International Journal of Computer Applications (0975 – 8887)
Volume 184 – No.7, April 2022

Mining Data Streams using Clustering Techniques

Manal Mansour Alharthi
King Abdulaziz University
Jeddah
Saudi Arabia

Manal Abdulaziz Abdullah
King Abdulaziz University
Jeddah
Saudi Arabia

ABSTRACT

This research highlights the significant aspects to consider when building a framework for mining the data streams. The main aspects include the methods of data summarizing and creating synopsis. Besides, the main approaches of clustering the data synopses. The research also shows some real applications and tools for mining the streaming data. The main goal is to present the ClusKmeans model as an example for data stream clustering, and to study its performance with different scenarios of data speed, noise levels, and horizon ranges.

Keywords

Data Streams, Data Stream Mining, ClusKmeans, Data synopsis, Pyramidal window, Micro-clusters

1. INTRODUCTION

The amounts of existed data globally almost reached to 64 zettabytes by the end of 2020, and the forecasts indicate that it will grow to 175 zettabytes by the end of 2025[1]. Thisis leadsto produce new data mining techniques that could process the huge infinite data flows coming from different sources such as sensors data, network traffic, call center records, social media posts, ATM and credit card operations, and web searches. All these sources produce continuous sequences of massive data that generated in high speed rate and known as data stream[2].

A data stream consists of infinite sequence of elements that keep flowing with high speed rate. Every element of a stream is a pair of (Xi,Ti) where Xi is a dimensional vector arrives at time stamp Ti which identifies when the element occurred[3].

Mining the streaming data is a continuous process of extracting knowledge from infinite and rapid data signals. The main goal is to predict the class or the value of the new elements using machine learning techniques such as clustering, classification, regression, etc[2]. All these techniques work differently with streaming data because of their different characteristics such as data volume which keep increasing continually, data volatility which reflects the different types of data values, and data velocity that reach very high ranges. Therefore, the system requiresspecific methods to summarize, store, and process the data stream[3].

This paper will study the whole process of mining the data stream, starting with the data generators, the different methods of summarizing the data stream, and the different approaches of clustering the data stream. It will also review some real applications of mining the data stream and some contributions in the related work. The goal is to study the process of mining the data stream and present a model for data stream clustering.

2. GENERAL MODEL FOR MINING DATA STREAMS

Many applications were developed for mining the streaming data, and there are common steps in their architectures which are: data generating, initializing, sampling and processing. These common steps combined to define the general data streams model presented in Figure 1.

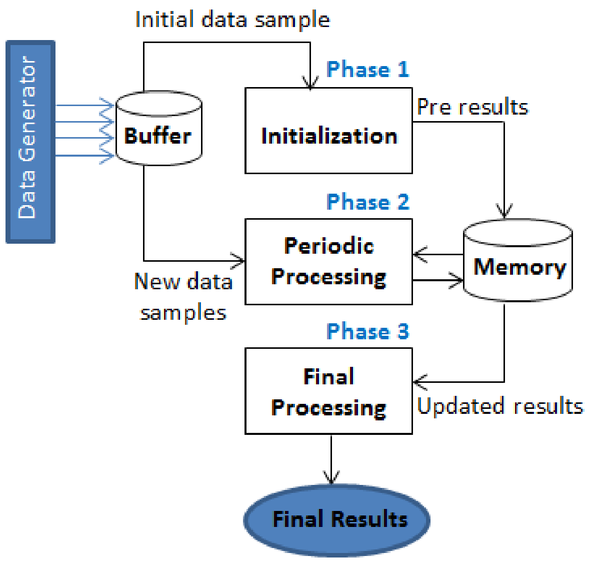

Figure 1 General Model for mining data streams

The general framework presents the process of mining data streams following the common steps of the existing models. First, streaming data flows to the system continuously in massive volumes and high velocity. Since the system is unable to process the whole stream, it takes some initial data, processes it, produce approximated results as a summary, and store it in memory. Then, the system takes different samples from the new coming data periodically, and updates the data summary stored in memory. Finally, only the updated summaries will be processed to introduce the final results instead of using the raw data comes from the stream[4].

3. DATA STREAM SYNOPSIS

The characteristics of streaming data make it impossible to store the whole stream in memory. Instead, the data synopses areconstructedas a summary of data stream. Next sections will review some methods of constructing the data synopsis.

3.1 Sampling

Sampling is one of the simplest methods for constructing synopsis from data streams. The process of sampling data stream is based on taking a subset of data elements that estimate the whole data stream statistical factors, like mean, variance, probability distribution, etc [2].

Random sampling is the simplest example, where all data instances have equal probability to be included in the sample. One technique for random sampling is the reservoir algorithm [5] which randomly collects data sample of size k and keep updating this sample. Every new instance in the data flow has a probability p=1/n to replace an old instance, where n is the number of instances passed in the stream. Other sampling techniques include systematic sampling [6], stratified sampling[7], and cluster sampling[8].

4.2 Windowing

Data stream windowing is a process of segmenting the data stream into small packets at certain time t with specific size w [2]. Windowing approach helps to deals with concept drift which reflects the change of the data stream over time. The content of the window continues to change and gives updated results [9]. Figure 2shows the windows types and how it works.

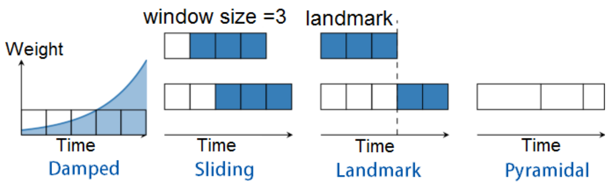

Figure 2: Types of Data Stream Windowing [9]

All types of windows commonly update the data periodically, but they differ in the updating procedure as follow:

- **Damped Window**[10] gives specific weight value for each arrived data element based on time. New data elements have higher weight values which keep decreasing over time based on ageing function.
- **Sliding Window** is more suitable for applications that interested in the newest information because it considers only the recent data of the stream [10]. The window is updated by deleting the old data and replacing it with newest data. All the data elements in the updated window have the same weight which means having the same importance. The small size windows give better accuracy in high changing environment, while the larger size works better in the stable data environment [9].
- **Landmark Window**works by segmenting the data stream into chunks based on landmarks. Landmark could reflect the data events or the passed time. For example, the landmark defined as the starting point that could be (every 100 elements) which means updating the window after receiving 100 new data points and removed all the previous 100 data points. The window summarizes all data points that came after the landmark and gives them the same level of importance [9]
- **Pyramidal Window**also known as tilted window. It registers the data at different times with different levels of granularity. The new data points are added to the window and aggregated with the old data. In contrast of sliding window, titled window does not completely discard the old data, but integrate all the data and give more importance to the recent data [9][10].

3.2 Histograms

Histogramsare based on capturing the frequency distribution of the attributes values in the stream. Instead of store the frequency for each data value, the domain of data values are divided into subsets called buckets. Then, the average frequency is computed and stored for each bucket [2].

Different types of histograms are developed with different manners of dividing the data stream domain into buckets. The histogram with less variance is better, but it requires more space capabilities to implants. The variance equals the weighted sum of the original values for each bucket where the bucket's weight equals the number of values in that bucket[11][12]. Types of histograms are described as follow:

- **Equi-width histogram** is based on dividing the data domain into equal width buckets which means that every bucket has the same range of data pints numbers. It reduces the storage requirements and works well with simple skewed data distribution, but it is difficult to estimate the errors [11].
- **Equi-depth Histogram**is based on dividing the data domain into similar height buckets which mean that data points in certain bucket have almost the same frequencies. It requires the same storage of Equi-width histograms, but it is more complex to maintain. It Works well with low skewed data distribution and gives better accuracy than Equi-width histograms, but variance within a bucket may still high [11].
- **Variance-optimal Histogram**is based on dividing the data domain into buckets where the data frequencies in every bucket is either all greater than or all less than the frequencies of other buckets. It works well with high skewed data distributions and gives the minimum histogram variance, but it requires high memory capabilities[12].
- **MaxDiff Histogram**is based on dividing the data points with the highest frequencies and lowest frequencies in individual separated buckets, while the data with middle frequencies are all combined with one bucket. It stores only the frequencies of the attribute values in the individual buckets which require smaller storage than Variance-optimal histograms [12].
- **Compressed Histogram**is based on dividing the data points with high frequencies in singleton buckets while the rest data is dividedas equi-depth histogram. It produces great accuracy with skewed data distributions [12].

4.4 Sketching

The data sketch is a data structure used to record some information about some data samples in order to answer predefined queries [13]. The stored information could be the number of distinct values appears in the stream like introduced in Flajolet Martin Algorithm (FM Sketch[14], the most frequent elements seen in the stream like introduced in Count Sketch Algorithm [15], or frequencies of data events like introduced in Count–Min Algorithm (CM Sketch) [16]. The task of recording values frequencies is shared in CM sketching and histograms techniques discussed before, but histogram considers the frequencies of specific data sample, while CM sketching updates the frequencies of a data sample over the stream [13].

Sketching is much related to sampling technique since it uses some initials samples as basis to build the sketch. Every coming data point will be discarded after being used to update the sketch which means not storing all the data dimensions, but only the interested dimensions. This technique presents the sketching as a powerful method to compress data stream with high dimensions [13].

4. DATA STREAMS CLUSTERING

Clustering is a process of finding homogeneous groups among unlabeled data. The process of clustering the data streams consist of two phases: online and offline. In the online phase, the algorithm performs one pass over some current data streams, computes a set of micro-clusters, store them in memory, and update it using new subsets of the coming new data. In the offline phase, the algorithm has several passes only over the stored micro-clusters, processes it using traditional batch clustering [3]. Most popular algorithms for clustering data stream are:

- **Hierarchical Clustering** which works by grouping the data into a tree of clusters such as Clustree Algorithm[17], and Brich Algorithm[18].
- **Partitioning Clustering** which works by grouping the data into number of partitions which reflects the clusters. It works based on distance functions like k-means which used in StreamKM++ Algorithm [19].
- **Density Based Clustering** works based on density functions which assume dense areas of data points as clusters. DenSteam Algorithm[20]is a popular example of this approach which works based on summarizing clusters using dense micro-clusters called core-micro-cluster.
- **Grid Based Clustering** is based on computing the grid density. D-Stream Algorithm [21] is an example which works by segmenting the data space into finite number of cells assumed as grid structure. Then, mapping the recorded data to that grid andgrouping it based on their density.
- **Growing Neural Gas Clustering (GNG)** is based on topology learning that models the data space using interconnected units that located on the densest areas of that space. It produces a graph that can be represented on a two-dimensional plane showing the cluster data patterns [22].

5. RELATED WORK

The data stream analysis is a rich field and always required for further improvements. Several contributions were adopted to create, develop, and improve the algorithms of clustering data stream which reviewed in table 1.

Table 1.Algorithms for Data Stream Mining

Algorithm	Main Ideas
Growing neural gas based Algorithm (G-Stream) [22]	Using fading and exponential functions to weight the nodes and edges. Effective to discover clusters of arbitrary shape
Hyper-Ellipsoidal Clustering for Evolving data Stream (HECES) [23]	Using covariance shrinkage for data estimation, and heuristic technique for the initial clusters
A Privacy-Preserving Distinct Counting Scheme for Mobile Sensing (PPDC) [24]	Integrating homomorphic encryption into Flajolet-Martin (FM) sketch
Clustering Algorithm for geographical data streams (GeoDenStream) **[25]**	Enhancing DenStream Algorithm to support entity-based mining in geographical space
FlexSketch Algorithm [26]	Estimating the probability density for data streams based on histograms ensemble and updating the model using sketches

6. CLUSKMEANS MODEL

Cluskmeans is one of the powerful partitioning techniques for data streams clustering. It works through different phases start with initializing micro-clusters, maintaining those micro-clusters and finally creating the final macro-clusters.These previousclustering phases offer good flexibility to explore the evolution nature of the clusters over different time periods which means handling the data streams over time rather than clustering the whole stream at one time. In addition, it provides the cluster characteristics over different time horizons. Figure 3shows the detailed phases of the ClusKmens model.

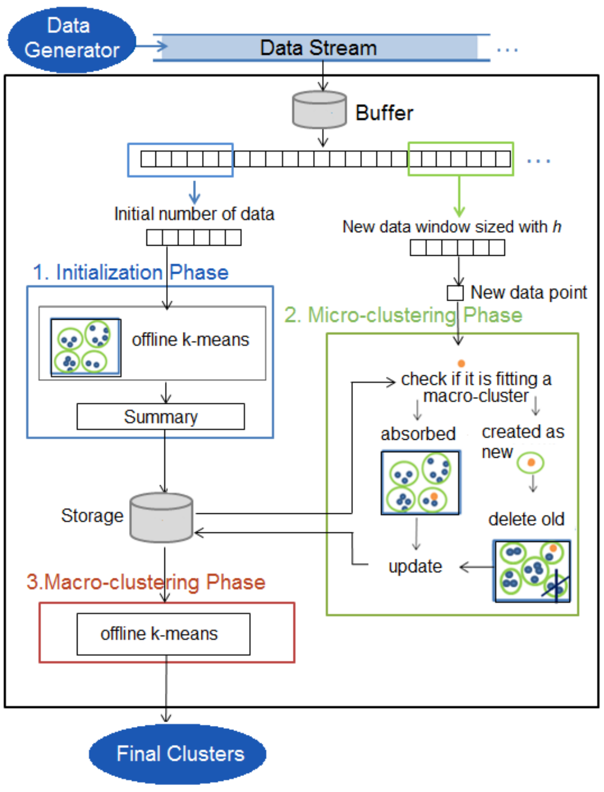

Figure 3: ClusKmeans Detail Model

6.1 The Initialization Phase

The initialization phase starts with processing some premier data at the beginning of clustering process to formulate the initial micro-clusters. This is done by applying offline k-means on a sized window of the initial data stream. The size of the initial data window is determined by fixed number which reflects the number of data instances to include in the window sample at the beginning of the initialization process. This process produces a predefined number of micro-clusters which used to store the data summary.

6.2 Micro-clustering phase

This phase processes a sized window of a new data points comes from the stream, stores them as snapshots at particular times, and updates the summary statistics of micro-clusters stored in the memory. Each instance comes from the stream is clustered using k-mean algorithm. The new generated micro-cluster is either absorbed by an existing micro-cluster, i.e. it is already exists in the k number of micro clusters, or it is created as a new micro-cluster. This new micro-cluster requires freeing some space in the memory and keeps k constant all over the process. This is done by deleting an old micro-cluster.

6.3 Macro-clustering phase

This phase applies a weighted k-means algorithm on the micro-clusters stored in the memory for specific time horizon instead of dealing with the running input stream. The time horizon reflects certain period of time which determined by the amount of stored data that arrived during specific time. Then, the results of this phase are the macro-clusters which present the final clustering result of the streaming data for this time horizon.

Previous phases shows how to take an initial data points from input data stream to start the initialization of micro-clusters, how to include the new data points to maintain the stored micro-clusters, and finally how to use the stored micro-clusters to introduce the final macro-clusters. All these phases are also described as steps in Algorithm1 which clarify the algorithm of ClusKmean model.

```
Algorithm 1: ClusKmean Pseudocode
Input➔
    • Time horizon $h$ – range of window
    • Max_num_kernals $m$ - max num of micro-clusters
    • Kernel_radi_factor $t$ – Multiplier for radius
    • $k$ - Number of macro cluster to be detected
➢ Initialize
    - Apply q-means over initial points
    - Built summary for each cluster
    - Store summary statistics
➢ Micro-clusters Creating
    - Micro-cluster maintenance as a new point p arrive
      from the stream $p$
    - Find the closest micro-cluster $clu$ by computing
      distance between $p$ and each of $q$ micro-clusters
      centers
        o If p is within the max boundary of $clu$, p is
          absorbed by $clu$
        o Otherwise, a new cluster is created with $p$
    - The number of micro-clusters should not exceed $m$
        o Delete old micro-cluster or merge the two
          closest clusters
➢ Periodic
    - Store micro-clusters snapshots in particular times
➢ Macro-clusters Creating
    - Locate the valid micro-clusters during $h$
    - Apply k-means upon these micro-clusters
  Output ➔
    • k macro-clusters
```

7. ClLUSKMEANS PERFORMANCE

The implementation of Cluskmeans studies the model performance based on different clustering measures such as purity, SSQ, homogametic, completeness, Etc. The study focuses on the impact of the data stream nature and how it affects the clustering quality. Besides, the effect of the horizon range that used during processing phase.

7.1 Clustering Evaluation Measures

Clustering evaluation metrics aim to assign a score to the clustering algorithms and study their performance. MOA provides important measurement such as purity, SSQ, homogametic, completeness, Etc.

7.1.1 Purity

Purity is a measure of the extent to which clusters contain a single class. It can be calculated by count the number of data points for each cluster from the most common class in said cluster. Then, sum the results over all clusters and divide by the total number of data points. The higher the better, where c is the number of clusters and N is the number of all points in the data set[3].

7.1.2 SSQ Measurement

Sum of Squares SSQ reflects how much the members are belonging to the cluster. It is computed by the sum of squared distances of the data items to their cluster centers. The lower distances are better, so the smallest SSQ is better [3].

7.1.3 Homogeneity

Homogeneity reflects that each cluster contains only members of a single class. In another words, the homogeneity represents how much the instances in a cluster are similar. The lower bound is 0.0 and the upper bound is 1.0 (higher is better). When all samples in cluster k have the same label c, the homogeneity equals 1[27].

7.1.4 Completeness

Completeness means that all members of a given class are assigned to the same cluster. In another words, completeness represents how much similar samples are put together by the clustering algorithm. The lower bound is 0.0 and the upper bound is 1.0 (higher is better). When all samples of kind c have been assigned to the same cluster k, the completeness equals 1[27].

7.2 Experiment Setup

ClusKmeans implementation is done using MOA toolwith RBF (Radial Basis Function) data generator which simulates the characteristics of the data stream nature.

The settings of the generator enable the model to simulate the real world conditions and control the data speed and noise levels. The parameter numCluster reflects the number of centroids in the model which defaults to 5.Data speed default to 500 which means that kernels move a predefine distance of 0.01 every 500 points. The noise levels reflect the amount of useless data which defaults to 0.1.

The settings of ClusKmean enable the model to control the horizon which means the amount of pointsto include in the data window during the processing phase. The maxNumKernals parameter reflects the number of micro-clusters created and maintained during processing phase. The parameter kernalRadiFactor reflects the multiplier for the kernel radius. The default settings are shown in table 2.

Table 2 Experiment Default Settings

Generator Settings		**ClusKmean Settings**	
numCluster	5	horizon	1000
speed	500	maxNumKernals	100
Noise_level	0.1	kernalRadiFactor	2

7.3 Experiments Results

The experiments process 100,000 instances through different scenarios related to the data stream nature and some factors of the clustering process. Due the changing nature of data, results may differ with every run process. Therefore, all experiments are tested and repeated 10 times to take the average results.

7.3.1 Scenario 1: Impact of Speed

This scenario studies the impact of raising the data speed. Table 3 shows the results while raisingthe speed incrementally from 100 to 1000 while keeping the other parameters as default. The experiment shows wobbly results of all measurements. The purity was vacillated between 0.73 and 0.98. Homogeneity was between 0.41 and 0.90. Completeness was between 0.69 and 0.95. SSQ was between 6.44 and 11.73. Figure 4present a column chart of the results values.

Table 3. Speed of Data Stream Impact

Experiment Fixed Sittings:
Noise=0.1, Horizon=1000, Number of Micro-clusters= 100

Speed	Purity	SSQ	Homogeneity	Completeness
100	0.98	7.71	0.90	0.90
200	0.85	8.49	0.76	0.85
300	0.79	7.97	0.50	0.80
400	0.87	11.73	0.49	0.79
500	0.89	7.23	0.73	0.83
600	0.73	10.60	0.41	0.69
700	0.98	7.10	0.69	0.69
800	0.90	6.94	0.76	0.85
900	0. 78	8.17	0.77	0.88
1000	0.90	6.44	0.81	0.95

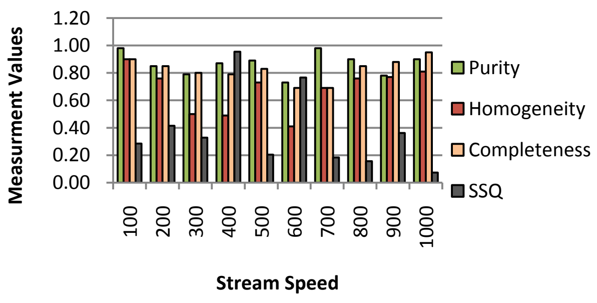

Figure 4: Speed Impact Colum Chart

7.3.2 Scenario 2: Impact of Noise level

Noise level reflects the amount of corrupt data which means the data contains large amounts of meaningless information, and the data that cannot be properly understood or interpreted. The results indicate that the higher the noise level become, the lower values of measurements are resulting.Table 4 shows the values of clustering measurements while raising the noise level. The purity equals 1 when removing all the noise and becomes worst while increasing the noise level. The higher noise levels affect all measurements negatively and give their worst values. Figure 5 shows the column chart of the noise impact on the measurement values. SSQ values being normalized to 0~1 values in order to cope with other measurement values in the line chart shown in Figure 6.

Table 4. Noise Level Impact

Experiment Fixed Sittings:
Speed= 500, Horizon=1000, Number of Micro-clusters= 100

Noise	Purity	SSQ	Homogeneity	Completeness
0	1.00	1.60	0.75	0.64
0.1	0.89	7.23	0.73	0.83
0.22	0.82	14.87	0.69	0.79
0.333	0.70	18.70	0.56	0.67
0.4444	0.69	21.77	0.63	0.54
0.5555	0.54	27.58	0.57	0.29

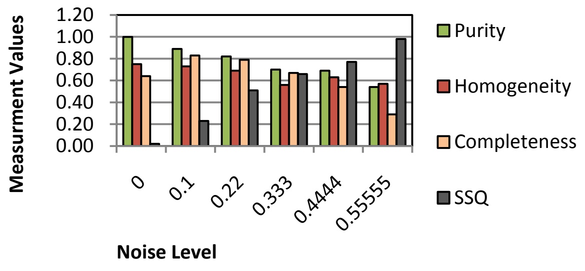

Figure 5: Noise Level Impact Colum Chart

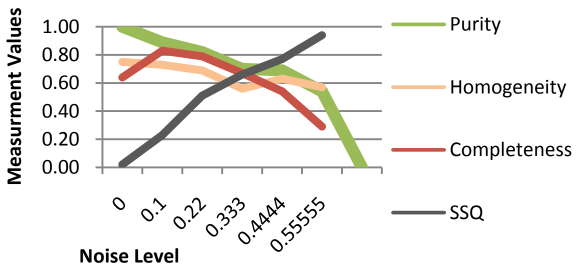

Figure 6: Noise Level Impact Line Chart

7.3.3 Scenario 4: Impact of Horizon

This scenario focuses on the change of the horizon range which reflects the window size of the data being processed with every iteration round. Table 5shows the experiment of running the ClusKmeans model with the default generator settings to notice the impact of change the window size. The experiment shows better results of purity, homogeneity, and completeness when using a window with size range between: 500-600. SSQ results were enhanced while raising the window size to 900. Figure 7 presents all the results in a column chart.

Table 5. Horizon Impact 1

Experiment Fixed Sittings:
Speed= 500, Noise=0.1, Number of Micro-clusters= 100

Range of window	Purity	SSQ	Homogeneity	Completeness
100	0.91	10.35	0.64	0.44
200	0.86	8.64	0.73	0.72
300	0.85	8.67	0.71	0.72
400	0.88	8.52	0.76	0.77
500	**0.91**	**8.61**	**0.84**	**0.85**
600	**0.92**	**8.53**	**0.83**	**0.83**
700	0.89	7.28	0.74	0.85
800	0.89	7.26	0.74	0.83

900	**0.89**	**7.20**	**0.74**	**0.83**
1000	0.89	7.23	0.73	0.83
1100	0.90	7.26	0.73	0.84
1200	0.89	7.22	0.74	0.84

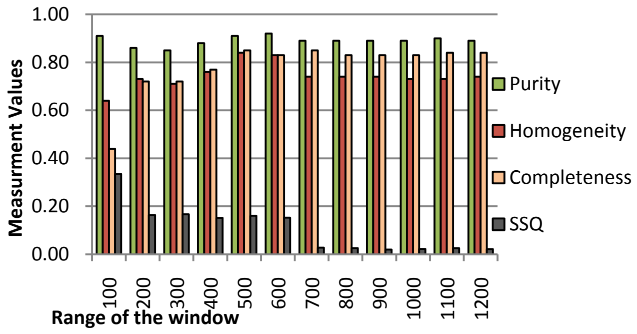

Figure 7: Horizon Impact Colum Chart 1

Table 6 shows the experiment of the horizon impact while running the ClusKmeans model with the default generator settings but raising the speed to 1000. The experiment shows better results in all measurement while using a window of size of 1000. Figure 8 presents all the results in a column chart.

Table 6. Horizon Impact 2

Experiment Sittings:
Speed=1000, Noise=0.1, Number of Micro-clusters= 100

Range of window	Purity	SSQ	Homogeneity	Completeness
100	0.85	11.67	0.55	0.84
200	0.89	7.79	0.65	0.79
300	0.89	7.13	0.75	0.86
400	0.89	6.53	0.78	0.92
500	0.89	6.49	0.75	0.87
600	0.89	6.52	0.77	0.89
700	0.88	6.47	0.75	0.86
800	0.88	6.55	0.75	0.87
900	0.89	6.50	0.77	0.91
1000	**0.90**	**6.44**	**0.81**	**0.95**
1100	0.90	6.44	0.77	0.89
1200	0.82	10.15	0.48	0.77

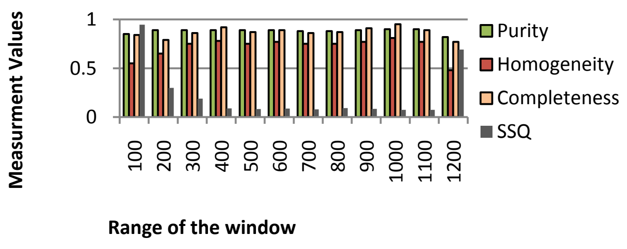

Figure 8: Horizon Impact Colum Chart 2

Table 7 shows the experiment of the horizon impact while running the ClusKmeans model with the default generator settings but, raising the speed to 1000 and the noise level to 0.333. The experiment results were wobble values. The purity was vacillated between 0.72 and 0.83. Homogeneity was between 0.52 and 0.71. Completeness was between 0.59 and 0.64. SSQ was between 23.29 and 29.48. Figure 9 present all the results in a column chart.

Table 7. Horizon Impact 3

Experiment Sitting:
Speed=1000, Noise=0.333, Number of Micro-clusters= 100

Range of window	Purity	SSQ	Homogeneity	Completeness
100	0.77	**29.48**	0.54	**0.59**
200	**0.83**	26.27	**0.71**	0.60
300	0.78	23.97	**0.52**	0.60
400	**0.72**	23.81	0.55	0.62
500	0.75	23.61	0.54	0.63
600	0.79	23.53	0.54	0.61
700	0.76	23.57	0.54	0.61
800	0.74	24.63	0.54	0.60
900	0.76	**23.29**	0.55	0.62
1000	0.74	24.18	0.54	0.55
1100	0.77	23.71	0.56	0.61
1200	0.73	24.03	0.54	**0.64**

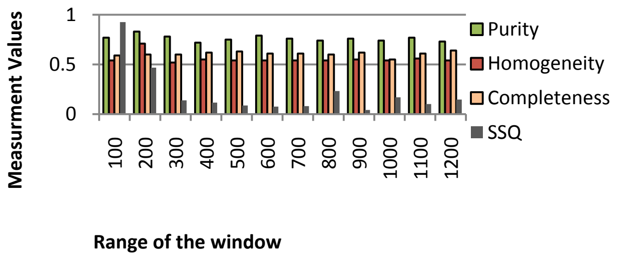

Figure 9: Horizon Impact Colum Chart 3

8. CONCLUSION AND FINDINGS

ClusKmeans model ispresented as an example for creating clusters among the data streams. It works through different phases: initialization, micro-clustering, and macro-clustering.

The Implementation of the ClusKmeans model is done using MOA tool based on different scenarios. The experiments scenarios focus on the impact of the data speed, noise levels, and the horizon. The results show that data speed doesn't affect the clusters quality as much as the high noise level do where it highly affects the clustering measures negatively. It may be concluded that when the noise increased by 55%, the purity decreased by 0.46%, the homogeneity by 0.24%, and the completeness by 0.54%. The SSQ also gives worst results where it increased by 1623%.On the aspect of horizon ranges, the experiments show better results when using a window with size range between: 500-600to create the data synopsis while the data speed is 500, and need to increase it while raising the speed.

The process of clustering the data stream keeps developing to enhance the clustering results. There are some suggestions that are recommended to extend the work:

- Applying the ClusKmeans model in a real data set and compare the results to assess the model performance
- Evaluating the ClusKmeans model with more than 100,000 instances
- Enhancing the initialization and micro-clustering phases by using other methods for data summarization and creating synopsis
- Studying the effects of data nature and the clustering factors with other evaluation metrics

9. REFERENCES

[1] "How Much Data Is Created Every Day," 28 October 2021. [Online]. Available: https://seedscientific.com/how-much-data-is-created-every-day/. [Accessed 18 12 2021].

[2] M. Garofalakis, J. Gehrke and R. Rastogi, in Data Stream Management: Processing High-Speed Data Streams, Springer, 2016.

[3] A. Bifet, R. Gavalda, G. Holmes and B. Pfahringer, Machine Learning for Data Streams:with Practical Examples in MOA, MIT Press, 2018.

[4] P. K. SRIMANI and M. M. PATIL, "Mining data streams with concept drift in massive online analysis frame work," WSEAS Trans. Comput, vol. 15, pp. 133-142, 2016.

[5] Gaber, Mohamed Medhat; Gama, Jo˜ao ; Krishnaswamy, Shonali; Gomes, Jo˜ao B´ artolo ; Stahl, Frederic;, "Data stream mining in ubiquitous environments: state-of-the-art and current directions," WIREs: Data Mining & Knowledge Discovery, vol. 4, no. 2, pp. 116-138, 2014.

[6] D. J. Brus and N. Saby, "Approximating the variance of estimated means for systematic random sampling, illustrated with data of the French Soil Monitoring Network," Elsevier, vol. 279, pp. 77-86, 2016.

[7] S. S. Ramkrishna and . P. S. Housila, "Efficient classes of estimators in stratified random," Statistical Papers, vol. 56, no. 1, pp. 83-103, 2015.

[8] W. Li , "Joint Image-Text News Topic Detection and Tracking by Multimodal Topic And-Or Graph," IEEE transactions on multimedia, vol. 19, no. 2, pp. 367-381, 2017.

[9] M. Carnein and H. Trautmann , "Optimizing data stream representation: An extensive survey on stream clustering algorithms," Business & Information Systems Engineering, vol. 61, no. 3, p. 277–297, 2019.

[10] E. Ntoutsi, N. Pelekis and Y. Theodoridis, "An evaluation of data stream clustering algorithms," Statistical Analysis and Data Mining, vol. 11, no. 4, pp. 167-187, 2018.

[11] Y. Ioannidis, "The history of histograms (abridged)," Proceedings, pp. 19-30, 2003.

[12] J. Gama and T. Mendonça, "Constructing fading histograms from data streams," PROGRESS IN ARTIFICIAL INTELLIGENCE, vol. 3, no. 1, pp. 15-28, 2014.

[13] R. Jayaram, "Sketching and Sampling Algorithms for," Carnegie Mellon University Pittsburgh, Pittsburgh, 2021.

[14] P. Flajolet and G. N. Martin, "Probabilistic counting algorithms for data base applications," Journal of computer and system sciences, vol. 31, no. 2, pp. 182-209, 1985.

[15] M. Charikar, K. Chen and M. Farach-Colton, "Finding frequent items in data streams," International Colloquium on Automata, Languages, and Programming, vol. 2380, pp. 693-703, 2002.

[16] D. Ting, "Count-Min: Optimal Estimation and Tight Error Bounds using Empirical Error Distributions," in Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018.

[17] J. Zgraja and M. Woźniak, "Drifted data stream clustering based on ClusTree algorithm," International Conference on Hybrid Artificial Intelligence Systems, vol. 10870, pp. 338-349, 2018.

[18] G. Pitolli, L. Aniello, G. Laurenza, L. Querzoni and R. Baldoni, "Malware family identification with BIRCH clustering," International Carnahan Conference on Security Technology (ICCST), pp. 1-6, 2017.

[19] M. R. Ackermann, M. Märtens, C. Raupach, K. Swierkot, C. Lammersen and C. Sohler, "StreamKM++: A Clustering Algorithms for Data Streams," Journal of Experimental Algorithmics (JEA), vol. 17, pp. 2-1, 2012.

[20] F. Cao, M. Estert, W. Qian and A. Zhou, "Density-Based Clustering over an Evolving Data Stream with Noise," in Proceedings of the 2006 SIAM International Conference on Data Mining (SDM).

[21] M. H. Ali, A. Sundus, W. Qaiser, Z. Ahmed and Z. Halim, "Applicative implementation of D-stream clustering algorithm for the real-time data of telecom sector," in International conference on computer networks and information technology, Abbottabad, Pakistan, 293-297.

[22] M. Ghesmoune, M. Lebbah and H. Azzag, "A new Growing Neural Gas for clustering data streams," Neural Networks, vol. 78, pp. 36-50, 2016.

[23] M. Z.-u. Rehman, T. Li, Y. Yang and H. Wang, "Hyper-ellipsoidal clustering technique for evolving data stream," Knowledge-Based Systems," Knowledge-Based Systems, vol. 70, pp. 3-14, 2014.

[24] X. Yang , M. Xu, S. Fu and Y. Luo , "PPDC: A Privacy-Preserving Distinct Counting Scheme for Mobile Sensing," Applied Sciences, vol. 9(18), p. 3695, 2019.

[25] M. Li, A. Croitoru and S. Yue, "GeoDenStream: An improved DenStream clustering method for managing entity data within geographical data streams," Computers & Geosciences, vol. 144, p. 104563, 2020.

[26] N. Park and S. Kim, "FlexSketch: Estimation of Probability Density for Stationary and Non-Stationary Data Streams," Sensors, vol. 21, no. 4, p. 1080, 2021.

[27] J. Han, M. Kamber and J. Pei, "10 - Cluster Analysis: Basic Concepts and Methods," in Data Mining (Third Edition), ISBN 9780123814791, 2012, pp. 443-495.