Online Clustering on Uncertain Data Stream

A Makhmutova¹, I Anikin²
Information Security Systems Department
Kazan National Research Technical University named after A.N. Tupolev-KAI,
420111, Kazan, Russia

phd.makhmutova@griat.kai.ru¹, anikinigor777@mail.ru²

Abstract. Importance of distributed systems nowadays cannot be overstated due to of enormously fast grown of the data to process. This data of different types comes from many devises in continuous flow, usually specified as data stream. In most cases data comes with part of uncertainty, because of the inaccuracy of hardware, unstable network, delays etc. Distribution and uncertainty create challenges for implementing data mining techniques for knowledge discovery. There are modern open source technologies and tools, for instance, products of Apache foundation like Storm, Samza, Spark Streaming and Flink, which allow work with data streams without concerns of technical boundaries, but they are not provide built-in methods for data mining of inaccurate data. Thereby, this work mainly focuses on research of online part of one of data mining technique, such as clustering, for uncertain data stream in distributed systems.

1. Introduction
The creation of distributed processing systems (DPS) in early 1980’s [1] was closely connected with the fact that single machine cannot compute huge amount of information with low level of latency. There are different types of distributed computing [2]:

- cluster computing – group of similar computers or workstations with alike configuration and one master node with management property, and computing nodes as “slaves”;
- grid computing - federation of systems, each of which has its own purpose (resources, communication, management, access control and usage);
- cloud computing – usage of outsource resources and distribution of computational power. A scalable system that can include many logical layers.

Multiple ideas and solutions was created since that time, but presented in 2004 by Google MapReduce concept made Big Data processing believable and real. Main principle of parallelization based on three steps: map, shuffle and reduces. On map step some function is applied on splitted data set of [key, value] pairs, which produces result, shuffle step will group by the keys and later on reduce step the function calculate the output. There are many frameworks started to developing after MapReduce introduction, basically organized on cluster type distribution. Also there are some solutions for real-time processing, or usually called, streams processing. MapReduce can be assigned to batch-oriented data processing paradigm, which is slightly different from true stream processing. A real-time processing application has to have of replicable and predictable results [3]. Some differences between distributed batch and stream processing systems are presented in Table 1.
Table 1. Differences between batch and stream DPS.

| Batch                                      | Stream                                      |
|--------------------------------------------|---------------------------------------------|
| Compute function on all data in batch      | Computes function of single coming data element |
| Function might be big and complex          | Function needs to be relatively simple       |
| More considerations on throughput of individual components of the computation | Independent computations                     |
| Synchronized - data source has time to produce another batch | Asynchronous - source of data doesn’t interact with the stream processing directly (waiting for an answer) |

Relevance of a knowledge discovery task is growing from year to year because of the increased amount of data and complexity of existing systems. Well performing data mining techniques and machine learning algorithms must be adapted for work in distributed way and in conditions of real-time. For satisfying performance of approaches in data streams also the following requirements must be fulfilled [4]: quick processing of arriving data record, compact representation, identification and handling of outliers. In reality for described requirements one more difficulty is added – quite often applications provide data with part of inaccuracy, imprecision, and sometimes even with lack of information – such streams can be called as uncertain. There are many sources of origin of uncertainty: sensor network and their inaccuracies in measurements, delays of network and data transmission, incorrect tracking and so on. In databases may exist tuple-level uncertainty (question about existence of tuple in relation) or attribute-level uncertainty (question about reliability of obtained value). In case of first type of uncertainty possible world semantics method exist [5], which expands original database by all possible meanings and calculate probability for each of them. For attribute-based uncertainty it’s possible to determine Probability Density Function (PDF) or calculate acceptable deviation of values, which can be expressed through standard error $r$, which means the average deviation of the data ($x_i \pm r$).

From all above follows, that processing of data streams nowadays is challenging operation not only because of requirements and limitations of hardware, but also volume, variety and veracity of coming data. We are faced with the task of adapting knowledge discovery algorithms for modern distributed systems in case of obtaining data with some level of uncertainty. Also it’s necessary to compare work of two clustering algorithms – for deterministic and uncertain data - and make a decision about their efficiency in distributed environment.

2. Machine learning in Distributed Stream Processing Platforms

Data mining techniques imply the preliminary collection of data, carried out by a person with some goal in mind. They can be divided into two large subgroups: mining techniques with and without pre-processing phase [6]. In real-time processing, initially, characteristics of data in stream may not exist, and through some time records may evolve with time and be different from initial. Clustering is genuine choice for such systems, because it allows not only to identify similar records and group them in clusters without any knowledge about coming data [5], but also to adjust for the new conditions.

With the increasing amount of data streams and frequent use of the term Big data in its processing scope, different platforms for real-time processing was developed. Nowadays there are already four stable open-source framework exists under Apache license - Samza, Storm, Spark Streaming and Flink. In Table 2 differences between this systems is represented. There are few libraries or frameworks with implemented clustering techniques for this systems – Spark’s ML API [7], FlinkML, Apache SAMOA [8], StreamDM [9]. Despite the number of decisions, none of them implemented algorithm for uncertain stream clustering.
Table 2. Comparison of distributed stream processing tools.

| Characteristics          | Samza     | Storm     | Spark Streaming | Flink     |
|--------------------------|-----------|-----------|-----------------|-----------|
| Processing type          | Stream-only | Stream-only | Hybrid (batch and stream) | Hybrid (batch and stream) |
| Message delivery guarantee | at-least-once | at-least-once | exactly-once | exactly-once |
| Latency                  | sub-second | sub-second | seconds         | sub-second |
| Throughput               | high      | low       | high            | high      |
| Programming models       | compositional | compositional | declarative | declarative |
| Fault tolerance          | yes       | yes       | yes             | yes       |
| Stateful processing      | yes       | yes       | yes             | yes       |
| In-memory                | yes       | yes       | yes             | yes       |
| Iterations               | no        | no        | yes             | yes       |

3. Clustering of uncertain data stream

In most of data streams approaches clustering process divided into two phases - online and offline. Large amount of coming records cannot be stored in memory, consequently, online part allows process information in single pass [10], due to collection of summary information about the received record. The algorithm in offline phase is selected based on type of preserved information in online phase. Online phase process information of coming records and summarize it into so-called micro-clusters [11], extended concept of Cluster Feature Vector (CFV). There are density-based algorithms DenStream [12] and DCUSStream [13]. Distance-based methods also uses a general model, which came from deterministic stream processing e.g. Umicro [14], LuMicro [15] and EMicro [16].

In the scope of this work it was decided to compare work of two similar algorithms CluStream [17] and UMicro [14]. The results of the computing capacity evaluation criteria (each algorithm work with different levels of noise on varying computing capacity - local and distributed) represented on the Figure 1 and demonstrate tendency, that with the adding computational capacity to the system, UMicro approach will shows better greater result on data processing. The Table 3 contain result of evaluation of the standard deviation of cluster centers with adding noise from original “clean” centers for each dimensions (the deviation was calculated for 7 different noise levels and was calculated by simple standard deviation formula for obtained cluster center and original centers). From the result of evaluation, it is possible to conclude, that UMicro shows a less deviation from the initial “clean” results on the data with uncertainty, than CluStream algorithm.

Table 3. Sample standard deviation for cluster’s center for CluStream and UMicro

| Number of clusters | CluStream |       | UMicro |       |
|--------------------|----------|------|--------|------|
|                    |         | 1    | 2      | 3    |
|                   |         | 1    | 2      | 3    |
|                   |         | 1    | 2      | 3    |
|                   |         | 1    | 2      | 3    |
|                   |         | 1    | 2      | 3    |
|                   |         | 1    | 2      | 3    |
|                   |         | 1    | 2      | 3    |
|                   |         | 1    | 2      | 3    |
|                   |         | 1    | 2      | 3    |
|                   |         | 1    | 2      | 3    |

3
4. Conclusion
In both evaluation criteria – accuracy of clustering and time performance - implemented UMicro algorithm demonstrated the greater performance: it is provide accurate cluster formation despite the level of noise in the data in distributed system. So we can make an assumption about possibility of expansion of machine learning algorithms in distributed areas.

![Different computing power on varying noise levels](image)

**Figure 1.** Varying the number of computing capacity on different levels of noise.

References
[1] Probert R L, Fischer M J, and Santoro N 1982 Tsymposium on principles of distributed computing *Proceedings of the First ACM SIGACT-SIGOPS*
[2] Tanenbaum A S and Steen M V 2013 *Distributed Systems* (Amsterdam: Pearsons) p 25
[3] Stonebraker M, Cetintemel U, and Zdonik S 2005 The 8 requirements of real-time stream *Processing ACM SIGMOD Record* 34(4) 4247
[4] Barbar D 2002 Requirements for clustering data streams. *ACM SIGKDD Explorations Newsletter* 3(2) 23–27
[5] Guha S and Motwani R 2002 Clustering data streams: theory and practice. *IEEE Transactions on Knowledge and Data Engineering* 15(3) 515–528
[6] Reddy V S, Rao T V, and Govardhan A 2017 Data mining techniques for data streams mining *Review Of Computer Engineering Studies* 4(1) 31–35
[7] Mllib: Rdd-based api. [Online] (https://spark.apache.org/docs/latest/mllib-guide.html)
[8] Morales G D F and Bifet A 2015 Samoa: Scalable advanced massive online analysis *The Journal of Machine Learning Research* 16(1) 149153
[9] Bifet A, Maniu S, Qian J, Tian G, He C, and Fan W 2015 Streamdm: Advanced data mining in spark streaming *IEEE International Conference on Data Mining Workshop (ICDMW)* p 1608
[10] Aggarwal C C and Reddy C K 2013 *Data Clustering: Algorithms and Applications* (Chapman and Hall/CRC)
[11] Zhang T, Ramakrishnan R, and Livny M 1996 Birch: An efficient data clustering method for very large databases. *Proceedings of the 1996 ACM SIGMOD international conference on Management of data* pp 103–114
[12] Cao F, Ester M, Qian W, and Zhou A 2006. Density-based clustering over an evolving data stream with noise *Proceeding The 2006 SIAM Conference on Data Mining* pp 328–339
[13] Yang Y, Liu Z, Zhang J, and Yang J 2012 Dynamic density-based clustering algorithm over uncertain data streams Proceedings. In 9th International Conference on Fuzzy Systems and Knowledge Discovery pp 2664–2670

[14] Aggarwal C and Yu P S 2008 A framework for clustering uncertain data streams Proceedings of the 2008 IEEE 24th International Conference on Data Engineering pp 150–159

[15] Zhang C, Gao M, and Zhou A 2009 Tracking high quality clusters over uncertain data streams Proceedings. of the 2009 IEEE 25th International Conference on Data Engineering pp 1641–1648

[16] Zhang C, Jin C-Q, and Zhou A-Y 2010 Clustering algorithm over uncertain data streams Journal of Software 21(9) 2173–2182

[17] Aggarwal C C, Han J, Wang J, and Yu P S 2003 A framework for clustering evolving data streams Proceedings of the International Conference on Very Large Data Bases 29 81–92