Anomaly Detection in Data Streams Based on Graph Coloring Density Coefficients

Achyut Mani Tripathi
Department of Computer Science
& Engineering
Indian Institute of Technology Guwahati
Guwahati-781039, Assam, India
Email: t.achyut@iitg.ernet.in

Rashmi Dutta Baruah
Department of Computer Science
& Engineering
Indian Institute of Technology Guwahati
Guwahati-781039, Assam, India
Email: r.duttabaruah@iitg.ernet.in

***Abstract*—Anomaly or outlier detection is a fundamental task of data mining and widely used in various application domains. The main aim of anomaly detection is to identify all the data points with significant deviation from other normal data points. Mining the outliers become more challenging in environments where data is received at extreme pace. Such environments demand detection of outliers on-the-fly mode. The existing anomaly detection methods focus more towards the identification of point anomalies, very few of them explore the problem of collective anomaly detection over data streams. In this paper, we present a mutual density based anomaly detection algorithm, which can identify collective anomalies in data streams. The concept of mutual density is adopted from the area of graph theory.**

I. Introduction

Recent advancements in hardware and software technology encourage the use of monitoring applications in numerous fields. Various applications use sensor-based monitoring architecture, which generates continuous data streams. These continuous data streams are rich with information and can be used to determine various unseen patterns that describe all possible behaviours of the monitored system. Mining information from such data streams that arrive at extreme pace is a challenging task of data mining due to continuous change in underlying data distribution. Some of the major challenges associated with mining task of data streams are quick response time, less memory overhead, low false alarm rate. Further, previously detected anomalies in a data stream may turn to normal on the arrival of new data.

A monitoring application fundamentally requires detecting flaws commonly known as anomalies, abnormal patterns, outliers in the data. Anomaly detection is adopted by various application domain as major component like detection of suspicious activities in video frames for video surveillance applications [1], detection of cyber attack requires continuous monitoring of high dimensional data streams [2], condition of equipments and structure damage in industries can be monitored by analyzing the pattern of data streams received through sensors [3][4], security applications using speech for authentication use anomaly detection for unauthorized speaker identification [5], military equipments like unattended ground sensors (UGS) are used for continuous monitoring of activities across the border areas. Monitoring through these equipments follow the mechanism of extraction of abnormal patterns from data streams generated by UGS sensors [6], health care monitoring application perform the monitoring task by measuring the change in different data streams received by wearable sensors [7], other fields that are using anomaly detection are robot navigation [8], text mining [9], stock market prediction [10], ecological disturbance [11] etc.

There are mainly three types of anomalies [12]: Point anomaly, Collective anomaly, and Contextual anomaly. Point anomaly is an indidual data instance showing significant deviation with respect to the rest of the data whereas collective anomaly is a collection of data instances that are together anomalous with respect to the entire data set. Data instances that are anomalous within a specific context constitutes contextual anomalies. This paper focusses on detection of collective anomalies over data streams. The proposed method is window-based and relies on graphical model. The key contribution of this paper is adaptation of vertex coloring concept of graph theory as a feature for each node of graph for detecting the collective anomalies in data streams.

The remainder of the paper is organized as follows: related work is discussed in section II, section III describes the proposed methodology used for anomaly detection in data streams, section IV presents the experiments and results, and finally section V gives the conclusion and future work.

II. Related Work

Anomaly detection methods can be subdivided into two broad categories: statistical methods and machine-learning based methods. Earlier work on outlier detection was performed by statistical community, extreme value based outlier analysis [13], projecting outliers on the surface of convex hull [14] but later these methods became less popular as they assume that the data is generated by a particular data distribution. Techniques based on machine learning approaches for identification of outliers can be further divided into two sub categories supervised learning techniques and unsupervised techniques. Supervised techniques generate models based on available labeled training data. The data label associated with the data instance represents the behavior of data as normal or outlier. Some of the widely used supervised anomaly

detection techniques include principle component analysis (PCA)[15], fuzzy system for anomalies [16], neural network and regression based anomaly detection [17]. These techniques are primarily for point anomaly detection. The main issue with supervised techniques is of obtaining the labels for the data. For such techniques, labeling of data is usually performed manually by experts which is expensive, time consuming, and tedious. Further obtaining the training data set which contains all possible labeled anomalous data patterns is more difficult for real time applications than getting label for normal behavior. Semi-supervised anomaly detection methods have been proposed as these methods requires labelled data only for normal data instances [18],[19],[20]. These techniques are useful when it is difficult to obtain the data for all possible anomalous behavior.

Unsupervised anomaly detection methods do not require labeled training data for anomaly detection. Most of the proposed unsupervised methods are distance-based or density-based methods. One of the early works based on distance measure, for large datasets, is presented in [21]. The anomaly detection mehtods for data streams proposed in [22] and [23] are also distance-based methods. All distance based-techniques require various user defined parameters that highly influence the performance of such methods. Breuing et.al [24] proposed density-based approach, they introduced local outlier factor (LOF) to represent the degree of outlierness of each data point of data set. In [25] variation of LOF is proposed where correlation integrals (LOCI) is used to represent outlierness of each data point. Other variations of LOF are, representation of outlierness in terms of local outlier probability (LoOP)[26]. In[27] cluster-based technique is used for clustering the data before identifying the LOF of each data point for the detection of outliers (CBLOF). Aggarwal and YU[28] proposed outlier detection technique for high dimensional data by assuming a point is outlier, if it belongs to lower density region, when data in projected to lower dimension. Clustering-based outlier detection methods have been proposed in [29] and [30]. The data stream clustering methods do not detect outliers, they represent outliers as a bi-product of clustering algorithm. Further, these methods focus on detection of point anomalies. In [31], the authors proposed a window-based collective and contextual anomaly detection method over multiple data streams that requires user to define anomaly threshold.

All data points cannot be always considered as independent points lying on multi dimensional space, their exists inter dependencies in between points that can be used to define their abnormal behavior. Graph is one of the powerful mechanism that effectively denotes the inter dependencies between data points. Various graph based anomaly detection techniques have been proposed to identify the anomalies. Graph-based anomaly detection techniques basically address the anomalies by inspecting the change in structural or associated attribute features of each edge or vertex of a graph. Proximity measurement is one of the most used feature to define the closeness of nodes in a graph by exploiting the graph structure. The well-known page rank algorithm defines proximity measurement based on random walk between nodes of graph [32]. The Community-based method is similar to clustering method in which densely connected nodes are grouped together to form communities. In [33] bipartite graph has been used to exploit the community feature of graph. Probabilistic graph based method is proposed in in [34] for outlier detection. In [35],[36] neighborhood nodes are ranked to detect the anomalies.

Time series data can be represented as time variant dynamic graph and change in the features of nodes can be used to define the outlier data in time series. The main idea is to extract good summary from each snapshot of input graph and after finding the measurement of similarity or dissimilarity anomalous time stamp are identified [37]. Common features that define the substructure of a graph are diameter of a graph, shortest path length, degree of vertex, edge weight, clustering coefficients. However, calculation of some of these features like shortest path length, cluster coefficients etc. will add additional complexity to the algorithm. In [38] time series data is represented as a time variant undirected k-nearest neighbor graph that changes with respect to time. In this paper, we use the concept of vertex coloring as a feature to define the closeness of nodes in time variant graph. To the best of our knowledge, it has not been applied so far as a feature in time variant graphs.

Most of the above mentioned methods are capable of identifying outliers in a static or a dynamic environment by assuming that outliers are very rare as compared to normal data. However, they are unable to discover collective anomalies present in data streams.

III. METHODOLOGY

In this section, we first describe the terms that have been used to develop the anomaly detection algorithm.

k-nearest neighbor set (KNN set): k-nearest neighbor graph represents the relationship between data and its all k nearest neighbors. KNN graph is widely used to represent the relationship between data points because KNN graph effectively captures and represents the local properties of data. Before constructing KNN graph of the data set, it is required to identify the k-nearest neighbors set of each data point. k-nearest neighbors set of any data point p_i can be defined by

$$KNN_{set}(p_i) = \left\{ p_j \mid\mid \| p_j - p_i \|_2 \leq d_i^k \right\} \tag{1}$$

where $\| p_i - p_j \|_2$ is euclidean distance between p_i and p_j, d_i^k is distance of p_i from its k^{th} nearest data point. The next step is to construct KNN graph based on KNN nearest neighbor set of data points obtained from the equation (1).

k-nearest neighbors graph (KNN graph): A k-nearest neighbors graph of N data points is obtained as follows. An edge E_{ij} is drawn in between p_i and p_j:

$$E_{ij} = \begin{cases} 1 & p_i \epsilon KNN_{set}(p_j) \ or \ p_j \epsilon KNN_{set}(p_i) \\ 0 & otherwise \end{cases} \tag{2}$$

The resulting KNN graph is undirected and free from self loop

Mutual nearest neighbors set (MNN set): Mutual nearest neighbors set efficiently defines the local information of neighborhood data points [39] [40]. Two points p_i and p_j are mutual nearest neighbors of each other if both are present in their KNN_{set} at the same time.

$$\mu(p_i) = (p_j \mid p_j \epsilon KNN_{set}(p_i) \; and \; p_i \epsilon KNN_{set}(p_j)) \quad (3)$$

Vertex coloring of graph: Vertex of a graph is colored such that no two adjacent vertices will be given same color. The minimum number of color used for coloring is known as chromatic number of graph[41]. Here we used greedy algorithm to perform coloring of KNN graph.

For example, consider the set of 10 data points. A = (69.88, 71.22, 70.87, 68.95, 69.28, 70.06, 69.27, 69.36, 69.16, 68.98). Table I represents the 3 nearest neighbors of all the data point of set A. Table II shows the mutual nearest neighbors set of data points of set A. Fig.1 represents the KNN graph constructed by using equation (1) and (2) and each vertex is colored such that no two adjacent vertices are having same color. Color of each node is represented by a number, each different number denotes one color.

TABLE I: Nearest Neighbors of Set A, for k=3

Index	Data Point	Index of k^{th} Nearest Neighbour		
		k=1	k=2	k=3
1	69.88	6	8	5
2	71.22	3	6	1
3	70.87	2	6	1
4	68.95	10	9	7
5	69.28	7	8	9
6	70.06	1	8	5
7	69.27	5	8	9
8	69.36	5	7	9
9	69.16	7	5	10
10	68.98	4	9	7

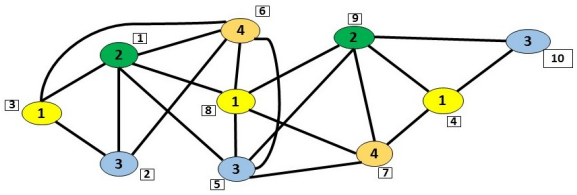


Fig. 1: KNN graph and vertex coloring of KNN graph

Mutual neighborhood and vertex coloring based density coefficients (VCC): To identify the data points belonging to local dense region of data set, we are proposing density coefficients based on mutual nearest neighbor set and vertex color of each node. Here we define a new density coefficient, which describes the local density of each data point. Proposed density coefficient not only identifies the connectivity of data points to its mutual neighbors, but also finds the connectivity of mutual neighbors to themselves.

Index	Index of Mutual Nearest Neighbor Set
1	(6)
2	(3)
3	(2)
4	(10)
5	(7, 8, 9)
6	(1)
7	(5, 8, 9)
8	(5,7)
9	(7, 5, 10)
10	(4, 9)

TABLE II: Mutual nearest neighbor set of A

Density of each vertex of a graph can be calculated by following rules:

$$VCC(p_i) = C_1^{ij}(p_i) + C_2^{jk}(p_i), (p_j, p_k) \epsilon \mu(p_i), k < j \quad (4)$$

$$C_1^{ij}(p_i) = \begin{cases} 1 & [color(p_i) \neq color(p_j)]_{\forall p_j \epsilon \mu(p_i)} \\ 0 & otherwise \end{cases} \quad (5)$$

$$C_2^{jk}(p_i) = \begin{cases} 1 & [(color(p_j) \neq (color(p_k)) \wedge (p_j \epsilon \mu(p_k)) \\ & \wedge (p_k \epsilon \mu(p_j))]_{\forall (p_j, p_k \epsilon \mu(p_i))} \\ 0 & otherwise \end{cases} \quad (6)$$

C_1^{ij} is vertex coloring density coefficient for all j mutual neighbors of p_i node of graph, C_2^{jk} is vertex coloring density coefficient in between all j and k mutual neighbors in $\mu(p_i)$. VCC is vertex coloring density coefficient of node w.r.t. all its mutual neighbors. Table III denotes the VCC coefficients for all data points of set A. Fig. 2 and Fig.3 show data points of set A and corresponding VCC coefficients of each data point respectively. From Fig. 2 and Fig.3 it is clear that data points outside the dense region are having low value of VCC coefficients.

TABLE III: VCC coefficients for A

Index	C_1	C_2	VCC
1	1	0	1
2	1	0	1
3	1	0	1
4	1	0	1
5	3	2	5
6	1	0	1
7	3	2	5
8	2	1	3
9	3	1	4
10	2	0	2

Anomaly Detection Strategy: User is more interested to inspect the recent data, for this purpose sliding window mechanism is used. Basic strategy for mining outliers from data streams is done by detailed inspection of recent data against summarized old data. Here we are using sliding

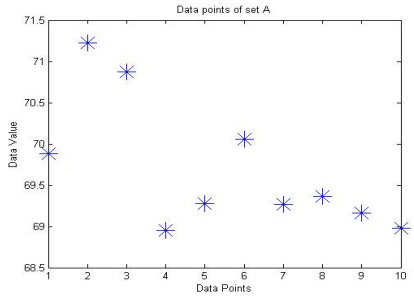


Fig. 2: Data points of set A

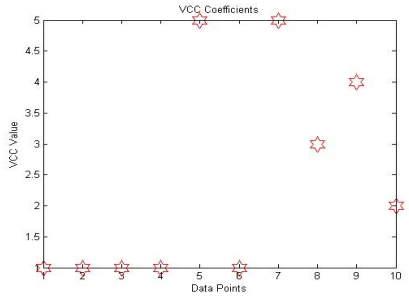


Fig. 3: VCC coefficients of set A

window mechanism to analyze the variation in VCC value of summarized data on the arrival of new data. Based on the variation on VCC value of data, anomalies are identified. Anomaly detection is a two-step process: VCC coefficients calculation and VCC change detection.

VCC computation module:

1. Windowing is performed over the time series data.
2. All the data points in a window are considered as nodes of a graph and KNN graph is constructed using equation (1) and (2).
3. MNN set of nodes is determined by using equation (3).
4. Vertex coloring is performed over the graph using greedy algorithm [41].
5. VCC coefficients of each node is calculated using equation (4),(5), and (6).
6. The Output of VCC computation module consists of first-M smaller VCC coefficients and associated data points. Fig. 4 represent the steps of this module.

VCC change detection module:

1. VCC coefficients are calculated for previous window using VCC computation module.
2. Output of VCC computation module will produce first-M smaller VCC values and associated data points.
3. Now, these data points with smaller VCC values from previous window (step 2) are merged with the data of the current window and again new VCC values are calculated for the same data points.
4. If less than 50% (say, τ)of M data points VCC value changes then the collection of M data points is declared as anomaly, otherwise change in data distribution is triggered.

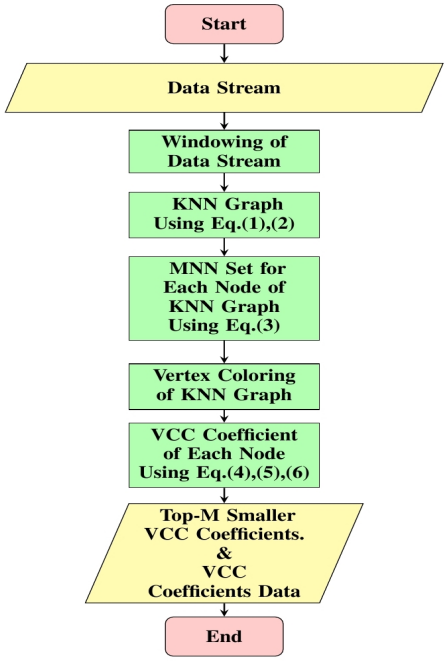


Fig. 4: VCC computation module

5. Steps 1 to 4 are repeated for all windows to detect the collective anomalies present in the data stream.

Fig.5 represents the steps to detect change in VCC value.

IV. EXPERIMENTAL RESULTS

To test the proposed method we used Numenta anomaly benchmark labeled data set [42]. The data set consists of 58 time series with both point and collective anomalies. For our experiments, we selected 10 time series with collective anomalies. The experiments were conducted using MATLAB 2013a, on a system with AMD Phenom(tm), X3710 processor, 4GB RAM and running Windows 8.1 operating system.

We compared the results obtained by our method with INS factor algorithm [43] that is a window-based collective anomaly detection approach.Instability factor analysis method is inspired from the concept of center of gravity, each outlier data point shifts the center of mass by large margin as compare to normal data points. This can identify all local and global anomalies exists in a data set. Window based INS factor algorithm identifies all collective anomalies present in a given data set. The results presented here are initial results only and we are in the process of implementing other existing graph-based anomaly detection algorithms for a detailed comparative analysis. Table IV gives the name of the time series from the data set that are used for testing purpose, and also presents the true and detected starting and ending index (in terms of samples)of anomalies in each time series for our approach as well as for INS factor algorithm. Fig.8 shows the collective anomalies detected by proposed

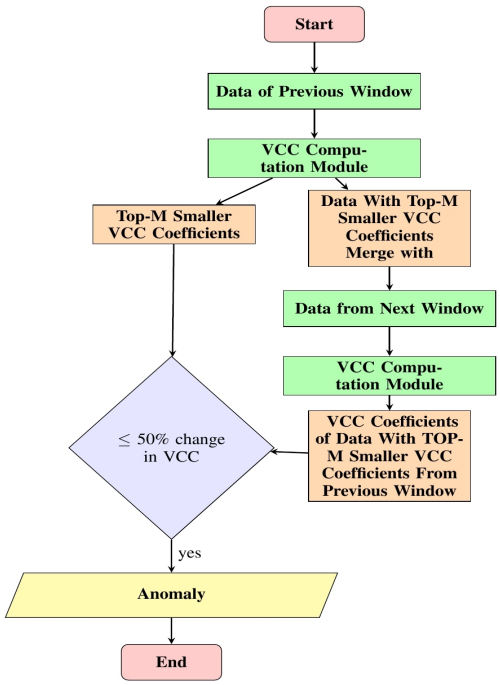


Fig. 5: VCC change detection module

algorithm (denoted by blue dotted windows) and by INS factor algorithm (denoted by green windows). Fig.6 And Fig 7 the start index and end index of true collective anomaly window, anomalies identified by the proposed method, and anomalies identified by INS algorithm. It is apparent from Table IV and figures 6,7 that the start index and end index of the collective anomaly window given by the proposed approach is closer to the true start index and end index. Whereas, INS factor algorithm captures the normal data as anomalies.

The results obtained from the proposed approach are sensitive towards the value of parameter (k). Increase or decrease in k will influence the results and sometimes anomalies are not detected by the proposed algorithm. We performed the sensitivity analysis for the parameter k and observed that it affects the percentage of number of points that change their VCC values (τ). The value of k is varied from 6 to 14. When k=6, out of M data points around 60% - 100% data points VCC value changes, similarly for k=7 the range is 60% - 95%. Table V summarizes the change in τ with respect to value of k. As anomaly is detected only when τ is less than or equal to 50%, when the value of k is set to 6, 7, and 8, no anomaly will be flagged. Same behavior is observed for remaining 9 time series and found that when k is in the range 9 to 11 the algorithm detects the collective anomalies in all the time series. Fig. 9 represents the detected anomalies in Art_daily_flatmiddle time series when the value of k is varied

S.No.	Time Series	True Label (Sample No.)		Detected Label (VCC coff)		Detected Label (INS Factor)	
		start	End	Start	End	Start	End
1	Art_daily_flatmiddle	2681	3191	2620	3200	2500	3100
2	Art_daily_jumpsdown	2789	3191	2646	3220	2552	3300
3	Art_daily_jumpsup	2789	3191	2700	3220	2600	3111
4	Art_daily_nojumps	2789	3191	2689	3200	2661	3250
5	Ec2_cpu_utilization_5f5s33	2932	4000	2800	3500	2700	3450
6	Ec2_cpu_utilization_825cc2	1528	1890	1500	1920	1491	1800
7	Grok_asg_anomaly	3678	4500	3600	4080	3554	3920
8	Lio_us_east_ia2ebled9_Network	208	372	200	380	180	420
9	Rdu_cpu_utilization_cc0c53	3082	4000	2950	3500	2663	3300
10	Rdu_cpu_utilization_ea7h3b	2587	3581	2559	3411	2500	3450

TABLE IV: True anomaly and detected anomaly labels

from 6 to 14. For k =9, the detected range is (2600-3250) Fig. 9 (a), for k=10 the range is (2620-3200) Fig. 9(b) and for k=11 range is (2800-3400) Fig. 9(c). However, for $k \leq 8$ and $k \geq 12$ no anomalies are detected as shown in Fig. 9(d).

The time complexity of the proposed algorithm is $O(n^2)$, where n is the total number of data points buffered in one window. The time taken by KNN graph construction step is $O(kd)$, where k is the number of nearest neighbors and d is the dimension of data stream. Further, the time complexity for MNN set computation is $O(kn^2)$ and $O(n^2)$ for graph coloring step. During the experiments, the algorithm took 0.78 seconds on average with average window size of 200 samples.

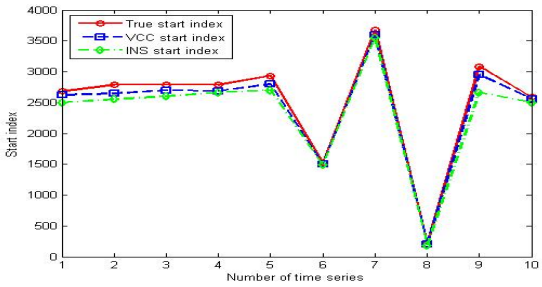


Fig. 6: Detected start index

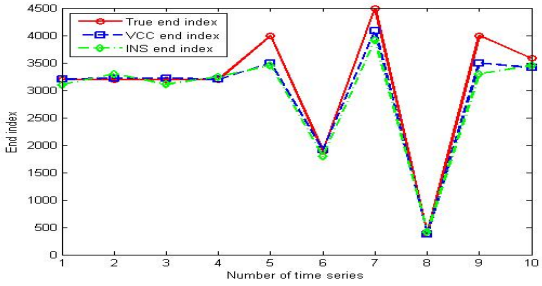


Fig. 7: Detected end index

V. CONCLUSIONS AND FUTURE WORK

In this paper, an unsupervised anomaly detection method for data streams based on graph coloring density coefficients is proposed and evaluated. The proposed method introduces the chromatic number of graph as one of the features to represent the density of data points. The method is capable of identifying the collective anomalies present in a data stream, and it also identifies the change in the data distribution within the data

S.No	Value of k	Range of vote %
1	k=6	60-100
2	k=7	60-95
3	k=8	55-95
4	k=9	40-95
5	k=10	45-90
6	k=11	50-95
7	k=12	60-100
8	k=13	60-100
9	k=14	70-100

TABLE V: Change in vote % with k for Art_daily_flatmiddle time series

stream with respect to time. However, the results obtained from the proposed method depend on various parameters like window size, k nearest neighbors. In future our focus will be to extend the work such that anomaly detection phase becomes self-adaptive in nature.

REFERENCES

[1] C. E. Au, S. Skaff, and J. J. Clark, "Anomaly detection for video surveillance applications," in *18th International Conference on Pattern Recognition (ICPR'06)*, vol. 4. IEEE, 2006, pp. 888–891.

[2] M. Roesch *et al.*, "Snort: Lightweight intrusion detection for networks." in *LISA*, vol. 99, no. 1, 1999, pp. 229–238.

[3] D. Dasgupta and S. Forrest, "Artificial immune systems in industrial applications," in *Intelligent Processing and Manufacturing of Materials, 1999. IPMM'99. Proceedings of the Second International Conference on*, vol. 1. IEEE, 1999, pp. 257–267.

[4] J. M. Brownjohn, "Structural health monitoring of civil infrastructure," *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 365, no. 1851, pp. 589–622, 2007.

[5] L. Osterhout and P. J. Holcomb, "Event-related potentials and syntactic anomaly: Evidence of anomaly detection during the perception of continuous speech," *Language and Cognitive Processes*, vol. 8, no. 4, pp. 413–437, 1993.

[6] Z. Sun, P. Wang, M. C. Vuran, M. A. Al-Rodhaan, A. M. Al-Dhelaan, and I. F. Akyildiz, "Bordersense: Border patrol through advanced wireless sensor networks," *Ad Hoc Networks*, vol. 9, no. 3, pp. 468–477, 2011.

[7] S. C. Jacobsen, T. J. Petelenz, and S. C. Peterson, "Wireless health monitoring system," Dec. 12 2000, uS Patent 6,160,478.

[8] P. Chakravarty, A. M. Zhang, R. Jarvis, and L. Kleeman, "Anomaly detection and tracking for a patrolling robot," in *Australasian Conference on Robotics and Automation (ACRA)*. Citeseer, 2007.

[9] L. AlSumait, D. Barbará, and C. Domeniconi, "On-line lda: Adaptive topic models for mining text streams with applications to topic detection and tracking," in *2008 eighth IEEE international conference on data mining*. IEEE, 2008, pp. 3–12.

[10] G. W. Schwert, "Anomalies and market efficiency," *Handbook of the Economics of Finance*, vol. 1, pp. 939–974, 2003.

[11] L. M. Bettencourt, A. A. Hagberg, and L. B. Larkey, "Separating the wheat from the chaff: practical anomaly detection schemes in ecological applications of distributed sensor networks," in *International Conference on Distributed Computing in Sensor Systems*. Springer, 2007, pp. 223–239.

[12] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, vol. 41, no. 3, p. 15, 2009.

[13] C. C. Aggarwal, "Outlier analysis," in *Data Mining*. Springer, 2015, pp. 237–263.

[14] S. Zani, M. Riani, and A. Corbellini, "Robust bivariate boxplots and multiple outlier detection," *Computational Statistics & Data Analysis*, vol. 28, no. 3, pp. 257–270, 1998.

[15] M.-L. Shyu, S.-C. Chen, K. Sarinnapakorn, and L. Chang, "A novel anomaly detection scheme based on principal component classifier," DTIC Document, Tech. Rep., 2003.

[16] F. Li, D. Zheng, T. Zhao, and W. Pedrycz, "A novel approach for anomaly detection in data streams: Fuzzy-statistical detection mode," *Journal of Intelligent & Fuzzy Systems*, no. Preprint, pp. 1–12, 2016.

[17] M. Schlechtingen and I. F. Santos, "Comparative analysis of neural network and regression based condition monitoring approaches for wind turbine fault detection," *Mechanical systems and signal processing*, vol. 25, no. 5, pp. 1849–1875, 2011.

[18] R. A. R. Ashfaq, X.-Z. Wang, J. Z. Huang, H. Abbas, and Y.-L. He, "Fuzziness based semi-supervised learning approach for intrusion detection system," *Information Sciences*, 2016.

[19] R. Kozik, M. Choraś, R. Renk, and W. Hołubowicz, "Semi-unsupervised machine learning for anomaly detection in http traffic," in *Proceedings of the 9th International Conference on Computer Recognition Systems CORES 2015*. Springer, 2016, pp. 767–775.

[20] M. Gupta, J. Gao, C. Aggarwal, and J. Han, "Outlier detection for temporal data," *Synthesis Lectures on Data Mining and Knowledge Discovery*, vol. 5, no. 1, pp. 1–129, 2014.

[21] E. M. Knox and R. T. Ng, "Algorithms for mining distancebased outliers in large datasets," in *Proceedings of the International Conference on Very Large Data Bases*. Citeseer, 1998, pp. 392–403.

[22] F. Angiulli and F. Fassetti, "Detecting distance-based outliers in streams of data," in *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*. ACM, 2007, pp. 811–820.

[23] D. Pokrajac, A. Lazarevic, and L. J. Latecki, "Incremental local outlier detection for data streams," in *Computational Intelligence and Data Mining, 2007. CIDM 2007. IEEE Symposium on*. IEEE, 2007, pp. 504–515.

[24] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: identifying density-based local outliers," in *ACM sigmod record*, vol. 29, no. 2. ACM, 2000, pp. 93–104.

[25] S. Papadimitriou, H. Kitagawa, P. B. Gibbons, and C. Faloutsos, "Loci: Fast outlier detection using the local correlation integral," in *Data Engineering, 2003. Proceedings. 19th International Conference on*. IEEE, 2003, pp. 315–326.

[26] H.-P. Kriegel, P. Kröger, E. Schubert, and A. Zimek, "Loop: local outlier probabilities," in *Proceedings of the 18th ACM conference on Information and knowledge management*. ACM, 2009, pp. 1649–1652.

[27] Z. He, X. Xu, and S. Deng, "Discovering cluster-based local outliers," *Pattern Recognition Letters*, vol. 24, no. 9, pp. 1641–1650, 2003.

[28] C. C. Aggarwal and P. S. Yu, "Outlier detection for high dimensional data," in *ACM Sigmod Record*, vol. 30, no. 2. ACM, 2001, pp. 37–46.

[29] H. Moradi Koupaie, S. Ibrahim, and J. Hosseinkhani, "Outlier detection in stream data by clustering method," *International Journal of Advanced Computer Science and Information Technology (IJACSIT) Vol*, vol. 2, pp. 25–34, 2014.

[30] H. Izakian and W. Pedrycz, "Anomaly detection and characterization in spatial time series data: A cluster-centric approach," *IEEE Transactions on Fuzzy Systems*, vol. 22, no. 6, pp. 1612–1624, 2014.

[31] Y. Jiang, C. Zeng, J. Xu, and T. Li, "Real time contextual collective anomaly detection over multiple data streams," *Proceedings of the ODD*, pp. 23–30, 2014.

[32] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: bringing order to the web." 1999.

[33] J. Sun, H. Qu, D. Chakrabarti, and C. Faloutsos, "Neighborhood formation and anomaly detection in bipartite graphs," in *Fifth IEEE International Conference on Data Mining (ICDM'05)*. IEEE, 2005, pp. 8–pp.

[34] L. Peel and A. Clauset, "Detecting change points in the large-scale structure of evolving networks," *arXiv preprint arXiv:1403.0989*, 2014.

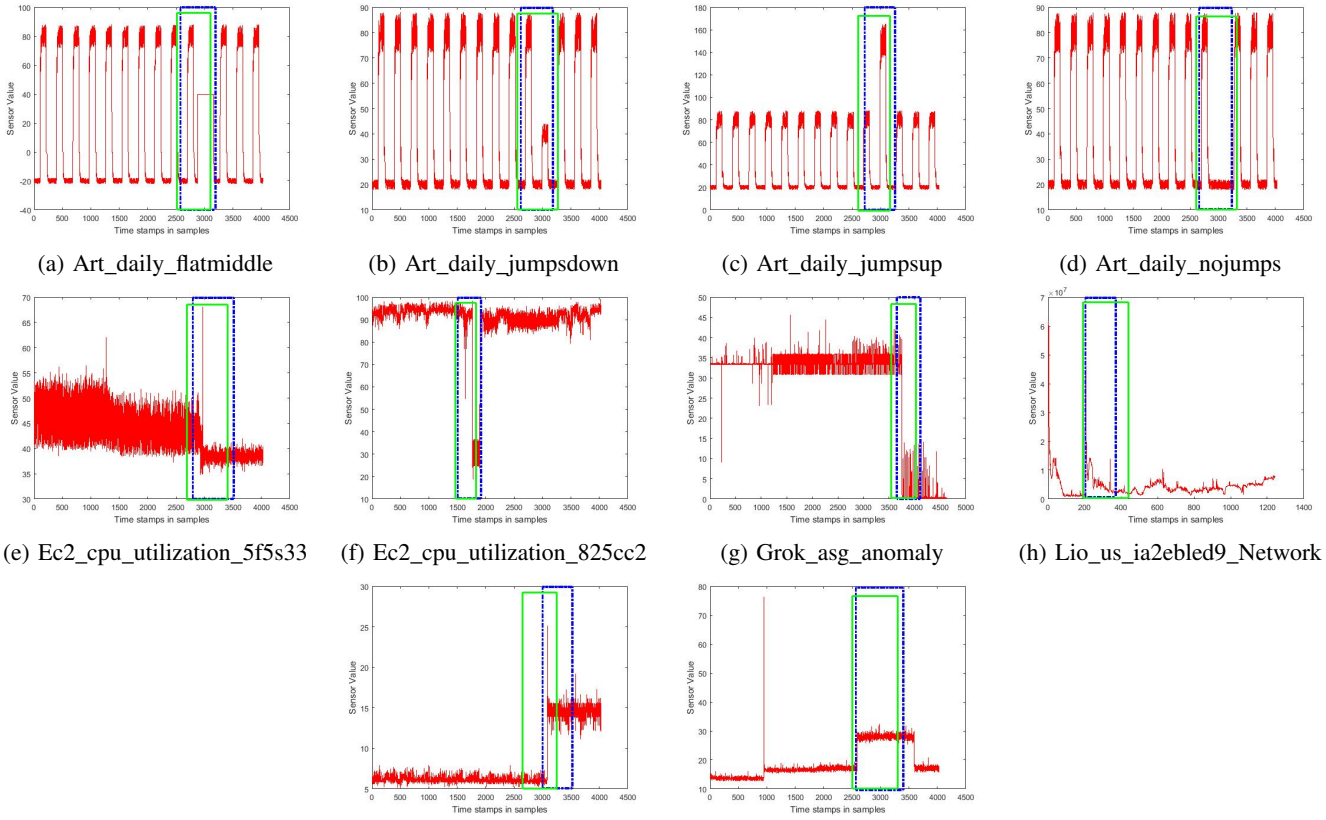
[35] B. Perozzi and L. Akoglu, "Scalable anomaly ranking of attributed neighborhoods," *arXiv preprint arXiv:1601.06711*, 2016.

[36] B. Perozzi, L. Akoglu, P. Iglesias Sánchez, and E. Müller, "Focused clustering and outlier detection in large attributed graphs," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2014, pp. 1346–1355.

[37] S.-H. Cha, "Comprehensive survey on distance/similarity measures between probability density functions," *City*, vol. 1, no. 2, p. 1, 2007.

[38] H. Bunke, P. Dickinson, A. Humm, C. Irniger, and M. Kraetzl, "Computer network monitoring and abnormal event detection using graph matching and multidimensional scaling," in *Industrial Conference on Data Mining*. Springer, 2006, pp. 576–590.

[39] C. Ding and X. He, "K-nearest-neighbor consistency in data clustering: incorporating local information into global optimization," in *Proceedings*



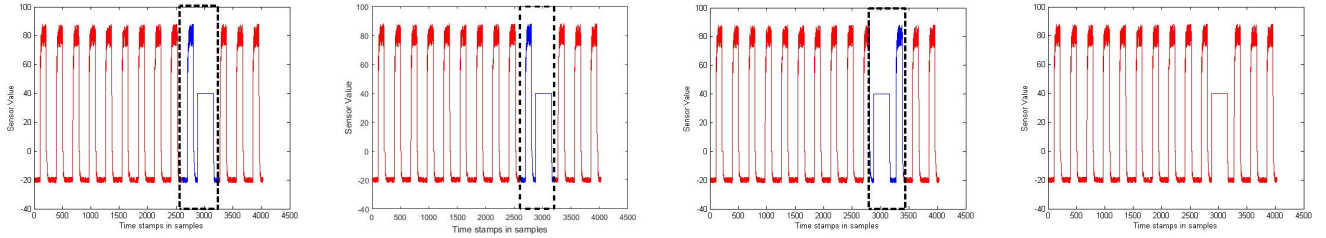
(a) Art_daily_flatmiddle (b) Art_daily_jumpsdown (c) Art_daily_jumpsup (d) Art_daily_nojumps

(e) Ec2_cpu_utilization_5f5s33 (f) Ec2_cpu_utilization_825cc2 (g) Grok_asg_anomaly (h) Lio_us_ia2ebled9_Network

(i) Rdu_cpu_utilization_cc0c53 (j) Rdu_cpu_utilization_ea7h3b

Fig. 8: Collective anomalies detected in 10 time Series of NUMENTA data Set

Fig. 9: Anomalies detected with change in value of parameter k



(a) For k=9 (b) For k=10 (c) For k=11 (d) For k $\leq$ 8 & k $\geq$12

of the 2004 ACM symposium on Applied computing. ACM, 2004, pp. 584–589.

[40] J.-S. Lee and S. Olafsson, "Data clustering by minimizing disconnectivity," *Information Sciences*, vol. 181, no. 4, pp. 732–746, 2011.

[41] N. Christofides, "An algorithm for the chromatic number of a graph," *The Computer Journal*, vol. 14, no. 1, pp. 38–39, 1971.

[42] A. Lavin and S. Ahmad, "Evaluating real-time anomaly detection algorithms–the numenta anomaly benchmark," in *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2015, pp. 38–44.

[43] J. Ha, S. Seok, and J.-S. Lee, "Robust outlier detection using the instability factor," *Knowledge-Based Systems*, vol. 63, pp. 15–23, 2014.