, Sulur, Coimbatore, Tamil Nadu, Email: chitra.k@rvsgroup.com*
- *Maheswari. D, Head, Research Coordinator, School of Computer Studies- PG, Rathnavel Subramaniam College of Arts and Science, Sulur, Coimbatore, Tamil Nadu, Email: maheswari@rvsgroup.com*

The rest of the paper is organized into different sections as follows. In section 2, related works in data stream clustering are discussed. In section 3, a brief description of the proposed GMSHEC-RTI technique is presented. Experimental setup and parameter settings are presented in section 4 using the dataset. In section 5, comparisons of the different metrics are presented using a table or graphical representation. Finally, the conclusion is presented in section 6.

2 RELATED WORKS

The Ant Colony Stream Clustering (ACSC) algorithm was designed in [1] for grouping the dynamic data streams. However, the dimensionality reduction was not performed with varying density of data. A Density-Based Self Organizing Incremental Neural Network (DenSOINN) was developed in [2] for data stream clustering. The designed method failed to improve the clustering accuracy. A neuro-fuzzy Kohonen network using possibilistic fuzzy clustering was developed in [3] for processing the data streams mining tasks. But the complexity of the algorithm remained unaddressed. A novel versatile hyper-elliptic clustering (VHEC) algorithm was introduced in [4] to group the streaming data. However, the algorithm provides incorrect clustering results when the dimension of data was larger. An incremental c-regression and c-varieties clustering technique were introduced in [5] for processing the streaming data. The performance of clustering algorithms was not improved. A divide and conquer approach was introduced in [6] for grouping the data stream using the vector model. But the approach failed to consider the arbitrary and non-convex shaped clusters. A Multi-Density Stream Clustering (MDSC) technique was developed in [7] for on-line grouping of dynamic data streams. Though the technique minimizes the complexity, the accurate clustering results were not obtained. The different stream clustering methods were developed in [8] to estimate the performance with the datasets. But the dimensionality reduction was not performed. A Multiview Data Stream (MVStream) clustering method was introduced in [9] for identifying the clusters of random shapes.

But the clustering performance was not enhanced using multiple streams of data. A Batch Capturing with Elliptic Function (BEstream) was developed in [10] for grouping the data streams. However, the complexity of this clustering technique was higher.A novel synchronization-based clustering method (SyncTree) was developed in [11] for evaluating the data streams. The method reduces the time as well as memory consumption but it failed to perform the synchronization for distributed data stream clustering. A Continuous Clustering of Trajectory Stream Data (CC-TRS) was developed in [12] to partition the stream of data. The CC-TRS algorithm consumed high memory space since the data structure of the temporal micro cluster has additional temporal fields.A fully online clustering method was introduced in [13] for partitioning the data into randomly shaped clusters. The designed clustering method did not have the ability to accurately cluster the data from evolving data streams. A buffer-based online clustering technique was introduced in [14] for processing the online data streams and providing the different shaped clusters. The designed algorithm was scalable to a high-dimensional data stream but it failed to minimize the space complexity. An Improved Streaming Affinity Propagation (ISTRAP) was developed in [15] to perform the data stream clustering with minimum time consumption. But the designed approach failed to effectively handle the high-dimensional data streams. Dynamic Feature Mask (DFM) scheme was developed in [16] for selecting the features to enhance the performance of data stream clustering and minimizing the processing time. But the scheme failed to minimize the space complexity while handling the high dimensional data streams.

A Fast Evolutionary Algorithm for Clustering data streams (FEAC-Stream) was presented in [17]. The algorithm failed to reduce the dimensionality since it did not merge the nearest clusters. A Dynamic Fitness Proportionate Sharing (DFPS) clustering method was developed in [18] for processing the data stream. The designed method reduces memory usage and execution time but the computational efficiency was not improved. A k-means-based clustering technique was introduced in [19] to concurrently group the high-dimensional data. But the accurate clustering results were not obtained with minimum false positive rate. Evolving the Fractal-based Clustering of Data Streams called (eFCDS) was developed in [20] to group the set of multivariate data streams by measuring the similarity between the attributes. But the performance of complexity was not minimized. The existing issues are overcome by introducing a novel technique called GMSHEC-RTI.

3 METHODOLOGY

A Gaussian Mean Shift Hyper-Ellipsoidal clustering-based R-Tree indexing (GMSHEC-RTI) technique is introduced for reducing the dimensionality of high-dimensional data. The GMSHEC-RTI technique considers the stream of high-dimensional data as input to perform the dimensionality reduction. After collecting the big data stream, the GMSHEC-RTI technique performs two major processes namely clustering and indexing. In the clustering process, the continuous streams of data are grouped into different clusters for further processing. Next, the R-tree index is applied for storing the clustered data in order to minimize the dimensionality of the data.

3.1. Gaussian Mean Shift Hyper-Ellipsoidal clustering of the data stream

The GMSHEC-RTI technique initially performs the big data stream clustering using Gaussian Mean Shift Hyper-Ellipsoidal clustering technique. The proposed clustering technique uses the sliding window concept to manage the incoming data streams for minimizing the time complexity in the clustering process. The existing data stream clustering algorithm uses the Euclidean distance for grouping similar objects and it provides the spherical shape of clusters. But it failed to handle the varying density of data. In order to overcome such kind of issue, the hyper-ellipsoidal clustering technique is introduced in the proposed GMSHEC-RTI technique for evolving the data stream with varying density. In addition, the hyper-ellipsoidal function is used for probably rotating in any direction and adjusting the size using a set of recursive functions.

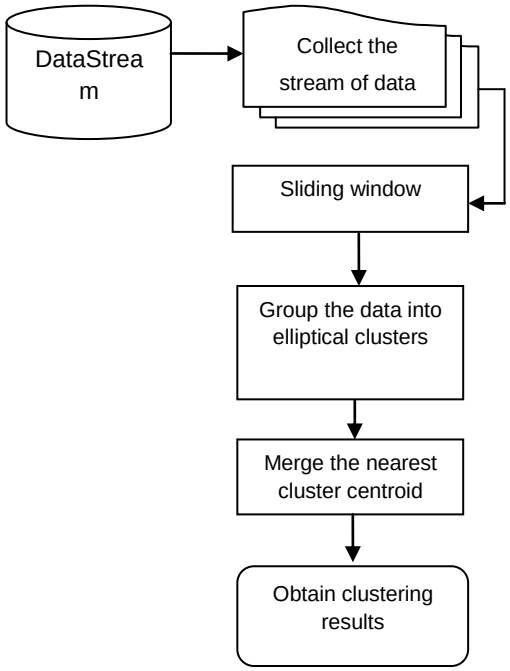

Fig. 1 flow process of Gaussian Mean Shift Hyper-Ellipsoidal clustering

Figure 1 illustrates the process of the clustering technique to group similar data into different clusters in the form of elliptical structure. Let us consider the big dataset D_b and the stream of data $Ds_1, Ds_2, Ds_3, \ldots . Ds_n$ collected from the dataset. The incoming streams of the data are partitioned into several small window slides $w_1, w_2, w_3, \ldots w_n$ based on the arrival order of each data. Therefore, the size of the window is specified based on time instead of data arrivals

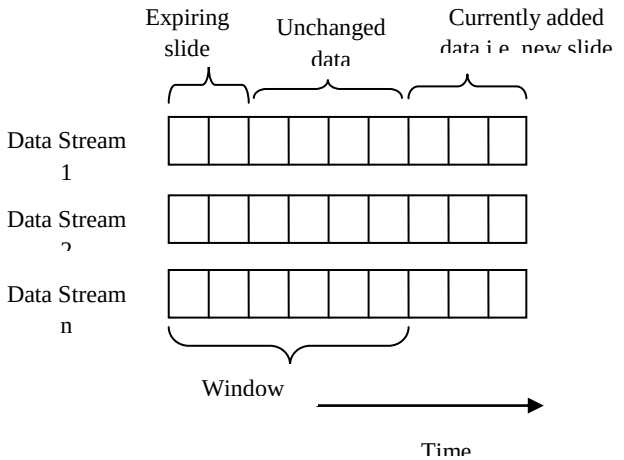

Fig. 2 structure of the sliding window

Figure 2 shows the structure of sliding window with the number of slides. The sliding window worked on the basis of time. The 'M' 'sized sliding window is shifting from the extreme left to the extreme right of the array. Every time the sliding window shifts right by one position. Timestamp are stored for every basic window. When the timestamp of the previous window expires, that the window is removed from the array and a new basic window is added into the array. After storing the data into the array of sliding window, the clustering is carried out using the Gaussian Mean Shift Hyper-Ellipsoidal clustering technique. The Gaussian mean shifted clustering partitions the incoming data into different clusters based on mean and deviation. Compared to an existing clustering algorithm, the Gaussian mean shift clustering technique did not intilize the number of clusters. The number of clusters is determined based on the number of data. Therefore, the number of incoming data streams is partitioned into a number of clusters. For each cluster, the mean is calculated based on the weighted sum of data streams.

$$\mu = \frac{\sum_{i=1}^{n} \beta_i \, Ds_i}{\sum_{i=1}^{n} \beta_i} \qquad (1)$$

Where,μ denotes a mean of the cluster, β_i denotes a weight, Ds_i represents the number of data. For each mean (i.e. cluster center), the nearby data are grouped into the cluster based on the Gaussian distribution function.

$$Y = \exp\left(-\frac{1}{2\,\sigma^2} * \left\|d_{ij}\right\|^2\right) \qquad (2)$$

Where Y represents a Gaussian distribution function, σ denotes a deviation from its mean, $\left\|d_{ij}\right\|^2$ denotes a Mahalanobis distance which is measured as follows,

$$d_{ij} = \left(Ds_i - \mu_j\right)^T M_j \left(Ds_i - \mu_j\right) \quad (3)$$

Where, μ_j denotes a mean (i.e. center) of the 'j' th cluster and Ds_i represents the dataM_j denotes a symmetric positive definite covariance matrix which provides a strength of the relationship between the data and cluster mean, 'T' denotes a transpose vector. For each iteration, the streams of data are assigned into the nearest cluster mean. For each cluster, the volume of a cluster is defined in the structure of Hyper-Ellipsoidal which enclosing all the data into the cluster. If Mahalanobis distance is lesser than one it is said to be an ellipsoid structure based cluster which is mathematically defined as follows,

$$e\left(c_j, M\right) = \left\{ Ds_i \in D_b : d_{ij} \leq 1 \right\} \qquad (4)$$

Where, e denotes an ellipsoid structure based cluster, Ds_i is the stream of data, D_b denotes a dataset, d_{ij} denotes a Mahalanobis distance. For each iteration, the cluster center is updated. This process is iterated until the number of data is moved into the clusters. Finally, the ellipsoid-shaped clusters are obtained in the given dimensional space. After clustering the stream of data, the merging process is carried out by the nearest mean concept. If the distance between the means of the cluster (i.e cluster center) is lesser than the threshold (Th_r), then the two clusters are merged into one cluster in the elliptical surface. After merging the clusters, the cluster center gets updated for the newly combined cluster. Finally, the well-partitioned data are shrunken to the newly updated cluster centroid. As a result, the proposed technique merges the clusters and to form a new cluster which results in minimizes the dimensionality of the cluster in given dimensional space.

Input: Big dataset, Stream of data $Ds_1, Ds_2, Ds_3, \ldots . Ds_n$
Output: Improve the clustering accuracy
Begin
1. **Collect** Stream of data $Ds_1, Ds_2, Ds_3, \ldots . Ds_n$
2. Partition the data into small window slides $w_1, w_2, w_3, \ldots w_n$
3. Partition Ds_i to form a number of clusters
4. **For** each cluster' j'
5. Compute mean μ
6. **end for**
7. **For each** Ds_i
8. **For** each mean μ_i
9. Calculate the Gaussian distribution function Y
10. Group Ds_i into clusters j based on the distance d_{ij}
11. **end for**
12. **end for**
13. **Compute distance (d_c) between two centers of cluster**
14. **If** $(d_c > Th_r)$ **then**
15. Merge the two clusters
16. Update the cluster center
17. Shrink the data to the new cluster center
18. **else**
19. **Remove the** ellipsoid having a lower cardinality
20. **end if**
21. Process repeated until no data moved
22. Obtain final clusters

end

Algorithm 1 Gaussian Mean Shift Hyper-Ellipsoidal data stream clustering

Algorithm 1 describes the step by step process of the clustering process. The incoming data streams are handled by using a sliding window technique to minimize the clustering time. After that, the data are grouped into the different elliptical-shaped clusters based on the distance between the cluster center and data. After clustering, the nearest cluster center of the ellipsoid-shaped clusters is merged into one cluster for minimizing the dimensionality. The threshold value is set to verify the distance between the two cluster centers. If the distance is lesser than the threshold, then the two clusters are merged. Followed by, the center of the cluster gets updated and all the data are shrunk to the updated cluster center. This process gets iterated until all the data are moved

into the clusters. This process improves the clustering accuracy of given high dimensional data.

3.2 R-tree indexing based data storage

After clustering the continuous incoming data, the R-tree indexing is applied for storing the multiple data for minimizing the space complexity. The R-tree is a binary search tree that contains the three nodes such as a single root node, internal nodes, and leaf nodes. It is the top-down approach where each node contains pointers to their child nodes where the region of child nodes entirely overlaps the regions of parent nodes. The leaf node contains the clustered data.

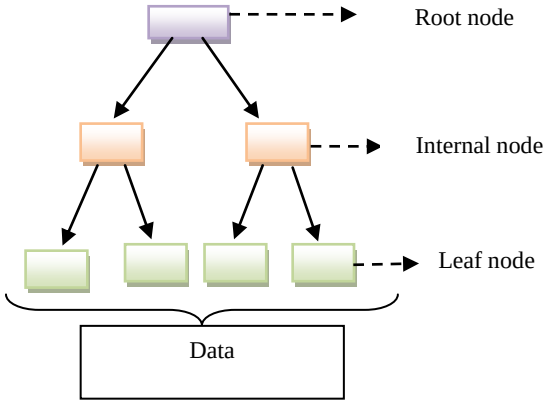

Fig. 3 Structure of R -tree

Figure 3 illustrates the R-tree for storing the multiple data in the leaf nodes. A root node of the tree includes at least two entries. The internal node store $n \leq B/2$ and B child where B denotes a child entries, n is the number of entries. Each entry is in the form (v, b) where 'v' denotes a pointer to a child node and b denotes a minimum bounding rectangles that spatially includes all the entries. Every leaf node stores the data between $n \leq B/2$ and B entries. Every entry is in the form of (ID, b) where, ID denotes a data identifier, b denotes a minimum bounding rectangles that spatially includes all the data. All leaves presented at the similar height in the tree. The maximum feasible height of the R-tree is expressed as follows,

$$H_m = [log_n\, Ds_n] - 1 \qquad (5)$$

Where, H_m denotes a maximum height of the tree, Ds_n represents 'n' number of data stored in the tree. Therefore, this helps to improve the dynamic data storage in the tree. The tree structure also performs data insertion and date deletion operations. If any data insert to the tree, a new entry is inserted into the leaves. Then the entire path from the root to a leaf node gets updated and increases the height of the tree by 1. In addition, the data deletion is carried out by deleting the entry from the leaf nodes. If any root has one child node, that root node is removed. Followed by, the entire path from the root to the leaf node gets updated. In this way, all the clustered data are stored in the tree with minimum space complexity. The algorithmic process of data storage is described as follows,

// R-tree based dynamic data storage
Input: Clustered data
Output: Minimize space Complexity
Begin
1. Construct R-tree with a maximum height H_m
// Insertion Operation
2. Insert a new entry E to the new leaf node L
3. Store the data Ds_i
4. update the path from the root to leaf
5. Increase the height of the tree
// Deletion Operation
6. **If** data wants to delete **then**
7. Search all entries of the root node
8. Find the entries of the root node
9. Remove the entries 'n' from L
10. **end if**
11. **if** the root has one child **then**
12. remove the root and set a new root
13. **end if**
14. Update the path from the root to leaf node
end

Algorithm 2 R-tree indexing based data storage

The above algorithm describes the step by step process of the dynamic data storage on the tree for minimizing the dimensionality. The tree is constructed with the root node, internal node and leaf node with the maximum possible height. The clustered data are stored in the leaf node. Then two operations are carried out such as insertion and deletion. During the insertion operation, the new entry is added into the leaf node and inserts the clustered data. Followed by, the path from the root to leaf gets updated. Similarly, the data deletion operation is performed to remove the data from the leaf node. Therefore, efficient data storage is performed with minimum space complexity. The above said two algorithmic processes are implemented in the experimental evaluation to show the performance of the proposed technique over the existing methods.

4 EXPERIMENTAL SETUP AND PARAMETER SETTINGS

Experimental evaluation of proposed GMSHEC-RTI technique and existing methods namely ACSC [1], DenSOINN [2] are implemented in Java language using El Nino Data Set taken from the UCI machine learning repository. The streams of the data are collected from the big weather dataset. The dataset comprises the 12 attributes and 178080 instances. The dataset comprises of different weather data like humidity, air temperature, sea surface temperature, and subsurface temperatures and so on. The attribute characteristics are integer and real and the dataset characteristics are Spatio-temporal. For the experimental consideration, the multiple streams of the weather data are taken from the dataset.

5 RESULTS AND DISCUSSION

The experimental result of the proposed GMSHEC-RTI technique and existing methods ACSC [1], DenSOINN [2] are discussed in this section. The different metrics are used for evaluating the results such as clustering accuracy, false-positive rate, time complexity and space complexity. The obtained results are discussed with the help of a table or graphical representation. For each section, the statistical calculation is given to show the performance of the proposed technique and conventional techniques.

5.1 Clustering Accuracy

Clustering accuracy is defined as the percentage ratio of a number of incoming stream data is correctly grouped into the clusters to the total number of data taken as input. Therefore, the clustering accuracy is mathematically calculated as follows,

$$C_{Acc} = \left(\frac{N_{CG}}{n}\right) * 100 \quad (6)$$

Where, C_{Acc} represents the clustering accuracy, N_{CG} denotes a number of data that are correctly grouped, 'n' denotes a total number of stream of data. The clustering accuracy is measured in the unit of percentage (%).

Sample mathematical Calculation:

Existing ACSC: Let us taken as a total number of data taken as input is 1000 and the number of data correctly grouped into the cluster is 860. Therefore, the clustering accuracy is mathematically calculated as follows,

$$C_{Acc} = \left(\frac{860}{1000}\right) * 100 = 86\ \%$$

Existing DenSOINN: Let us taken as a total number of data taken as input is 1000 and the number of data correctly grouped into the cluster is 840. Therefore, the clustering accuracy is mathematically calculated as follows,

$$C_{Acc} = \left(\frac{840}{1000}\right) * 100 = 84\ \%$$

Proposed GMSHEC-RTI: Let us taken as a total number of data taken as input is 1000 and the number of data correctly grouped into the cluster is 920. Therefore, the clustering accuracy is mathematically calculated as follows,

$$C_{Acc} = \left(\frac{920}{1000}\right) * 100 = 92\%$$

Table 1 clustering accuracy versus the number of data

Number of data	Clustering accuracy (%)		
	GMSHEC-RTI	ACSC	DenSOINN
1000	92	86	84
2000	96	88	83
3000	94	86	81
4000	95	89	83
5000	93	88	84
6000	92	87	83
7000	94	89	87
8000	95	88	85
9000	98	90	87
10000	96	89	86

Table 1 describes the clustering accuracy based on the number of data taken from the weather dataset. For the experimental consideration, the stream of data is taken in the range from 1000 to 10000. The table shows that the clustering accuracy of three clustering techniques namely GMSHEC-RTI, ACSC [1], DenSOINN [2]. The reported result evidently proves that the clustering accuracy is found to be improved using the GMSHEC-RTI technique as compared to existing methods. This improvement is achieved by applying the Gaussian mean shift hyper ellipsoidal clustering technique. The technique groups similar data based on the mean and standard deviation. The data which is closer to the mean is grouped into a particular cluster. The ellipsoidal structure of the cluster is formed in the GMSHEC-RTI technique for handling the more number of data in the given dimensional space. The nearest cluster center is merged and obtaining the final clustering results. Based on the clustering results, the input stream of weather data is correctly grouped into the different cluster. Therefore, the clustering results of the GMSHEC-RTI technique are said to be improved. Totally ten results are obtained for each clustering technique. The average comparison results show that the clustering accuracy of the GMSHEC-RTI technique is improved by 7% and 12% as compared to existing ACSC [1], DenSOINN [2] respectively.

5.2 False positive rate

The false-positive rate is mathematically defined as the percentage ratio of a number of incoming streams of data is incorrectly grouped into the clusters to the total number of data taken as input for conducting the experiments. The formula for calculating the false positive rate is expressed as follows,

$$FP_{rate} = \left(\frac{N_{ICG}}{n}\right) * 100 \quad (7)$$

Where, FP_{rate} represents the false positive rate, N_{ICG} denotes a number of data that are incorrectly grouped, 'n' denotes a total number of data. The false positive rate is measured in the unit of percentage (%).

Sample mathematical Calculation:

Existing ACSC: Let us taken as a total number of data taken as input is 1000 and the number of data incorrectly grouped into the cluster is 140. Therefore, the false positive rate is mathematically calculated as follows,

$$FP_{rate} = \left(\frac{140}{1000}\right) * 100 = 14\%$$

Existing DenSOINN: Let us taken as a total number of data taken as input is 1000 and the number of data incorrectly grouped into the cluster is 160. Therefore, the false positive rate is mathematically calculated as follows,

$$FP_{rate} = \left(\frac{160}{1000}\right) * 100 = 16\ \%$$

Proposed GMSHEC-RTI: Let us taken as a total number of data taken as input is 1000 and the number of data incorrectly grouped into the cluster is 80. Therefore, the false positive rate is mathematically calculated as follows,

$$FP_{rate} = \left(\frac{80}{1000}\right) * 100 = 8\%$$

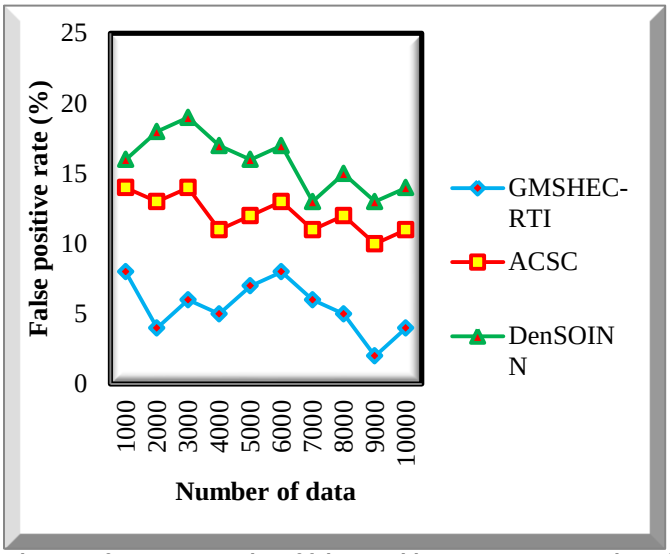

Fig. 4 Performance results of false-positive rate versus number of data

Figure 4 depicts the experimental results of the false-positive rate versus a number of data taken in the range from 1000 to 10000. The number of data is given to the 'x' axis and the results of the false positive rate are obtained at the 'y' axis. The above graphical results clearly shows that the proposed GMSHEC-RTI technique outperforms well and provides the accurate clustering results with minimum false positive rate. This is because of the GMSHEC-RTI technique updates the cluster for each iteration since the nearest cluster center is grouped. After that, the data are shrunken to the new cluster center. Based on the clustering results, the streams of weather data are correctly grouped into the particular cluster. This in turn helps to minimize the incorrect data clustering and this is proved by the mathematical calculation. Let us consider the 1000 data as input for computing the false positive rate. The experimental results of false positive rate using the GMSHEC-RTI technique is 8% whereas the false positive rate of ACSC [1], DenSOINN [2] are 14% and 16% respectively. Likewise, the remaining nine results are obtained with various input data. The observed result proves that the false positive rate is considerably minimized by 55% and 65% using the GMSHEC-RTI technique as compared to the existing clustering techniques.

5.3 Time complexity

Time complexity refers to the amount of time taken by the algorithm to group similar weather data into different clusters. The formula for calculating time complexity is mathematically expressed as follows,

$$T_{com} = n * t(grouping\ single\ data) \qquad (8)$$

Where, 'T_{com}' denotes a time complexity, n is the total number of data, t denotes a time to group a single data. The time complexity is estimated in the unit of milliseconds (ms).

Sample mathematical Calculation:

Existing ACSC: Let us taken as a total number of data taken as input is 1000 and the time for grouping single data is $0.032ms$. Therefore, the overall time complexity rate is mathematically calculated as follows,

$$T_{com} = 1000 * 0.032ms = 32ms$$

Existing DenSOINN: Let us taken as a total number of data taken as input is 1000 and the time for grouping single data is $0.038ms$. Therefore, the overall time complexity rate is mathematically calculated as follows,

$$T_{com} = 1000 * 0.038ms = 38ms$$

Proposed GMSHEC-RTI: Let us taken as a total number of data taken as input is 1000 and the time for grouping single data is $0.027ms$. Therefore, the overall time complexity rate is mathematically calculated as follows,

$$T_{com} = 1000 * 0.027ms = 27ms$$

Table 2 time complexity versus the number of data

Number of data	Time complexity (ms)		
	GMSHEC-RTI	ACSC	DenSOINN
1000	27	32	38
2000	30	36	40
3000	34	42	48
4000	40	46	52
5000	46	55	60
6000	51	60	66
7000	56	63	70
8000	62	68	76
9000	73	79	84
10000	80	90	95

As shown in table 2, the experimental results of time complexity with respect to a number of data are presented using various clustering techniques namely GMSHEC-RTI, ACSC [1], DenSOINN [2]. With the increasing number of data, the time complexity of three methods is found to be in the increasing trend. But comparatively, the time complexity gets minimized using the GMSHEC-RTI technique. This is due to the application of the sliding window-based clustering technique. The incoming data streams arrive at the different slides of the windows based on the arrival time. Therefore, the size of the window is specified based on time instead of data arrivals. Then the GMSHEC-RTI technique performs the clustering process and minimizes the time complexity. For example, 1000 datas are considered for calculating the time complexity. The GMSHEC-RTI technique groups the similar weather data by consuming $27ms$ of time whereas the time complexity of ACSC [1], DenSOINN [2] are $32ms$ and $38ms$. The above statistical analysis proves that the overall time complexity is minimized using GMSHEC-RTI technique by 13% when compared to ACSC [1], and 22% as compared to DenSOINN [2].

5.4 Space complexity

Space complexity refers to the amount of memory space taken by the algorithm to store similar weather data into the clusters. Mathematically, the space complexity is calculated as given below,

$$S_{com} = n * S(storing\ single\ data) \qquad (9)$$

Where, 'S_{com}' denotes a space complexity, n is the total number of data, S denotes a space to group a single data. The space complexity is estimated in the unit of megabytes (MB).

Sample mathematical Calculation:

Existing ACSC: Let us taken as a total number of data taken as input is 1000 and the space for grouping single data is $0.036MB$. Therefore, the overall space complexity is mathematically calculated as follows,

$$S_{com} = 1000 * 0.04MB = 40MB$$

Existing DenSOINN: Let us taken as a total number of data taken as input is 1000 and the space for grouping single data is $0.046MB$. Therefore, the overall space complexity is mathematically calculated as follows,

$$S_{com} = 1000 * 0.046MB = 46MB$$

Proposed GMSHEC-RTI: Let us taken as a total number of data taken as input is 1000 and the space for grouping single data is $0.036MB$. Therefore, the overall space complexity is mathematically calculated as follows,

$$S_{com} = 1000 * 0.036MB = 36MB$$

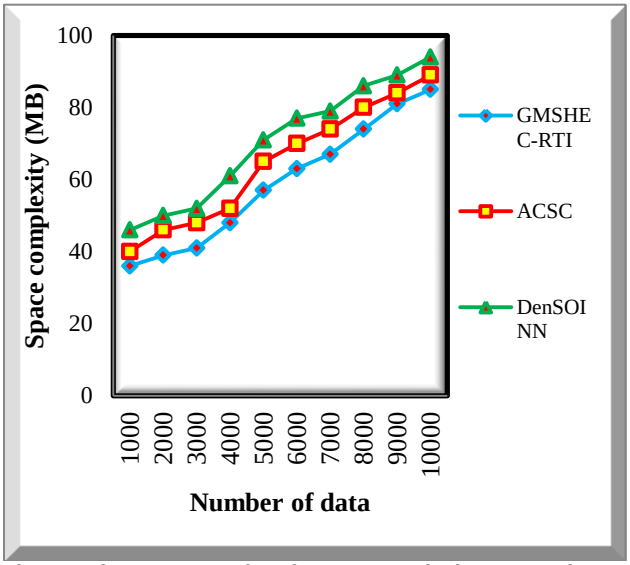

Fig. 5 performance results of space complexity versus the number of data

Figure 5 illustrates the performance results of space complexity with respect to a number of data using three clustering techniques. As shown in the above graph, the three different colors of lines such as blue, red and green are represented as the space complexity of three techniques namely GMSHEC-RTI, ACSC [1], DenSOINN [2] respectively. The graphical results show that the space complexity is minimized using the GMSHEC-RTI technique. This is because of applying the R tree indexing. After the clustering, the data are stored in the leaf node of R-tree with minimum storage space. If any data inserted or removed from the tree, the new leaf node is created or removed from the tree. Followed by, the entire tree structure gets updated. Therefore, the R-tree based technique reduces the space complexity of high dimensional data. Totally ten results of space complexity with various input data are obtained for three clustering techniques and the results of the proposed clustering are compared with the existing results. Then the average of ten various results confirms that the GMSHEC-RTI technique minimizes the space complexity by 9% when compared to the existing ACSC [1]. Similarly, the performance results of space complexity also minimized by 17% when compared to existing DenSOINN [2]. The above-discussed results show that the GMSHEC-RTI technique improves the clustering accuracy of the high dimensional data stream with minimum time-space complexity when compared to the state-of-the-art methods.

6 CONCLUSION

An efficient technique called GMSHEC-RTI is introduced for clustering the dynamic data streams with higher accuracy and minimum complexity. At first, the clustering of multidimensional data streams is carried out based on the Gaussian distribution function. The function uses the Mahalanobis distance measure between the data and the cluster mean. The data which is close to mean is assigned into the cluster. Similarly, the entire data are partitioned and obtain elliptical-shaped clusters. Then the nearest cluster centers are merged and obtain the final ellipsoid clusters. Secondly, the tree-based indexing is performed to store the multiple clustered data resulting in it minimizes the dimensionality. The experimental assessment is carried out using the Elnino dataset for clustering the weather data. The performance of the GMSHEC-RTI technique is evaluated with different metrics such as clustering accuracy, false-positive rate, time complexity, and space complexity. The observed results show that the GMSHEC-RTI technique achieves better clustering accuracy and minimum false positive rate, time complexity as well as space complexity than the state-of-the-art- methods.

REFERENCES

[1] Conor Fahy, Shengxiang Yang, Mario Gongora, "Ant Colony Stream Clustering: A Fast Density Clustering Algorithm for Dynamic Data Streams", IEEE Transactions on Cybernetics (Volume 49, Issue 6, 2019, Pages 2215 – 2228

[2] Baile Xu, Furao Shen, Jinxi Zhao, "A density-based competitive data stream clustering network with self-adaptive distance metric", Neural Networks, Elsevier, Volume 110, 2019, Pages 141-158

[3] Zhengbing Hu, Yevgeniy V. Bodyanskiy, Oleksii K. Tyshchenko, Olena O. Boiko, "A Neuro-Fuzzy Kohonen Network for Data Stream Possibilistic Clustering and Its Online Self-Learning Procedure", Applied Soft Computing, Elsevier, Volume 68, 2018, Pages 710-718

[4] Niwan Wattanakitrungroj, Saranya Maneeroj and Chidchanok Lursinsap "Versatile Hyper-Elliptic Clustering Approach for Streaming Data Based on One-Pass-Thrown-Away Learning", Journal of Classification, Springer, Volume 34, Issue 1, 2017, Pages 108–147

[5] Saso Blazic and Igor Skrjanc, "Incremental Fuzzy C-regression Clustering from Streaming Data for Local-model-network Identification", IEEE Transactions on Fuzzy Systems, 2019, Pages 1-10

[6] Madjid Khalilian, Norwati Mustapha & Nasir Sulaiman, “Data stream clustering by divide and conquer approach based on vector model”, Journal of Big Data, Springer, Volume 3, 2016, Pages 1-21
[7] Conor Fahy and Shengxiang Yang, “Finding and Tracking Multi-Density Clusters in Online Dynamic Data Streams”, IEEE Transactions on Big Data, 2019, Pages 1-15
[8] Stratos Mansalis, Eirini Ntoutsi, Nikos Pelekis, Yannis Theodoridis, “An evaluation of data stream clustering algorithms”, Statistical Analysis and Data Mining: The ASA Data Science Journal, Wiley, Volume 11, Issue 4, Pages 167–187
[9] Ling Huang, Chang-Dong Wang, Hong-Yang Chao, Philip S. Yu, “MVStream: Multiview Data Stream Clustering”, IEEE Transactions on Neural Networks and Learning Systems, 2019, Pages 1 – 15
[10] Niwan Wattanakitrungroj, Saranya Maneeroj, Chidchanok Lursinsap, “BEstream: Batch capturing with elliptic function for one-pass data stream clustering”, Data & Knowledge Engineering, Elsevier, Volume 117, 2018, Pages 53-70
[11] Junming Shao, Yue Tan, Lianli Gao, Qinli Yang, Claudia Plant, Ira Assent, “Synchronization-based Clustering on Evolving Data Stream”, Information Sciences, Elsevier, Volume 501, 2019, Pages 573-587
[12] Musaab Riyadh, Norwati Mustapha, Md. Nasir Sulaiman, and Nurfadhlina Binti Mohd Sharef, “CC-TRS: Continuous Clustering of Trajectory Stream Data Based on Micro Cluster Life”, Mathematical Problems in Engineering, Hindawi, Volume 2017, July 2017, Pages 1-9
[13] Richard Hyde, Plamen Angelov, A.R.MacKenzie, “Fully online clustering of evolving data streams into arbitrarily shaped clusters”, Information Sciences, Elsevier, Volumes 382–383, 2017, Pages 96-114
[14] Md. Kamrul Islam, Md. Manjur Ahmed, Kamal Z. Zamli, “A buffer-based online clustering for evolving data stream”, Information Sciences, Elsevier, Volume 489, 2019, Pages 113–135
[15] Jinping Sui, Zhen Liu, Alexander Jung, Li Liu, Xiang Li, “Dynamic Clustering Scheme for Evolving Data Streams Based on Improved STRAP”, IEEE Access, Volume 6, 2018, Pages 46157 – 46166
[16] Conor Fahy and Shengxiang Yang, “Dynamic Feature Selection for Clustering High Dimensional Data Streams”, IEEE Access, Volume 7, 2019, Pages 127128 - 127140
[17] Jonathan de Andrade Silva, Eduardo Raul Hruschka, João Gama, “An evolutionary algorithm for clustering data streams with a variable number of clusters”, Expert Systems With Applications, Elsevier, Volume 67, 2017, Pages 228-238
[18] Xuyang Yan, Mohammad Razeghi-Jahromi, Abdollah Homaifar, Berat A. Erol, Abenezer Girma, Edward Tunstel, “A Novel Streaming Data Clustering Algorithm based on Fitness Proportionate Sharing”, IEEE Access, Volume 4, 2016, Pages 1-16
[19] Šárka Brodinová, Peter Filzmoser, Thomas Ortner, Christian Breiteneder, Maia Rohm, “Robust and sparse k-means clustering for high-dimensional data”, Advances in Data Analysis and Classification. Springer, Volume 13, Issue 4, 2019, Pages 905–932
[20] Christian C. Bones, Luciana A. S. Romani, and Elaine P. M. de Sousa, “Improving Multivariate Data Streams Clustering”, Procedia Computer Science, Elsevier, Volume 80, 2016, Pages 461–471