CMC, vol.60, no.1, pp.245-257, 2019

Stream-Based Data Sampling Mechanism for Process Object

Yongzheng Lin[1], Hong Liu[1,*], Zhenxiang Chen[2], Kun Zhang[2] and Kun Ma[2]

Abstract: Process object is the instance of process. Vertexes and edges are in the graph of process object. There are different types of the object itself and the associations between object. For the large-scale data, there are many changes reflected. Recently, how to find appropriate real-time data for process object becomes a hot research topic. Data sampling is a kind of finding changes of process objects. There is requirements for sampling to be adaptive to underlying distribution of data stream. In this paper, we have proposed a adaptive data sampling mechanism to find appropriate data to modeling. First of all, we use concept drift to make the partition of the life cycle of process object. Then, entity community detection is proposed to find changes. Finally, we propose stream-based real-time optimization of data sampling. Contributions of this paper are concept drift, community detection, and stream-based real-time computing. Experiments show the effectiveness and feasibility of our proposed adaptive data sampling mechanism for process object.

Keywords: Process object, data sampling, big data, data stream, clustering, stream processing.

1 Introduction

Process object is the instance of process [Du, Qu and Hua (2017)]. Vertexes and edges are in the graph of process object with the dependency and constraint on the entity and association [Srikant and Agrawal (1996)]. There are different of process objects, such as social networking, logistics exchange, industrial process in factory. In the area of learning process at school, data of process object are generated from the teaching and learning [Xia, Yang, Wang et al. (2012)]. Hidden and latent knowledge and rule could be analyzed and extracted. For the improvement, the mechanism should be researched. In this paper, we attempt to improve the performance of data sampling of process object and reducing the workload of the management.

In the process graph, there are implicit as well as explicit associations among entities [Stefan, Athitsos and Das (2013)]. Explicit association could be found using experiences

[1] School of Information Science and Engineering, Shandong Normal University, Jinan, China.

[2] School of Information Science and Engineering, University of Jinan, Jinan, China.

[*] Corresponding Author: Hong Liu. Email: lhsdcn@126.com.

CMC. doi:10.32604/cmc.2019.04322

and rules. However, the challenging is how to model implicit associations from real-time data. For big data, the sampling is more challenging to be adaptive to the current underlying distribution of data stream.

In this paper, an adaptive real-time data sampling mechanism to find appropriate data is proposed. First of all, we use concept drift to make the partition of the life cycle of process object. The state of process object is delivered to different evolution stages. The process object should be sampled for modeling the evolution. In the process object graph, association occurs between entities. Then, entity community detection is proposed to find changes. Relevant entities are gathered together to create a community with the properties of behaviour similarity and associations in a same evolution stage. Next, the knowledge could be discovered from the communities.

This paper is an extended version from the conference paper [Lin, Liu, Chen et al. (2018)]. We further develop the work on community detection and stream-based real-time computing. First, we have reorganized the structure of the paper, which would help the readers to gain insight into our motivations. Second, we propose stream-based data sampling mechanism using stream computing paradigm.

The organization of the paper is as follows. In Section 2, we present the related work of data sampling for process object. Section 3 introduces the architecture of adaptive sample mechanism for process object. First, we partition the life cycle of process object using concept drift. Second, we propose entity community detection in the same evolution stages. The third is our improved stream-based real-time data sampling using stream processing paradigm. Compared with traditional implementation, stream-based implementation is faster. Experimental results are shown to prove the effectiveness. Conclusions and future work is outlined in the last section.

2 Related works

Similar with workflow, process object is the instance of process. In the process object graph, there are entities and associations among them.

2.1 Association discovery of process object

Association discovery of process object is a hot research area [Song, Guo, Wang et al. (2014)] [Zhu, Du, Qu et al. (2016)]. Recently, there are many methods to find explicit and implicit knowledge among correlations and associations [Du, Qu and Hua (2017)] [Hua, Du, Qu et al. (2017)]. A scheme which can discover the state association rules of process object is proposed [Song, Guo, Wang et al. (2014)]. The hidden close relationships of different links in process object can be found using this method. The state association rules that can be obtained in accordance with rules. Another improvement of this work is using the local density of each sample point and the distance from the other sample points to determine the number of clustering centers in the grid [Hua, Du, Qu et al. (2017); Zhu, Du, Qu et al. (2016)]. However, the evolution stage of process object is not considered. With more data, evolution is common in reality. Therefore, we aim to make stage modeling of process object.

2.2 Concept drift of process object

Concept drift means that the statistical properties of the target variable [Gama, Žliobaitė, Bifet et al. (2014)]. Concept drift in data stream is used to find appropriate partition of life cycle of evolved process object. The generation of concept drift is caused by the not identical dynamic data streams. This causes problems because the predictions become less accurate as time passes. The detection of concept drift becomes a hot research topic [Ross, Tasoulis and Adams (2011)]. Some methods discussed blocking input data in big data streams. But the challenges are how to find changes in high-dimensional data space. In this situation, it will lead to more errors in data sampling. Another method is a sequential change detection model. It can offer statistically sound guarantees on false positive and false negative by using reservoir sampling [Pears, Sakthithasan and Koh (2014)]. Another method is Just-In-Time (JIT) classifiers [Wang, Park, Yeon et al. (2017)]. It is used to detect concept drifts contains two ways based on Intersection of confidence intervals (ICI). Therefore, it is necessary to research concept drift to address high speed issues. Recent work focused on combining method with a grouping attribute of constraint and penalty regression using concept drift [Wang, Park, Yeon et al. (2017)]. Following methods could be used in evolution stages partition [Tennant, Stahl, Rana et al. (2017); Sidhu and Bhatia (2017); Sethi and Kantardzic (2018); Duda, Jaworski and Rutkowski (2017); Sethi and Kantardzic (2018)].

2.3 Clustering optimization

Clustering optimization cab be used to find similar entities in the same evolution stages [Gong, Zhang and Yu (2017); Hyde, Angelov and MacKenzie (2017); Puschmann, Barnaghi and Tafazolli (2017); Bodyanskiy, Tyshchenko and Kopaliani (2017)]. There are some latest work on data stream clustering in entity community detection [de Andrade Silva, Hruschka and Gama (2017); Hyde, Angelov and MacKenzie (2017); Yarlagadda, Jonnalagedda and Munaga (2018)].

The size of the sliding windows are used according to the window itself [Bifet and Gavalda (2007)]. It recalculated based on the observed rate of the data. Other methods are algorithm for both infinite windows and sliding windows cases [Chen and Zhang (2018); Gemulla and Lehner (2008)]. They aim to address the issue of sampling without replacement for timestamp-based windows.

There are more work process clustering object modeling [Song, Guo, Wang et al. (2014); Hua, Du, Qu et al. (2017); Du, Qu and Hua (2017)]. However, they lack on the evolution stage of process object. The similar entities in the same evolution stages of process object should be considered. Another real-time clustering technique is stream computing. Several researchers have improved current method using stream computing framework [Ma and Yang (2017)]. But few research focus on data sampling with stream processing. In stream computing paradigm, it means accepting incoming data, and processing data in the business processing unit.

3 Adaptive data sampling

The architecture of our adaptive data sampling is proposed in this section. First, we present the basic definition of process object. Then, we present the adaptive data sampling process.

3.1 Basic definition

Process object is the instance of process. In the process graph, the process object consists of n links, and each link contains one or more sampling points. $X_i \, (i \in (1, n))$ is a link edge of the process object, and the generall system sampling period is T, and $t_i \, (i \in (1, m))$ is sampling time and $T = t_{i+1} - t_i$. The process object can be defined as

$$\chi = \left\{ \begin{matrix} \chi_1(x_1(t_1), x_1(t_2), ..., x_1(t_m)) \\ \chi_2(x_2(t_1), x_2(t_2), ..., x_2(t_m)) \\ ... \\ \chi_n(x_n(t_1), x_n(t_2), ..., x_n(t_m)) \end{matrix} \right\} \tag{1}$$

where $x_i(t_m)$ is sample value of i-th link edge with sampling time t_m. In reality, delay time exists between links in process objects during propagation. For any two links X_i and X_j, a change in link X_i will cause corresponding change in link X_j when X_i is in upstream and is an ancestor of X_j.

Links in the process object has some relations with other entities. It means that the link edge in process object is generalized to entity. That is to say that the entity will respond according to the changes of neighbors. This reveals some underlying evolution rule of process object.

Process object is also the instance of a group entities and its associations. The explicit or implicit association among entities could affect the evolution of the global process object. The relationship strength of association could change over time. Local changes might affect the global evolution of process object. That is to say that local changes could be detected by the global status change of process object. For a individual entity, its status could be defined by its observed value.

Definition 1 (Process Object)**.** Process object is the instance of a group entities and its associations, which could be modeled as a heterogeneous information network $G_{ProcessObject}(X_{entity}, E_{assocation})$. Given a fixed time interval, the status of links and entities are monitored and collected. The local status of entity x_i could be represented as a vector $\chi_i = \{x_i(t_1), x_i(t_2), ..., x_i(t_m), ...\}$. The global status of process object at time t_i could be represented as a vector $\omega_{t_i} = \{x_1(t_i), x_2(t_i), ..., x_n(t_i)\}$, where n is the number of entities in process object. If there is a relationship from entity χ_i to χ_j, there is a changeable function $\chi_j = f(\chi_i)$. This function is one attribute of the association.

The status of certain entity might be affected by a association chain of entities. Changes in one certain entity may affect other entities, and the change and evolution of local

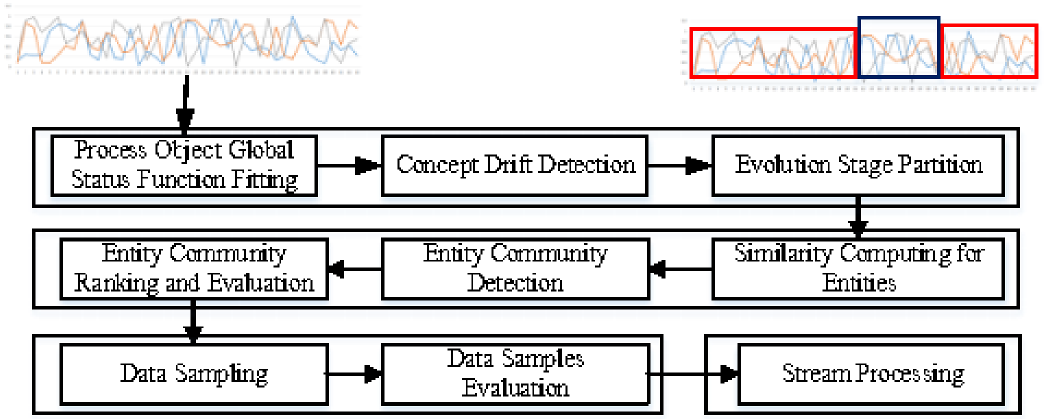
Process Object Global Status Function Fitting
Concept Drift Detection
Evolution Stage Partition
Similarity Computing for Entities
Entity Community Detection
Entity Community Ranking and Evaluation
Data Sampling
Data Samples Evaluation
Stream Processing

Figure 1: Process of adaptive data sampling

entities may cause the global process object evolve with time. Besides, it may evolve through different stages when process object is running. For example, the performance and association between entities may be different when process object lies in different stages. There might be an evolution of local association among entities. This instable state makes the complexity of data sampling.

Next, we propose the adaptive data sampling mechanism for modeling the evolution of process object dynamically and precisely. The motivation of our methods is to find the appropriate samples representing the evolution of process object according to the dynamic feature of big data stream.

3.2 Data sampling process

The adaptive data sampling process is shown in Fig. 1. It consists of the following steps.

3.2.1 Life cycle recognition and evolution stages partition

The life cycle of process object contains different continuous evolution stage. The performance and status of process object in different evoluton stage is different. For modeling the process object accurately, data should be sampled from different evolution stages. So the first step is to recognize the life cycle of process object and partition the life cycle into different evolution stages according to the status of process object.

3.2.2 Entity community detection and ranking

In different evolution stages, the association between entities is different. The impact and role of entity in different evolution stage is different. Then in a certain evolution stage, the relevant entities are partitioned into community based on their behaviour similarity and associations. These communities are the sources of data sampling.

3.2.3 Data sampling and evaluation

After evolution stages partition and entity community detection, data sampling is processed to sample appropriate data for modeling process object.

3.2.4 Stream-based sata sampling and evaluation

For big data stream environment of process object, a streaming processing architecture is proposed for sampling efficiently. More details are shown in Section 4.

3.3 Evolution stages partition

The process object evolves along with time. There are many stages where the performance of process object is different. The underlying data representing the evolution stage is also different. For modeling the evolution of process object, the stages of evolution should be partitioned. Then data could be sampled through different evolution stages accordingly.

For data stream, concept drift could be detected and the evolution of data stream could be partitioned into different stages. For process object, the evolution life cycle could be partitioned into different stages. In a certain evolution stage, the process object lies in a steady state. When concept drift occurs, it means that process object move from one steady state to another new steady state. The data belonging to the same steady state could be collected and sampled for modeling the steady state of process object. Based on this idea, a evolution stage partition mechanism is proposed, as shown in Algorithm 3.1.

Using the topology of process object, the exit entity could be found. Based on concept drift in data stream, a fitting function could be used to detect change of underlying data distribution. When the fitting function is no longer available for the newly data vector of process object, it means that the process object is moving from one old steady state to another new state. A new evolution stage is found. Along with the time, the evolution stages is recorded into the set $StagePartitions$.

3.4 Entity community detection

Data could be sampled from the different evolution stage. However, in the same evolution stage, the importance and influence of entities are different. So the entities in the same evolution stage could be grouped into different clusters. Data could be sampled from these clusters to model in a certain steady stage.

For finding the similar entities in the same evolution stage, a entity community detection algorithm is proposed, as shown in Algorithm 3.2.

Consider the χ_i is a time-series data about entity X_i. In process object, association between entities are not in real-time fashion. There is time delay when change propogate [Du, Qu and Hua (2017); Hua, Du, Qu et al. (2017); Song, Guo, Wang et al. (2014); Zhu, Du, Qu et al. (2016)]. So when compute the similarity between entities, it is important to take into account the time delay of associations. Based on previous work [Du, Qu and

Algorithm 3.1 Evolution Stage Partition Algorithm for Process Object.

```
Input:
    The continuous data vector from process object view, $\{\omega_{t_i}\}$;
    The continuous data vector from entity view, $\{\chi_i\}$;
    Stage Similarity Threshold, $T_{StageThreshold}$;
Output:
    The Stages Partition Set, $StagePartitions$;
$StagePartitions$ = Ø;
Get the exit entity $x_n$ according to the topology of process object;
Initialize $currentStage$;
Fitting a initial function $\chi_n = f(\chi_1, \chi_2, ..., \chi_{n-1})$;
while new data vector $\omega_{t_i}$ arrived do
    if SIMILARITY between $\chi_{n,t_i}$ and $\chi_n{}' = f(\chi_1, \chi_2, ..., \chi_{n-1})$ below $T_{threshold}$ then
        Process object is still in a steady state;
        Prepare for next data vector $\omega_{t_{i+1}}$;
    else
        Process object is moving to a new steady state;
        Prepare for fitting a new function for new steady state;
        Add the old steady state into set $StagePartitions$;
    end if
end while
return $StagePartitions$;
```

Hua (2017); Hua, Du, Qu et al. (2017); Song, Guo, Wang et al. (2014); Zhu, Du, Qu et al. (2016)], the time delay between entities could be found. Then the similarity matrix based on Dynamic Time Warping(DTW) could be adjusted into a more appropriate state. Then k-means clustering methods could be used to find the entity communities in a certain evolution stage.

3.5 Data sampling

After the evolution stages partition and entity community detection, the whole big data stream could be partitioned into different groups, which contain rich information about evolution of process object. Then the next step is to choose appropriate data for modeling.

For finding appropriate data for process object modeling, a adaptive sampling algorithm is proposed, as shown in Algorithm 3.3.

The proposed algorithm firstly use Algorithm 3.1 and Algorithm 3.2 to get the evolution stage partition and the community set in each evolution stage. Then based on the average of dispersion degree of community set in certain evolution stage, the sliding window mechanism is proposed to get the appropriate window that contain the data varies greatly. It means that it contain much more rich information, which could be use for process modeling.

```
Algorithm 3.2 Entity community detection for process object.
Input:
    Entity vector set in a certain evolution stage $S_j$, $\{\chi_i | \chi_i = (x_{i,t_m}, x_{i,t_{m+1}}, ..., x_{i,t_n}), [t_m, t_n] = TimeRange(S_j)\}$;
    The number of entity community, $m$;
Output:
    Entity community set, $EntityCommunity$;
1: Compute similarity matrix of $\{\chi_i\}$ using Dynamic Time Warping;
2: Correct the Similarity Matrix by time delay influence;
3: Choose m points in $\{\chi_i\}$;
4: Using classification such as k-means to finding cluster $EntityCommunity$
   considering time delay;
5: Return $EntityCommunity$;
```

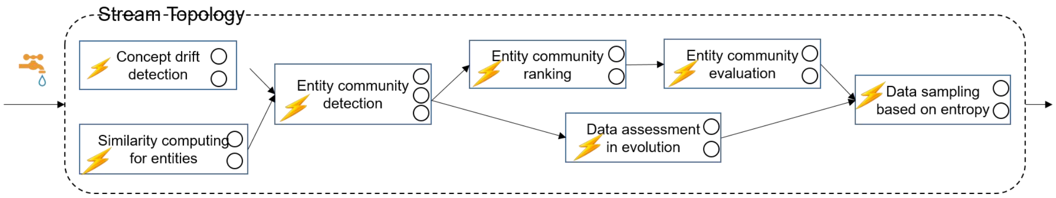
Stream Topology
Concept drift detection
Similarity computing for entities
Entity community detection
Entity community ranking
Entity community evaluation
Data assessment in evolution
Data sampling based on entropy

Figure 2: Stream topology of data sampling.

4 Stream-based real-time optimization of adaptive data sampling

4.1 Stream topology

We design a stream-based topology to implement real-time computing of data sampling. The stream topology is shown in Fig. 2. In streaming computing, a stream topology includes several processing element of data sampling. Each processing element of data sampling contains specific implementation of processing logic, and links between nodes contains how the relation how in-stream data is passed. The input is time series stream data some point, including the collecting digital data from sensors and a timestamp. Stream data in this topology is an unbounded sequence of tuples. The output of stream of stream computing in the topology is the sampling series stream data. Business of processing is capsulated in top-level abstraction of processing element. The processing stream is passed by the emition of the input, the it is processed by processing element. For stream-based real-time optimization of adaptive data sampling. For data migration strategy, processing elements include *concept drift detection*, *similarity computing for entities*, *entity community detection*, *entity community ranking*, *entity community evaluation*, *data assessment in evolution*, and *data sampling based on entropy*.

Algorithm 3.3 Adaptive data sampling for process object.

```
Input:
    The continuous data vector from process object view, $\{\omega_{t_i}\}$;
    The continuous data vector from entity view, $\{\chi_i\}$;
    Stage Similarity Threshold, $T_{StageThreshold}$;
    The number of entity community in each evolution stage, $m$;
Output:
    Sample Data Set, $SampleData$;
    Call Algorithm 3.1 to get the stages partition set $StagePartition = \{Stage_i\}$;
    Call Algorithm 3.2 to get the entity community set $EntityCommunity_i$ in evolution
    stage $Stage_i$;
    for each community $c_{m,i}$in $EntityCommunity_i$ in every stage $Stage_i$ do
        Initialize a sliding window;
        Get the average dispersion degree of communities in $EntityCommunity_i$;
        Slide the window to make the dispersion degree maxmum;
        Get data from the max dispersion degree window as $sample_i$ for $Stage_i$;
        Add $sample_i$ to $SampleData$
    end for
    Return $SampleData$;
```

4.2 Processing element

These processing elements have abstract interfaces to help developers add the implementation of business logic to complete the sampling processing of process object. A source of data streams from sensor process object with timestamps is emitted to the stream processing framework, then grouped to make adaptive data sampling. An input element reads tuples of sensor process objects, and emits them as a stream. The subsequent processing element of the stream processing framework consumes any number of input streams to implement adaptive data sampling discusses in Section 3, and possibly emits new streams. Complex adaptive data sampling requires multiple steps and processing elements to generate the sampling result.

In the stream topology, there are several processing elements that are implementing the abstract interface of stream computing. In order to accelerate the parallel sampling, we group process object before emitting to the stream topology. Concept drift detection is to discover the change of stable state of process object, and similarity computing for entities is to discover the variation tendency of process object. Then, entity community is detected. Afterwards, entity community is ranked and evaluated. Finally, the sampling result is got based on the entropy.

5 Experiments and analysis

In this section, experiments and analysis are made to demonstrate the effectiveness of the proposed adaptive data sampling mechanism.

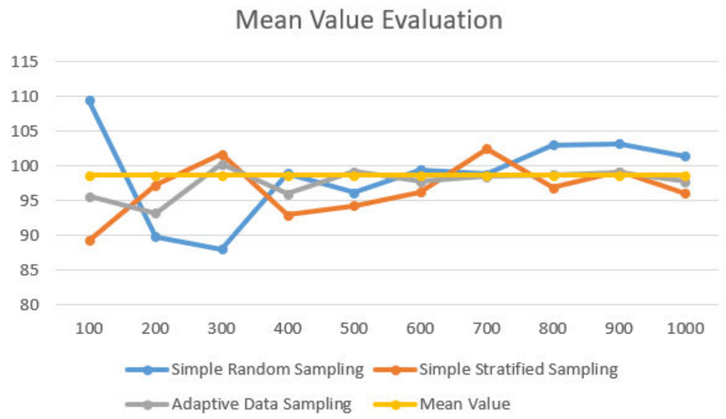

Figure 3: Mean Value Evaluation.

There has been a lot of public data set on the web, such as UCI machine learning repository [Dheeru and Karra Taniskidou (2017). The PM2.5 Data Set [Liang, Zou, Guo et al. (2015) in UCI machine learning repository is chose for experiments. The data time period is between Jan 1st, 2010 to Dec 31st, 2014. The attributes include row number, year, month, day, hour, PM2.5 concentration, dew point, temperature, pressure, combined wind direction, cumulated hours of snow and cumulated hours of rain.

Considering the PM2.5 data set, the PM2.5 concentration, dew point and temperature interact. In the proposed approach, these could be modeled as entities in process object. The association between entities could be modeled through process object. In our setting, the PM2.5 concentration is the focus, which could be the exit entity or dependent variable in process object. Other entities could be independent variables.

Given the simple random sampling, simple stratified sampling, and adaptive data sampling, the accuracy of sampling is compared. Data samples should preserve the synopsis of all data. Consider the mean value and median value, experiments are made to make a comparison between these three data samplings, as shown in Fig. 3 and Fig. 4.

The PM2.5 data set contains 41,757 data records. The three sampling mechanism is used to sample data at different scales, from 100 samples to 1000 samples. As we can see from the experiment result, the adaptive data sampling could get a more accurate synopsis of the PM2.5 data set.

6 Conclusion

In the evolution of different stages, the status of process object is not stable. Data should be sampled from incoming streams for modeling. In this paper, we attempt to propose a adaptive data sampling mechanism. Although experimental results are effective, some future work, such as the role of entity and association among entities, is also needed.

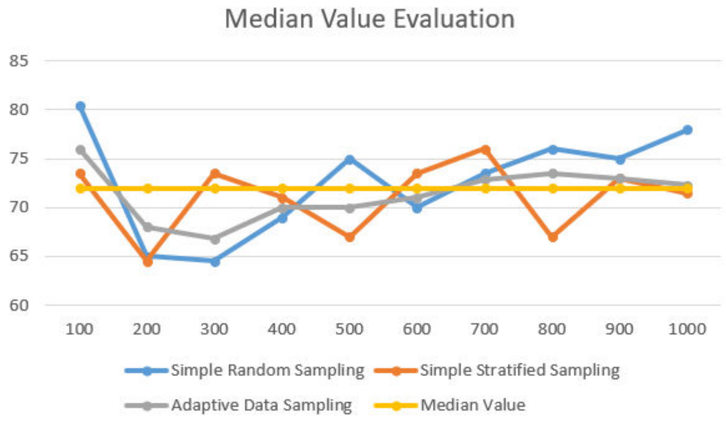

Figure 4: Median Value Evaluation.

Acknowledgement: This work was supported by the National Natural Science Foundation of China (No. 61472232), Natural Science Foundation of Shandong Province of China (No. ZR2017BF016), and the Science and Technology Program of University of Jinan (No. XKY1623).

References

Bifet, A.; Gavalda, R. (2007): Learning from time-changing data with adaptive windowing. *Proceedings of the 2007 SIAM International Conference on Data Mining*, pp. 443-448.

Bodyanskiy, Y. V.; Tyshchenko, O. K.; Kopaliani, D. S. (2017): An evolving connectionist system for data stream fuzzy clustering and its online learning. *Neurocomputing*, vol. 262, pp. 41-56.

Chen, J.; Zhang, Q. (2018): Distinct sampling on streaming data with near-duplicates. *Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, pp. 369-382.

de Andrade Silva, J.; Hruschka, E. R.; Gama, J. (2017): An evolutionary algorithm for clustering data streams with a variable number of clusters. *Expert Systems with Applications*, vol. 67, pp. 228-238.

Dheeru, D.; Karra Taniskidou, E. (2017): UCI machine learning repository. http://archive.ics.uci.edu/ml.

Du, T.; Qu, S.; Hua, Z. (2017): A novel timing series calculation algorithm based on statistical extremum for process object. *9th International Conference on Computer and Automation Engineering*, pp. 94-98.

Duda, P.; Jaworski, M.; Rutkowski, L. (2017): On ensemble components selection in data streams scenario with reoccurring concept-drift. *IEEE Symposium Series on Computational Intelligence*, pp. 1-7.

Gama, J.; Žliobaitė, I.; Bifet, A.; Pechenizkiy, M.; Bouchachia, A. (2014): A survey on concept drift adaptation. *ACM computing surveys*, vol. 46, no. 4, pp. 44.

Gemulla, R.; Lehner, W. (2008): Sampling time-based sliding windows in bounded space. *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, pp. 379-392.

Gong, S.; Zhang, Y.; Yu, G. (2017): Clustering stream data by exploring the evolution of density mountain. arxiv:1710.00867.

Hua, Z.; Du, T.; Qu, S.; Mou, G. (2017): A data stream clustering algorithm based on density and extended grid. *13th International Conference on Intelligent Computing Theories and Application*, pp. 689-699.

Hyde, R.; Angelov, P.; MacKenzie, A. (2017): Fully online clustering of evolving data streams into arbitrarily shaped clusters. *Information Sciences*, vol. 382, pp. 96-114.

Liang, X.; Zou, T.; Guo, B.; Li, S.; Zhang, H. et al. (2015): Assessing beijing's pm2.5 pollution: severity, weather impact, apec and winter heating. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 471, no. 2182.

Lin, Y.; Liu, H.; Chen, Z.; Zhang, K.; Ma, K. (2018): Adaptive data sampling mechanism for process object. *International Conference on Algorithms and Architectures for Parallel Processing*, pp. 257-266.

Ma, K.; Yang, B. (2017): Stream-based live data replication approach of in-memory cache. *Concurrency and Computation: Practice and Experience*, vol. 29, no. 11.

Pears, R.; Sakthithasan, S.; Koh, Y. S. (2014): Detecting concept change in dynamic data streams. *Machine Learning*, vol. 97, no. 3, pp. 259-293.

Puschmann, D.; Barnaghi, P.; Tafazolli, R. (2017): Adaptive clustering for dynamic IOT data streams. *IEEE Internet of Things Journal*, vol. 4, no. 1, pp. 64-74.

Ross, G. J.; Tasoulis, D. K.; Adams, N. M. (2011): Nonparametric monitoring of data streams for changes in location and scale. *Technometrics*, vol. 53, no. 4, pp. 379-389.

Sethi, T. S.; Kantardzic, M. (2018): Handling adversarial concept drift in streaming data. *Expert Systems with Applications*, vol. 97, pp. 18-40.

Sidhu, P.; Bhatia, M. (2017): Online approach to handle concept drifting data streams using diversity. *International Arab Journal of Information Technology*, vol. 14, no. 3.

Song, Q.; Guo, Q.; Wang, K.; Du, T.; Qu, S.; orhers (2014): A scheme for mining state association rules of process object based on big data. *Journal of Computer and Communications*, vol. 2, no. 14, pp. 17-24.

Srikant, R.; Agrawal, R. (1996): Mining quantitative association rules in large relational tables. *ACM Sigmod Record*, vol. 25, no. 2, pp. 1-12.

Stefan, A.; Athitsos, V.; Das, G. (2013): The move-split-merge metric for time series. *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 6, pp. 1425-1438.

Tennant, M.; Stahl, F.; Rana, O.; Gomes, J. B. (2017): Scalable real-time classification of data streams with concept drift. *Future Generation Computer Systems*, vol. 75, pp. 187-199.

Wang, L.-Y.; Park, C.; Yeon, K.; Choi, H. (2017): Tracking concept drift using a constrained penalized regression combiner. *Computational Statistics & Data Analysis*, vol. 108, pp. 52-69.

Xia, F.; Yang, L. T.; Wang, L.; Vinel, A. (2012): Internet of things. *International Journal of Communication Systems*, vol. 25, no. 9, pp. 1101-1102.

Yarlagadda, A.; Jonnalagedda, M.; Munaga, K. (2018): Clustering based on correlation fractal dimension over an evolving data stream. *International Arab Journal of Information Technology*, vol. 15, no. 1, pp. 1-9.

Zhu, T.; Du, T.; Qu, S.; Zhu, L. (2016): A novel timing calculation algorithm based on statistical extremum for the time series of process object. *Hans Journal of Data Mining*, vol. 6, no. 4, pp. 179-191.