https://doi.org/10.29042/2020-10-5-124-128

Helix (2020) 10 (5): 124-128

Testing of Streaming Data Clustering Algorithm Effectiveness

✉ [1] Anis Fuatovich Galimyanov, [2] Nurgayaz Farhatovich Garifyanov, [3] Chulpan Bakievna Minnegalieva
[1] Department of Bilingual and Digital Education, Institute of Philology and Intercultural Communication
anis_59@mail.ru
[2] Institute of Computational Mathematics and Information Technology. Kazan Federal University
nf.garifyanov@gmail.com
[3] Department of Information Systems, Institute of Computational Mathematics and Information Technologies
Kazan Federal University
mchulpan@gmail.com

Received: 21st August 2020, Accepted: 14th September 2020, Published: 31st October 2020

Abstract
This article describes the task of streaming data clustering. The task of streaming data processing becomes more and more urgent with the device number increase that produces and process new data. Such devices create endless streams of data at tremendous speed. This article gives the examples of such data streams and the rationale for their processing need. Cluster flow analysis algorithms differ from classical algorithms due to RAM limitations of a computing device. Both artificial data sets and experimental observations were chosen for stream algorithm testing. The data of chemical gas sensors, as well as information about network connections in the local network, were chosen as such observations. Means and tools were chosen for comparisons between the algorithms. For these purposes, the WEKA and Massive Online Analysis software packages were selected. The article describes the process of working with this software. The data preprocessing process is demonstrated using WEKA. Several algorithms have been tested working with data streams. Clustering results were evaluated using an external quality measure. At the end of the work, they presented the graphs of this indicator changes during flow clustering.

Keywords
Clustering, Data Flow, MOA, WEKA

Introduction

The development of information technology, which includes both software improvement and computing power increase, ultimately leads to the number of areas increase that use them. Modern technologies can reduce their cost for the production of electronic devices. The decrease of semiconductor circuit manufacturing process can reduce their energy consumption and, accordingly, heat dissipation. All this contributes to the increase of different devices that create and process a large amount of data.

Processing and analysis of data allow us to gain new knowledge, which gives an advantage in the field to which this data relates. But this amount of data is too large to be processed manually, because the speed of its creation is very high. Thus, automatic methods are needed to process such data.

One of the methods for new knowledge gaining is cluster analysis. Cluster analysis is a statistical procedure, the purpose of which is to search for such groups of objects where the similarity of objects in one group would be maximum. Accordingly, the similarity between the objects of different groups should be minimal. Common applications include profitable market segment identification, anomaly detection, or sensor reading analysis. Most clustering algorithms require a static dataset and process each data point several times to create clusters.

In practice, many systems generate data constantly as a stream. The data stream can be represented as an infinite sequence $X = (x_1, x_2, \dots, x_N)$, where $x_t = (x_{t1}, x_{t2}, \dots, x_{td})$ – the observation data of dimension d at time t.

For example, sensors can produce thousands of observations per second, and social networks generate a huge number of interactions. To take into account new data points and possible changes in the structure of clusters, it is necessary to restart the clustering algorithms when new data arrives. This requires large computational costs, and at the same time, it is necessary to store the corresponding data to start the clustering process periodically. A more efficient approach is to upgrade existing clusters and integrate new observations into the existing model by new structure identification and gradual removal of obsolete structures (Carnein and Trautmann 2019).

The objective of this paper is to test various clustering algorithms. To perform this task, it is necessary to select various data sets, all the necessary means and tools.

 Helix E-ISSN: 2319-5592; P-ISSN: 2277-3495

Methods and Tools

The first step will be the selection of data on which we will test the effectiveness of clustering algorithms. We use artificially generated data as such data. We will also use the data obtained in the course of the real world phenomenon observations.

The next step will be data preprocessing, if it is the data obtained from the real world. The files containing the data for experiments will have ARFF format (Attribute-Relation File Format). To prepare data for cluster analysis, it is necessary to bring all the attributes to numerical and normalize these attributes so that their value range makes (Carnein and Trautmann 2019).

For this purpose, we will use the WEKA tool. This set is free software written in JAVA language, which is the means of visualization, processing, and data analysis. We will use this software to preprocess the data and prepare to use this data in the MOA package.

We use the Massive Online Analysis (MOA) framework to cluster streaming data. MOA is an open source software that enables machine learning or data mining experiments on evolving data streams. It includes a set of tools for testing various models, as well as stream generators that can be used from the graphical user interface, command line and Java API (Bifet et al. 2010).

1. Datasets

As was indicated above, we will use artificially generated data as the data set for experiments. To do this, select the Clustering tab, and specify Random RBF Generator Events as the data stream. This generator creates a data stream based on a radial basis function.

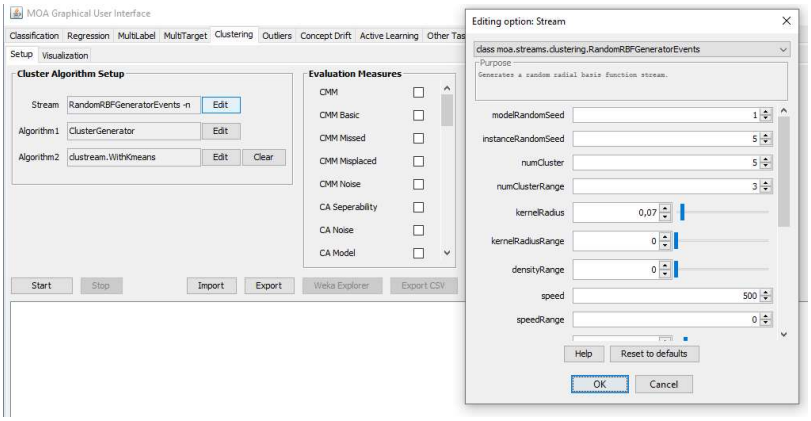

Fig. 1: MOA Interface. Data Flow Generator Parameter Settings

It is possible to specify various parameters for data generation. In this program, you can see the visualization of both generated and downloaded data streams.

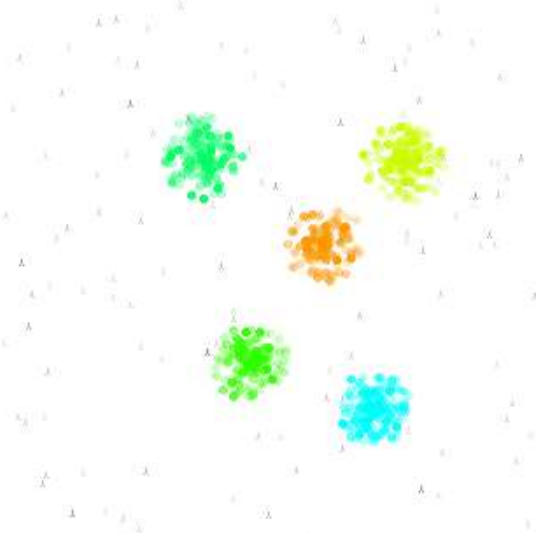

Fig.2: Data Stream Visualization

As real data, we take the readings of chemical gas sensors. This data set contains the readings of 16 sensors, which include 8 different parameters in each sensor. Thus, we get 128 attributes (Vergara et al. 2012).

 Helix E-ISSN: 2319-5592; P-ISSN: 2277-3495

The next step is to preprocess this data for use in the MOA package. To do this, we use the Explorer section. Next, you need to download the file with the data format ARFF or CSV. Next, we need to apply filters to this data. The first filter is normalization (unsupervised → attributes → Normalize). Since the data has a large dimension (128 features), it is necessary to reduce the data volume. To do this, we use the principal component method (PCA) (unsupervised → attributes → Principal Components). After these operations, we received the data set containing 13 attributes. Now you can save the finished data in the ARFF format.

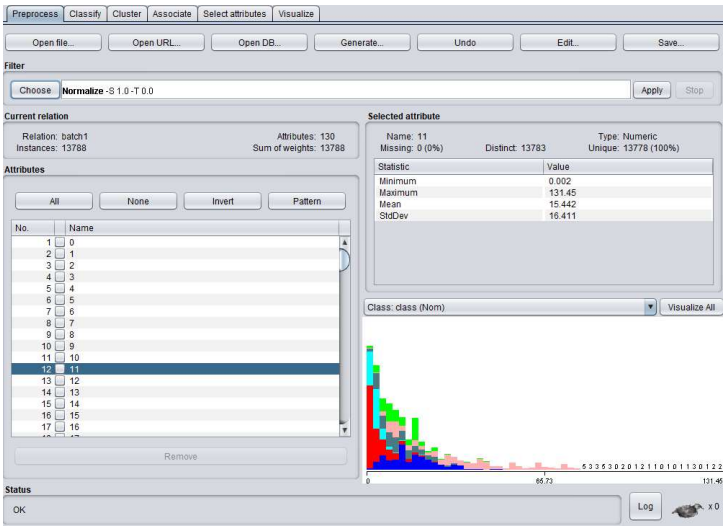

Fig.3: WEKA Interface for Data Preprocessing

KDD Cup 99 as another data set to test algorithms could be used. It is a dump of TCP data in the local network. Each entry represents connection information with 42 attributes. To preprocess this data set, we also carry out the procedure described above. After the removal of non-numeric attributes and this procedure performance, we received the data set containing 19 attributes.

2. Clustering Quality Evaluation

Algorithms will be compared based on clustering quality metrics. In our case, exact groupings are known for all data sets. This allows us to use an external quality assessment metric. We will use Purity as such a metric, which is calculated as follows:

$$Purity = \frac{1}{N}\sum_{i=1}^{k} \max_{j} |c_i \cap t_{ij}|, \quad (1)$$

Where N is the number of objects, k is the number of clusters, c_i are the objects in the i-th cluster, and t_{ij} is the number of objects of class j in the cluster i. Thus, clusters are associated with such class numbers in which the number of this class of objects is maximum.

3. Clustering Algorithms Comparison

To test the algorithms in the MOA, select the Clustering tab. In the Algorithm field, select the clustering algorithm we need, and select Purity in the Evaluation measures field. After starting the clustering process, you can go to the Visualization tab and see the following result:

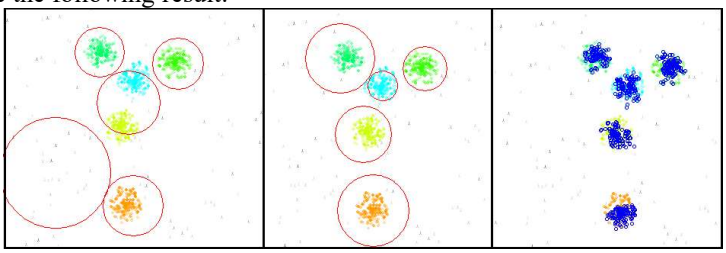

Fig. 4: CluStream, Clustree, DenStream

 Helix E-ISSN: 2319-5592; P-ISSN: 2277-3495

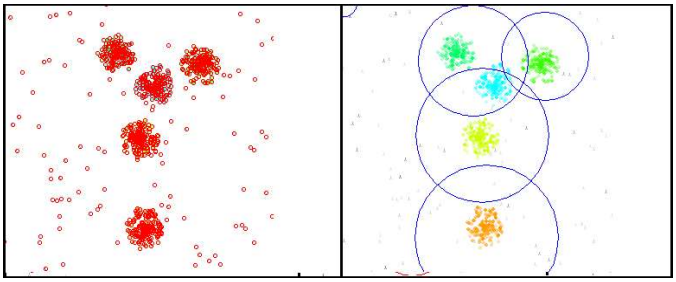

Fig.5: DStream, StreamKM

The visualization demonstrates that the DenStream and Clustree algorithms showed the best results with artificial data clustering task. The following is the graph of the Purity score change during data stream clustering of 20,000 elements for each of the algorithms.

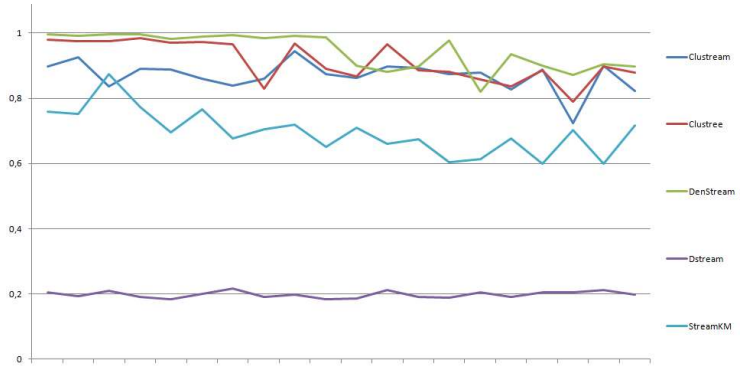

Fig. 6: Purity Score Change Graph for Artificial Data

Now let's move on to the comparison of these algorithms with real data. They are stored in ARFF format files. Before starting the clustering process, you must change the data source in Stream to clustering. FileStream. Here are the Purity evaluation graphs obtained after the clustering process:

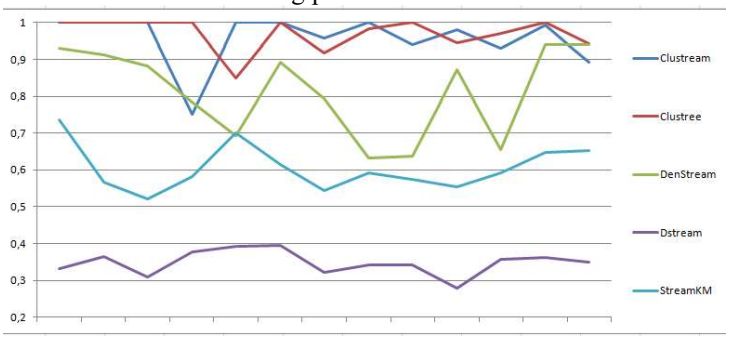

Fig.7: Purity values. Gas Sensor Data (13 Attributes, ~13000 Points)

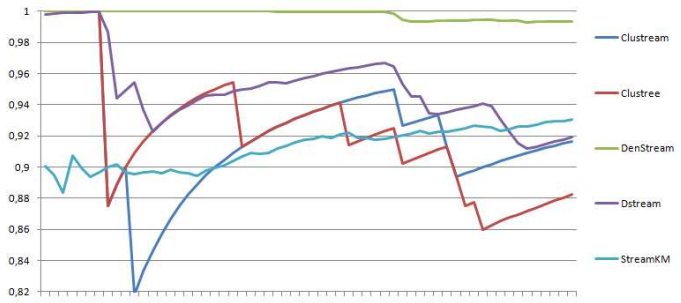

Fig.8: Purity Values. Network Data (19 Attributes, ~60000 Points)

 Helix E-ISSN: 2319-5592; P-ISSN: 2277-3495

Conclusions

This paper compares various clustering methods for streaming data. A comparison of the used methods for streaming data processing is performed using clustering quality metrics. The tools that were used to test these methods were described.

We tested these algorithms on various data sets. All three tests showed that the DenStream algorithm provided good results, but it is computationally expensive. Good results were shown by CluStream and Clustree algorithms, which show higher processing speed than DenStream. DStream may show good results when processing some data. StreamKM has good efficiency for these tasks, but lower performance, and requires setting the number of clusters, as in the case with Clustream.

Acknowledgements

The work is performed according to the Russian Government Program of Competitive Growth of Kazan Federal University.

References

[1] Bifet, Albert, Geoff Holmes, Bernhard Pfahringer, Philipp Kranen, Hardy Kremer, Timm Jansen, and Thomas Seidl. 2010. "Moa: Massive online analysis, a framework for stream classification and clustering." In *Proceedings of the First Workshop on Applications of Pattern Analysis*, pp. 44-50. PMLR.

[2] Bifet, Albert, Ricard Gavaldà, Geoff Holmes, and Bernhard Pfahringer. 2018. "*Machine learning for data streams: with practical examples in MOA*." MIT Press,

[3] Carnein, Matthias, and Heike Trautmann. 2019. "Optimizing data stream representation: An extensive survey on stream clustering algorithms." *Business & Information Systems Engineering* 61(3): 277-297.

[4] Nasraoui, Olfa, and Chiheb-Eddine Ben N'Cir. 2019. "*Clustering Methods for Big Data Analytics*." Springer International Publishing: Berlin/Heidelberg, Germany.

[5] Vergara, Alexander, Shankar Vembu, Tuba Ayhan, Margaret A. Ryan, Margie L. Homer, and Ramón Huerta. 2012. "Chemical gas sensor drift compensation using classifier ensembles." *Sensors and Actuators B: Chemical* 166: 320-329.

[6] Witten, Ian H., and Eibe Frank. 2002. "Data mining: practical machine learning tools and techniques with Java implementations." *Acm Sigmod Record* 31(1): 76-77.

  Helix E-ISSN: 2319-5592; P-ISSN: 2277-3495