A Dynamic Clustering Algorithm for Context Change Detection in Sensor-Based Data Stream System

Nitesh Funde, Meera Dhabu and Umesh Balande

Abstract Sensor-based monitoring systems are growing enormously which lead to generation of real-time sensor data to a great extent. The classification and clustering of this data are a challenging task within the limited memory and time constraints. The overall distribution of data is changing over the time, which makes the task even more difficult. This paper proposes a dynamic clustering algorithm to find and detect the different contexts in a sensor-based system. It mines dynamically changing sensor streams for different contexts of the system. It can be used for detecting the current context as well as in predicting the coming context of a sensor-based system. The algorithm is able to find context states of different length in an online and unsupervised manner which plays a vital role in identifying the behavior of sensor-based system. The experiments results on real-world high-dimensional datasets justify the effectiveness of the proposed clustering algorithm. Further, discussion on how the proposed clustering algorithm works in sensor-based system is provided which will be helpful for domain experts.

Keywords Clustering · Context state · Data streams · Principal component analysis (PCA) · Sensor-based system

1 Introduction

Nowadays, infinite volume of data stream is generated by the industry processes equipped with sensors, real-time monitoring systems, online transactions, social networks, and Internet of Things (IoT). In contrast to the traditional data, sensor

N. Funde (✉) · M. Dhabu · U. Balande
Department of Computer Science and Engineering,
Visvesvaraya National Institute of Technology, Nagpur, Maharashtra, India
e-mail: nitesh.funde@gmail.com

D. Reddy Edla et al. (eds.), *Advances in Machine Learning and Data Science*,
Advances in Intelligent Systems and Computing 705,
https://doi.org/10.1007/978-981-10-8569-7_5

data streams are continuous, ordered, potentially infinite, and require fast real-time response. Almost all the real-time sensor-based monitoring systems produce a fast, huge amount of data which need to be analyzed for real-time decision making. Data stream mining is the process of analyzing continuous, fast records within limited storage and time constraints. For more efficient data stream mining technique, sensor data streams must be processed online in single-scan, and it should incorporate the ability to handle multidimensional data [1]. Consider an example of a satellite remote sensing system, in which a sensor constantly produces a data over a period of time. The statistical properties of data distribution are often changes in an unexpected ways in sensor data streams. This property is called as a concept drift [2].

The data stream clustering is a challenging problem in sensor-based system applications because of its operational constraints. In this paper, the data stream clustering for context change detection in sensor-based system is addressed. Context is an information which can be used to discover the behavior of a sensor-based system [3]. Although many data stream clustering algorithms are available, they are not suitable for detecting the current context state of sensor-based system. The context state of a sensor-based system is constantly changing which represents the different modes or states. This behavior of system can further modeled in order to develop some useful applications. For example, smartphone nowadays consists of in-built sensors such as accelerometer and gyrometer. In the near future, it is easily possible for the sensor-based system to provide more personalized services. The behavior analysis of smartphone user will be useful for developing personalized recommender system.

This paper is organized as follows. Section 2 discusses the related work of data stream clustering algorithm. Section 3 describes the proposed dynamic clustering algorithm. Section 4 provides experimental details, results, and discussion, and at last, Sect. 5 gives conclusion.

2 Related Work

Silva et al. presented a detailed survey of different existing data stream clustering algorithms such as Clustream, D-stream, Denstream [4–7]. The experimental methodology and temporal aspects of data stream clustering are discussed in this survey. There are large number of references provided regarding applications of data stream clustering in the different fields such as network intrusion detection, stock market analysis, and sensor networks. Authors also presented the information regarding various datasets and software packages. The most important part of the survey is the discussion of open challenges and issues regarding the design of algorithms with or without using ad hoc user-defined parameters such as number of clusters, window length. Zhang et al. proposed Strap clustering algorithm by combining affinity propagation (AP) algorithm with Page-Hinkley test [8]. The Strap is validated on a real-world application and performed well in data stream environments.

Qahtan et al. discussed the framework for change detection in high-dimensional data streams using PCA [9]. Here, change detection is performed by comparing data distribution in a test window with reference window which uses divergence metrics and density estimators. This framework has many benefits over the existing approaches of reduction in computational costs of density estimator. The threshold value is dynamically calculated by using Page-Hinkley test. However, the clustering procedure is yet to be experimented in sensor-based system. Mirsky et al. proposed the pcStream which is dynamically detecting the sequential temporal context of an entity from sensor-fused datasets [10].

In contrast to the above-mentioned algorithms, the proposed algorithm transforms the data into the suitable representation for handling high-dimensional data. It is capable of identifying different changing context states of various length in sensor-based system in an online and unsupervised manner within limited memory and time constraints. The context can be of any repeating nature or representing the current state of an operation of a system. The experiments are performed on two real-world dataset such as KDD’99 and home activity monitoring.

3 Proposed Dynamic Clustering Algorithm (Dclust)

The proposed dynamic clustering algorithm uses the principal component analysis (PCA) for modeling the data distribution in high-dimensional data streams. PCA is a popular technique for dimensionality reduction [11]. The proposed clustering algorithm exploits PCA to capture the set of uncorrelated information, i.e., principal components (eigenvectors) from the original data such that it better represents the transformed data. PCA on original data S gives the eigenvectors and eigenvalues. The advantage of using PCA is as follows: It allows the algorithm to detect changing context state in the form of mean, variance, and correlation. The original data distribution changes are reflected in the PC projections over a period of time.

Table 1 Notations

Notations	Descriptions
S	An unbounded data stream, $S = \{x_1, x_2, \ldots, x_n\}$ and $x_i \in R^d$
$\|Clust\|$	Set of clusters representing the different context states of system
δ	Threshold value
p	Number of stream instances
β	Minimum context state drift size (# drifting instances that represent new context state)
P_v	Percentage of variance required to select principal components
B_d	A buffer with at most β consecutive drifting instances
m	The maximum # records a system can process (window size)

It also reduces the computational cost by removing the principal components which is having smaller variance.

The notations used in the proposed algorithm are as shown in Table 1. The basic idea of this algorithm is to follow data stream distribution for identifying the different context states of the sensor-based system. The statistical similarity between a stream instance and known context is calculated using Mahalanobis distance method by taking only selected PCs of that data distribution which satisfies the criteria of variance P_v. As long as the data streams fit within a known context‘s data distribution, we assign it to that existing context state. Otherwise, new context state is defined for those data points. Each context state has window memory size m for concept drift. Finally, algorithm’s different parameters such as β, δ, and P_v are adjusted accordingly to identify different context state in the system. The code of the algorithm is as shown below.

At first, all parameters are initialized: p is initialized with the number of rows which is equal to min context state drift size β, $|Clust| = \phi$, $m = 100$, and $P_v = 0.98$. Then, by applying function ClusterModel to first p instances of S for initial model building, the first context state is obtained. The ClusterModel function calculates the eigenvectors, eigenvalues and mean using P_v. The memory window of the context model is a maximum length m which forgets older observations. As the stream instance arrives, the statistical similarity is calculated by using selected eigenvectors to existing **clusters (contexts)**. If it fits within that known context distribution by checking with δ, then that instance is assigned to the best model in $|Clust|$ and B_d is emptied. If it does not fit, then we add stream instance to B_d and check if it is equal to β (full). If B_d is full, then new context state has been discovered in sensor-based system. In this case, new context model C_j is added to $|Clust|$. In this way, different contexts of the sensor-based system are discovered.

```
Algorithm: A Dynamic Clustering Algorithm
Input: An unbounded data streams $S$
Parameters: Threshold value δ, min context state drift size β, variance $P_v$ .
Output: Clusters in $|Clust|$ representing different context state of a system
Initialization: $p = \beta$ ; $|Clust| = \phi$ ; $m = 100$; countDrift=0
ClusterModel($S\ (1{:}p), P_v$) ;
 numClust = numClust+1;
AddModel($C_1, |Clust|$);
for $i = p+1$ to $size(S)$ do
   for $j = 1$ to $|Clust|$ do
     Calculate dist =$[(S_i)$ -Centroid($j$) × eigenvector($j$)];
     scores($S_i$) = dist × dist′;
   end
   score =sqrt(dist);
   bestModel= min(scores);
   if min(scores) >  δ
      countDrifters = countDrifters+1;
      $B_d$(countDrifters,:)=$S\ (i,:)$;
      if countDrifters == β
         numClust=numClust+1;
         AddModel($C_j$,|Clust|);
         ClusterModel($B_d$ , $P_v$ );
      end
   else
      if countDrifters >0
         Add new instance to the S;
         Check the window size m of instances;
         ClusterModel($B_d$ , $P_v$ );
         $S_i$ =bestModel; // $S_i$ belongs to bestModel
      end
   end
end

ClusterModel (S, $P_v$){
   [EigenVectors, EigenValues] =PCA(S);
   Centroid=Mean(S);
}
```

The contextual sensor stream can be illustrated as a sequence of attributes in geometric space. The clusters C_1, C_2, and C_3 with their principal components and mean values in geometric space are as shown on Fig. 1. The different clusters are considered to be as different context state in sensor-based system.

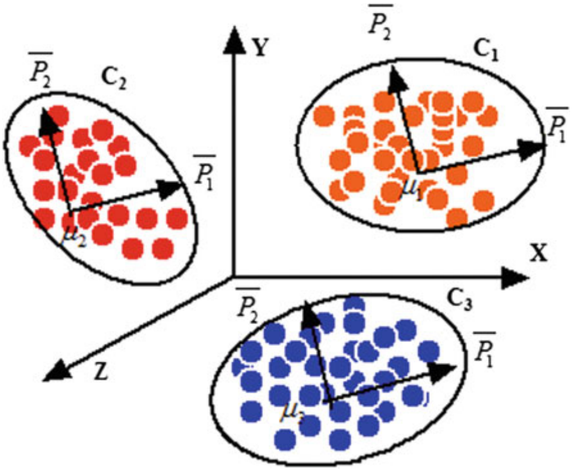


Fig. 1 Clusters with its principal components

4 Experiments

4.1 Datasets

We used two real-world datasets to demonstrate the clustering in sensor-based system. The first is the network intrusion dataset (KDD) and second is home activity monitoring dataset (HAM) [12, 13]. The aim of KDD'99 is to differentiate various attacks from normal connection. The sequence of TCP packets starting from source to target is considered as a connection. There are approximately 4,900,000 records and 41 numeric attributes. We normalized dataset and used 38 attributes for the proposed clustering algorithm. The objective of using this dataset is to evaluate the proposed algorithm on high-dimensional data. There are 23 classes for each connection for the normal and the particular type of attack such as ftp_write, neptune, and buffer_overflow. Overall, the dataset includes normal connection and specific attacks which are categorized in major type such as user to root attack (U2R), denial of service attack (DoS), probing attack, and remote to local attack (R2L). These major types can be considered as different contexts.

The HAM dataset has recordings of 10 sensors, i.e., 8 MOX gas sensors, 1 temperature, and 1 humidity sensor over a period of time. These sensors are exposed to different conditions in home such as normal background and two stimuli: wine and banana. The aim of HAM dataset is to build an application like electronic nose which differentiates among background, banana and wine.

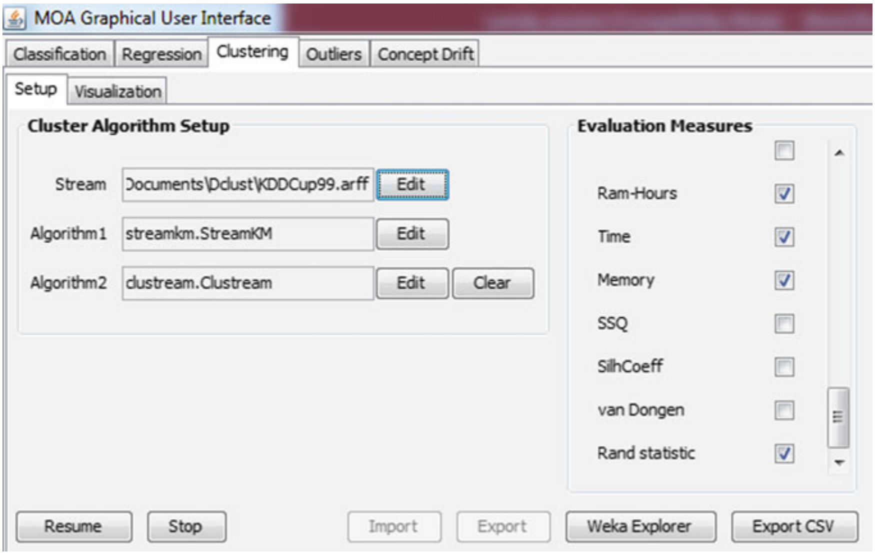
4.2 Experimental Setting

The dynamic clustering algorithm is implemented in R which is a language and environment for statistical computing and graphics. We have used Massive Online Analysis (MOA) tool for comparison of our proposed clustering algorithm (Dclust) with other algorithms such as Clustream, Denstream, StreamKm, and Strap. MOA is an open-source framework for data stream mining which includes various machine learning algorithms and tool for evaluation. The setup for evaluation of clustering algorithms is shown in Fig. 2.

The MOA tool also provides visualization of clustering over a period of time and user can control the different parameters such as visualization speed and evaluation measures. The evolving clusters at T_1 and T_2 are as shown in Fig. 3a and Fig. 3b, respectively. The different colors represent different clusters.

4.3 Results and Discussion

The clustering performance of proposed algorithm is evaluated in terms of adjusted Rand index (ARI) because of overlapping criteria of clusters (context states) in the system. The performance of clustering algorithms is measured by calculating ARI between dataset’s labels and algorithm’s clustering assignment. First, the data



Stream: Documents\Dclust\KDDCup99.arff

Algorithm1: streamkm.StreamKM

Algorithm2: clustream.Clustream

☐

Ram-Hours ☑

Time ☑

Memory ☑

SSQ ☐

SilhCoeff ☐

van Dongen ☐

Rand statistic ☑

Fig. 2 MOA setup for clustering

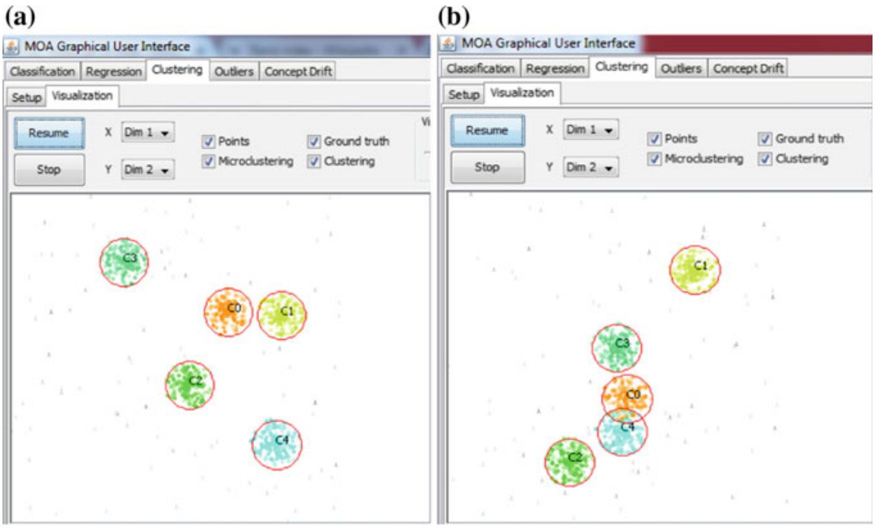


Fig. 3 **a** Evolving clusters at T_1, **b** evolving clusters at T_2

stream clustering is performed on a dataset, and then ARI is calculated using dataset's labels and respective clustering assignments.

Figure 4 shows the comparison of dynamic clustering (Dclust) algorithm with the other algorithms such as Clustream, Denstream, StreamKM, and Strap on two dataset. It shows that the dynamic clustering algorithm performs better than other clustering algorithms using all feasible input parameters of algorithms.

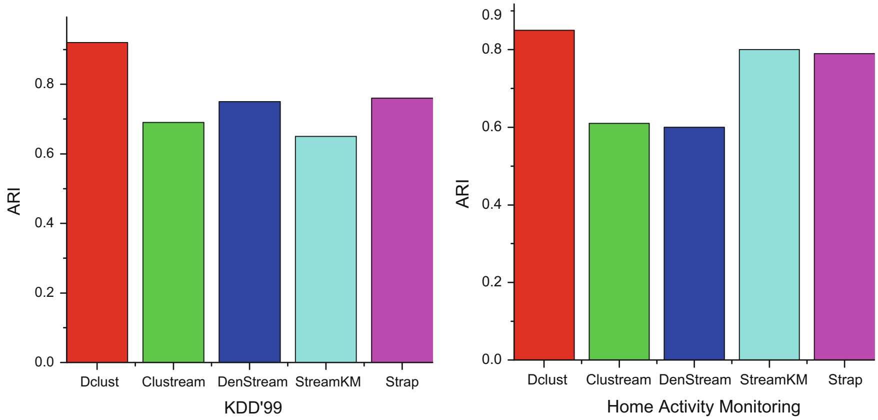


Fig. 4 Comparison of dynamic clustering (Dclust) algorithm with other clustering algorithms on two dataset, i.e., KDD'99 and home activity monitoring

5 Conclusion

This paper presented the dynamic clustering algorithm for detecting the different changing context states in sensor-based system. The algorithm is based on the concept of PCA. The different context states of the sensor-based system are on or off mode, specific working mode and not working. The context states can be of any length which represents the current property of the system. This dynamic clustering algorithm can be used for recognizing the current context and predicting the coming contexts of the system. The performance results on high-dimensional real-world dataset show that the algorithm outperforms the existing stream clustering algorithms. There are different user-defined parameters which control the proposed algorithm such as threshold value and window size. As the data distribution changes, threshold value should also be dynamic. We will consider the combining of dynamic threshold settings approach with this algorithm in real-time application as our future work.

References

1. Han, J., Kamber, M., Pei, J.: Data Mining: Concepts and Techniques. Elsevier (2011)
2. Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., Bouchachia, A.: A survey on concept drift adaptation. ACM Comput. Surv. (CSUR) **46**(4), 44 (2014)
3. Dey, A.K.: Providing architectural support for building context-aware applications. Doctoral Dissertation, Georgia Institute of Technology (2000)
4. Aggarwal, C.C., Han, J., Wang, J., Yu, P.S.: A framework for clustering evolving data streams. In: Proceedings of the 29th International Conference on Very Large Data Bases, vol. 29, pp. 81–92. VLDB Endowment (2003)
5. Cao, F., Estert, M., Qian, W., Zhou, A.: Density-based clustering over an evolving data stream with noise. In: Proceedings of the 2006 SIAM International Conference on Data Mining, pp. 328–339. Society for Industrial and Applied Mathematics (2006)
6. Chen, Y., Tu, L.: Density-based clustering for real-time stream data. In: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 133–142. ACM (2007)
7. Silva, J.A., Faria, E.R., Barros, R.C., Hruschka, E.R., de Carvalho, A.C., Gama, J.: Data stream clustering: a survey. ACM Comput. Surv. (CSUR) 46–53 (2013)
8. Zhang, X., Furtlehner, C., Germain-Renaud, C., Sebag, M.: Data stream clustering with affinity propagation. IEEE Trans. Knowl. Data Eng. **26**(7), 1644–1656 (2014)
9. Qahtan, A.A., Alharbi, B., Wang, S., Zhang, X.: A pca-based change detection framework for multidimensional data streams: change detection in multidimensional data streams. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM (2015)
10. Mirsky, Y., Shapira, B., Rokach, L., Elovici, Y.: pcstream: a stream clustering algorithm for dynamically detecting and managing temporal contexts. In: Advances in Knowledge Discovery and Data Mining, vol. 2015, pp. 119–133. Springer (2015)
11. Jolliffe, I.T., Cadima, J.: Principal component analysis: a review and recent developments. Philos. Trans. R. Soc. A **374**(2065), 20150202 (2016)
12. Bache, K., Lichman, M.: UCI Machine Learning Repository. http://archive.ics.uci.edu/ml (2013)
13. Huerta, R., Mosqueiro, T., Fonollosa, J., Rulkov, N., Rodriguez-Lujan, I.: Online decorrelation of humidity and temperature in chemical sensors for continuous monitoring. Chemom. Intell. Lab. Syst. **157**, 169–176 (2016)