Available online at http://www.ijcrls.com

International Journal of Current Research in Life Sciences
Vol. 07, No. 02, pp.1204-1207, February, 2018

ISSN: 2319-9490

RESEARCH ARTICLE

OUTLIER DETECTION IN BIG DATA USING OPTIMIZED ORION TECHNIQUE

***Astha Tripathi and Dr. Raghav Yadav**

Department of Computer Science & Information Technology Sam Higginbottom University of Agriculture, Technology and Sciences (SHUATS), Allahabad, UP, India

Received 25th December, 2017; Accepted 18th January, 2018; Published Online 28th February, 2018

ABSTRACT

Big data is data set that cannot reasonably be handled by our traditional database or tools due to the large volume, velocity and variety. In this vast and complex data set outlier detection is very crucial aspect. A failure to detect outliers or their ineffective handling can have serious ramifications on the strength of the inferences drained from the exercise. This dissertation proposes a reliable and high throughput outlier detection technique which attempts to detect projected outlier in high dimensional data stream. Optimize ORION algorithm employs an innovative window based time model in capturing dynamic statistics from stream data, and novel data structure containing a set of top sparse subspaces to detect projected outlier effectively in low process time and less memory storage. This algorithm able to identifies a data point as an outlier if it resides in low density region. Optimized Orion Increase throughput 20% and enhance stability 25%.

Key words: Data mining, Outlier detection, Orion, synthetic dataset, clustering, numerical data

Citation: Astha Tripathi and Dr. Raghav Yadav, 2018. "Outlier detection in big data using optimized Orion technique" *International Journal of Current Research in Life Sciences*, 7, (02), 1204-1207.

INTRODUCTION

Big data is often characterized by 3Vs the extreme volume of data, the wide variety of data types and the velocity at which the data must be processed. Although big data doesn't equate to any specific volume of data, the term is often used to describe terabytes, petabytes and even exabytes of data captured over time. And Outlier points indicate faulty data or certain set of data that might not be valid (Varun Chandola and Banerjee and Kumar). Detecting outliers gets even more difficult when the data is highly variable, the surface your data sits on is not flat, or your data exists in a three-dimensional setting. The bigger your dataset, the greater your chance of stumbling into an outlier. It's practically a certainty you'll find isolated, unexpected, and possibly bizarre data you never expected to see in your data. But how you respond to these outliers could mean the difference between big data success and failure. Outliers can be critically important to big data project. Depending on the context, it may be actively hunting for outliers, or may be trying to subdue them. In big data project, first need to detect the outliers. Taking the time to explore which approach works best for detection will give the best chance of finding success with big data project (Desai, 2011). Outliers existing in high-dimensional data streams are embedded in some lower-dimensional subspaces. Here, a subspace refers to as the data space consisting of a subset of attributes. These outliers are termed projected outliers in the high-dimensional space. The existence of projected outliers is due to the fact that, as the dimensionality of data goes up, data tend to become equally distant from each other. As a result, the difference of data points' outlier-ness will become increasingly weak and thus undistinguishable (Vijayarni and Nithya, 2011). Only in moderate or low dimensional subspaces can significant outlier-ness of data be observed. Outlier detection for single streams compares a data point in a stream with respect to the history data points from that same stream in order to identify whether the data point is an outlier. In case of multiple data streams, such identification can be done either by (1) comparing the data point with the history data points from the same stream that carries the data point, (2) comparing the data point with the data points from the other correlated streams, or (3) using a combination of both (1) and (2). The opportunity of having multiple data streams to compare allows richer semantics across the data streams to be taken into consideration which would lead to better detection accuracy. The outliers may be of particular interest, such as in the case of fraud detection, where outliers may indicate fraudulent activity. Thus, outlier detection and analysis is an interesting data mining task, referred to as outlier mining or outlier analysis (Jae-Gil). A data stream is an infinite sequence of data points $\{Dt \mid t \geq 0\}$ with explicit or implicit timestamps. Many data stream applications involve monitoring, so that a particular data point is interesting only for a specific amount of time. Therefore, every data point has to be processed in time.

Corresponding author: **Astha Tripathi**
Department of Computer Science & Information Technology Sam Higginbottom University of Agriculture, Technology and Sciences (SHUATS), Allahabad, UP, India

Data streams are often characterized by uncertainty because of unreliable communication and error (Hendrik Fichtenberger, 2013). For multi-dimensional data streams, the number of dimensions makes outlier detection even more complicated. The most important problem in detecting multi-dimensional outliers is similarity measurement. Calculating the distances between data points is a popular approach to similarity measurement. This paper proposes an outlier detection algorithm for data streams, called Optimized Orion that addresses all their issues related to outlier detection in case of big data including transiency, temporal relation among data points, waiting time of data stream, throughput and overall storage cost (Pedro Pereir a Rodrigues *et al.,* 2008).

Background

There are various techniques to detect outliers in big data. Orion technique is one of them. Orion detect a data point is an outliers if the data point has drastically different volume compared to other data point. Orion processes each data point from one stream.

Orion goes through three phases.

- Finding an appropriate projected dimension i.e. p-dimension.
- Computing the outlier metrics for data point DT.
- Based on these metrics, determining if DT is an outliers or not.

To find p-dimension, Orion uses an Evolutionary Algorithm (EA). Each data point has a value along p-dimension. If this value has less number of neighbors than the values of other data point thus it is very likely outliers. After that Orion adds new p-dimensions and removing the old ones and these new p-dimensions again compare the data point to other data points of the data stream. So when any data point DT arrives, Orion picks the p-dimension that has smallest neighbor density to reveal the outlierness of DT. Orion uses two outlier metrics for DT.

- Neighbor density
- K-distance

If data points have much fewer neighbor means it has smallest ND and largest K-distance compare to other data points then it must be an outlier. When a data point arrives, Orion updates the DDFs of all *p*-dimensions. Each *p*-dimension has a DDF based on the data points arrived after its creation. The DDF of a *p* dimension follows the DDF proposed in [7], which addresses transiency, temporal relation, uncertainty and concept drift for single-dimensional data streams, but is modified for a dynamic implementation that does not require the range of values to be known in advance. It is based on a kernel density estimator that estimates the DDF based on the projection of the data values on a *p*-dimension. For any *p*-dimension, we use the DDF. Crossover finds two parent *p*-dimensions with high fitness and creates a new individual that performs better than its parents. These two individuals are selected according to the rank selection scheme [8], in which the probability of selecting an individual is proportional to its fitness. According to Lemma 3, the linear combination of two *p*-dimensions produces a new *p*-dimension that has a smaller SD than at least one parent.

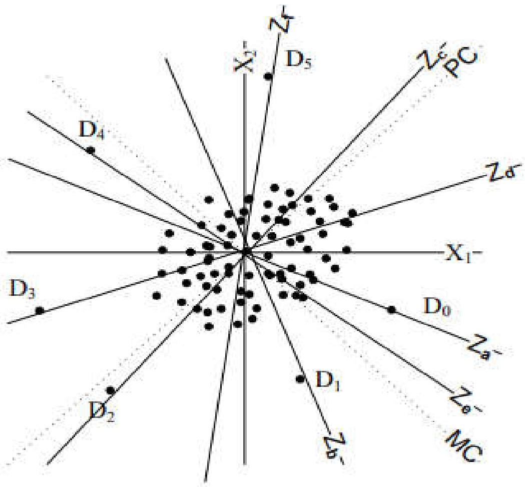

Fig. 1. Outlier detection with p-dimensions

Proposed work

After having a review on the traditional work that was done in the field of outlier detection in big data it looks simple but is highly challenging due to following reasons:

- This process hold data stream for long time period to detect p-dimension for every data point.
- In case of big data velocity of data stream is very high so the problem of network jamming can also occur.
- Waiting time of data stream also a big challenge.
- In case of Orion, storage cost of data stream in detection process is also very high.

First motivation of Optimized Orion is to perform operations on replica of data stream i.e. generally treated as waste in database. The second motivation of Optimized Orion is to avoid the effect of outliers by using roll back concept.

Overview: To determine if a incoming data point DT is an outlier Optimized Orion goes through these phases: (1) Analyzing the data stream (2) Create replica of data stream RDS. (3) Make Free the original data for further processing (4) find out p-dimensions for replica RDT. (5) Using metrics to determine outlierness of data point DT. (6) If DT is an outlier then roll back that data point.

Algorithm of Optimized Orion

1. Initialize data stream DS.
2. Create replica (RDS) of data stream DS.
3. Make free original data stream DS for further processing.
4. Determine appropriate p-dimension for RDS.
5. Calculate neighbor density for each data point of the RDS.
6. Prepare the density list.
7. Calculate k-distance for data point.
8. k- integral ← k- integral (DT, k).’
9. If detect outlier (n den, k-dis, h-center, v-center)
10. Then is outlier ← T else is outlier F ←
11. End if
12. If outlier ← T then roll back DT else not.
13. End if
14. END.

Create replica of whole data stream and apply all the metrics on this replica and make free the original data stream for further processing and when data point will found guilty (Outlier) then roll back that data point. With the help of this algorithm delay will not occur and process is also safe because of roll back concept and do not have need to store the replica of data stream. We improve the overall throughput with less storage cost and low network traffic.

Performance analysis

Based on the result, it would essential to discuss about the performance of the proposed algorithm and other aspects related with the work. Here ten data stream is to be taken for both Optimized Orion and Orion protocols, which is implemented; number of sample data points 4000, maximum degree of freedom is 8000, and stability of network life time is also changed during the whole process. The parameter values for different configurations are given in Table 1.

Table 1.

Parameters	Values	Remarks for Optimized Orion
Implemented data stream	10	For both protocol
Number of data points	4000	For both protocol
Degree of freedom	8000	For both protocol
Stability	25%	Enhanced
Throughput	20%	Increase
Waiting Time	42%	Reduce

Waiting Time: Figure 2 shows the waiting time of the data streams using Optimized Orion technique. Waiting time For DS4 is 12 ms as shown in figure. The waiting times of data stream ensure that how much time consumes by pervious data stream for process. From the figure it can be clearly shown that the delay in the processing of data stream is quite low.

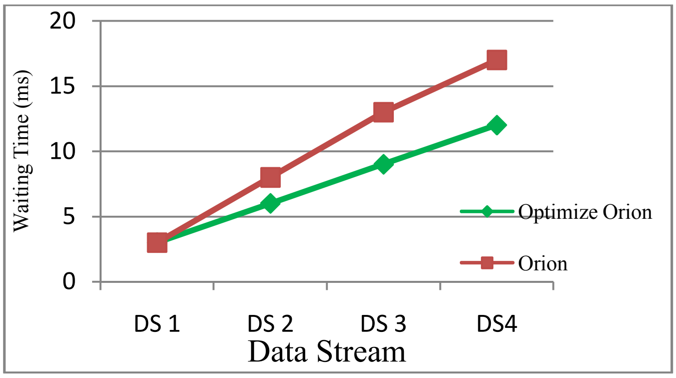

Fig. 2. Waiting Time Graph

Execution Time: Run time or execution time is the time during which program is running, in respect to other program lifecycle phases such as compile time, link time, and load time. Figure 3 shows the execution time graph**.** The time spends by the job actively using processor resources is its execution time.

Throughput: Throughput is an important indicator of performance and quality of network connection. A high ratio of unsuccessful data packets will ultimately lead to lower throughput and degraded the performance. Figure 4 shows the throughput graph of Optimized Orion. With the help of Optimized Orion more data stream can be processed in comparison to Orion. And it also reduces the chance of the fault because Optimized Orion uses the roll back concept.

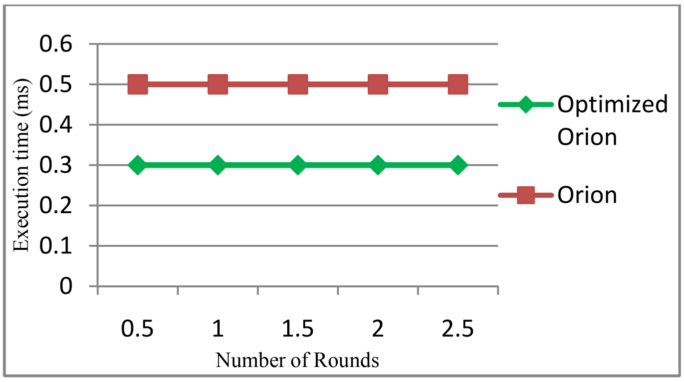

Fig. 3. Execution time graph

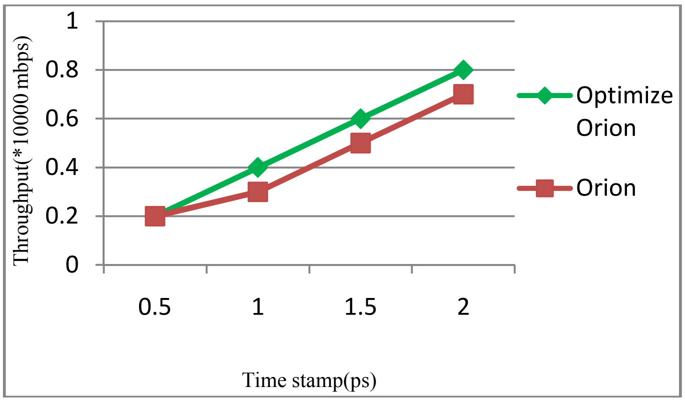

Fig. 4. Overall throughputs

Overall Storage Cost

Proposed Optimized Orion also reduce the overall cost of storage because we do not store the replica of data stream for future use and also release the original data stream. So the overall cost of storage is also reduces as shown in figure 5

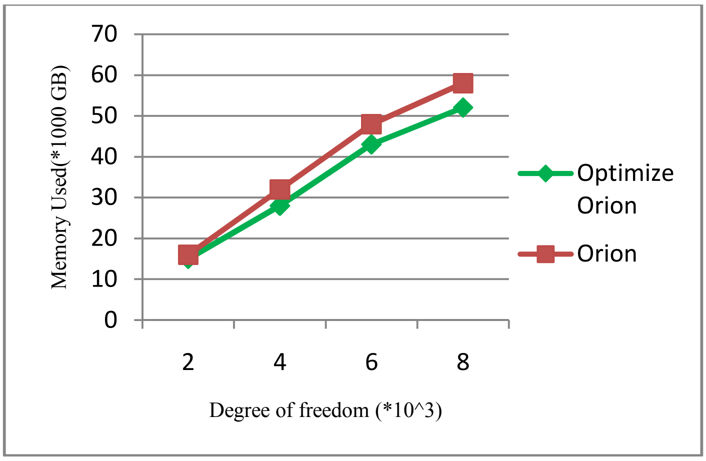

Fig. 5. Cost of Storage

Conclusion

Outlier detection for multi-dimensional data stream is relatively new area of research. Outlier detection for multi-dimensional data stream posses' critical challenges. Outlier detection requires similarity among data points but here we deal with big data base which deals with high verity of data points. In this dissertation we have proposed an effective and efficient outlier detection technique with high throughput and less waiting time, Optimized Orion. Optimized Orion is designed for multiple data streams that may or may not be correlated. Unlike other approaches, Optimized Orion does not assume equality correlation among the data points from multiple streams.

REFERENCES

Chuang-Cheng Chiu and Chieh-Yuan, T sai, 2007. A k-Anonymity Clustering Method for Effective Data

De Andrade Silva, J, Extending k-Means-Based Algorithms for Evolving Data Streams with Variable Number of Clusters .IEEE, Published in: Machine Learning and Applications and Workshops (ICMLA), 2011 10th International Conference on Volume: 2

Desai, H. 2011. "Comparative Study of K-means Type Algorithms", UNIASCIT, Vol. 2.

Hendrik Fichtenberger, Marc Gillé, Melanie Schmid, in Algorithms–ESA2013, Volume 8125, 2013, pp 481-492

Jae-Gil, "Trajectory Outlier Detection: A Partition-and-Detect Framework", Department Of Computer Science, University of Illinois at Urbana-Champaign Urbana, IL 61801, USA.

Jian Wang, Yongcheng Luo, Yan Zhao Jiajin Le, 2009. "A Survey on Privacy Preserving Data Mining", First International Workshop on Database Technology and Applications.

MdZahidul Islam, Ljiljana Brankovic, 2011. "Privacy preserving data mining: A noise addition framework using a novel clustering technique", Elsevier.

Mira A. and S. Saharia, 2012. "A Robust Outlier Detection Using Hybrid Approach", *Aamerican Journal of Intelligent System* 2012.

Mohd - Al- Zoubi, 2010. "New Outlier Detection Method Based On Fuzzy Clustering", (IJAR) Vol.4.

Pachgade, S. D. and S.S. Dhande, 2012. "Outlier detection Over Data Set Using Cluster Based and Distance - Based Approach", (IJARCSSE), Volume 2, Issue6.

Parneeta Dhaliwal, MPS Bhatia and Priti Bansal, 2010. "A Cluster-based Approach for Outlier Detection in Dynamic Data Streams (KORM: k-median OutlieR Miner)", Journal of Computing, Vol. 2, No. 2, ISSN 2151-9617, pp. 74-80.

Pedro Pereir a Rodrigues, João Gama, João Pedro Pedroso, 2008. "Hierarchical clustering of Time series Data Streams", IEEE Transactions on Knowledge and data engineering, Vol 20,no.5,pp. 615-627.

Varun Chandola and Banerjee and Kumar, "Outlier Detection: A Survey".

Vijayarni S.and S. Nithya, 2011. "An Efficient Clustering Algorithm for Outlier Detection", (IJCS) Vol.32.

Ville Hautamaki, Svetlana Cherednichenko, Ismo Karkkainen, Tomi Kinnunen, and Pa si Fr anti, "Improving k-Means by Outlier Removal", SCIA, LNCS 3540, 2005, pp. 978–987

****