Egyptian Informatics Journal (2013) **14**, 157–164

Cairo University

Egyptian Informatics Journal

www.elsevier.com/locate/eij
www.sciencedirect.com

ORIGINAL ARTICLE

Outliers detection and classification in wireless sensor networks

Asmaa Fawzy [a], **Hoda M.O. Mokhtar** [b,*], **Osman Hegazy** [b]

[a] *Faculty of Science and Arts, The Northern Borders University, Saudi Arabia*
[b] *Faculty of Computers and Information, Cairo University, Egypt*

Received 25 March 2013; revised 4 June 2013; accepted 10 June 2013
Available online 10 July 2013

KEYWORDS

Outlier detection;
Wireless sensor networks;
Data mining;
Clustering

Abstract In the past few years, many wireless sensor networks had been deployed in the real world to collect large amounts of raw sensed data. However, the key challenge is to extract high-level knowledge from such raw data. In the applications of sensor networks, outlier/anomaly detection has been paid more and more attention. Outlier detection can be used to filter noisy data, find faulty nodes, and discover interesting events. In this paper we propose a novel in-network knowledge discovery approach that provides outlier detection and data clustering simultaneously. Our approach is capable to distinguish between an error due to faulty sensor and an error due to an event (probably an environmental event) which characterize the spatial and temporal correlations between events observed by sensor nodes in a confined network neighborhood. Experiments on both synthetic and real datasets show that the proposed algorithm outperforms other techniques in both effectiveness and efficiency.

1. Introduction

Outlier detection, also known as deviation detection or data cleansing, is a necessary pre-processing step in any data analysis application [1]. Outlier detection in wireless sensor networks (WSNs) is the process of identifying those data instances that deviate from the rest of the data patterns based on a certain measure [2]. The observations whose characteristics differ significantly from the normal profile are declared as outliers [3]. Wireless sensor networks (WSNs) consists of hundreds or thousands of tiny, low-cost sensor nodes, integrated with sensing, computational power, and short-range wireless communication capabilities, and have strong resource constraints in terms of energy, memory, computational capacity, and communication bandwidth. The large-scale and high density vision of the WSN implies that the network usually has to operate in a harsh and unattended environment [4]. Moreover WSNs are vulnerable to faults and malicious attacks; this in turn causes inaccurate and unreliable sensor readings [5]. Consequently, several factors make wireless sensor networks

* Corresponding author. Tel.: +20 109911734.
E-mail addresses: asmaa.fawzy25@gmail.com (A. Fawzy), h.mokhtar@fci-cu.edu.eg (H.M.O. Mokhtar), o.hegazy@gmail.com (O. Hegazy).
Peer review under responsibility of Faculty of Computers and Information, Cairo University.

Production and hosting by Elsevier

1110-8665 © 2013 Production and hosting by Elsevier B.V. on behalf of Faculty of Computers and Information, Cairo University.
http://dx.doi.org/10.1016/j.eij.2013.06.001

(WSNs) prone to outliers [6] among those factors are: (1) WSNs report the monitored data from the real world using imperfect sensing devices, (2) such devices are battery powered and thus their performance tends to deplete as power is exhausted, (3) if WSN is deployed for military and security uses, sensors are exposed to manipulation by adversaries, and (4) since these networks may include a large number of sensors, this number may reach an extremely high value that can reach to million nodes depending on the application, hence the chance of error is more than that in traditional networks. Consequently, traditional outlier detection techniques are not directly applicable to wireless sensor networks due to their particular requirements, dynamic nature, and resource limitations [7]. An appropriate outlier detection technique for the WSN should pay attention to computing power, communication and storage limitations of the network and deal with the distributed nature of -data analysis. The main objective of outlier detection in WSNs thus is to identify outliers in the distributed streaming data in an online manner with high detection accuracy while maintaining the resource consumption of the network to a minimum [8].

Distinguishing between sources of outliers is a vital matter which in turn gives an insight on how to handle the detected outlier [9]. Generally speaking if the detected outlier is an error or noisy data, it should be removed from the sensed data to ensure high data quality and accuracy; and to save energy consumption by eliminating communication load. Otherwise, if the outlier is caused by an event (e.g. fire or chemical spills), eliminating the outlier will lead to loss of important hidden information about events, which may have undesired penalty [3]. However, many researches tend to deal with outliers and events as similar conditions by treating events as some sorts of outliers (i.e. events are one of the causes of outliers). Due to the fact that there exist spatio-temporal correlation among neighboring node readings, this observation enables us to distinguish between events and errors. This is based on the fact that noisy measurements and sensor faults are likely to be stochastically unrelated, while event measurements are likely to be spatially correlated [10,11].

In our research, we investigate how to classify outliers; in other words we aim to classify outlier readings as either erroneous data or due to an actual event that occurred in the physical world. Once this classification is achieved, we start to focus on real readings and cluster them into groups in order to further classify the sensors as trustful or outlier sensors. In this paper, the clustering-based algorithm and nearest neighbor outlier detection method are combined for efficient clustering and outlier detection. We develop a technique for outlier detection that proceeds in more than one phase. The first phase is the in-network clustering-based approach. In this step, we cluster sensor data into normal clusters and outlier clusters in a distributed manner to save energy and communication overhead. The second phase concentrates on the outlier clusters to discover the source of outlier by examining the spatial correlations (i.e. neighboring node readings), and temporal correlations (i.e. readings time-stamps); if the surrounding nodes readings also give outlier values as well as the time period between readings is short this implies that there is an event. This is based on the fact that an event usually lasts for a relatively long period of time and changes historical pattern of sensor data. Otherwise, the algorithm considers such reading as erroneous or noisy data produced due to, for instance, low power.

The main contributions of this paper are as following:

1. We present a novel in-network knowledge discovery approach for outlier detection in sensor network.
2. We propose a novel approach that is capable to classify outliers as erroneous value or interesting event.
3. We conduct an intensive experimental evaluation for our algorithm based on both real data set gathered from Intel Lab and synthetic dataset that demonstrate that our approach for classification of outlier detection outperforms both in effectiveness and efficiency.

The remainder of this paper is organized as follows. Section 2 reviews the literature survey done. In section 3, we introduce some necessary background definitions. Section 4 presents the proposed approach and methodology. In Section 5, our approach for measuring sensor trustfulness is proposed. Section 6 reports the experimental evaluation on both synthetic and real datasets. Finally, Section 7 concludes the paper.

2. Related work

Outlier detection is a well-studied problem in sensor network [2,6,12,16,18,22], but further research is still needed. In 2006, Subramaniam et al. [12] proposed an online outlier detection technique for sensor networks. The authors proposed a framework that computes in a distributed fashion an approximation of multi-dimensional data distributions in order to enable complex applications in resource-constrained sensor networks. In addition, Rajasegarar et al. [13] proposed a global outlier detection technique based on clustering-based technique to identify outlier measurements in sensor nodes. This technique clusters the sensor measurements and merging clusters before communicating with other nodes. This method did not detect the interesting events in the network. Also in [14] the authors study the outlier detection problem; authors use wavelet approximation to clean local sudden-burst outliers in every node, and uses dynamic time wrapping to detect and remove outliers of small range (2 hops) which last for a long period. This approach takes the advantage of spatio-temporal correlations of sensor data to identify outliers. However, it can only give an approximate result of outliers in small area, but not the exact global result in the whole network. In addition, Branch et al. [15] proposed a technique based on distance similarity to identify global outliers in sensor networks. This technique attempts to reduce the communication overhead by a set of representative data exchanges among neighboring nodes. Each node uses distance similarity to locally identify outliers and then broadcasts the outliers to neighboring nodes for verification. The neighboring nodes repeat the procedure until the entire sensor nodes in the network eventually agree on the global outliers. This technique can be flexible in respect to multiple existing distance-based outlier detection techniques. However, the technique does not adopt any network structure so that every node uses broadcast to communicate with other nodes in the network, which causes too much communication overhead. Thus, it does not scale well to large-scale networks. In 2007, Sheng et al. [16] present a histogram-based technique to detect global outliers in data collection applications of sensor networks. This technique attempts to reduce communication cost by collecting histogram information rather than

collecting raw data for centralized processing. The sink uses histogram information to extract data distribution from the network and filters out the non-outliers. Nevertheless, Zhuang et al. [17] propose a method for cleaning sensor data in a hierarchical topology of sensor nodes. The proposed method maintains a weighted moving average that quickly reflects rapid changes in the data distribution. The weighted moving average is computed at the sink and provides clean data which can be used for query answering. This algorithm does not identify outliers and does not provide a formal definition about what an outlier is. Moreover, Zhang et al. [18] proposed a distance-based technique to identify n global outliers in snapshot and continuous query processing applications of sensor networks. This technique reduces communication overhead as it adopts the structure of aggregation tree and prevents broadcasting of each node in the network. This technique considers only one dimensional data and the aggregation tree used may not be stable due to the dynamic changes of network topology. In 2010, Verma et al. [19] proposed a statistical modeling technique that is based on the approximation of the sensor data distribution using kernel density function to transform the raw sensor readings into meaningful information. However, this approach takes into consideration only one dimensional data and does not deal with multi-dimension data. In [20], authors introduce machine learning technique for distributed event detection in wireless sensor networks and evaluate the performance and applicability for early detection of disasters, particularly residential fires. They present a distributed event detection approach incorporating a novel reputation-based voting and the decision tree. On the other hand, Al-Zoubi et al. [21] presented a fuzzy clustering method to detect outliers. The rest of the outliers were then determined by computing the difference between the objective function values and when a noticeable change is noticed, the points are considered as outliers. In 2011, Mohamed and Kavitha [22] presented a real time network outlier detection technique in WSNs based on the classification approach. Authors proposed a technique to classify the sensor node data as local outlier or cluster outlier or network outlier using Support Vector Machine (SVM) classification method. If the data is classified as network outlier then it may be due an event otherwise if it is classified as a cluster outlier then it is an error in the cluster due to some environmental factor or network otherwise is an error in the sensor node due to some defects in that sensor. This technique suffers from some computational complexity because of updating the training set in real time. This approach differs from our approach that, our approach has the advantage of spatio-temporal correlations of sensor data to identify errors and events.

3. Preliminaries

In this section, we provide necessary background definitions and mechanisms.

3.1. Outliers

Most of existing work in outlier detection stem from the field of statistics [6], initially, Hawkins [23] defined the term ''outlier'' as:

> *''An outlier is an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism''.*

However, Barnett and Lewis [24] defined it as:

> *''An outlier is an observation or subset of observations that appears to be inconsistent with the rest of the set of data''.*

From a knowledge discovery point of view, outliers are often more interesting than the normal ones since they contain useful information underlying the abnormal behavior.

Outliers arise due to mechanical faults, changes in system behavior, fraudulent behavior, human error, instrument error or simply through natural deviations in populations [8].

3.2. Source of outliers

1. *Error*: An error refers to a noise-related measurement coming from a faulty sensor. Outliers caused by errors may occur frequently, while outliers caused by events tend to have extremely smaller probability of occurrence [7].
2. *Event*: An event is defined as a particular phenomenon that changes the real-world state, e.g., forest fire, air pollution, etc. This kind of outliers usually lasts for a relatively long period of time and changes historical pattern of sensor data.

3.3. Clustering-based approach

Clustering, is a popular approach in the data mining community, used to group similar data points or objects in groups or clusters with similar behavior [25]. Clustering is an important tool for outlier analysis [1]. Several clustering-based outlier detection techniques have been developed, most of which rely on the key assumption that normal objects belong to large and dense clusters, while outliers form very small clusters or do not belong to any cluster [1,25]. There are several advantages of using cluster-based approach as mentioned in [2,7,21], among them are the following:

1. It is easily adaptable to incremental mode (i.e. after learning the clusters, new points can be inserted into the system and tested for outliers).
2. It does not have to be supervised.
3. It is suitable for anomaly detection from temporal data.
4. The testing phase for clustering based techniques is fast since the number of clusters against which every test instance needs to be compared is a small constant.

3.4. Nearest neighbor-based approach

Nearest neighbor-based approach is a widely used approach to analyze a data instance with respect to its nearest neighbors in the data mining and machine learning community [1]. It uses several well-defined distance notions to compute the distance or similarity measure between two data instance, for instance, Euclidean Distance is a popular choice for univariate and multivariate continuous attributes [26,27]. A data instance is declared as an outlier if it is located far from its neighbors [2]. Advantages of nearest neighbor-based approach are:

1. It is unsupervised in nature and does not make any assumptions regarding the underlying distribution for the data.
2. Adapting nearest neighbor-based techniques to a different data type is straightforward, and primarily requires defining an appropriate distance measure for the given data.

3.5. Outlier detection mechanisms

Outlier detection in wireless sensor networks (WSNs) is mainly processed by two mechanisms; either the centralized mechanism, or the in-network/distributed mechanism [7]. In the centralized outlier detection, both the clustering algorithm and outlier detection algorithm are performed after all data from each sensor node is transmitted to the sink node.

On the other hand, in the in-network/distributed mechanism, the clustering algorithm is moved down to the network level where each sensor node performs the clustering algorithm on its own data then produces the clusters and the parent nodes combine its own clusters with the clusters from its intermediate children. Finally, the outlier detection algorithm is performed in the gateway node to detect the outlier cluster(s).

It is clear that the in-network process is preferred than the centralized mechanism as it does not require much communication overhead between nodes which significantly saves the life time of the network. Therefore, due to the advantages of the in-network approach, our algorithms adopt the in-network paradigm

4. Proposed approach and methodology

In this section, the proposed approach is introduced in details.

Several outlier detection techniques have been developed, however they did not take into account the interesting events.

On the other hand, many recent researches were developed that are interested only in events and did not care about erroneous data. In this paper, a new clustering-based approach combined with nearest neighbor-based approach is proposed to classify outliers, i.e. noisy data or interesting events. Our methodology consists of the following four steps:

- *Pre-processing (clustering)*: First, the clustering algorithm is applied on all the sensory data to group data into clusters.
- *Outlier Detection*: In the second step, for each produced cluster, the outlier detection algorithm is applied to label each cluster as normal cluster or outlier cluster.
- *Outlier Classification*: The third step is to classify the degree of outlier value (error or event).
- *Measuring Sensor Trustfulness*: The last step is to compute the trustfulness of each sensor node to increase our certainty in trusting a specific node.

In what follows, each step is introduced in more details.

4.1. Pre-processing (clustering)

First, we execute an in-network clustering algorithm on each sensor node, using the fixed-width clustering algorithm [28,29], to cluster the measured data into clusters. In fixed-width clustering algorithm, each sensor reading is assigned to the cluster whose center is within a pre-specified distance to it. If no such cluster exists then a new cluster with the reading as the center is created. This process produces a set of fixed radius clusters in the feature space.

4.2. Outlier detection

After applying the fixed-width clustering algorithm, we have to label the produced clusters as normal clusters or outlier clusters. Therefore, we determine which clusters are anomalies based on their distance from other clusters (i.e. in our work, we use the Euclidean distance as the dissimilarity measure which is calculated as Eq. (1)). Hence, a cluster is identified as outlier if its average inter-cluster distance is more than one standard deviation of the inter-cluster distance from the mean inter-cluster distance [13].

$$D(x,y) = \sqrt{(x-y)^2} \tag{1}$$

4.3. Outlier classification

In this step, we aim to know the source of the values labeled as outlier values. Thus, we are concerned only with the outlier clusters to be able to classify these values, and consequently acquire more knowledge about their contributing sensors. Consequently, there are two possible options; either this outlier value is due to an error, which may be, for instance, because of low battery or network damage; or this outlier value is due to an event or phenomena in the surrounding environment. Our idea is based on the following observation:

> *Sensor faults are probable to be spatially unrelated while event measurements are likely to be spatially correlated.*

On the other hand, sensor data tends to be correlated in both time and space. Hence, we employed this fact by using data from neighboring nodes to assist measuring the spatial correlations, and used time stamps between readings to assist measuring the temporal correlations. In more details, the algorithm detects the neighboring nodes of the "outlier" node. If those nodes produce similar values or values larger than the outlier reading, in addition those neighboring nodes readings are within the same time range, this indicates that it is an interesting event in the physical world. Otherwise, it is likely to be an erroneous data. In our work, we assume that a sensor node (x) is considered to be a neighbor of another node (y) if x is within y's communication range, and vice versa.

Our proposed procedure proceeds as follows:

Step 1: *Perform fixed-width clustering algorithm to produce a set of C clusters*
Step 2: *Determine outlier clusters and consider the points (data instants) that belong to these clusters*
For *each outlier cluster j (determined in Step 2)*
Begin
For *each point i in cluster j*
trace all readings of neighboring nodes of point i
If *(those readings* $\geqslant$ *point i* ***and*** *timestamp of these readings is close)*
Then *classify point i as an event;*
Else *classify it as an error;*
End

4.4. Measuring sensor trustfulness:

After the outlier classification phase is finalized; we compute a measure that measures the degree of trustfulness of the readings obtained from each sensor. This measure is used to increase our confidence in labeling a sensor as trustful or outlier in terms of the produced erroneous readings.

We calculate a trustfulness measure, as shown in Eq. (2), which indicates to what extent this sensor can be trustful based on the number of erroneous readings of this sensor divided by total number of the sensor readings [30]. So basically our measure is the percentage of erroneous readings obtained from each sensor. The higher the percentage, the less we trust the readings obtained from this sensor.

$$Trust(s_i) = \left(1 - \left(\frac{N_{oi}}{N_i}\right) \times 100\right) \quad (2)$$

where $Trust(s_i)$ is the trustfulness measure for sensor s_i which refers to the percentage of erroneous readings to the whole readings, N_{oi} is number of erroneous readings of sensor s_i, and N_i is the total number of readings (i.e. either normal or erroneous readings) for sensor s_i. In order to increase the accuracy of our measure, we store the readings acquired from each sensor in the network for a period of time in a temporal database or in a data warehouse. Then, we use this historical data collection to determine some characteristics and features that characterize each node through analyzing those readings. This heuristic approach helps us to have more confidence in our decisions.

5. Experimental results

In this section, we investigate the effectiveness of our proposed approach when applied on both the real dataset from Intel Berkeley Research lab [31], and synthetic dataset. We compare the accuracy of our algorithm with another event detection method called CollECT method [32], which is based on a distributed collaboration among neighbor nodes. We evaluate the accuracy and the scalability of the proposed method against the CollECT method on both real dataset and synthetic dataset.

5.1. Real-life dataset

This dataset contains information about data collected from 54 sensors deployed in the Intel Berkeley Research lab between February 28th and April 5th, 2004. This dataset was collected with epoch duration of about 30 s (resulting in a total of about 65,000 epochs) and it contains about 2.3 million readings [31]. The dataset schema is as shown in Table 1.

In the first set of experiments, we evaluate the proposed algorithm, particularly the clustering phase, in terms of algorithm execution time. Additionally, we evaluate our proposed algorithm when varying the value of k-nearest neighbor (k-NN) parameter to be 8, 10 and 12. As shown in Fig. 1, the execution time of the proposed algorithm is reasonable especially when the k-NN parameter is equal to 12.

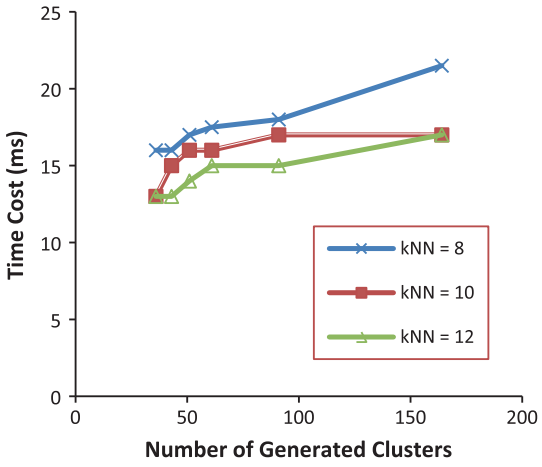

Figure 1 Number of generated clusters vs. execution time.

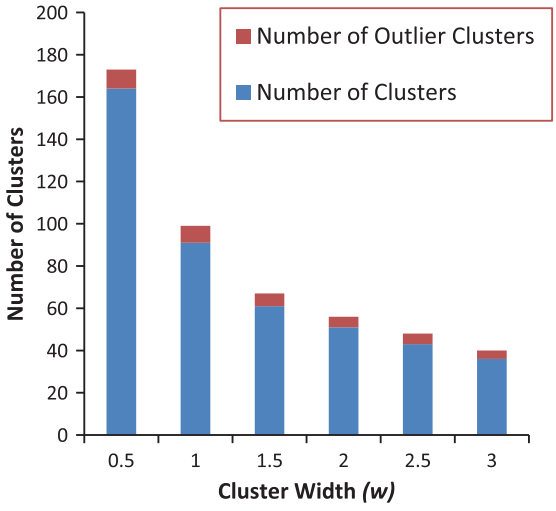

Figure 2 Total amount of produced clusters and outlier clusters vs. cluster width.

In the second experiment, we examine the number of reported outlier clusters when changing the k-NN parameter from 8, 10 and 12 as shown in Fig. 2 and vary the cluster width (w) parameter from 0.5 to 3.0 in step = 0.5. As expected, it is observed that when the width is low, number of produced clusters (i.e. normal and outlier clusters) is greater and vice versa. The result of this experiment matches our expectation for the relation between the cluster width and the number of outliers.

Outlier detection algorithms are usually evaluated using the detection rate and the false alarm rate. In order to define these

Table 1 Dataset schema.

Date	Time	Epoch	Moteid	Temp	Humidity	Light	Voltage
(yy-mm-dd)	(hh:mm:ss.xxx)	(int)	(int)	(real)	(real)	(real)	(real)

Table 2 Confusion matrix.

	Predicted outlier	Predicted normal
Actual outlier	True positive (TP)	False negative (FN)
Actual normal	False positive (FP)	True negative (TN)

metrics, we take the confusion matrix in [33], illustrated in Table 2.

Therefore, detection rate and false alarm rate could be defined as follows:

$$\text{Detection rate} = \text{TP}/(\text{TP} + \text{FN}) \quad (3)$$

$$\text{False alarm rate} = \text{FP}/(\text{FP} + \text{TN}) \quad (4)$$

In the third experiment, we evaluate the ability to find outliers (i.e. events) using our proposed algorithm and compare it with CollECT. The results are shown in Table 3. Table 3 demonstrates that our approach has the detection rate of 100%, which means that it finds all outlier (i.e. either events or erroneous data), while CollECT just has the detect rate of 92.45%. Moreover, the execution time of our approach is less than that of CollECT.

5.2. Synthetic dataset

In this section, we compare our approach against CollECT method on synthetic dataset and the results are shown in Table 4. Additionally, we evaluate the ability to classify outliers of the proposed algorithm. In this set of experiments, we use a data generator to produce a 2-dimensional datasets with 5000 objects, with 53 outliers included where 3 events are involved. In our datasets, there are four clusters and we fixed the cluster width to be 2.75. The original datasets are shown in Fig. 3.

Tables 4 and 5 show that CollECT tends to break down with large datasets, because of the curse of scalability. Meanwhile, dealing with large scale datasets is an advantage of our algorithm.

Our proposed algorithm was able to detect 53 outliers from the synthetic dataset where 50 of them are noisy data and the other 3 are interesting events.

Table 3 Detecting events on real life dataset (our approach vs. CollECT).

Algorithm	CollECT	Our approach
Execution time	21	16
Detection rate%	92.45%	100%
False alarm rate%	0.08%	0.10%

Table 4 Detection results in synthetic dataset (CollECT).

Dataset size	Execution time (ms)	Detection rate (%)	False alarm rate (%)
1500	7	95.40	0.04
3000	11	89.76	0.76
5000	13	76.25	0.98

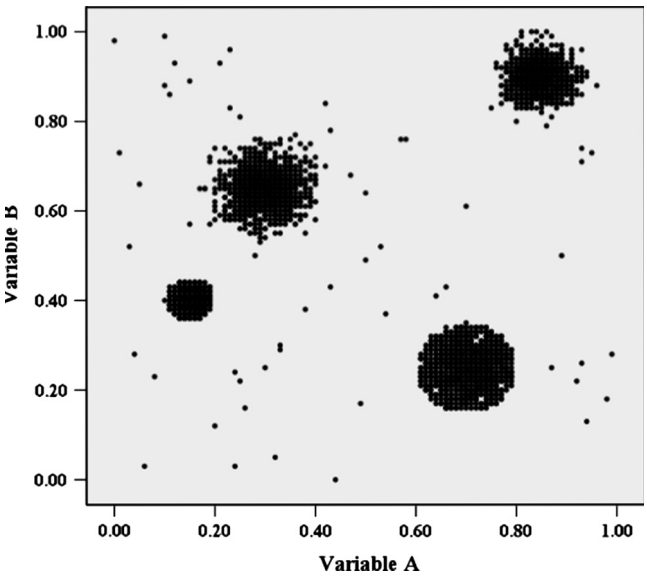

Figure 3 The synthetic dataset.

Table 5 Detection results in synthetic dataset (our approach).

Dataset size	Execution time (ms)	Detection rate (%)	False alarm rate (%)
1500	3	100	0.02
3000	5	100	0.06
5000	8	100	0.09

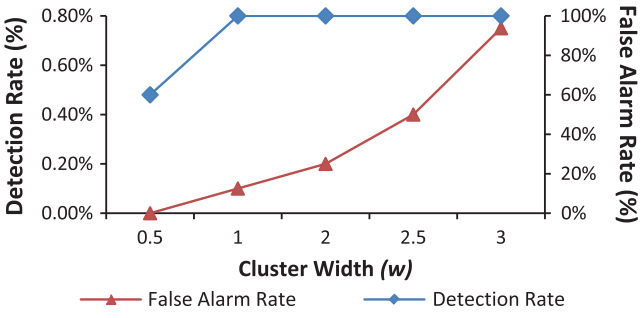

Figure 4 Impact of parameterization on the accuracy of the algorithm.

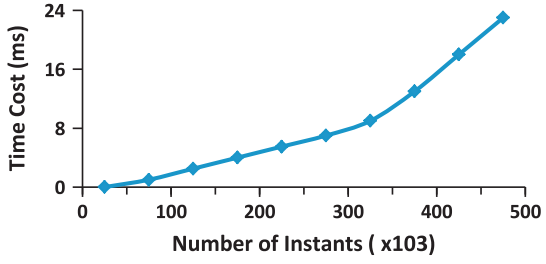

Figure 5 The scalability of the algorithm (size of dataset).

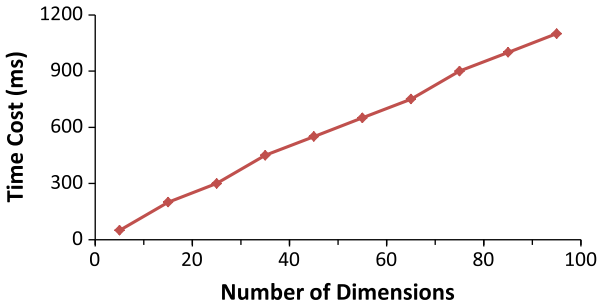

Figure 6 The scalability of the algorithm (dimensionality).

In the next set of experiments, we evaluate the efficiency of our approach on synthetic dataset. We test the detection rate and false alarm rate metrics, as shown above in Eqs. (3) and (4). In Fig. 4, if cluster width (w) is reduced to a value less than 1.0, the detection rate drops significantly, whereas the false alarm rate rises significantly if cluster width (w) is increased.

Fig. 5 shows the scalability as the size of the datasets is increased from 5,000 to 50,000. As expected, the curve exhibits likely quadratic behavior.

Fig. 6 shows the scalability as the dimensionality of the data space is increased from 2 to 20. The datasets has 5000 instances. The run time scales linearly with the dimensionality of the datasets.

6. Conclusions and future work

In this paper, we focus on the outlier detection problem in wireless sensor networks. Conventional outlier detection techniques are not suitable for wireless sensor networks due to the special characteristics and resource limitations of the networks. Rather than working on the raw sensor readings at the base station node, an in-network clustering algorithm is proposed to save the resource consumption. The proposed algorithm is capable to detect outlier values and classify them to either noisy data or interesting event in the physical environment; therefore this algorithm offers a more reliable way to gain insight into the physical phenomena under monitoring. Our approach takes into consideration various characteristics and features of streaming sensor data.

We evaluated our proposed approach with a set of experiments with both real life dataset obtained from Intel Berkeley research lab and synthetic dataset. The experimental evaluation shows that our approach can achieve high accuracy rate for identifying outliers, and demonstrate the effectiveness of the proposed approach. In addition, the experimental results show that our approach can detect interesting events as well as noisy data.

For future work, we plan to evaluate the algorithm performance on larger dataset. Also, we aim to enhance the proposed approach to deal with the phenomenon in which the event enlarges or disappears as time elapses. Future studies can also explore the solutions to sensor node deployment, data dissemination, and routing in WSNs.

References

[1] Han J, Kamber M. Data mining: concepts and techniques. Morgan Kaufmann; 2001.
[2] Chandola V, Banerjee A, Kumar V. Anomaly detection – a survey. ACM Comput Surv 2009;4(3):1–58.
[3] Tan PN. Knowledge discovery from sensor data. Sensors 2006.
[4] Akyildiz I, Su W, Sankarasubramaniam Y, Cayirci E. A survey on sensor networks. Comput Networks J 2002:38.
[5] Tian B, Yang Y, Li D, Li Q, Xin Y. A security framework for wireless sensor networks. J China U Posts Telecommun 2010;17(2):118–22.
[6] Hodge VJ, Austin J. A survey of outlier detection methodologies. Artif Intell Rev 2004;22:85–126.
[7] Zhang Y, Meratnia N, Havinga P. Outlier detection techniques for wireless sensor networks: a survey. IEEE Commun Surv Tutorials 2010;12(2):159–70.
[8] Gaber MM. Data stream processing in sensor networks; 2007. p. 41–8.
[9] Bahrepour M, Zhang Y. Use of event detection approaches for outlier detection in wireless sensor networks. In: Proceedings of symposium on theoretical and practical aspects of large-scale wireless sensor networks. Melbourne, Australia; 2009.
[10] Luo X, Dong M, Huang Y. On distributed fault-tolerant detection in wireless sensor networks. IEEE Trans Comput 2006;55(1):58–70.
[11] Elnahrawy E, Nath B. Context-aware sensors. Berlin, Germany: EWSN; 2004, pp. 77–93.
[12] Subramaniam S, Palpanas T, Papadopoulos D, Kalogeraki V, Gunopulos D. Online outlier detection in sensor data using non-parametric models. Seoul, Korea: VLDB; 2006, pp. 187–198.
[13] Rajasegarar S, Leckie C, Palaniswami M, Bezdek JC. Distributed anomaly detection in wireless sensor networks. UK: IEEE, ICCS; 2006, pp. 12–16.
[14] Zhuang Y, Chen L. In-network outlier cleaning for data collection in sensor networks. In: Proceedings of VLDB; 2006.
[15] Branch J, Szymanski B, Giannella C, Wolff R. In-network outlier detection in wireless sensor networks. In: Proceedings of IEEE ICDCS; 2006.
[16] Sheng B, Li Q, Mao W, Jin W. Outlier detection in sensor networks. QC, Canada: MobiHoc; 2007, pp. 219–228.
[17] Zhuang Y, Chen L, Wang X, Lian J. A weighted moving average-based approach for cleaning sensor data. IEEE ICDCS; 2007.
[18] Zhang K, Shi S, Gao H, Li J. Unsupervised outlier detection in sensor networks using aggregation tree. In: Proceedings of ADMA; 2007.
[19] Verma V, Kumar S, Harsh K. Outlier detection of data in wireless sensor networks using kernel density estimation. Int J Comput Appl August 2010;5(7):28–32.
[20] Bahrepour M, Meratnia N, Poel M, Taghikhaki Z, Havinga PJM. Distributed event detection in wireless sensor networks for disaster management. Intell Networking Collaborative Syst 2010:507–12.
[21] Al-Zoubi M, Al-Dahoud A, Yahya AA. New outlier detection method based on fuzzy clustering. WSEAS Trans Inf Sci Appl 2010:681–90.
[22] Mohamed MS, Kavitha T. Outlier detection using support vector machine in wireless sensor network real time data. Int J Soft Comput Eng 2011;1(2).
[23] Hawkins DM. Identification of outliers. London: Chapman and Hall; 1980.
[24] Barnett V, Lewis T. Outliers in statistical data. New York: John Wiley Sons; 1994.
[25] Jain AK, Dubes RC. Algorithms for clustering data. Englewood Cliffs, NJ: Prentice-Hall; 1988.
[26] Knorr E, Ng R, Tucakov V. Distance-based outliers: algorithms and applications. VLDB J 2000;8(3–4):237–53.
[27] Ramaswamy S, Rastogi R, Shim K. Efficient algorithms for mining outliers from large data sets. Texas, USA: ACM SIGMOD; 2000, pp. 427–438.
[28] Eskin E, Arnold A, Prerau M, Portnoy L, Stolfo S. A geometric framework for unsupervised anomaly detection: detecting intru-

sions in unlabeled data. In: Applications of data mining in computer security. Netherlands: Kluwer; 2002. p. 77–101.
[29] Ma X, Yang D, Tang S, Luo Q, Zhang D, Li S. Online mining in sensor networks. In: Lecture notes in computer science. Springer; 2004. p. 544–50.
[30] Hassan AF, Mokhtar HMO, Hegazy O. A heuristic approach for sensor network outlier detection. Int J Res Rev Wireless Sens Networks 2011;1(4).
[31] Madden S. Intel Berkeley Research Lab. <http://db.lcs.mit.edu/labdata/labdata.html>.
[32] Wang K, Yang S, Chang P, Shih C. CollECT: collaborative event detection and tracking in wireless heterogeneous sensor networks. In: Proceedings of the 11th IEEE Symposium on Computers and Communications (ISCC); 2006. p. 935–40.
[33] Lazarevic A, Kumar V. Feature bagging for outlier detection. In: Proceedings of the ACM SIGMOD international conference on knowledge discovery and data mining. Chicago, USA; 2005.