Dynamic Classifier and Sensor Using Small Memory Buffers

R. Gelbard and A. Khalemsky

Information Systems, Bar-Ilan University, Ramat-Gan 5290002, Israel
gelbardr@mail.biu.ac.il, anna.khalemsky@gmail.com

Abstract. A model presented in current paper designed for dynamic classifying of real time cases received in a stream of big sensing data. The model comprises multiple remote autonomous sensing systems; each generates a classification scheme comprising a plurality of parameters. The classification engine of each sensing system is based on small data buffers, which include a limited set of "representative" cases for each class (case-buffers). Upon receiving a new case, the sensing system determines whether it may be classified into an existing class or it should evoke a change in the classification scheme. Based on a threshold of segmentation error parameter, one or more case-buffers are dynamically regrouped into a new composition of buffers, according to a criterion of segmentation quality.

Keywords: Dynamic classifier · Dynamic rules · Big data · Sensing data
Memory buffers · Clustering · Classification

1 Introduction

Sensors are located in environments that change dynamically and are required not only to detect the values of all parameters measured, but also to assess the situation and alert accordingly, based on predefined rules managed by a "Classifier-engine". Since the environments are dynamically changed and there may be new situations that were not known in advance, which are reflected in new combinations of parameter values, there is a need for a dynamic updating of the sensor's classifier.

One exemplary application for such dynamic classification unit (DCU) is a screening gate including biometric sensors that screen travelers entering a high security area such as an airport. The sensors may be configured to test multiple parameters of a traveler, such as heart rate, heart pressure, perspiration, etc. The classification system may be set to measure two classes of travelers, the bulk of travelers who have "normal" parameters and should pass the biometric screening without interference, and those who should be checked by security personnel. Upon receiving a new case, the sensing system determines whether it may be classified into one of the existing classes, or it should evoke a change in the classification scheme. Thus, over the course of a day, environmental conditions may change; ranges of values that haven't been observed before may appear causing dynamic changes-updates in sensor's classifier.

Changes in the sensor's classifier (i.e. classification scheme) are triggered based on a threshold of segmentation error parameter. The sensor's classifier is based on small

P. Perner (Ed.): ICDM 2018, LNAI 10933, pp. 173–182, 2018.
https://doi.org/10.1007/978-3-319-95786-9_13

data buffers and collects remembers a limited set of "representative" cases for each class (case-buffers). As a result of the trigger's appeal, one or more case-buffers are dynamically regrouped into a new composition of buffers, according to segmentation quality criteria.

The novelty of this real-time mechanism lies in the fact that the entire process is based on the use of limited memory buffers. In addition, each DCU, which is a remote autonomous sensing system, can communicate with multiple additional remote autonomous sensing systems. In such situations, the case buffers, as well as the case history, can be synchronized and managed via a central controller. Furthermore, in a distributed environment, regardless the existence of a central controller, the contents of the case buffers and the classifier scheme of each DCU can be synchronized between the multiple remote autonomous agents (sensing-systems). Synchronization may be performed after each regrouping process. That is to say that each incremental updating at any local DCU may initiate synchronization among all connected autonomous agents.

2 Related Work

In the reality of the dynamic data environment, when a huge amount of raw data and information flows ceaselessly, the main purpose of individuals and organizations is discovering the optimal way to find a hidden potential in it, through the constant cooperation of human intelligence and machine capabilities. The techniques and models that successfully functioned in stable data environment are outdated and need to be corrected to deal with dynamic data environment. "Databases are growing in size to a stage where traditional techniques for analysis and visualization of the data are breaking down" [1, 2]. Because of the constant increase in data volume, interpreting of similarities of different sub-populations becomes the new dimension of data mining goal. The data usually flows from different sources and has to be handled and processed simultaneously [3].The development of new and advanced techniques in data mining in dynamic data environment covers more and more fields, for instance, computer sciences, medicine [4], security systems [5] and social networks [6, 7]. And it is not just an application of existing algorithmic tools in these fields, but the inclusion of elements and logic and even tools, that were created purposefully for them.

As a result of the constant need to get real-time solutions, the research is naturally directed into a new field – incremental data processing. The motivation is to maximize the quality of solutions through minimizing the process cost [8–10]. The algorithmic tools have to be adjusted to dynamic data environment and be capable to absorb significant amounts of data, possibly to handle with the Big Data environment. The main idea of incremental techniques is to use small segments of data and not the whole historical data [11–13].

One of the commonly used directions in data mining is classification process, in which the objects are classified into homogeneous groups, with a maximal diversity between groups and minimal within groups. The proximity of an object to group centroid is usually measured by similarity measures, such as RMSE (root mean square error), used in the current paper [14]. The classification tasks are usually divided into two main types: if the target attribute is previously known, the process is called "classification", and if the target attribute is not known, it is called "clustering" [2, 10, 15, 16]. In the case of clustering problems, the interpretation of achieved clusters is one of the main challenges. For example, if a higher education consultant has to recommend the future student what is the best faculty for him, he will probably pick the faculty name from the existing list in the university. On the other hand, if the security system bank controller needs to identify the type of a new financial fraud trend, he needs to be very open-minded and be able to classify the action undertaken by a fraudster to an absolutely different type and give it an appropriate description. In some cases, there is a need to create a set of groups/classes based on items/customers/actions that are needed to be classified without any information about the target attribute. Different kinds of classification/clustering tasks in dynamic data environment in combination of existing and new techniques became the basis for extensive research [17–19].

The current research presents a dynamic classifier based on incremental dynamic clustering process. It permits the use of small data buffers that represent existing groups. This approach is significantly different from other approaches and methods, considered in the literature.

The dynamic classifier, proposed in the current paper, functions as a sensor. It works as a screening gate that distinguishes between "regular" items that are close enough to at least on of existing groups and alerts when the relatively "different" item occurs. The model permits not just an alert in such situation, but the action required to classify the item. Lots of studies combine different sensors in decision making processes [20–23].

3 Model Architecture

Figure 1 presents a schematic architecture of the proposed system, showing one DCU connected to a central controller as well as to other remote autonomous sensors-agents.

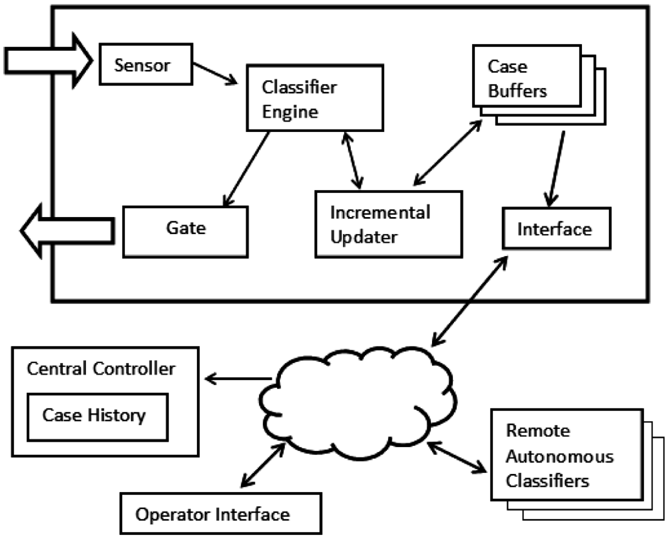

Fig. 1. Schematic architecture of the system.

The real-time data flows through the “Sensor” (Fig. 1, first component) to the “Classifier-engine”. The “Classier-engine” performs a decision process based on a classification scheme, as described in the flow diagram in Fig. 2. The “Gate” component represents the output, or, in other words, the decision regarding the classification of object. Based on a threshold of segmentation error parameter and segmentation quality criteria, the DCU incrementally updates the population of the relevant “Case-buffers”. The mechanism that manages the population (i.e. cases) stored in each buffer can use diverse policies, such as FIFO policy (First-In First-Out), or a selection policy that may store extreme-farthest cases of each group (“outliers” that are still classified to that group).

The term “Sensor” represents the “funnel” through which the data stream flows. Thus, a sensor can be a physical object, as well as a logical handshake through which the data flows into the system. The flow chart, illustrated in Fig. 2, presents the real-time decision-making process for each new sensing data element (i.e. each new case).

The version control is managed by sequential numbering approach.

Table 1 presents the notations used in the flowchart:

Table 1. Notations

X_i	New case
R_s	The closest group
e	Minimal distance measure RMSE between a new case and existing groups centroids
δ	Threshold that determines the decision of update
R	New group
$R_s(n)$	Number of cases in the buffer
Z_{min}	Minimal number of cases in buffer that justifies the update of the buffer
Z_{max}	The buffer size – maximal number of cases stored in the buffer

The mechanism presented in Fig. 2 works as follows: The new item X_i passes through the sensor, the distance measure between the new item and the centroids of existing groups are calculated and the minimal value e is registered. The threshold level δ, the maximal buffer size Zmax and the rest of parameters have to be determined at this point of time. If the minimal distance e is less than a threshold, no rearrangement needed and a new case X_i joins the closest group (completion). If e exceeds the threshold level, the number of items in the buffer is checked. If there are not enough cases (the number of cases is less than Zmin), the new case creates an absolutely new group. If there are enough cases in the buffer, the new case removes the oldest case and a new distribution is created (by splitting or merging of the existing groups).

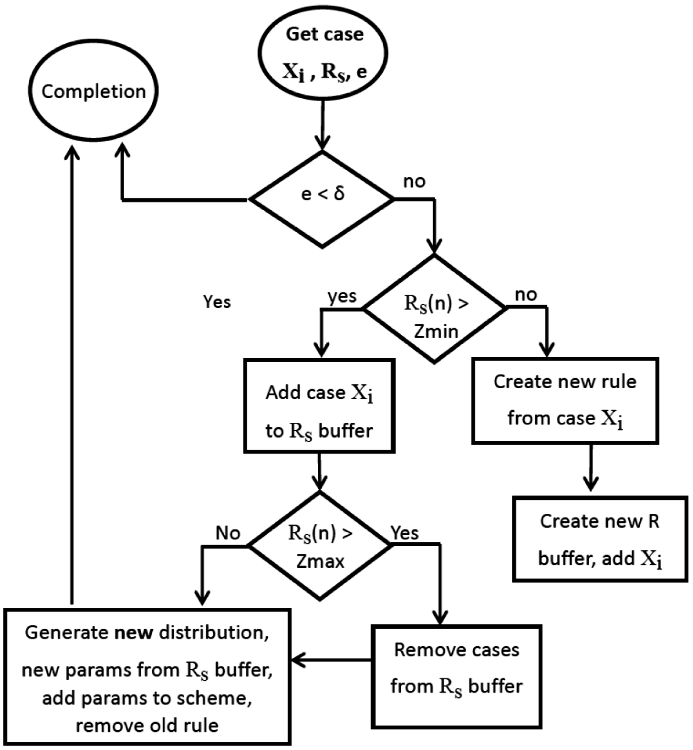
Get case
X_i, R_S, e
Completion
e < δ
no
Yes
yes
R_S(n) > Zmin
no
Add case X_i to R_S buffer
Create new rule from case X_i
Create new R buffer, add X_i
No
R_S(n) > Zmax
Yes
Generate new distribution, new params from R_S buffer, add params to scheme, remove old rule
Remove cases from R_S buffer

Fig. 2. A flow diagram of a process for real-time data classification based on dynamic updating of sensor's classifiers.

4 Model Validation

Since the model deals with a stream of real-time data, which is a continuous flow of new cases, the validation was based on datasets of classification problems. The model is implemented by the code developed in Python and combines the k-means algorithm package [24].

The following datasets were used: (1) "ERA" dataset, donated by Prof. Ben-David [25]. This data set was originally gathered during the academic decision-making experiment. Input attributes are candidates' characteristics (such as past experience, verbal skills etc.), output attribute is a subjective judgement of a decision-maker to which degree he/she tends to accept the applicant to the job or to reject him. All the input and output attributes have ordinal values. The data set contains 1000 instances, four input attributes and one output attribute. (2) "Car Evaluation" dataset that was retrieved from the UCI Machine Learning Repository [26–28]. Input attributes are cars properties and an output attribute is a class value (unacceptable, acceptable, good and very good). The data set contains 1728 instances.

4.1 Optimal Situation as a Baseline

The theoretical optima in such case is the situation in which the algorithm runs across the entire dataset. Thus, based on the results obtained by the clustering k-means algorithm, while analyzing all the records in the dataset, we can find the best set of rules, and the required total number of rules, that achieve the best classification accuracy.

4.2 The Initial Stage

According to widely used methodology in machine learning, each dataset was divided into training set (with about two thirds of data) and test set (with about one third of data). The training set provides the initial groups and the test set simulates a new data stream. Worth mention that the initial stage is mainly used to shorten the "reset-cycle" of the decision-making process. In cases where there is no urgency, the system can stat with no decision rule at all, and with totally empty "Case-buffers".

4.3 The "Dynamic-Flow" Stage

The test set was used in an unsupervised mode (while hiding the target-labeled field). The records flowed through the "Sensor" to the "Classifier-engine" without any information regarding the right classification-filtering.

- The "Classifier-engine" and the "Incremental-updater" used the flow diagram mechanism described in Fig. 2.
- The delta symbol (δ), in Fig. 2, represents Root-mean-square error (RMSE) that was used as a threshold.
- The parameters in each experiment were set as follows:
  ERA data-set: three threshold levels: 2, 2.25, 2.5; initial number of groups: 10; buffer size: 25; training set = 600, test set = 400.

Car evaluation data-set: three threshold levels: 0.8, 0.9, 1; initial number of groups: 15; buffer size: 25; training set = 1400, test set = 328.

In accordance with the schematic architecture of the system (illustrated in Fig. 1), a case is either directly classified, or initiates an incremental reevaluation (supported by the "Incremental-updater" component) till the threshold is satisfied, then the "Case-buffers" and the "Classifier-engine" are dynamically updated.

5 Results and Discussion

As shown in Figs. 3, 4 and in Table 2, we can see that although the learning mechanism uses only small data increments, it succeeds to perform good and consistent results. Figures 3 and 4 represent the dynamics of group set updating for different threshold levels. The process converges in both data sets for all sensitivity levels.

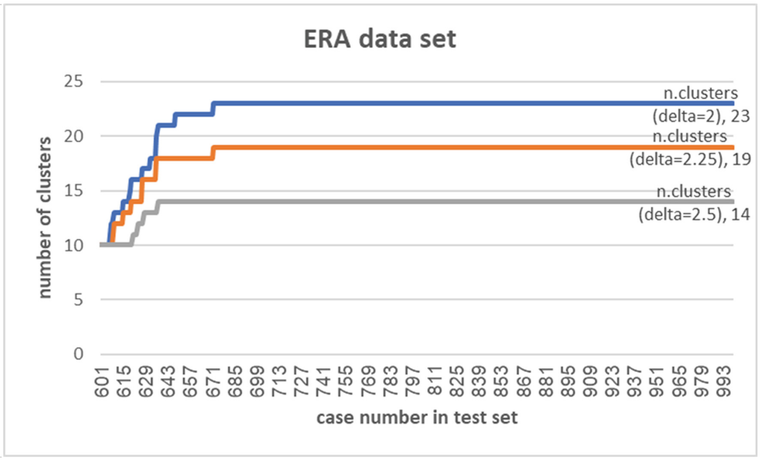

Fig. 3. Rules convergence using k-means with "ERA" dataset.

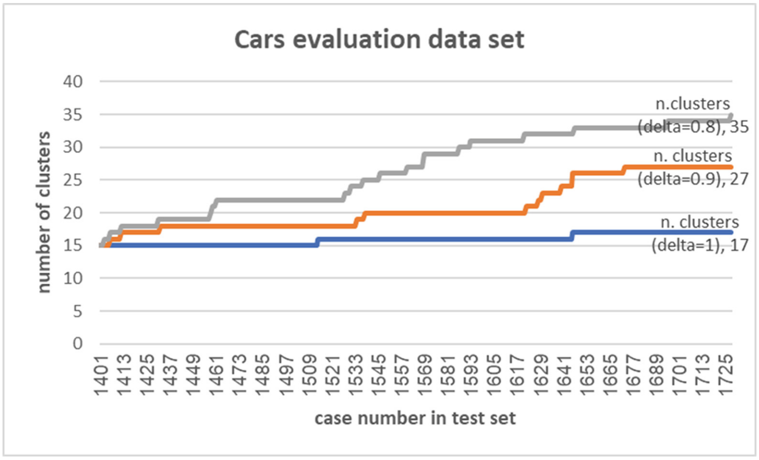

Fig. 4. Rules convergence using k-means with "Car evaluation" dataset.

We can see that in all three threshold values a reach a convergence of the classification process. In order to trace the dependence of aggregate rate of total number of groups on the sensitivity level, we chose three threshold levels for each data set. We can see that the convergence is faster as the threshold refers to lower accuracy value, but even at a high accuracy threshold, a relatively rapid convergence was achieved. The application of this result is very practical: on one hand, the dynamic data environment dictates us to act in real time, that is why we use small increments of data to be able to classify objects immediately; on the other hand, we need to provide good classification results and identify new trends or significant changes in data distribution. The convergence of classification process shows the ability of the proposed model to catch the critical moments when an update is needed, without too much computational effort. The updated groups set becomes more and more representative, that is why the periods of time between every two updates lengthens.

Table 2 presents the numerical results of all experiments in two data sets. The distance measure RMSE was calculated for each classified item (in most cases the distance between the item and at least one of the existing groups is less than a threshold level, so the item is joins the existing group; sometimes the threshold is achieved and the update is needed). The average and standard deviation of all minimal RMSE values are calculated for each experiment. The total number of groups in the end of each experiment is presented in addition. As sensitivity of a threshold level decreases (higher values of δ), the average distance measure grows. This result is expected: if a threshold level is relatively high, less items are defined as “far” or “non-similar” and more items succeed to join existing groups. Their minimal RMSE value is weighted into the calculation of average RMSE and we get bigger result. The same effect usually happens in standard deviation.

Table 2. The dynamic incremental updating of group set, according to threshold level.

Data set	δ	Average RMSE for classified instances	Std. dev.	Number of clusters
ERA Initial number of clusters = 10	2	0.9048	0.6349	23
	2.25	1.1967	0.6677	19
	2.5	1.4455	0.5842	14
Cars evaluation Initial number of clusters = 15	0.8	0.6	0.1133	35
	0.9	0.6871	0.1184	27
	1	0.7433	0.1199	17

In the conclusion of the above facts, we can see that the proposed incremental dynamic mechanism succeeds to achieve good results, that can be adopted in industry or in academical research as well.

6 Conclusions

Dynamic incremental classifier presented in this paper is designed to improve the classification process in state of dynamic data environment. The constant changes in data characteristics and preferences require from the mechanism immediate solutions. In addition to this obligatory condition, the process has to be economic. There is no dispute that the most qualitative solution will be obtained through the update of whole relevant data, but it is not possible in dynamic data environment. We assume that it is not possible to revise all previous data, so we choose to demonstrate the incremental mechanism that functions using small data buffers.

Experiments with different data sets showed that the loss of quality in classification results is not significant and the mechanism succeeds to identify the important changes in data stream and converges during the process.

The further research is planned in different possible directions: dealing with a big data sets that simulate big data environment; new trend and outlier detection; text data processing etc.

Acknowledgment. This work was supported in part by a grant from the MAGNET program of the Israeli Innovation Authority; who also submitted this work as a patent application.

References

1. Fayyad, U., Stolorz, P.: Data mining and KDD: promise and challenges. Future Gener. Comput. Syst. **13**, 99–115 (1997). https://doi.org/10.1016/S0167-739X(97)00015-0
2. Gelbard, R., Goldman, O., Spieger, I.: Investigating diversity of clustering methods: an empirical comparison. Data Knowl. Eng. **63**, 155–166 (2007)
3. Fan, J., Han, F., Han, L.: Challenges of big data analysis. Natl. Sci. Rev., 293–314 (2014)
4. Darema, F.: Dynamic data driven applications systems: a new paradigm for application simulations and measurements. In: Bubak, M., van Albada, G.D., Sloot, P.M.A., Dongarra, J. (eds.) ICCS 2004. LNCS, vol. 3038, pp. 662–669. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24688-6_86
5. Ren, K., Wang, C., Wang, Q.: Security challenges for the public cloud. IEEE Internet Comput. **16**, 69–73 (2012). https://doi.org/10.1109/MIC.2012.14
6. Li, J., Xu, H.: Suggest what to tag: recommending more precise hashtags based on users' dynamic interests and streaming tweet content. Knowl.-Based Syst. **106**, 196–205 (2016). https://doi.org/10.1016/j.knosys.2016.05.047
7. Miller, Z., Dickinson, B., Deitrick, W., et al.: Twitter spammer detection using data stream clustering. Inf. Sci. **260**, 64–73 (2014). https://doi.org/10.1016/j.ins.2013.11.016
8. Siddharth, S., Chauhan, N.C., Bhandery, S.D.: Incremental mining of association rules: a survey. Int. J. Comput. Sci. Inf. Technol. **3**, 4041–4074 (2012)
9. Cheung, D.W., Han, J., Ng, V.T., Wong, C.: Maintenance of discovered association rules in large databases: An Incremental Updating Technique, pp. 106–114 (1996)
10. Grira, N., Crucianu, M., Boujemaa, N.: Unsupervised and semi-supervised clustering: a brief survey. In: Review of Machine Learning Techniques for Processing Multimedia Content (2005)
11. Sreedhar, G.: Web Data Mining and the Development of Knowledge-Based Decision Support Systems. IGI Global, Hershey (2016)
12. Song, Y.C., Meng, H.D., Wang, S.L., et al.: Dynamic and incremental clustering based on density reachable. In: Fifth International Joint Conference on INC, IMS and IDC, 2009. NCM 2009, pp. 1307–1310 (2009)
13. Mishra, N., Hsu, M., Dayal, U.: Computer implemented scalable, incremental and parallel clustering based on divide and conquer (2002)
14. Deza, M.M., Deza, E.: Encyclopedia of Distances. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44342-2
15. Jain, A.K., Murty, M.N., Flynn, P.L.: Data clustering: a survey. ACM Comput. Surv. **31**, 264–323 (1999)
16. Vempala, S., Wang, G., Kannan, R., Cheng, D.: Techniques for clustering a set of objects (2010)
17. Thomas, S., Bodagala, S., Alsabti, K., Ranka, S.: An efficient algorithm for the incremental updation of association rules in large databases (1997)

18. Zhang, C-Q., Ou, Y.: Method for data clustering and classification by a graph theory model - network partition into high density subgraphs (2009)
19. Lughofer, E.: A dynamic split-and-merge approach for evolving cluster models. Evol. Syst. **3**, 135–151 (2012). https://doi.org/10.1007/s12530-012-9046-5
20. Toth, C.K., Grejner-Brzezinska, D.: Extracting dynamic spatial data from airborne imaging sensors to support traffic flow estimation. ISPRS J. Photogramm. Remote Sens. **61**, 137–148 (2006). https://doi.org/10.1016/j.isprsjprs.2006.09.010
21. Zhang, Y., He, S., Chen, J.: Data gathering optimization by dynamic sensing and routing in rechargeable sensor networks. IEEE/ACM Trans. Netw. **24**, 1632–1646 (2016). https://doi.org/10.1109/TNET.2015.2425146
22. Okeyo, G., Chen, L., Wang, H., Sterritt, R.: Dynamic sensor data segmentation for real-time knowledge-driven activity recognition. Pervasive Mob. Comput. **10**, 155–172 (2014). https://doi.org/10.1016/j.pmcj.2012.11.004
23. Ni, Q., Patterson, T., Cleland, I., Nugent, C.: Dynamic detection of window starting positions and its implementation within an activity recognition framework. J. Biomed. Inform. **62**, 171–180 (2016). https://doi.org/10.1016/j.jbi.2016.07.005
24. Download Python. In: Python.org. https://www.python.org/downloads/. Accessed 20 Apr 2018
25. Ben-David, A.: Automatic generation of symbolic multiattribute ordinal knowledge - based DSS's: methodology and applications. Decis. Sci. **23**, 1357–1372 (1992)
26. UCI Machine Learning Repository: Data Sets. https://archive.ics.uci.edu/ml/datasets.html. Accessed 10 Dec 2016
27. Bohanec, M., Rajkovic, V.: Knowledge acquisition and explanation for multi-attribute decision making, pp. 1–19 (1988)
28. Bittmann, R.M., Gelbard, R.: Visualization of multi-algorithm clustering for better economic decisions—the case of car pricing. Decis. Support Syst. **47**, 42–50 (2009). https://doi.org/10.1016/j.dss.2008.12.012