International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING

ISSN:2147-6799 www.ijisae.org Original Research Paper

An Approach for Incremental Parallel Mining of Interesting Clustering Patterns in Big Data

Ahmed S. Al-Hegami*[1], Akram A. M. Mustafa[2], Abdulmajed A. G. Al-Khulidi[3]

Submitted:13/03/2024 **Revised**: 28/04/2024 **Accepted**: 05/05/2024

Abstract*:* Clustering algorithms are a significant problem in data mining. Researchers were motivated to propose incremental and parallel clustering algorithms To handle the ever-increasing size of data in real-world databases in order to discover interesting patterns. This is because traditional clustering algorithms, which require the entire dataset to be present before the clustering process can begin, can be computationally expensive and time-consuming to run on large datasets. Incremental clustering algorithms, on the other hand, can be used to cluster data that is being added to a dataset incrementally, which can be much more efficient. Incremental clustering allows new data points to be added to an existing clustering model without having to reprocess all the data. It is useful when dealing with big data that are continuously growing or changing, as it allows the clustering model to be updated without incurring the computational cost of reprocessing all the data. In this paper, an incremental and parallel Clustering mining approach that integrates interestingness criterion during the discovery process of the model is proposed. The approach efficiently discovers interesting patterns from big data. The user's prior knowledge about the domain is essential for the patterns to be interesting. The approach uses MapReduce to process big data in parallel. Parallel and incremental clustering algorithms that consider changing data trends and user attitudes are promising for making the mining process more effective for decision making.

Keywords: *data mining, incremental clustering, parallel mining, machine learning, novelty measure*

1. Introduction

Clustering is the task of grouping data points together based on their similarity. It is a powerful tool for data mining. It can be used to solve a variety of problems, such as market segmentation, fraud detection and Recommender systems [1], [2], [3], [4].

Clustering big data is a challenging problem because the data sets can be very large, making it difficult to process them using traditional clustering algorithms. For example, the possible number of clusters in a database with n records is $\frac{1}{k!}\sum_{i=1}^{k}(-1)^{k-i}\binom{k}{i}(i)^{n}$[5],6]. Also, Big data sets can contain mixed types of data, such as numerical, categorical, images, and audio and videos. This makes it difficult to find a clustering algorithm that can work well on all types of data.

Traditional approaches for clustering algorithms typically make the assumption that data is static, which is not true when data is evolving. Since the user domain knowledge is monotonically augmented with time, the conventional clustering algorithms not only waste computational, communication and I/O resources [7] but also become ineffective in terms of pattern interestingness criteria. Design of efficient incremental and parallel clustering algorithms in dynamic environments is an important area of research [8],[9] ,[11],[12].

The parallel and incremental clustering algorithms that can dynamically adjust to the evolving nature of data and user preferences are more likely to be successful in data mining[2],[3],[4] ,[12]. Parallel clustering algorithms may be used to cluster big data by dividing the data set into smaller parts and clustering each part on a different machine. This can significantly improve the performance of clustering algorithms for large data sets. Incremental clustering algorithms updates the clustering results as new data becomes available without incurring the computational cost of reprocessing all the data [13],[14],[15],[16],[17],[19]. This is useful for applications where the data is constantly changing, such as social media analysis or financial forecasting. By considering changing data trends and user attitudes, the clustering algorithms can be made more effective for decision making [20],[21].

In this paper, an incremental and parallel clustering mining approach that incorporates interestingness criteria into the model building process is proposed. The incorporation of the novelty measure of interestingness proposed in [5],[20] helps in reducing the size, maintaining an up-to-date clusters and efficiently discover novel patterns from big data. The approach uses MapReduce [20],[24],[25] to

[1] *Professor of AI and Intelligent Systems, Faculty of Computer & IT, University of Sana'a, Yemen.*
ORCID ID : 0000-0002-0904-0741
Emai:alhegami@su.edu.ye
[2] *Ph.D sholar at Faculty of Computer & IT, University of Sana'a, Yemen.*
ORCID ID : 0009-0008-0315-3180
Emai:akram.mustafa@su.edu.ye
[3] *Professor of Software Engineering, Faculty of Computer & IT, University of Sana'a, Yemen.*
ORCID ID : 0000-0001-6843-1155
Emai:alkhulaidi@su.edu.ye
** Corresponding Author Email: alhegami@su.edu.ye*

process big data in parallel. MapReduce is a programming model that breaks down a large data processing task into smaller tasks that can be executed in parallel on a distributed system. It is designed to process large data sets much faster and more efficient, which can make the mining process much faster and more efficient. In addition, the proposed approach addresses the issues of dealing with mixed data, changing user beliefs and model maintenance in an environment consisting of evolving data.

2. Motivations

Classical clustering algorithms have a major drawback: they do not take into account the temporal ordering of data. Data is typically acquired in a streaming fashion and the amount of data continues to grow. Subsequently, the potential benefits of will become more significant [25],[26]. There are a number of solutions that have been proposed to address these challenges, such as cloud computing, parallel processing, and machine learning [1],[19].Parallel processing is a powerful tool that can be used to handle big data efficiently and effectively.

As data continues to grow, traditional algorithms added new data to the existing data set continuously. In such a scenario, a new model is created from the ground up using both old and new data. This can lead to the loss of Previously Discovered Knowledge (PDK).

Researchers have proposed techniques for incrementally updating models, rather than retraining them from scratch[14],[15],[20]. This lead to the incremental algorithm which update the model as new data arrives, while traditional algorithms retrain the model from scratch [16],[17],[21]. Incremental clustering algorithms are able to adapt to changes in data and user preferences, which can make the mining process much faster and more efficient. Our proposed incremental clustering algorithm is based on the assumption that the patterns discovered from one data set are likely to be similar to those discovered from another data set, unless the underlying data generation process has changed dramatically [1],19],[20],[29].

3. Problem Statement

The big data presents significant challenges for traditional data processing methods. The need for efficient and scalable solutions has led to the exploration of parallel processing techniques. However, most existing approaches focus on batch processing, which limits their ability to handle real-time data streams and incremental updates effectively.

This paper aims to develop an approach that leverages parallel processors to manage big data in an incremental manner. Our objective is to design an approach that can dynamically accommodate new data in an incremental way to discover interesting pattern. By addressing these challenges, our approach seeks to enhance the efficiency of big data analytics and provide timely insights necessary for decision-making in various applications.

4. Related Works

Several researches deal with the problem of incremental clustering and address challenges such as real-time processing, adaptability to evolving clusters, and scalability.

In [30],[31], an online K-means clustering algorithm that can generate approximately O(k) clusters with a K-means cost of approximately O(W*) is proposed. Experimental results show that the algorithm performs similarly to k-means++ in a more constrained computational model. In [32], an incremental clustering algorithm called Streaming K-means++ which extends the K-means++ algorithm to handle data streams is proposed. It maintains a representative set of centers called "coresets" that adaptively captures the evolving clusters. The algorithm incrementally updates the coreset as new data arrives, allowing for efficient and scalable stream clustering.

In [33], DenStream is presented which is an incremental clustering algorithm designed specifically for data streams. It employs a density-based approach to identify micro-clusters that represent the evolving clusters. It dynamically updates the micro-clusters as new data arrives, allowing for efficient and scalable stream clustering.

In [34],[35], CluStream and DenStream algorithms that are proposed. They are incremental clustering algorithm designed to handle data streams with concept drift. Density-based approaches using micro-clusters to capture the evolving clusters in the stream are employed. DenStream dynamically adjusts the micro-clusters based on the arrival of new data and the concept drift detection. The algorithms provide flexibility in handling different density distributions and is suitable for real-time stream clustering.

In [36], MuDi-Stream is a density-based clustering algorithm for evolving data streams that improves clustering quality in multi-density environments. It does this by keeping summary information about the evolving data stream in the form of core mini-clusters in the online phase. In the offline phase, the algorithm uses a modified density-based clustering algorithm to generate the final clusters.

In [37], a new incremental density-based clustering algorithm has been proposed that uses Non-dominated Sorting Genetic Algorithm II (NSGA-II) to improve clustering precision. The algorithm adjusts the two input parameters (MinPts and Eps) iteratively using fitness functions, and the optimization process is parallelized to improve efficiency. This results in a significant speed-up compared to the serial version of the algorithm.

In [38], a clustering approach called SyncTree for evolving data is proposed. It works by maintaining the entire micro-clusters at several levels of granularity. It summarizes the constantly arriving data points sequentially in a batch way. This allows us to examine the structure of cluster between any two time stamps in the earlier period.

In [29], an incremental K-Means for massive multidimensional datasets is presented. It is designed for evolving databases. The algorithm measures the new cluster cenroids by computing the new data from the means of the earlier discovered clusters. This approach is more efficient than rerunning the K-means algorithm, which can be computationally expensive.

In [39], m-BIRCH is presented. It is an online clustering algorithm that is designed for incremental clustering large datasets of features used in computer vision. It is efficient because it only uses a fraction of the dataset memory, and it is effective because it can handle varying density regions in the feature space. m-BIRCH is a publicly available clustering tool that can be used to cluster data from a variety of sources.

In [40], an incremental clustering which is based on OPTICS algorithm is proposed. It works by ordering the cluster structure that looks like the structure of OPTICS. However, ICA does not require the user to pre-set the parameters ε and MinPts, and it uses a simpler distance measure called Distance. This makes ICA more efficient than OPTICS.

Other recent incremental partitioning clustering algorithms are Incremental PAM, Incremental CLARANS, Incremental CHAMELEON whish are the extended versions of PAM, CLARANS and CHAMELEON respectively. These algorithms are efficient for streaming data, can handle large datasets, robust to noise and can handle clusters of arbitrary. However, they are less efficient than Incremental k-means [26],[41].

In [42], an incremental clustering algorithm based on nearness is proposed. The algorithm does not comprises the quality of data. The main feature of this algorithm is its ability to update the cluster incrementally and saving computing complexity.

In [43], a clustering approach is proposed. it makes use of an incremental clustering method and a pairwise clustering method to reduce the dimension of the dataset first and subsequently. clustering the dataset to a predetermined number of clusters . the approach is tested using large number of documents and the results shown promising.

In [44], an incremental clustering algorithm, based on a new distance measure is proposed. It deals with both numerical and categorical attributes. The incremental clustering is performed in two stages, In the first stage, the traditional k-means is engaged for initial clustering of the static data set. In the second stage, the distance measure is used to generate the appropriate cluster for the incremental data points. The results of evaluation of the approach shows improving the accuracy and the computational time of clustering.

In [24],[45], a parallel k-means clustering algorithm based on MapReduce is proposed. It efficiently processes large datasets on commodity hardware.

In [46], A novel hybrid clustering algorithms is proposed. It is based on incremental clustering and initial selection to tune up Fuzzy C-Means (FCM) for the Big Data problem.

In [47], a parallel and incremental clustering algorithm called pi- Lisco is proposed. It uses sliding windows to clusters point-cloud data in. It reuses the clusters from overlapping portions of the data to facilitate single-window processing. By combining these key ideas, pi-Lisco achieves significant performance improvements over the state-of-the-art algorithms.

In [23], StreamSW is proposed. It is a density-based clustering algorithm for data stream over a sliding window. It consists of two-steps approach to discover clusters first clusters and then refines the clusters in the second step. This approach exhibits better speed and quality than current approaches.

In [27], a parallel incremental k-means clustering technique is proposed. It clusters the data into a set of k clusters, and then using a forecasting method to predict the future values of each cluster. The clustering is performed incrementally, so that new data can be added to the model without having to re-cluster the entire dataset.

In [10], a parallel incremental density based clustering algorithm called (IncAnyDBC) is introduced. It works by partitioning the data into a set of partitions. Then, the algorithm clusters each partitions in parallel and finally, the clusters from different partitions are merged to form the final clusters.

In [22], a modified version of k-means for mining big data under Hadoop parallel approach is introduced. It works by partitioning the data into a set of partitions, and clusters are generated for each partition in parallel manner. The results from the individual partitions are then merged to form the final clustering solution.

In [28], an algorithm is proposed to enhance the incremental DBSCAN clustering algorithm. It works by reducing the search space rather than the entire dataset. It builds and updates the shaped clusters in large datasets.

In [18], an incremental k-means clustering algorithm is proposed for large multidimensional datasets in dynamic environments where data may be frequently updated. This approach measures the new cluster centers by directly computing them from the means of the existing clusters, instead of rerunning the k-means algorithm from scratch.

In [13], a parallel clustering approach of high dimensional data is proposed. It fits well for interactive online clustering and facilitates incremental clustering because chunks of instances are clustered as separate sets. Subsequently, the results are merged with present clusters.

In [14], an incremental anomaly detection approach is proposed that is based on Gaussian Mixture Model (GMM) to identify regular patterns and discover outliers in from flight data. The presented approach update its clusters incrementally based on new data, rather than running the clustering algorithm from scratch. This makes it more efficient and scalable for clustering large and streaming data.

In [15], a paper presents a modified streaming k-means algorithm that is dynamic, incremental, and can take into account long-term patterns of data. Compared to streaming k- means clustering, the modified streaming k-means clustering has better convergence ability and more stable results.

In [16], an incremental clustering algorithm called LIMBO (Learning Incremental Model- Based Online Clustering) is proposed. It uses a model-based approach to cluster the data points and updates the clustering model as new data points arrive and adjusts the clustering accordingly. The strengths of LIMBO include its ability to handle large datasets and its ability to learn the data distribution. However, its weakness is that it may not be suitable for datasets with varying densities.

In [17], a density-based incremental clustering algorithm called DIStream (Density- Incremental Stream Clustering) is proposed. DIStream can handle data streams with concept drift, and it updates the clustering model as new data points arrive. DIStream adjusts the clustering based on the density of the data points. The strengths of DIStream include its ability to handle data streams with concept drift and its ability to detect noise. However, its weakness is that it may not be suitable for datasets with varying densities.

The proposed algorithm is based on the idea that patterns discovered in one dataset are likely to be similar to patterns discovered in other datasets, to varying degrees. This means that the algorithm can learn from previous datasets and use that knowledge to find patterns in new datasets. Subsequently, the model is continuously updated to reflects the changing data and user beliefs which makes the overall mining process more effective.

5. The Proposed Approach

This paper proposes an efficient parallel and incremental approach for mining interesting clustering patterns from large datasets. The approach integrates interestingness measures into the mining process to discover patterns that are significant to the interests of usrers. It uses MapReduce [20],[24] to distribute the data across multiple machines in a parallel way. The novelty measure of interestingness proposed in [5],[20],[21] is integrated with a clustering algorithm. Integration of the novelty measure helps in reducing the size , and maintaining an up-to-date clusters in an environment where the training set is constantly changing.

K-means and its varieties typically makes the assumption that, data is static, which is not true when data is evolving. Since the user domain knowledge is monotonically augmented with time, the conventional clustering algorithms not only waste computational, communication and I/O resources but also become ineffetctive in terms of pattern interestingness criteria. Our proposed approach is similar to the K-means algorithm except that, it addresses the issues of model maintenance in an environment consisting of evolving data, changing user beliefs, handling the big data in parallel way, dealing with mixed data and using pattern interestingness criteria. The approach design an efficient incremental clustering algorithms in dynamic environments which has the capabilities to:

(1) Utilize domain knowledge,

(2) Acquire additional knowledge when new datasets are introduced,

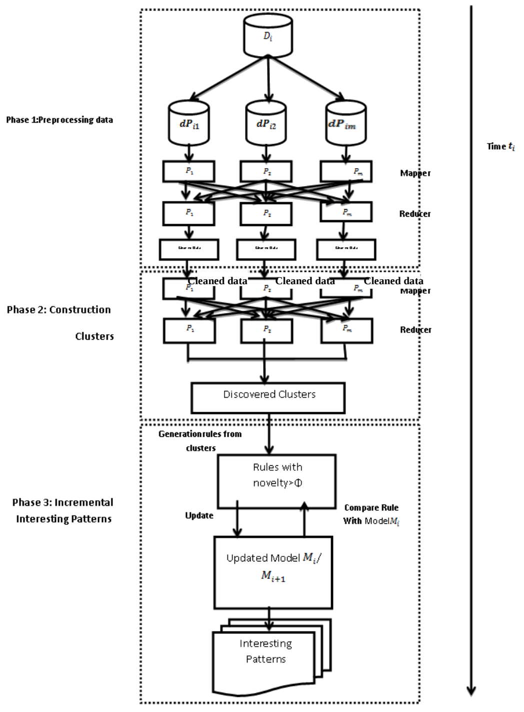
(3) Retain previously discovered knowledge, which are relevant for current dataset,

(4) Ignore previously discovered knowledge, that is not relevant at current time.

(5) Handle big data sets,

(6) Deal with incremental mixed data

The proposed approach is illustrated in Figure 1. At time instance *Ti*, database *Di* is partitioned into *m* parts. where *m* is set manually, and distributed on processors $P_{i,}$ using MapReduce to analyze and mine data in parallel. The clustering algorithm takes the pre- processed data and the existing model MTi as input. The model MTi represents the known patterns and domain knowledge. It then computes the similarity and novelty measures of the new data with respect to MTi. Uninteresting patterns that are not of interest to the user are pruned. The discovered interesting patterns are then presented to the user and used to update the model MT_i resulting in model MT_{i+1}.The following subsections explains the proposed approach:



D_i
Phase 1:Preprocessing data
dP_{i1}
dP_{i2}
dP_{im}
Time t_i
P_1
P_2
P_m
Mapper
Reducer
Cleaned data
Cleaned data
Cleaned data
Phase 2: Construction Clusters
Discovered Clusters
Generationrules from clusters
Rules with novelty>Φ
Phase 3: Incremental Interesting Patterns
Update
Compare Rule With ModelM_i
Updated Model M_i/ M_{i+1}
Interesting Patterns

Fig 1. The Approach for Incremental Parallel Mining of Interesting Clustering Patterns for Big Data

5.1 Pre-processing data

The data pre-processing stage is used to clean, transform, and format the data so that it is ready for clustering algorithm. This is important because the quality of the data can have a significant impact on the accuracy of the model. Pre-process big data, ensuring it is in a suitable format for MapReduce processing. This might involve cleaning, transforming, and scaling the data. Data. This might involve cleaning, transforming, and scaling the data. The proposed framework deals with different types of data. This step involves handling missing values, normalizing the numerical variables , unimportant attributes are removed using a dimensionality reduction technique and feature selection to selects a minimum set of attributes that is sufficient for the clustering task. To pre-process mixed data used in the proposed approach, the following steps are performed:

1. Data understanding: Understand the nature of the mixed data. Mixed data typically consists of both numerical and categorical variables.
2. Variable selection: Identify the variables that are relevant for analysis. Ignore or remove any variables that are not important or contain missing values.
3. Handle missing values: To handle missing values in the dataset, the missing values for numerical variables are replaced with the mean, median, or another statistical measures. For categorical variables, the missing values are replaced with the mode or create an additional category for missing values.
4. Encoding categorical variables: Since k-means is a distance-based algorithm, categorical variables are converted into numerical format. Techniques like one-hot encoding or label encoding are used for this purpose.
5. Scaling numerical variables: Scale numerical variables to a similar range to avoid dominance by variables with higher magnitudes. Techniques like standardization or normalization ate used to achieve this.
6. Combine pre-processed data: Combine the pre-processed numerical and encoded categorical variables into a single dataset.

These steps will ensure that the mixed data is properly pre-processed and suitable for applying the proposed incremental parallel clustering algorithm.

5.2. Domain Knowledge

The proposed approach is able to learn from the user and improve its performance over time. This is done by capturing the user's background knowledge and monotonically augmenting it with new information. This allows the approach to provide more accurate and personalized predictions and recommendations. Domain knowledge represents the user's background knowledge, and it is expressed in the form of a set of implications, Å→ C where the antecedent (Å) represents the conditions and the consequent (C) represents the conclusion. The antecedent and the consequent are both expressed in conjunctive normal form (CNF), which means that they are a disjunction of conjunctions.

The proposed algorithm is a self-learning algorithm that can continuously improve its performance by learning from new data and user's experience. This is done by representing the user's domain knowledge in the rule format and using it to compare new data to the existing model. This allows the algorithm to discover novel patterns from clusters and to improve its performance over time.(the patterns in the clusters are converted to rules). A feedback mechanism is provided to update domain knowledge when required by the user.

5.3. Computational of Similarity/Dissimilarity Measures

The traditional k-means algorithm is a distance-based algorithm. It is designed for batch processing. It assumes that, the dataset is static and available at once. However, when considering incremental mixed big data that arrive over time, it is more appropriate to use an incremental and parallel version of the k-means algorithm [6],[25],[26],[41]. The algorithm isn't directly applicable to mixed and therefore the Euclidean distance function isn't really meaningful.

As the proposed approach allows for continuous learning and clustering of mixed data points, appropriate distance measures are used. In this work, the Gower's distance measure (GD) is applied.. The Gower's distanceis a hybrid measure that handles both numerical and categorical data. It measures the similarity between the two mixed data points for clustering of similar data points without compromising of clustering accuracy. It is a metric that indicates how different two data points are. The metric ranges from 0 to 1, with 0 representing no difference and 1 representing maximum difference. It is calculated based on the partial similarities of any two samples. The partial similarity (*ps*) is calculated differently depending on whether the data type is numerical or categorical. For categorical data, ps = 1 if the values are the same, and ps = 0 if they are different. For numerical data, *ps* is calculated as follows:

First, the range of the feature column is determined.

$$R_f = max(f) - min(f) \qquad (1)$$

Where *f* is a feature under consideration.

Second, the *ps* is calculated for any two samples in the data.

$$\mathrm{ps}_{\mathrm{ij}}^{f} = \frac{|xif|-|xjf|}{R_f} \qquad (2)$$

Third, Gower Similarity (*GS*) is calculated by taking the arithmetic mean of all partial similarities.

$$GS_{ij} = \frac{1}{m}\sum_{f=1}^{m} \mathrm{ps}_{\mathrm{ij}}^{f} \qquad (3)$$

Finally, the similarity metric (*GS*) is converted to a distance metric (*GD*) by subtracting it from 1.

$$\mathrm{GD} = 1\text{-}\mathrm{GS} \qquad (4)$$

For better understanding, consider the following data:

Table 1. The similarity distance using Gower's distance Measure

	Pclass	Sex	Age	Sibsp	Parch	Fare	Embarked
0	3	Male	21.0	0	0	7.7750	S
1	3	Male	9.0	0	1	3.1708	S
2	2	Female	12.0	0	0	15.7500	S
3	3	Male	25.0	0	0	7.9250	S
4	2	female	20.0	0	0	36.7500	S

Looking at tupples 0 and 3, one immediately recognizes that these two tupples are remarkably similar, with only slight differences in *Age* and *Fare*. As a result, we would expect the *GD* to be very low (closer to zero). In fact, the exact *GD* is 0.0092 (see Table 4 below). More dissimilar samples, such as tupples 0 and 2, have higher distance values — in this case, 0.24.

The proposed algorithm first performs distance calculations between the data points and randomly selected cluster centroids. The data points are then clustered by assigning them to the cluster with the closest centroid. It utilizes the modes of each cluster to find the new cluster centroid. For categorical data, it takes the value that appears the most frequent. For numeric data, it takes the median value. If the same occurrence of values, the first one is taken. The advantage of using the median and mode is their ability to generate accurate results with presence of outliers.

5.4 Construction of Clusters

In this phase, the MapReduce parallel programming model is used to divide Big Data partitioned into m parts, and each part is assigned to a processor for mining and analyzing of data. At each processor *Pi*, the Map step reads the input data and divides it into multiple partitions. Then, it assigns each data point to the closest initial centroid based on similarity measures. The algorithm emits key-value pairs with the centroid as the key and the data point as the value. In the Reduce step, the new centroid position at each centroid is computed by averaging the coordinates of all the data points assigned to it. The algorithm then emits key- value pairs with the updated centroid as the key and the data point as the value. The Map and Reduce steps are repeated until convergence is achieved, that is, in the Map step, the new centroids obtained from the previous iteration is used and In the Reduce step, the cluster centroids are updated to reflect the new data points. The phase outputs the final centroids and their corresponding data points in the clusters. Fig.2 shows the proposed parallel clustering algorithm.

```
Input:
  data: Input dataset
  K: Number of clusters
Output:
        A set of k clusters
Steps:  Step 1: randomly select K centroids
        Step 2: Distribute the data points among the
        available processors
Step 3: Repeat the following until convergence criteria is
met:
        Step 3.1: For each data point x in each processor:
```

```
      Step 3.1.1: Find the nearest centroid c from the current set of Centroids
      Step 3.1.2: Assign data point x to cluster c which has the closest centroid;
    Step 3.2: Collect the assigned data points from all processors
    Step 3.3: Recalculate the centroids for each cluster:
      For each cluster c:
        Step 3.3.1: Compute the new centroid by taking the mode/median of all data points
    Step 3.4: Broadcast the updated centroids to all processors
Step 4: Merge all centroids on different processor to form the final
```

Fig 2. The pseudo-code of the proposed parallel clustering algorithm.

Regarding evaluation the performance of clusters, in terms of accuracy, we should keep in mind that the clustering methods are not a predictive task such as classification, rather, they are descriptive tool**s,** thus analyzing accuracy is meaningless. A clustering task is supposed to find a groups of data which minimizes the cluster distances in each group. It does not use any labeled data for training. Consequently, the proposed approach follow this observation.

5.5 Building Incremental Interesting Model

In this phase, the algorithm extracts clustering patterns from the clusters. The patterns are then evaluated for novelty using a novelty criterion [5],[20]. The novelty criterion considers the existing model *Mi*, which represents the known knowledge (*KK*+*DK*). This ensures that only new and interesting patterns are added to the incremental model M_{i+1}.For each cluster, the class that is assigned to the majority of the data points in the cluster is determined. This is done by counting the number of data points in each class and then identifying the class with the highest number of data points.

As the proposed approach deals with mixed data that may contains only categorical and numerical features, the following steps are performed:

If the mixed data contains only categorical features, the majority class label within that cluster is determined by counting the frequency of each class label in the categorical features of the data points in the cluster and selecting the class label with the highest frequency.

If the data is mixed, with both categorical and numerical features, then for each cluster, the categorical features of the data points in the cluster are considered. The frequency of each class label in the categorical features is then counted. The class label with the highest frequency is selected as the most frequent class for that cluster.

After extracting a set of rules from each cluster, the approach applies a novelty measure (NM) to identify novel rules. The novel rules are then used to update the model M_{i+1}. The novelty measure (NM) is calculated for each rule against the existing model M_i(known knowledge (KK)) and domain knowledge (DK) as shown in the following equation:

$$NM = \frac{\{|S1|+|S2|-2*\mathrm{k}\}+\sum_{i=1}^{k} \Delta(c_1^i, \mathrm{c}i_2)}{|S1|+|S2|} \qquad (5)$$

Where, S1 and S2 be two conjunct sets, K= the pairs of compatible conjuncts between S1 and S2. (C_1^1, C_2^i) is the ith pair of compatible conjuncts. The detailed description of computation of novelty measure is presented in [5],[20]. A rule is considered novel if its novelty measure (NM) is greater than the threshold value. The novelty measure (NM) is a measure of how different the rule is from the existing knowledge. The following are the steps required to build the incremental interesting model:

1. Determine the class label within that each cluster.
2. Generation the rules R_i from each cluster.
3. Compute the novelty measure (*NM*) of rules R_i with respect to Model M_i
4. If *NM(* R_i*)*> Φ Go step 5 else Go step3.
5. Update Model M_i/M_{i+1}

The algorithm for building the incremental interesting model is presented in Algorithm 3. Algorithm 3 takes the set of clusters as input, the novelty threshold value, and the existing knowledge (Model M_i). It outputs the incremental interesting model (M_{i+1}).

Input:

Data: set of clusters

Model M_i

Φ: (Novelty threshold).

DK: Domain knowledge

Output:

Updated incremental interesting model M_{i+1}

```
Steps:
Step 1: for each cluster, determine theclass label.
Step 2: extract the rules $R_i$ from each cluster.
Step 3 :For each rules $R_i$:
    Step 3.1: Calculate the novelty measure (NM) of rules
        Ri with respect to model Mi and DK.
    Step 3.2: If NM( $R_i$)> $\Phi$  Go step 4 else Go step3.
Step 4: Update Model $M_i/M_{i+1}$
```

Fig 3. The pseudo-code of the proposed incremental interesting algorithm.

6. A Detailed Example

To better understand our approach, let's consider a Big Data D that arrived at time T1. We will denote this data as D1. It contains 5 transactions, as shown in Table 2. Suppose that the number of clusters is 2 (k=2) and D1 is divided into 2 parts to achieve parallel mining. The following steps are performed by the proposed approach:

Table 2. A dataset arrived at Time T_1

Tupple_No	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	3	Male	21	0	0	7.8	S
1	3	Male	9	0	1	3.2	S
2	2	Female	12	0	0	15.8	S
3	3	Male	25	0	0	7.9	S
4	2	Female	20	0	0	36.7	S

Step 1: take K data points at random and use them as the centroids of the clusters. Suppo**se** tupple**0** andtupple2 are selected randomly as shown in Table 3.

Table 3. Initial centroids are selected randomly

Tupple_No	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	3	Male	21	0	0	7.8	S
2	2	Female	12	0	0	15.8	S

Step 2: Compute the similarities between each data point and each cluster centroid using the Gower measure**.** Then, assign each data point to the cluster with the closest centroid. Repeat this process iteratively, comparing the data points to each of the centroids using the Gower measure. Data points that are similar to a centroid will have a similarity score of 0**,** while data points that are dissimilar to a centroid will have a similarity score of 1**.** For more clarification, comparing the centroid tupple 0 to the data point in tupple 0 gives 0. While Comparing tupple 0 to the data point in tupple 1 gives 0.066 dissimilarities. In the same fashion, compute all the distances between each centroids and data points. Table 4 shows the distances of each data points with each other.

Table 4. The similarity distance using Gower's distance measure between data points

	0	1	2	3	4
0	**0**	0.066	0.24	0.0092	0.23
1	0.066	**0**	0.26	0.075	0.29
2	0.24	0.26	**0**	0.25	0.03
3	0.0092	0.075	0.25	**0**	0.24
4	0.23	0.29	0.03	0.25	**0**

Notice that, the distance between tupple 0 and tupple 3 is 0.0092 and that of between tuppls 0 and tupple 2 is 0.24. This indicates that tupple 0 is the most similar to tupple 3. We can verify the same as Age and Gender of the tupples are same and preTestScore, postTestScore as well as available_credit are closely related. In the same way, distinction between tupple 0 and tupple 2 can also be made. Table 5 shows the distances of centroids with data points.

Table 5. The similarity distance using Gower's distance measure between data points and centroids..

	Tupple0	Tupple1	Tupple2	Tupple3	Tupple4
Centroid1	0	0.66	0.24	0.009	0.23
Centroid2	0.24	0.26	0	0.25	0.03

Step 3:The data points are clustered by assigning them to the cluster with the closest centroid. The data points Tupple 0,

Tupple 1 and Tupple 3 are assigned to cluster 1; Tupple 2, Tupple 4 are assigned to Cluster 2 as shown in Table 6.

Table 6. The assignment of data points to clusters

Tupple _No	Cluster 1 (centroid Tupple 0)	Cluster 2 (centroid Tupple 2)	Cluster
0	0	0.24	Cluster 1
1	0.066	0.26	Cluster 1
2	0.24	0	Cluster 2
3	0.0092	0.25	Cluster 1
4	0.23	0.03	Cluster 2

Step 4: Determine new centroids for the clusters**.** For the data poins in each cluster, the modes and median are utilizes to find the new cluster centroid. For categorical data, the mode is used, which is the most frequent value. For numeric data, the median is used, which is the middle value when the data is sorted from least to greatest. If there are multiple values with the same frequency, the first one is taken. The data points in each cluster are assigned a different color to make them easier to distinguish. Cluster 1 is colored blue, and Cluster 2 is colored red. This is shown in Table 7.

Table 7. Determining the occurrence of attributes in each cluster

Rec_No	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	3	Male	21	0	0	7.8	S
1	3	Male	9	0	1	3.2	S
2	2	Female	12	0	0	15.8	S
3	3	Male	25	0	0	7.9	S
4	2	Female	20	0	0	36.7	S

Note that, as in Table 7, the cluster 1 data points (Tupple 0, Tupple 1, Tupple 3 has two categorical attribute namely, Sex and Embarked which have Male value as the most observed Sex and S as the most observed Embarked. Cluster 1 also has 5 numerical attributes, namely, Pclass, Age, SibSp, Parch and Fare. For the numerical attributes, the Mean is used. This results in the following new centroids as shown in Table 8.

Table 8. The new centroids of clusters

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
Centroid (Cluster1)	2	Male	21	0	0	7.8	S
Centroid (Cluster2)	2	Female	16	0	0	26.25	S

Finally**,** The steps 2–4 are repeated iteratively until the clusters converge.

Once the clusters are generated, clustering patterns are identified from each cluster. The patterns are then evaluated for their interestingness using a novelty measure. Patterns with a high novelty score are considered to be interesting. The following steps are performed:

Step 1. Determine the most frequent class label for each cluster. This can be done by counting the frequency of each class label in the categorical features of the data points in that cluster. The class label with the highest frequency is selected as the most frequent class for that cluster.

For cluster 1, the most frequent class label is "Sex=male". Therefore, this attribute and its value, "male", become the class label of this cluster. Similarly, the most frequent class label for cluster 2 is "Sex=female", so this attribute and its value, "female", become the class label of this cluster.

Step 2. Extract the rules from each clusters as follows:

Rule 1: Pclass=3, Age=21, Fare=7.8, Embarked=S →Sex = Male

Rule 2: Pclass=3, Age=9, Fare=3.2, Embarked=S →Sex = Male

Rule 3: Pclass=3, Age=25, Fare=7.9, Embarked=S → Sex = Male

Rule 4: Pclass=2, Age=12, Fare=15.8, Embarked=S → Sex = Female

Rule 5: Pclass=2, Age=20, Fare=36.7, Embarked=S → Sex = Female

Step 3. Compute the novelty measure of each rule with regard to known knowledge and domain knowledge assuming that the novelty threshold Φ=0.5.

Table 9 . Evaluating patterns in terms of their interest using the novelty Measure

Rule No.	Discovered Rules	Accept	Compare with Rule	Novelty(NM)	NM> Φ=0.5 Add to M_i/M_{i+1}
R1	Pclass=3,Age=21, Fare=7.8, Embarked=S □Sex = Male	Yes	-------	$\frac{1}{1} = 1$	Yes

R2	Pclass=3, Age=9, Fare=3.2, Embarked=S □Sex = Male	No	R1	$\frac{4+4-2*4+0.11}{4+4} =$ 0.013	No
R3	Pclass=3, Age=25, Fare=7.9, Embarked=S □Sex = Male	No	R1	$\frac{4+4-2*4+0.04}{4+4} = .0$	No
R4	Pclass=2, Age=12, Fare=15.8, Embarked=S □Sex = Female	Yes	R1	$\frac{1}{1} = 1$	Yes
R5	Pclass=2, Age=20, Fare=36.7, Embarked=S □Sex = Female	No	R1, R4	$\frac{4+4-2*4+019}{4+4} = .0$	No

Note that, due to high value of novelty threshold, only two rules (R1 and R4) are considered novel and therefore used to update the model at time T1.

7. Experimental Results

In this section, experimental results are presented to show the performance of the proposed approach. The proposed approach is written in python programming language and implemented on Hadoop. The experiments were conducted using public datasets available at [URL:https://www.kaggle.com/ and https://archive.ics.uci.edu/dataset/]. The datasets are considered to be evolving over time. They are divided into three chunks, named D1, D2, and D3. Dataset D1 arrived at time *T*1, dataset D2 arrived at time *T*2, and dataset D3 arrived at time *T*3. The proposed approach is compared with several incremental clustering algorithms. The following subsections provide details of the experiments that were performed.

7.1 Experiment 1

The first experiment is conducted on the Titanic dataset available at [https://www.kaggle.com/c/titanic] to show the performance of the proposed approach in terms of accuracy, running time, and speedup against Serial K-Means, Parallel K-Means (1 thread/core) and Parallel K-Means (2 thread/core). This dataset contains 891 instances and 12 mixed attributes. The performance of the proposed approach against the well-known parallel clustering algorithms are shown in Table10.

Table 10 . The performance of the proposed approach against the well-known parallel clustering algorithms.

Algorithm	Accuracy	Running time (s)	Speedup
Serial K-Means	0.81	100	1
Parallel K-Means (1 thread/core)	0.81	50	2
Parallel K-Means (2 threads/core)	0.81	25	4
The proposed algorithm (2 threads/core)	0.81	25	4

As you can see in Table10, the proposed parallel clustering algorithm and traditional parallel K-Means algorithm achieve the same clustering accuracy as the serial K-Means algorithm, but they run much faster. The speedup increases with the number of threads used. For example, the proposed parallel clustering algorithm and parallel K-Means with 2 threads/core is 4 times faster than serial K-Means. The results confirm that the proposed parallel clustering algorithm is a very effective way to improve the performance of clustering algorithms on large datasets.

The proposed parallel clustering algorithm achieve the same clustering accuracy as the serial K-Means algorithm and traditional parallel K-Means algorithm. This is because the parallel algorithms are able to divide the data into smaller sub-problems that can be processed independently. Also, the parallel clustering algorithms run much faster than the serial K- Means algorithm. The speedup increases with the number of threads used. This is because the parallel algorithms can take advantage of multiple cores to process the data simultaneously.

Further more, the parallel clustering algorithms are more scalable than the serial K-Means algorithm. This means that they can be used to cluster larger datasets without a significant decrease in performance.

7.2 Experiment 2

The second experiment shows the results the proposed approach against several incremental clustering algorithms on the Titanic dataset as shown in Table 11.

Table 11. shows the results the proposed approach against several incremental clustering algorithms on the Titanic dataset

the Algorithm	Accuracy	Time (seconds)
DBSCAN	81.5%	100
BIRCH	80.5%	50
CURE	79.5%	25
OPTICS	81.0%	75
ROCK	79.0%	30

The proposed algorithm	81.5%	22

As you can see in Table 11, the proposed algorithm and DBSCAN achieved the highest accuracy, followed by OPTICS, BIRCH, CURE, and ROCK. However, DBSCAN was also the slowest algorithm, taking 100 seconds to cluster the Titanic dataset. OPTICS was slightly slower than DBSCAN, taking 75 seconds. BIRCH, ROCK, and CURE were much faster, taking 50, 30 and 25 seconds respectively. The proposed approach was the fastest algorithm, taking only 22 seconds. It is important to note that these results are just for one particular dataset. The performance of different incremental clustering algorithms may vary depending on the dataset.

7.3 Experiment 3

The third experiment aims to demonstrate the effectiveness of our framework in reducing the number of discovered rules compared to the Incremental K-means++ algorithm. We used four datasets and considered them to be evolving over time. We divided them into three increments: D1, D2,and D3, which were mined at times T1, T2, and T3, respectively. Suppose that the number of clusters required is 3 (k=3) and we varied the novelty threshold Φ. The number of interesting rules are decreased in our approach, in contrast to the number of rules discovered by the Incremental K-means++ algorithm at T1, T2, and T3.

Intuitively, the interesting rules discovered by our approach at time T1 are no longer interesting at time T2, and the interesting rules discovered at time T2 are no longer interesting at time T3.

Consequently, as the value of the novelty threshold Φ increases, the number of discovered interesting rules decreases at each time, as expected. Table 12 shows the comparison between our approach against the Incremental K-means++ algorithm in terms of number of discovered rules.

8. Conclusions and Future Works

We propose a clustering mining approach that can be adapted to changes in evolving data, scale to large data sets, and incorporate interestingness criteria. This is done by incrementally clustering the data and by using parallel processing to speed up the clustering process. The interestingness criteria are used to guide the clustering process and to identify clusters that are of the users' interest to the user. Our future work will focus on enhancing our approach to create a clustering model that can adapt to data stream environments. This means that the model will be able to learn from new data as it arrives and update the clusters accordingly. This is important for applications where the data is constantly changing, such as financial trading or sensor networks.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this paper.

Author contributions

Ahmed S. Al-Hegami: Conceptualization, Methodology, review.

Akram A. M. Mustafa: Data collection, Methodology, Writing-Original draft preparation, Software, Validation, Investigation, implementation of software writing the draft.

Abdulmajed A. G. Al-Khulidi: Visualization, Reviewing and Editing.

Table 12 . The comparison between our approach against the Incremental K-means++ algorithm in terms of number of discovered rules

Dataset	Size of dataset	Incremental K- means++	Novelty threshold Φ	The proposed approach		
		No. of discovered patterns		No. of novel patterns		
		T1+T2+T3		T1	T2	T3
Adult	48,842	5194	0.5	4012	2612	835
			0.6	3124	2212	723
			0.7	2712	1234	543
			0.8	827	728	423
			0.9	176	155	102
Customer	200,000	93087	0.5	27596	16896	9833
			0.6	19283	8013	6234

Segmentation			0.7	9834	4321	3432
			0.8	2008	1934	2192
			0.9	725	634	433
Online Retail	541909	133878	0.5	80754	33233	14233
			0.6	65744	14622	3454
			0.7	44333	6223	1342
			0.8	11922	2034	879
			0.9	2349	1523	221
Social Media Sentiment	1000000	332614	0.5	99233	56433	23965
			0.6	74231	43212	12864
			0.7	49111	21876	6235
			0.8	18222	12986	2001

References

[1] Absalom E. Ezugwu, Abiodun M. Ikotun, Olaide O. Oyelade, Laith Abualigah, Jeffery O. Agushaka, Christopher I. Eke, Andronicus A. Akinyelu,(2022).A comprehensive survey of clustering algorithms: State-of- the-art machine learning applications, taxonomy, challenges, and future research prospects, Engineering Applications of Artificial Intelligence, Volume 110,2022, 104743,ISSN 0952-1976, https://doi.org/10.1016/j.engappai.2022.104743.

[2] Amber Abernathy, M. Emre Celebi, (2022). The incremental online k-means clustering algorithm and its application to color quantization. Expert Systems with Applications, Volume 207, 2022, 17927,ISSN 0957-4174, https://doi.org/10.1016/j.eswa.2022.117927.

[3] Abiodun M. Ikotun, Absalom E. Ezugwu, Laith Abualigah, Belal Abuhaija, Jia Heming. (2023).K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data,Information Sciences, Volume 622, Pages 178-210, ISSN 0020-0255,https://doi.org/10.1016/j.ins.2022.11.139.

[4] Redha Mutar, J. (2022). A Review of Clustering Algorithms. International Journal of Computer Scienceand Mobile Applications (IJCSMA), 10(10), 44–50. https://doi.org/10.5281/zenodo.7243829.

[5] Bhatnagar, V. Al-Hegami, A. S. and Kumar, N., (2005). A hybrid approach for Quantification of Novelty in Rule Discovery", In Proceedings of International Conference on Artificial Learning and Data Mining (ALDM'05).

[6] Cicirelli, F., Nigro, L., Pupo, F. (2023). Performance of Parallel K-Means Based on Theatre. In: Yang, XS., Sherratt, S., Dey, N., Joshi, A. (eds) Proceedings of Seventh International Congress on Information and Communication Technology, DOI:10.1007/978-981-19-2397-5_24.

[7] N Kerdprasop, N., Kerdprasop, K. (2003), Data partitioning for incremental data mining. In proceedings of 1st International Forum on Information and Computer Science, 114-118.

[8] Rasim M. Alguliyev, Ramiz M. Aliguliyev, Lyudmila V. Sukhostat, (2021). Parallel batch k-means for Big data clustering", Computers & Industrial Engineering, Volume 152.

[9] Bagirov, A.M. &Ordin, Burak & Ozturk, Gurkan& Xavier, Adilson. (2015). An incremental clustering algorithm based on hyperbolic smoothing. Computational Optimization and Applications. 61. 10.1007/s10589- 014-9711-7.

[10] Mai, S. T., Jacobsen, J. , Amer-Yahia, S. Spence, I. and Nhat-Phuong, T. (2022). Incremental Density-Based Clustering on Multicore Processors, in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 3, pp. 1338-1356.

[11] Prasad, R.K., Sarmah, R., Chakraborty, S. (2019). Incremental k-Means Method In: Deka, B., Maji, P., Mitra, S., Bhattacharyya, D., Bora, P., Pal, S.(eds) Pattern Recognition and Machine Intelligence. PReMI 2019. Lecture Notes in Computer Science(), vol 11941. Springer, Cham.

[12] Kushwah, A.P., Jaloree, S., & Thakur, R.S. (2021). A Comparative Review of Incremental Clustering ٧٥٩ Methods for Large Dataset. *International Journal of Advanced Trends in Computer Science and*

Engineering. Volume 10, No.2.

[13] Özyer, T., & Alhajj, R. (2009). Parallel clustering of high dimensional data by integrating multi-objective genetic algorithm with divide and conquer. Applied Intelligence, 31, 318-331.

[14] Zhao, W., Li, L., Alam, S., Wang, Y. (2021). An incremental clustering method for anomaly detection in flight data, Transportation Research Part C: Emerging Technologies, Volume 132, 2021,103406.

[15] Zhu, W., Yu, W., Kan, B., & Liu, G. (2017). Smart Meter Data Analytics Based on Modified Streaming k- Means. 2017 3rd International Conference on Big Data Computing and Communications (BIGCOM), 328-333. https://doi.org/10.1109/BIGCOM.2017.49.

[16] Chen, L., & Cao, L. (2018). Learning incremental model-based online clustering. IEEE Transactions on Neural Networks and Learning Systems, 30(6), 1795–1809.

[17] Qian, W., Cao, F., & Ester, M. (2016). Incremental density-based clustering over evolving data streams with concept drift. IEEE Transactions on Knowledge and Data Engineering, 28(7), 1729–1743.

[18] Chakraborty, S., Nagwani, N.K. (2011). Analysis and Study of Incremental K-Means Clustering Algorithm. In: Mantri, A., Nandi, S., Kumar, G., Kumar, S.(eds) High Performance Architecture and GridComputing. HPAGC 2011. Communications in Computer and Information Science, vol 169. Springer, Berlin.

[19] Al-Sai, Z. & Abdullah, R. & Husin, H. (2019). Big Data Impacts and Challenges: A Review. In Proceedings of conference: 2019 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT), Amman- Jordan DOI:10.1109/JEEIT.2019.8717484

[20] Alsaeedi, H. and Alhegami, A. S., (2022). An Incremental Interesting Maximal Frequent Itemset Mining Based on FP-Growth Algorithm ", Journal of Complexity, Volume 2022, Article ID 1942517.

[21] Alhegami, A. S., &Alsaeedi, H. (2020). A framework for incremental parallel mining of interesting association patterns for big data. *International Journal of Computing*, *19*(1), 106-117.

[22] Lu, W. (2020). Improved K-Means Clustering Algorithm for Big Data Mining under Hadoop Parallel Framework. *Journal of Grid Computing*, *18*, 239-250.

[23] Reddy, K., & Bindu, C. (2019). StreamSW: A density-based approach for clustering data streams over sliding windows. Measurement. https://doi.org/10.1016/J.MEASUREMENT.2018.11.041.

[24] Mudassirm K. & Aadarsh, M. & Mahtab, A. (2019). Map Reduce Clustering in Incremental Big Data Processing. International Journal of Innovative Technology and Exploring Engineering 9(2):2278-3075. 2019.

[25] Hayatu, I., H., Mohammed, A., BarroonIsma'eel, A. (2021). Big Data Clustering Techniques: Recent Advances and Survey. In: Chiroma, H., Abdulhamid, S.M., Fournier-Viger, P., Garcia, N.M. (eds) Machine Learning and Data Mining for Emerging Trend in Cyber Dynamics. Springer, Cham. https://doi.org/10.1007/978-3-030-66288-2_3

[26] Fawzia Omer, A., Mohammed, H.A., Awadallah, M.A., Khan, Z., Abrar, S.U., Shah, M.D. (2022). Big Data Mining Using K-Means and DBSCAN Clustering Techniques. In: Ouaissa, M., Boulouard, Z., Ouaissa, M., Khan, I.U., Kaosar, M. (eds) Big Data Analytics and Computational Intelligence for Cybersecurity. Studies in Big Data, vol 111. Springer.

[27] Sahoo, S. (2017). A Parallel Forecasting Approach Using Incremental K-means Clustering Technique. In:Behera, H., Mohapatra, D. (eds) Computational Intelligence in Data Mining. Avances in Intelligent Systems and Computing, vol 556. Springer, Singapore.

[28] Bakr, A., Ghanem, N., & Ismail, M. (2015). Efficient incremental density-based Algorithm for clustering large datasets. Alexandria engineering journal, 54, 1147-1154.

[29] Chakraborty, S., &Nagwani, N. (2011). Analysis and Study of Incremental K-Means Clustering Algorithm. , 338-341. https://doi.org/10.1007/978-3-642-22577-2_46.

[30] Liberty, E., Sriharsha, R., & Sviridenko, M., (2014). An Algorithm for Online K-Means Clustering, 81-89 , https://doi.org/10.1137/1.9781611974317.7.

[31] Liberty, Edo & Sriharsha, Ram & Sviridenko, Maxim. (2014). An Algorithm for Online K-Means Clustering. 10.1137/1.9781611974317.7.

[32] Li, Y., Macready, W. G., & Chen, G. (2015). Streaming k-means approximation. In Proceedings of the 2015 SIAM International Conference on Data Mining (SDM'15) (pp. 189-197).

[33] Cao, Feng & Ester, Martin & Qian, Weining& Zhou, Aoying. (2006). Density-Based Clustering over an Evolving Data Stream with Noise. In Proceedings of the 2006 SIAM International Conference on Data Mining (SDM)

[34] Charu C. Aggarwal, Philip S. Yu, Jiawei Han, Jianyong Wang, (2003). A Framework for Clustering Evolving Data Streams, In Proceeding of 29th International Conference on Very Large Data Bases, VLDB 2003

[35] Al-Khamees, H. Al-A'araji, A., N. and Al-Shamery, E. S., (2021). Survey: Clustering Techniques of Data Stream," *2021 1st Babylon International Conference on Information Technology and Science (BICITS)*, Babil, Iraq, 2021, pp. 113-119, doi: 10.1109/BICITS51482.2021.9509923.

[36] Amini, M, Saboohi, H., Herawan, T., Wah, Y. (2016). MuDi-Stream: A multi density clustering algorithm for evolving data stream, Journal of Network and Computer Applications, Volume 59, 2016,Pages 370-385

[37] Azhir, E., Jafari, N., N., Hosseinzadeh, M., Sharifi, A., and Darwesh, A., (2021). An efficient automated incremental density-based algorithm for clustering and classification", Future Generation Computer Systems,Volume 114, Pages 665-678

[38] Shao, J., Tan, Y., Gao, L., Yang, Q., Plant, C., & Assent, I. (2019). Synchronization-based clustering on evolving data stream. Inf. Sci., 501, 573-587. https://doi.org/10.1016/J.INS.2018.09.035.

[39] Madan, S., & Dana, K. (2015). m-BIRCH: An online clustering approach for computer vision applications., 9408. https://doi.org/10.1117/12.2078264.

[40] Fu, JS., Liu, Y. & Chao, HC. (2015). ICA: An Incremental Clustering Algorithm Based on OPTICS. *Wireless Pers Commun*84, 2151–2170 https://doi.org/10.1007/s11277-015-2517-9

[41] Abiodun, I., & Absalom, E., & Laith, A., & Belal, A., & Jia, H. (2022). K-means Clustering Algorithms: A Comprehensive Review, Variants Analysis, and Advances in the Era of Big Data. Information Sciences. 622.10.1016/j.ins.2022.11.139.

[42] Mulay, P., & Kulkarni, P. (2013). Knowledge augmentation via incremental clustering: new technology for effective knowledge management. Int. J. Bus. Inf. Syst., 12, 68

[43] Tran, T., Nayak, R., Bruza, P. (2008). Document Clustering Using Incremental and Pairwise Approaches. In: Fuhr, N., Kamps, J., Lalmas, M., Trotman, A. (eds) Focused Access to XML Documents. INEX 2007. Lecture Notes in Computer Science, vol 4862. Springer, Berlin, Heidelberg.

[44] Sowjanya, A. M., & Shashi, M. (2011). A cluster feature-based incremental clustering approach to mixed data. *Journal of Computer Science*, *7*(12), 1875.

[45] Zhao, W., Ma, H., He, Q. (2009). Parallel *K*-Means Clustering Based on MapReduce. In: Jaatun, M.G., Zhao, G., Rong, C. (eds) Cloud Computing. CloudCom 2009. Lecture Notes in Computer Science, vol 5931. Springer, Berlin, Heidelberg

[46] Son, L.H., Tien, N.D. (2017).Tune Up Fuzzy C-Means for Big Data: Some Novel Hybrid Clustering Algorithms Based on Initial Selection and Incremental Clustering. *Int. J. Fuzzy Syst.* 19, 1585–1602.

[47] Najdataei, H., Gulisano, V., Tsigas, P., &Papatriantafilou, M. (2022). pi-Lisco: parallel and incremental stream-based point-cloud clustering. *Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing*.