Volume : 4 | Issue : 6 | Special Issue June 2015 • ISSN No 2277 - 8179

# Continuous Outlier Detection Based on Sliding window on Continuous Data Streams

Engineering


| | |
|---|---|
| **SHIMNA T** | Computer science and engineering , VedaVyasa Institute of Technology, Kozhikode |
| **S.KAVITHA MURUGESAN** | HOD, Dept. computer science , VedaVyasa Institute of Technology, Kozhikode. |



## ABSTRACT

Anomaly detection is considered an important data mining at the discovery of element (also known as outliers) that slow significant diversion from the expected case. This paper studies the problem of outlier detection on continuous data streams. The proposed system MCOD(Micro –Cluster-Based Continuous Outlier Detection)algorithms for continuous outlier monitoring on deterministic data streams based on the sliding window .In this paper ,we design efficient algorithms for continuous monitoring of distance-based outliers, in sliding windows over data streams, aiming at the elimination of the limitations of previously proposed SVDD algorithms. Our primary concerns are efficiency improvement and storage consumption reduction. The proposed algorithms are based on an event-based framework that takes advantage of the expiration time of objects to avoid unnecessary computations. The MCOD algorithm is an outlier detection method based on micro-cluster. This technique is able to reduce the required storage overhead ,run faster than previously proposed SVDD technique and offers significant flexibility. Experiments performed on real-life as well as synthetic data sets.


### 1. NTRODUCTION

Intrusion detection is a very important topic of network security that has received much attention since potential cyber threats are making the organizations vulnerable. Intrusion Detection Systems (IDS) are intended to protect information systems against intrusions and attacks and are traditionally based on signatures of known attacks. The main drawback is that in case of an emerging attack, based on the recent discovery of a new security hole for instance, the IDS will ignore it since this new attack has not yet been listed in the base of signatures.

In this paper, we present efficient algorithms for the continuous and real-time monitoring of outliers on uncertain data streams over sliding windows. In summary, the major contributions of this work are as follows:

- A new algorithm Micro Continuous Uncertain Outlier Detection (MCOD) is designed for outlier detection on uncertain data streams. The algorithm is able to quickly determine the nature of an uncertain element by probabilistic pruning, to further improve the efficiency

- The propose a pruning approach Probability Pruning for Continuous Uncertain Outlier Detection (PMCOD) based on PMCOD by estimating the outlier probability. In this way, the amount of calculation is effectively reduced, thus meeting with the real-time criteria of processing uncertain data streams.

- As multiple levels of sensitivity may be required, MCOD is optimized such that parameter variable queries can be run concurrently

### 2. RELATED WORKS

Outlier detection has been studied in the literature, both in the context of multi-dimensional data sets [1] and in the more general case of metric spaces [2]. Usually, the proximity among objects is used to decide if an object is an outlier or not. However, specialized techniques may also be applied.

#### 2.1 Clustering-Based Approaches

They always conduct clustering-based techniques on the samples of data to characterize the local data behavior[6]. In general, the subclusters contain significantly less data points than other clusters, are considered as outliers. Since clustering based approaches are unsupervised without requiring any labeled training data, the performance of unsupervised outlier detection is limited.

2.2 Density-Based Approaches[7]

One of the representatives of this type of approaches are local outlier factor (LOF) [7] and variants. Based on the local density of each data instance, the LOF determines the degree of outlierness, which provides suspicious ranking scores for all samples [7][6]. The most important property of the LOF is the ability to estimate local data structure via density estimation.

### 3. PROBLEM DEFINISION

The problem has been studied extensively by the statistics community, where the objects are modeled as a distribution, and objects are marked as outliers depending on their deviation from this distribution. However, for large dimensionalities SVDD techniques [4] fail to model the distribution accurately, leading to performance degradation. In addition, these techniques do not scale well for large databases.

### 4. PROPOSED APPROACH

Data stream is a special data model in data processing, characterized by high data throughput rate and potentially large amounts of data. In this model, data does not take the form of persistent relations, but rather arrives in multiple continuous rapidly time-varying data streams. In this case, traditional algorithms fail to meet the needs of processing. Since the stream is continuously updated, it is impossible to maintain all tuples in local memory. A sliding window is used to track the most recent data

#### 4.1. Synthetic Data Generation

In the field of mathematical modeling, a radial basis function network is an artificial neural network that uses radial basis functions as activation functions. The output of the network is a linear combination of radial basis functions of the inputs and neuron parameters. Radial basis function networks have many uses, including function approximation, time series prediction, classification, and system control. The number of neurons in the hidden layer, is the center vector for neuron, and  is the weight of neuron  in the linear output neuron. Functions that depend only on the distance from a center vector are radially symmetric about that vector, hence the name radial basis function. In the basic form all inputs are connected to each hidden neuron. The norm is typically taken to be the Euclidean distance and the radial basis function is commonly taken to be Gaussian. The Gaussian basis functions are local to the center vector in the sense that changing parameters of one neuron has only a small effect for input values that are far away from the center of that neuron..

**4.2 Distance Based Model**

The proposed system design efficient algorithms for continuous monitoring of distance-based outliers [2][3], in sliding windows over data streams, aiming at the elimination of the limitations of previously proposed algorithms. The primary concerns are efficiency improvement and storage consumption reduction. The proposed algorithms are based on an event-based framework that takes advantage of the expiration time of objects to avoid unnecessary computations.

**4.3 Event Based Approach**

The system interested in tracking the outliers in a set of objects of a stream defined by a sliding window. In particular, a set of outliers is maintained subject to arrivals of new objects from the stream and departures of existing objects due to the restricted window size. The arrival and departure of objects has the effect of a continuously evolving set of outliers.

An event is the process of checking whether an inlier becomes an outlier due to departure of objects from the window. The expiration time of the objects is known whether we talk about time-based windows or for count-based windows. Thus, the time stamp of an event depends on the expiration time of objects. This forces a total order on the events which can be organized in an event queue

The event-based method for outliers employs an event queue to efficiently schedule the necessary checks that have to be made when objects depart. Thus, in the event queue there are only stored inliers since only these can be affected by the departure of an object. Arrival of new objects results in potential updates of the keys of some objects in the event queue. Additionally, existing outliers are checked as to whether they have become inliers and thus they should be inserted in the event queue.

**4.4 MCOD Outlier Detection**

Proposed methodology is based on the concept of evolving micro-clusters that correspond to regions containing inliers exclusively. The resulting algorithm is denoted as MCOD (Micro-cluster-based Continuous Outlier Detection).

However, each user comprehends the notion of outlier differently by varying values of R and k. Each pair of R and k determines a query q of distance-based outlier detection. Therefore D(q.R, q.k) denotes the outliers of query q from the set of all queries Q. In this section, we study the continuous evaluation of multiple queries. For simplicity, we discuss separately the case in which k varies and R remains constant and vice-versa.

When an object departs, if the examination of an object p, at p.ev time instance, reports p as outlier we check the other queries in Q whether p is also outlier in them. In particular, for each query q, if n°p + n+p < q.k, then p is outlier in q.

Queries are examined with decreasing order of k, and this procedure is terminated as soon as we reach a query for which p is inlier. Moreover, when a new object arrives, if object p 2 D°, q.kmax) and its counter n+p is increased, we check all the queries for a possible move of p from outlier set to inlier set. Notice that p is not necessarily outlier in all queries. For each query q, if p 2 D(R, q.k) and n°p + n+p q.k then p should be removed from D(R, q.k). The queries are examined again in decreasing order of k and the procedure is stopped when we reach a query in which p is not outlier.

We proceed now with the examination of the case of fixed k and varying R. In this case, two sets for each object p are maintained, the sets Pp and Sp (recall that we only stored the size of Sp) along with their distances from p, by taking into account the maximum distance Rmax = max{qi.R} (0 Σ i Σ |Q|). When R varies it is necessary to maintain Sp since the neighbors of an object depend on the radius of the query.

This may lead to high memory requirements, since in the worst case the number of neighbors can reach the number of active objects n. Assuming the maximum distance Rmax, if n+p > k we can maintain the k neighbors with the smaller distances from p. This is because neighbors with larger distances will not be used in any query. Therefore the size of Sp is limited to k objects.

The detailed steps of the modified algorithm after each window slide are as follows:

**Step 1:** The expired objects are purged after having updated the counters mcn of corresponding micro-clusters (if any), accordingly. Subsequently, steps 2 and 3 are performed for each new data object p; new objects are processed in the order of their arrival.

**Step 2:** For each p, we detect (i) the micro-cluster, the center of which is closest to that object, and (ii) all micro-clusters, the centers of which are within a 32 R range. Conflicts (i.e., when there are two centers with equal distance) are resolved arbitrarily.

**Step 3:** If the distance from the closest center is not greater than R/2, then: (3a-i) the new object is assigned to the corresponding micro cluster and the value of p.mc is updated; (3a-ii) the size of the corresponding micro-cluster is increased by one; (3a-iii) let MCi be the micro-cluster where the new object is inserted. We evaluate the distance between the new object and all objects in PD that contain MCi in their Rmc lists, to check (i) whether the number of succeeding neighbors of the latter should be increased and (ii) whether any previous reported outliers have become inliers; Otherwise, i.e., if the distance from the closest center is greater than R/2, no assignment takes place and the following process is applied: (3b-i) For the new object p that has not been assigned to a micro-cluster, we perform a range query taking into account only (i) the objects in PD and (ii) the objects in the microclusters for which the distance from their centers is not greater than 32 R (the relevant micro-clusters have been detected in Step 2). (3b-ii) If the number of neighbors from the PD set within R/2 distance exceeds µk, µ 12, then a new micro-cluster is created, with the new object as its center. All the corresponding objects are moved from PD to Imc. All objects still in PD that are less than 32 R apart update their Rmc lists with the identifier of the new micro-cluster. (3b-iii) Otherwise, the event-based algorithm described in the previous sections is applied. The objects in p.Rmc are the cluster identifiers for which the distance from their centers is not greater than 32 R.

**Step 4:** If the size of a micro-cluster shrinks below k + 1, then this micro-cluster is dissolved, and its former objects are treated in a way similar to that described in Step 3b. At the end of these steps, additional outliers are reported with the help of the event queue, which in MCOD does not include any object p 2 Imc. The main advantage compared to the algorithms in the previous sections is that the number of distance computations is reduced significantly.

**6. PERFORMANCE COMPARISONS**

The performance of outlier detection algorithms can be evaluated based on two error rates: detection rate and false alarm rate. Detection rate gives information about the number of correctly identified outliers, while the false alarm rate reports the number of outliers misclassified as normal data records. The detection rate and the false alarm are computed as follows: Detection rate = TP/TP + FN, False alarm rate =FP/FP + TN.

**6.1 CPU Time**

First, we study the performance of the methods for varying values of W in the range [10K; 1000K]. Depicts the results. For Slide = 1, the memory requirements for Abstract- C are very high. More specifically, Abstract-C stores W¢(W¡1) 2 counters, which corresponds to 74GB for W = 200K and 465GB for W = 500, assuming integers need 4 bytes.

Volume : 4 | Issue : 6 | Special Issue June 2015 • ISSN No 2277 - 8179

**Table 6.2 CPU Time vs different algorithms**

| Window Size (K) | STROM | LOF | SVDD | MCOD |
|---|---|---|---|---|
| 10 | 10 | 0.8 | 0.7 | 0.1 |
| 100 | 11 | 0.9 | 0.74 | 0.3 |
| 200 | 12.5 | 0.95 | 0.8 | 0.4 |
| 300 | 12.8 | 1.5 | 1 | 0.8 |
| 400 | 13 | 1.8 | 1.3 | 1 |
| 500 | 13.5 | 1.9 | 1.45 | 1.1 |

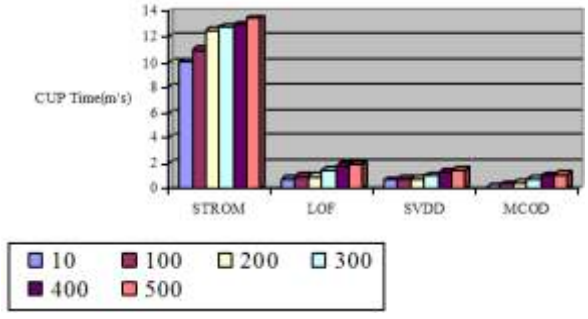


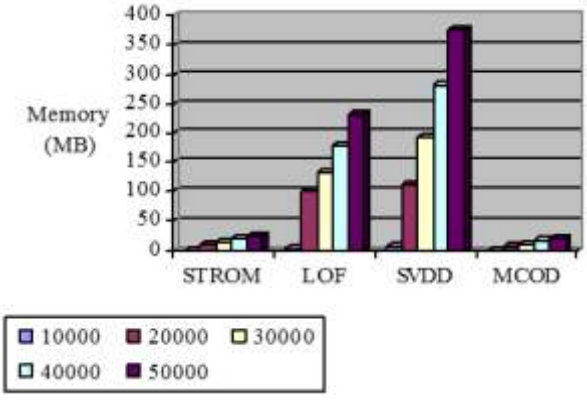

**Fig 6.1 CPU Time for outlier detection**

**6.2 Memory Consumption**

The memory consumption of the two real data sets for the experiment. The consumed memory corresponds to the memory needed to store the information for each active object the heap size used for the events prioritization, the outliers of all the queries and the micro-cluster information for SVDD. As can be seen, the required amount of memory is only a small fraction of the total memory available in modern machines, even for the MCOD method.

**Table 6.3 Memory usage vs different algorithm**

| W | STROM | LOF | SVDD | MCOD |
|---|---|---|---|---|
| 10,000 | 0.48 | 2.95 | 4.29 | 0.27 |
| 200,000 | 9.60 | 100.45 | 111.85 | 4.94 |
| 300,000 | 14.47 | 133.27 | 194.28 | 11.04 |
| 400,000 | 19.32 | 178.30 | 280.37 | 15.40 |
| 500,000 | 24.23 | 232.72 | 377.51 | 20.23 |




**REFERENCE**

[1] ] S. Ramaswamy, R. Rastogi, and K. Shim, "Efficient algorithms for mining outliers from large data sets," in SIGMOD Conference, 2000, pp. 427–438. | [2] E. Knorr, R. Ng, and V. Tucakov, "Distance-based outliers: algorithms and applications," The VLDB Journal, vol. 8, no. 3-4, pp. 237–253, 2000 | [3] D. Yang, E. Rundensteiner, and M. Ward, "Neighbor-based pattern detection for windows over streaming data," in EDBT, 2009, pp. 529–540 | [4] B. Liu, Y. Xiao "Svdd-based outlier detection on uncertain data" 2013. | [5] Y. Shi and L. Zhang "COID: A cluster-outlier iterative detection approach to multi-dimensional data analysis" 2011. | [6] F. Cao, M. Ester, W. Qian, and A. Zhou, "Density-based clustering over an evolving data stream with noise," in SDM, 2006. | [7] M. M. Breunig, H. P. Kriegel, R. T. Ng, and J. Sander, "LOF: Identifying density-based local outliers," in Proc. ACM SIGMOD Int. Conf. Manage. Data, New York, NY, USA, 2000, pp. 93–104.