3rd IFAC International Conference on Intelligent Control
and Automation Science.
September 2-4, 2013. Chengdu, China

An Anomaly Detection Approach Based on Isolation Forest Algorithm for Streaming Data using Sliding Window

Zhiguo Ding[*], Minrui Fei[**]

[*] *Shanghai Key Laboratory of Power Station Automation Technology, School of Mechatronics Engineering and Automation, Shanghai University, Shanghai, 200072 China (Tel:021-56331934; e-mail: dingzhiguo@shu.edu.cn)*. *College of Mathematics, Physics and Information Engineering, Zhejiang Normal University, Jinhua, Zhejiang, 321004, China*
[**] *Shanghai Key Laboratory of Power Station Automation Technology, School of Mechatronics Engineering and Automation, Shanghai University, Shanghai, 200072 China (Tel:021-56331934; e-mail: mrfei@staff.shu.edu.cn)*

Abstract: Anomalous behavior detection in many applications is becoming more and more important, such as computer security, sensor network and so on. However, the inherent characteristics of streaming data, such as generated quickly, data infinite, tremendous volume and the phenomenon of concept drift, imply that the anomaly detection in the streaming data is a challenge work. In this paper, using the frame of sliding windows and taking into account the concept drift phenomenon, a novel anomaly detection framework is presented and an adapted streaming data anomaly detection algorithm based on the *iForest* algorithm, namely *iForestASD* is proposed. The experiment results performed on four real-world datasets derived from the UCI repository demonstrate that the proposed algorithm can effective to detect anomalous instances for the streaming data.

Keywords: Anomaly Detection, Streaming Data, Concept Drift, Isolation Forest, Sliding Window

1. INTRODUCTION

Streaming data, where data flow in and out like streams, is a kind of novel and special data form, which usually are collected timely and acquired from some real applications, such as network security monitoring, telecommunication management, retail industry transaction, network traffic management, financial monitoring as well as wire/wireless sensors network. In those applications, monitoring inconsistent or unexpected data are playing an important role in order to assure the security and stability running of the target systems.

There are many researchers focus their attention on the anomaly detection of streaming data. In summary, anomaly detection in the streaming data has the following characteristics [Tan *et al*., 2011]. Firstly, streaming data usually generates quickly which implied that streaming data is infinite and volume tremendous, so any traditional off-line anomaly detection algorithm that attempt to store the whole dataset or to scan the dataset multiple times for anomaly detection will run out of memory space, which would be impossible and impractical to perform this work successfully. Secondly, because the anomaly behavior happened rarely, the streaming data usually contains mass normal data and rare anomalous points which made the anomaly detector training difficult and hard to learn the satisfied model for the significant unbalanced dataset. Thirdly, for the time-varying characteristic of the streaming data compared to the traditional ones, concept drift is a common phenomenon in the streaming data, which is an important consideration while dealing with anomaly phenomenon of streaming data [Zhou *et al*., 2009]. Once the concept drift occurred, it means that the previous training anomaly detection model may no longer consistent with the current concepts and unable to deal with such condition any more, if the trained detector can't update timely, the detection performance will be degraded. Consequently, in order to hold the detection effectiveness, re-training model or model updating based on the incoming new data set is necessary and urgently. Commonly, the anomaly detection model of streaming data is viewed as a generalization of the classical anomaly detection models when the data size grows infinitely. Consequently, Based on the classic anomaly detection method, namely *iForest* [Liu *et al*., 2008], which can deal with anomaly data points effective and efficient in the massive dataset and have lower computation complexity compared to other existing methods, an anomaly detection framework is proposed and an adapted algorithm, namely *iForestASD,* is present which is modified by the *iForest* algorithm and suited for anomaly detection for streaming data.

The main contributions of this paper as follows, firstly, we proposed an anomaly detection frame for the streaming data. Secondly, the adapted algorithm is proposed which is designed originally for the static dataset. Thirdly, the experiments were done to demonstrate the proposed algorithm available and effective.

The next of this paper is organized as follows: In section 2, the related work is described. In section3, the proposed method is demonstrated step by step, the general framework of anomaly detection for streaming data is demonstrated firstly. Secondly, the anomaly detection algorithm, *iForest*, is introduced and analyze; thirdly, the new adapted

algorithm, *iForestASD* , which can monitor the concept drift and update the un-proper trained model effectively, is presented. In section 4, the experiment datasets are firstly demonstrated and then experiment results are analyzed to validate our proposed method available. At the last, the conclusions and future research directions are provided.

2. RELATED WORK

Anomaly detection, or outlier detection, is an activity for finding the patterns in dataset which do not conform to the expected pattern or deviates from the expected behavior greatly. Anomaly detection is of great significance in many applications and there are many different methods available for outlier detection so far. In general, anomaly detection method can be categorized as distance-based method [Angiulli *et al*. 2009], density-based method *[Bruning et al*., 2008], modal-based method [He *et al*., 2003] and isolation-based method [Liu *et al*., 2008] so on. Those algorithms have their advantages in the specific field. Unfortunately, in the context of streaming data, those methods more or less have some drawbacks and are not directly applied to streaming data, such as poor adaptability and extensibility, inability to detection novel anomaly, high model updating cost and slow updating speed so on.

However, with the rapid development of information technology, the data is massive. Finding those unpredicted data points or pattern manually is not realistic corresponding to the more and more application in this field. For example, in order safeguarding the network security, we may detect network intrusions timely for the computer network based on the analyzing the great column monitoring data and finding the anomalous activity by the intentional vicious attacker [Kayitha *et al*., 2012][Palpanas *et al*.,2003]. We can monitor the sensor network and detect the occurring of the abnormal event [Subramaniam *et al*.,2006] [Davy *et al*., 2005]. As a result, some successful methods up to now have been proposed from the new perspective or adapted based on the traditional techniques and have been applied into some application scenes for anomaly detection of stream data. Based on the specific application, the employed techniques range from analysis of simple/individual time series to complex/multidimensional analysis of streaming data.

From the different perspective, some anomaly detection methods for the streaming data are present. Based on the semi-supervised technique, an anomaly detection algorithm is proposed for the characteristic of time-varying network traffic patterns [Yan *et al*., 2009], this algorithm uses data stream model to train detection model to solve the problem of change of traffic patterns. In order to detect the global and local anomaly simultaneous in time series data, two algorithms are proposed based on the distance-based anomaly detection techniques [Girish *et al*., 2005]. Considering the characteristic of limited memory and one-pass constraints for streaming data anomaly detection, another novel approach based on the distance metric technique is introduced to summarize the clusters with arbitrary shape for anomaly detection which is used to discovering clusters in an evolving data stream [Cao *et al*., 2006]. Considering the difficulty of obtaining the full dataset for the sensor networks or large online database due to the limitations, such as low bandwidth or memory, storage, or computing power, a framework for detecting anomaly from large-scale dataset is present, which investigated the information sampling technique and spectrum-based volume anomaly detection [Saha *et al*.,2009]. Besides above mentioned, there also have many successful methods proposed and some have been applied into the application scene for anomaly detection over stream data. Such as model-based [Yamanishi *et al*., 2006], clustering-based [Yamanishi *et al*., 2000][Burbeck *et a*l., 2007]and density-based methods [Pokrajac *et al*., 2007] [Palpanas *et al*.,2003][Subramaniam *et al*.,2006].

To detect the high-speed streaming data, one interesting research fruit is *Hoeffding trees*(HT), which is an incremental anytime decision tree induction algorithm [Hulten *et al*. 2001] and have been used for anomaly detection, but this algorithm require positive as well as negative class labels to be available for training, this precondition is no realistic because anomalous data is usually rare or not available for training , what's more, in the application, labeling the data also is an expensive job. Isolation Forest or *iForest* is another anomaly detection algorithm based on the assumption that the anomaly data points are always rare and far from the center of normal clusters[Liu *et al*.,2008], which is a new, efficient and effective anomaly detection technique based on the binary tree structures and building an ensemble of a series of *iTree*s for a given data set through random sampling. The main idea of isolation tree is to take advantages of anomalous instances being 'few and different'. In this paper, the main reasons of selecting the *iForest*-based anomaly detection compared to other existing algorithms describe follows. Firstly, building *iTree*s only need to select subset of the training set randomly, the research result show that the reasonable number of sub-samplings is set to 256, which is a relative small number and reduce the swamping and masking effects effectively. Secondly, *iForest* utilizes no distance or density measures to detect anomaly, this eliminates computational cost significantly compared to the distance-based methods and density-based methods. Thirdly, *iForest* has a linear time complexity with low constant and a low memory requirement. Last but not the least, *iForest* algorithm is based on the ensemble idea, even if the efficiency of some *iTree*s are not very high, the ensemble algorithm always can turn the weak algorithm into strong algorithm. For the significant merits of it, the *iForest* algorithm may be a promising algorithm and an appropriate selection for the anomaly detection in streaming datasets. In our paper, the important contribution is to adapt this algorithm to suit for the anomaly detection in the streaming data forms, the details can be described in section 3 step by step.

3. PROPOSED *iForestASD* ANOMALY DETECTION METHOD FOR STREAMING DATA

3.1 General Framework of Anomaly Detection for Streaming Data

Streaming data, just as its naming, flowing in and out like a stream and creating continuous and infinitely, is a growing

sequence of *N*-dimensional vectors [Pedro et al., 2010] and the general form of streaming data can be described as follows:

$$Z = \{z(1), z(2), \ldots, z(t), z(t+1), \ldots\}$$

where $z(t) \in R^N$ for $t \geq 1$.

At first, streaming training data set *Z* is divided into the data blocks having the same time interval and the same number of instances in a data block, which are named as window data block and can be described in detail as follows.

$Z=\{Z_1,Z_2,....Z_i,\ldots\}$ with the follow forms:

$$\begin{aligned} Z_1 &= z(1), z(2), \ldots, z(M) \\ Z_2 &= z(M+1), z(M+2), \ldots, z(2M) \\ &\vdots \\ Z_i &= z((i-1)M+1)), z((i-1)M+2)), \ldots, z(i*M) \end{aligned}$$

M denotes the size of the sliding window, Z_i the *i*-th streaming data block. The main task of anomaly detection for streaming data is to examine whether there exists the anomalous instances based on the anomaly detector. The general framework of anomaly detection proposed in this paper for streaming data is shown in Fig. 1.

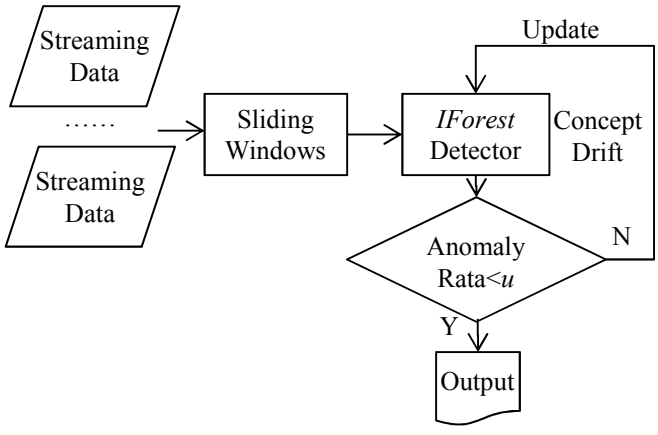

Fig. 1 General Framework of Anomaly Detection for Streaming Data

Firstly, when streaming data is coming, it will be feed into sliding window with the predefined size. The size of sliding window is our first challenge to consideration in the training stage of detector, which has not the specific theoretical guide and the parameter is selected by the prior knowledge and experiments. Based on the several windows data, an initial detector is trained based on the *iForest* algorithm. In the test stages, every instance in sliding window is examined by the anomaly detector to judge whether it is an anomaly point or not based on the anomalous score of instance. After the instances in one sliding windows are completed, the statistic result is acquired, if the anomaly rate in a sliding window data is less than the predefined threshold *u*, then the concept drift does not happen, the trained anomaly detector will not change. Otherwise, it implied that the concept shift is occurred and the trained anomaly detector need to modify and re-train, which will be updated and re-trained the detector based on all instances in current sliding window, so the previous detector is discard.

3.2 Building the initial anomaly detection model and prediction

For the streaming data set Z_i, the initial anomaly detection model is build, at first, we build the *iTree*s in terms of the bootstrap sampling from the Z_i . The ensemble detection model *E* is composed of *L* *iTree*s, namely, $E=\{E_1,E_2,\ldots E_L\}$,which is built from the data in *i*-th sliding window. The detail algorithms and evaluation of *iForest* can be seen in [Liu *et al*., 2008], here, we only describe the main part of *iForest* algorithm.

Algorithm 1: *iForest(X, L, N)*
Inputs: *X* - input dataset, *L* - number of trees, *N* – subsampling size in the sliding window
Output: a set of *L iTrees*
Step 1: **Initialize** *Forest*={ };
Step 2: set *iTree* height $h = ceiling(\log_2 N)$;
Step 3: **for** *i* = 1 **to** *L* **do**
X' ←*sample*(*X*,*N*);
Forest←Forest∪ *iTree*(*X'*, *0*, *h*);
Step 4: **return** *Forest;*

An *iForest* consists of multiple isolation trees, namely *iTree*, which are created by choosing attributes and the values of attributes randomly. At each node in the isolation trees, the instances set is divided into two parts based on the chosen attributes and its values. Here, the attributes is selected randomly and the split value for this selected attribute is selected randomly as well between the minimum value and maximum value of this selected attribute. Commonly, anomalous instances are those objects that their attributes values are very different from the normal instances and are easier to being divided than normal instances'. In the process of isolation, they are also closer to the root and more easily divided than the normal instances. In order to alleviate the effects imported by the random characteristic in the process of building the isolation forest, we calculate the average depth of the instance in the forest which is composed multiple isolation trees and use the average depth as the anomalous score of the instance. The lower score the instance has, the higher probability it an anomaly. In the above algorithm, the procedure of creating the *iTree* is a key step. The detail creating procedure of *iTree* algorithm can be seen in [Liu *et al*., 2008].

3.3 Detecting the concept drift and updating the un-proper trained model

Concept drift is a common phenomenon occurred in the process of streaming data, in order to detect the concept drift as soon as possible, the anomaly rate is calculated after the instances in one sliding windows are all been detected, we first create a ranking of all instances in a sliding window. Each instance has a score based on its likelihood of being an anomaly which is calculated as its average depth of *iTrees* in the isolate forest. The anomaly score *S(x, N)* is calculated by the formula (1).

$$S(x, N) = 2^{-\frac{E(h(x))}{c(N)}}$$
$$E(h(x)) = \frac{1}{L}\sum_{i=1}^{L} h_i(x) \quad (1)$$

N denotes the subsampling size. $h_i(x)$ denotes the length of the i-th *iTree*, $E(h(x))$ is the average of $h(x)$ from a collection of *iTrees*, and c(N) is the average of h(x) given N.

The statistic result is acquired about the anomaly rate in this sliding window. Assuming that a rough estimation of the anomaly rate in the dataset based on the prior knowledge can be acquired, when the anomaly rate in the current sliding window is not less than the predefined threshold u, the concept drift happens in the steaming data, that is to say, the previous anomaly detector is not proper any more. The current detector will be discarded immediately and the new detector be re-trained simultaneous. The detail procedure can be shown below.

Algorithm 2 : *iForestASD (Z, L, N, E, u)*
Inputs: Z - input data, $Z=\{Z_1,Z_2,....Z_i,\ldots\}$, L - number of trees, N – subsampling size in the sliding window,E - current anomaly detector; u: the predefined anomaly rate;
Output: E – new anomaly detector;
Step 1: E=*iForest*(Z_i,, L, N);
Step **2**: *Score*=$E(Z_{i+1})$;
Step 3: *Ranking*(*Score*);
if *Anomaly Rate(Ranking)* ≥ u
then i=i+1;**goto** step 1;
else i=i+1; **goto** step 2;
Step 4: **return** E;

Consideration of the concept drift, the fixed sliding widow size is not appropriate. If the sliding window size is too large, the trained model will not accurately represent the concept drift, on the other hand, if the sliding window size is too small, then there will not be enough data to construct an accurate model. Unfortunately, there is not appropriate method to set this parameter, the common method carried in the literature is by the trial and error for studying in the specific application. What's more, the threshold u is an important parameter, in this paper, for the special streaming data, prior knowledge about anomaly rate is consideration.

4. EXPERIMENT STUDIES

We have conducted extensive experiments with four real-world data sets coming from the UCI Machine Learning Repository to evaluate the performance of *iForest*-based anomaly detection algorithm for streaming data. We only select those datasets which the number of instances in datasets is more than 10000 by the reason of the characteristic of massive data for streaming data. Because *iForest* algorithm is an un-supervised method and does not need the category attributes during the procedure of anomaly detection, the anomalies' labels are only used in evaluating the final anomaly detection performance.

The experiments are conducted on a personal PC with Intel® Core™ 2 Duo CPU, P7450@2.13GHZ and 4GB memory. The operating system is Windows 7 professional. The algorithms described in section 3 are programmed by the C++ language with the Visual C++ software platform and the dataset pre-processing as well as result analysis are used the Matlab2010 platform.

4.1 Datasets

Detail information of four datasets is shown in Table 1. These Datasets are selected because they contain known anomaly classes as ground truth and these data sets are used in the literatures to evaluate anomaly detectors in the streaming data many times [Tan *et al*., 2011][Cao *et al*. 2006][Yamanishi *et al.*, 2000].

The **Http** dataset and the **Smtp** dataset are two subsets of the KDD-CUP99 network intrusion data. According the data pre-processing method mentioned in [Yamanishi *et al*., 2000], only four of the original 41 attributes (service, duration, src-bytes, dst-bytes) are selected because these four attributes were thought of as the most basic and important attributes. **ForestCover**, this dataset consists of 581012 instances originally and the number of attributes is 54 and a label totally. In our experiment, we select instances labeled 2 and 4 and 10 attributes. The **Shuttle** dataset contains 9 attributes. Approximately 80% of the data belongs to class 1 and we regard as the label 2,3,5,6,7 as the anomaly class. In each datasets, the label attribute are selected only use it to evaluate the proposed algorithm.

Because *iForest* algorithm only handles the numeric values of attribute, all nominal and binary attributes are removed, the attributes selected in our experiments are all continuous value. What's more, for simulating the streaming data, a reasonable assumption is constructed that taking the data input order as the order of stream for building the streaming datasets.

Table 1 Information of Four Datasets

Dataset Name	M(#instances)	N(#attributes)	Anomaly Threshold
Http	567498	3	0.39%
Smtp	95156	3	0.03%
ForestCover	286048	10	0.96%
Shuttle	49097	9	7.15%

4.2 Experiment settings and result analysis

After the streaming dataset is constructed, the first task is to train an anomaly detector based on the *iForestASD* algorithm under the framework of sliding window for streaming data.

Firstly, the initial anomaly detection model is built for the streaming dataset Z_i. We build the *iTree*s in terms of the bootstrap sampling from the Z_i. There are two important parameters to this *iForestASD* anomaly detection method, namely, the size of the sliding widow and the ensemble scale. For the former, because the size of sliding window is time-sensitive, there is not theoretic direction for the setting of parameter. In our experiment, the parameter value is described as in Fig. 2, The value is set from 32 to 4096 and increased exponentially.

For the latter, the number of *iTree*s including in the *iForest*

ensemble actually also is no theoretic direction for this value. Zhou et al. find that path lengths usually converge well before t=100. As a result, we also use t=100 as the default value in our experiment. The anomaly threshold u demonstrated in Table 1 is a prior knowledge derived from the history statistic result. If the anomaly rate is bigger than u, which implies that the concept drift phenomenon is occurring and need to update the detector.

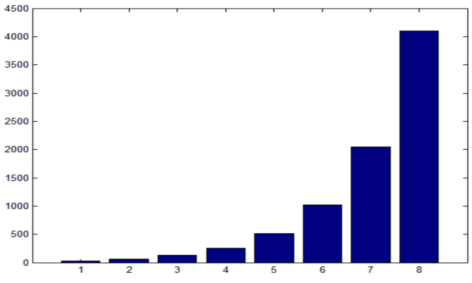

Fig. 2 The size of sliding widow (y-axis) versus the i-th experiment (x-axis)

In anomaly detection, AUC (Area under ROC Curve) [Tan *et al*., 2011] [Liu *et al*.,2008] is usually used to measure the overall performance of anomaly detector regardless of the threshold between the true positive and true negative. Anomalies are always treated as the positive class and normal points as negative class in the process of anomaly detection. The calculation formula is described as formula (2). n_a denotes number of true anomalies, n_b the number of true normal points. r_i is the rank of the i-th anomaly in the ranked list which descended according the anomaly scores.

$$S = \sum_{i=1}^{n_a} r_i$$
$$AUC = \frac{S - (n_a^2 + n_a)/2}{n_a * n_n} \quad (2)$$

In order to evaluate the performance of our proposed algorithm, we evaluated the detection performance of our proposed *iForestASD* anomaly detection algorithm for four datasets by the AUC. The result can be seen in Table 2 and Fig. 3.

Table 2 AUC of different Datasets for different sliding windows size

Dataset / Sliding Window size	Http	Smtp	ForestCover	Shuttle
32	0.53	0.51	0.51	0.86
64	0.89	0.78	0.57	0.89
128	0.93	0.74	0.63	0.95
256	0.95	0.76	0.63	0.96
512	0.94	0.83	0.81	0.98
1024	0.94	0.86	0.83	0.97
2048	0.95	0.85	0.84	0.98
4096	0.94	0.86	0.82	0.97

From the experiments result showed in Fig. 3(The value of x-axis is in log scale), we can easily find that when the size of sliding window increases, the value of AUC is increasing correspondingly. When the size of sliding window increase to a desired value, such as 256 for Http dataset, 1024 for Smtp dataset, 512 for Forest Cover dataset and Shuttle dataset, the *iForestASD* detection performance reaches reliably, that is to say, there is no need to increase it further because it increases processing time and memory size without any gain in detection performance. But an important fact that can't be ignored is the detection performances are all relatively low for four datasets compared to the results in [Tan *et al*., 2011][Liu *et al*.,2008], the fixed sizes of sliding windows which are not appropriate for the real application maybe the main reason, which is also a topic to be study.

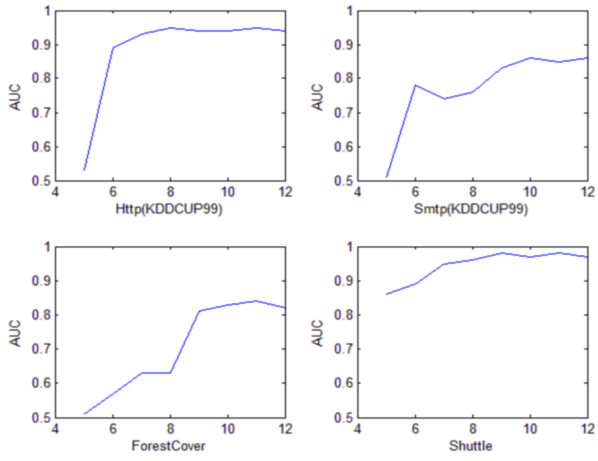

Fig. 3 AUC performance (y-axis) versus the different sliding window size in log scale (x-axis)

5. CONCLUSION AND FUTURE WORK

In this paper, considering the characteristics of streaming data, an *iForest*-based anomaly detection framework is firstly proposed under the sliding windows framework and a new streaming data anomaly detection algorithm, namely *iForestASD*, is proposed. The experiments results on four real world data sets demonstrate that proposed method is efficient. However, in our study, we only validate our proposed method suitable for the streaming data anomaly detection, but not compare it with the existing methods for the limitation of page. What's more, there are some problems appeared and need to explore further.

Our future works will focus on following three directions: Firstly, the predefined threshold u is needed to demonstrate the anomaly rate, but in the real application, this value may be not acquired accurately ahead, how to detection the occurring of concept drift in the streaming data still is a difficult work; Secondly, the size of sliding windows is fixed in this paper, which maybe sometimes is un-appropriate for the real application, especially when concept drift is occurred. The fixed size of sliding widows may not reflect the real streaming data characteristic and degrade the performance of anomaly detection. The next study will set the size of sliding windows automatically or self-adaptive for the real-world application. Thirdly, in our study, when the concept drift is occurred, our method is to re-train the detection model under the current datasets and discard the

previous ones completely. Because *iForestASD* is an ensemble method and the trained detection model was consist of multiple individual detection models, considering the consecutive characteristic of streaming data, when the detection model need to update, we may selectively discard the old ones and simultaneous adding the new ones, this maybe a good idea for the performance improvement of anomaly detection.

ACKNOWLEDGEMENTS

This work is supported by the National Natural Science Foundation of China under Grant No. 61074032, the Project of Science and Technology Commission of Shanghai Municipality under Grant No. 10JC1405000, the Key Project of Economic and Information Technology Commission of Shanghai Municipality under Grant No.11XI-31 and the Zhejiang Provincial Natural Science Foundation of China under Grant No. LY13F020015.

REFERANCES

Angiulli, F., Fassetti, F. Dolphin, (2009). An efficient algorithm for mining distance-based outlier in very large datasets. *ACM-TKDD* 3(1).

Bruning, M.M., Kriegel, H.P., Ng, R.T., Sander, J. (2008). LOF: Identifying density-based local outliers. *In: Proc. SIGMOD*.

Cao F, Ester M, Qian W, Zhou A. (2006). Density-based clustering over an evolving data stream with noise. *In Proceedings of the 2006 SIAM conference on data mining (SDM)*, Bethesda, MD, 328–339.

D. Pokrajac, A. Lazarevic, etc. (2007). Incremental local outlier detection for data streams. *In: Proceedings of IEEE Symposium on Computational Intelligence and Data Mining(CIDM)*. Honolulu: IEEE Computer Society Press, 504-515.

Girish Keshav Palshikar. (2005). Distance-Based Outliers in Sequences. *Lecture Notes in Computer Science Volume* 3816, 547-552.

G. Hulten, L. Spencer, and P. Domingos. (2001). Mining time changing data streams. *In Proceedings of the 7th ACM SIGKDD.*

He, Z., Xu, X., Deng, S. , (2003). Discovering cluster-based local outliers. *Pattern Recognition Letters* 24(9-10), 1641-1650.

Kavitha, C. (2012). Massive stream data processing to attain anomaly Intrusion Prevention Devices, Circuits and Systems, *In 2012 ICDCS*, 15-16 March, 572 -575.

K. Yamanishi and J. Takeuchi. (2006). A unifying framework for detecting outliers and change points from non-stationary time series data. *Knowledge and Data Enginnering*, 18(4): 482-492.

K. Yamanishi, J.I. Takeuchi, G.Williams, and P.Milne. (2000). On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms. *In Proceeding of the sixth ACM SIGKDD International conference on Knowledge discovery and data mining.* 320-324.

K. Burbeck and S. Nadjm-Tehrani. (2007). Anomaly Detection With fast Incremental Clustering. *Information Security Tech. Report*. 12(1): 56-67

Liu F.T., Ting K.M., and Zhou Z. H. (2008). Isolation forests. *In Proceedings of International Conference on Data Mining*, 413–422.

M. Davy, F. Desobry, A. Gretton, and C. Doncarli. (2005). An online support vector machine for abnormal events detection. Signal Processing, 86:2009-2025.

Pedro Henriques dos Santos Teixeira, Ruy Luiz Milidiú, (2010). Data stream anomaly detection through principal subspace tracking, *In Proceedings of the 2010 ACM Symposium on Applied Computing*, March 22-26. Sierre, Switzerland [doi>10.1145/1774088.1774434].

Saha Budhaditya, Duc-Son Pham, Mihai Lazarescu, Svetha Venkatesh. (2009). Effective Anomaly Detection in Sensor Networks Data Streams, I*n Proceedings of the 2009 Ninth IEEE International Conference on Data Mining*, December 06-09, 722-727.

S. Subramaniam, T. Palpanas, etc. (2006). Online outlier detection in sensor data using non-parametric models. *In:Proceedings of the 32nd international conference on Very large data bases (VLDB)*. Seoul : VLDB Endowment Press, 187-198.

T. Palpanas, D. Papadopoulos, etc. (2003). Distributed deviation detection in sensor networks. *ACM SPECIAL ISSUE: Special section on sensor network technology and sensor data management*, 32(4):77-82.

Tan S. C, Ting K. M., Liu F.T. (2011). Fast Anomaly Detection for Streaming Data [C]. *In Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*, 1511-1516.

Yan Yu, Shanqing Guo, Shaohua Lan, Tao Ban, (2009). Anomaly Intrusion Detection for Evolving Data Stream Based on Semi-supervised Learning ,*Advances in Neuro-Information Processing,Lecture Notes in Computer Science Volume* 5506, 571-578.

Zhou J.L., Fu Y., Wu Y. and Other. (2009). Anomaly detection over concept drifting data streams. *Journal of Computational Information Systems* v. 5(6), 1697-1703.