An Unsupervised Local Outlier Detection Method for Wireless Sensor Networks

Tianyu Zhang, Qian Zhao, Yoshihiro Shin and Yukikazu Nakamoto
*Graduate School of Applied Informatics University of Hyogo
Computational Science Center Building 5-7F
7-1-28 Minatojima-minamimachi Chuo-ku Kobe Hyogo Japan

***Abstract*—Recently, wireless sensor networks (WSNs) have provided many applications, which need precise sensing data analysis, in many areas. However, sensing datasets contain outliers sometimes. Although outliers rarely occur, they seriously reduce the precision of the sensing data analysis. In the past few years, many researches focused on outlier detection. However, many of them ignored one factor that WSNs are usually deployed in a dynamic environment that changes with time. Thus, we propose a new method, which is an unsupervised learning method based on mean-shift algorithm, for outlier detection that can be used in a dynamic environment for WSNs. To make our method adapt to a dynamic environment, we define two new distances for outlier detection. Moreover, the simulation shows that our method performed on real sensing dataset has ideal results; it finds outliers with a low false positive rate and has a high recall. For generality, we also test our method on different synthetic datasets.**

***Keywords*—Wireless sensor networks; outliers detection; unsupervised learning; mean-shift algorithm**

I. Introduction

In recent decades, wireless sensor networks (WSNs) have been widely used in various applications to improve people's lives including securing their properties and ensuring their safety. For example, sensor nodes are used in smart houses and other buildings to monitor and regulate the living environments to provide better living comfort and save energy. Sensor nodes are also deployed in vehicle systems to provide data required by system control. However, in such applications of WSNs, the sensing dataset may contain outliers due to, for example, low quality sensor nodes, damage to nodes caused by harsh environments, or malicious attacks from outside. The outliers make the analysis of sensing dataset imprecise, which affects the WSN performance and can even cause serious mistakes that lead to disasters. Therefore, outlier detection methods are very important to guarantee the effectiveness of applications provided by WSNs.

There are many researches [1], [2], [3], [4] about how to automatically detect outliers that need a previously collected sensing dataset. For example, statistic-based methods use a previously collected dataset to estimate a model that is an approximation for the underlying distribution that generates the dataset. After that, they detect outliers with the estimated model. However, the estimated model may become invalid when the environment changes because the underlying distribution changes with the environment as well. Supervised learning based methods have a similar weak point. They need training data where every data point in the dataset is previously labeled as normal or outlier to estimate a model. Similarly, the labels in training data may also become invalid when the environment changes. Moreover, preparing the training data is very time-consuming and expensive. Therefore, a method that can endure environment changes and automatically pick out outliers is needed in WSNs. In contrast, unsupervised learning based methods use raw data, which does not need to estimate a model from previously collected sensing dataset or prepare training data previously. Hence, unsupervised learning is more adaptable and convenient to WSNs. As a result of this property, we propose an unsupervised learning based method for detecting outliers.

Simply speaking, our outlier detection method first clusters the collected dataset and then uses the clustering result to detect outliers. We use the mean-shift algorithm to cluster the dataset because it can not only cluster the dataset but also find the mode (the mode is the most frequently occurring data point in a dataset) of each cluster as well. Then, the mode of each cluster and the median value of the sensing dataset can be used to detect which clusters are outliers. Moreover, to the best of our knowledge, this work is the first one to use the mean-shift algorithm to detect outliers in WSNs. Moreover, we simulated our method on the real sensor dataset of Intel Berkeley Research Laboratory and some synthetic datasets. We also compared our method with other unsupervised outlier detection methods [5], [6]. Simulation results shows that our method has a low FPR compared with related works, which indicates that our method outperforms than the related works in outlier detection.

The remainder of this paper is organized as follows. In Section 2, we present related researches and classify these researches into two classes that are model based and non-model based methods. Section 3 introduces preliminary knowledge related to our proposed method and the mean-shift algorithm used in our method. In Section 4, we presents the detail of our method. We test our method on real sensing dataset and synthetic dataset, and the results are shown in Section 5. Finally, Section 6 concludes this paper and provides a look at future work of our research.

II. Related Work

There are many surveys about outliers and abnormal detection, such as Y. Zhang et al. [7], Pimentel et al. [8], Chandola et al. [9], Xie et al. [10] and Gupta et al. [11]. In these reviews, outlier detection methods are all based on statistic or machine learning methods. Some of the statistic and machine learning based methods are similar. For example, parametric-based

methods in statistic-based methods are similar to supervised learning in machine learning based methods because both of them estimate a model from a previously collected dataset. Non-parametric-based methods in statistic-based methods are similar to unsupervised learning in machine learning based methods, in that they do not need to estimate a model. Hence, we classify a number of related works into model-based methods and non-model based methods.

A. Model Based Methods

As we described above, model based methods focus, for instance, on estimating a probability model and assume the model generates measured data points. If a data point has a low probability by the estimated model, the data point is considered to be an outlier.

The following three methods are based on statistics to estimate a model. Wu et al. [12] presented a localized algorithm to identify outlying sensors and event in sensor networks. They utilize the spatial relationship of neighbor sensor nodes' readings to detect outlying sensors and event. Bettencour et al. [13] proposed a local outlier detection method to detect outliers in WSNs. They also use the spatio-temporal correlation of measurements between a sensor and its neighbors to build a model. Palpanas et al. [14] proposed using kernel density estimators to estimate a sensing dataset model on the basis of the distance for online deviation detection in streaming data. This is the supervised learning based method that Rajasegarar et al. [15] used, and they presented a method for anomaly detection in WSNs based on a one-class quarter-sphere support vector machine (SVM). They use training data to fit a hypersurface, which is used to detect outliers.

B. Non-model Based Methods

Non-model based methods do not estimate a model. They use the relationship between data points, such as the distance between data points, and the density of the dataset.

These two methods are statistical non-model based methods. Subramaniam et al. [16] enhanced the work of Palpanas et al. [14] by detecting outliers online by approximating sensing data in a sliding window and using a local metrics-based algorithm to detect outliers in datasets that are hard to distinguish by distance. Sheng et al. [17] proposed a non-parametric-based method based on histogram information to detect outliers in WSNs. The biggest contribution of their method is that it reduces the communication cost by utilizing histogram information.

These are unsupervised learning methods in machine learning. Zhang et al. [5] presented an online local outlier detection method based on an unsupervised centered quarter-sphere SVM for WSN environmental monitoring applications. Fawzy et al. [6] presented a clustering based outlier detection method for WSNs. Similarly, Kiss et al. [18] also presented a clustering based outlier detection method. Other unsupervised learning based techniques include K-means approaches [19] and PCA-based approaches [20].

III. Preliminaries

In this section, we first introduce types of outliers and then introduce the related concepts and assumption in our proposed method. Finally, we introduce the clustering algorithm that we used in our method: "mean-shift algorithm".

A. Types of Outliers

Outliers are usually categorized as "global outliers" and "local outliers" (Fig. 1). Global outliers significantly deviate from the rest of the data points [21]. They are the simplest type of outliers and can be easily removed with some filters, such as "anchor data", that will be used in our method. On the other hand, local outliers are data points whose pattern significantly deviates from the pattern of the local area, so additional information of neighbor data points is needed for detecting local outliers. Therefore, detecting local outliers is more difficult than detecting global outliers.

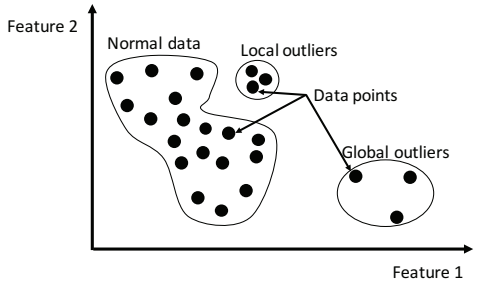

Fig. 1. Global outliers and local outliers.

B. Related Concepts and Assumption

There are three main indexes to show the center of a dataset: "mean value", "median value", and "mode".

The mean value is the average of the set of numbers, which can be easily calculated. However, it is easily affected by outliers because it becomes larger or smaller due to the effect of outliers.

The median value is the middle value in numerical order of a dataset. It is not observably affected by the outliers because if a dataset contains outliers, the median value is still decided by the majority of the non-outlier data points. Hence, most data points of a dataset are around the median value of the dataset.

The mode is a point that corresponds to the maximum probability density of a dataset. Hence, most data points are around the mode, which is similar to the median. However, calculating the mode of the dataset needs a lot of calculations. We can get an approximate value for the mode by using the median of the dataset.

In this paper, we assume that data points from a similar environment are generated by the same probability density function (PDF). Moreover, outliers are generated by other PDFs. The collected sensing dataset is mixed with normal data points and outliers. As stated above, the majority of data points should be around the center of the PDF. Moreover, the probability of outliers occurring is very low [22]. Hence, most of the data points in the dataset can be considered as normal data points, and they are around the center of the PDF. We choose the median value of the dataset to approximately represent the center of the PDF that generated the normal data points.

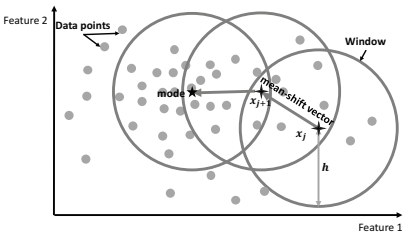

Fig. 2. Mean-shift migration from $\mathbf{x}_j$ to mode.

C. Mean-Shift Algorithm

The mean-shift algorithm [23] is an unsupervised learning based cluster algorithm developed by Fukunaga and Hostetler [24] in 1975. It is an intuitive "mode" seeking method. Cheng et al. [25] showed that the mean-shift algorithm procedure is equivalent to the gradient ascent by kernel density estimation. The result of kernel density estimation is the mode.

First, we introduce the general idea of the mean-shift algorithm. Assuming that a dataset contains N data points in an M-dimension Euclidean space, each data point contains M features, such as $\mathbf{x}_i = (x_{i1}, \ldots, x_{iM}), i = (1, \ldots, N)$. We now explain the window, radius, mean-shift vector, and mode in the mean-shift algorithm.

The window is a subset of the dataset that has center $\mathbf{x}_j$ and radius h (Fig. 2). It contains data points within a radius of h. The window notation in this paper is $win(\mathbf{x}_j, h)$. Every data point in a dataset can be considered as a center; hence, every data point can generate a window with radius h when initiating a mean-shift algorithm.

The radius h of a window is the only parameter of the mean-shift algorithm. The appropriate radius h is calculated by the standard deviation of the dataset [26]. Moreover, a stable dataset density is needed to get radius h to adapt to the dynamic environment. Hence, we introduce anchor data points (see Section 4(A) for details).

The mean-shift vector is calculated within a window. It decides the distance (length of mean-shift vector) and direction for moving the window from the previous center ($\mathbf{x}_j$) to the next center ($\mathbf{x}_{j+1}$). At the next center, the mean-shift repeats to make a new window and calculate the mean-shift vector of the new window. This process will terminate when the length of the mean-shift vector approaches zero. The mean-shift vector is calculated with the density gradient of the kernel density estimator according to Chengs study [25]. We show the derivations in the following subsection.

The mode is the center where a window finally stops moving. Data points swept by the movement of the window are contained in the same cluster because they have the same mode (center). Moreover, if some windows share the same mode (i.e. the modes are very close together), clusters generated by those windows are merged into one cluster. Fig. 2 shows the moving window procedure. The mode window is indicated by $win(\mathbf{c}_l, h)$, where $\mathbf{c}_l$ is called the mode of cluster l.

1) Kernel Density Estimator for Window: By referring to Fig. 2, the total kernel density estimation of probability density at window $win(\mathbf{x}_j, h)$ [27] is

$$p(\mathbf{x}_j) = \frac{1}{n^{(j)} h^M} \sum_{i=1}^{n^{(j)}} K\left(\frac{\mathbf{x}_j - \mathbf{x}_i}{h}\right), \tag{1}$$

where, $n^{(j)}$ is the total number of data points in $win(\mathbf{x}_j, h)$.

$K(\bullet)$ is defined as the kernel function. In accordance with the radially symmetric mentioned by Cheng [25], we are only interested in kernel function $K(\mathbf{u})$ that satisfies

$$K(\mathbf{u}) = ck\left(\|\mathbf{u}\|^2\right), \tag{2}$$

where, $k\left(\|\mathbf{u}\|^2\right)$ is called $profile$ of $K(\bullet)$. c is the positive normalization constant that assures kernel function $K(\mathbf{u})$ equals one. By utilizing the profile, we have

$$p(\mathbf{x}_j) = \frac{c}{n^{(j)} h^M} \sum_{i=1}^{n^{(j)}} k\left(\left\|\frac{\mathbf{x}_j - \mathbf{x}_i}{h}\right\|^2\right) \tag{3}$$

This is the kernel density estimator at $win(\mathbf{x}_j, h)$.

2) Calculating Mean-shift Vector of Window by using Density Gradient: To calculate the mean-shift vector of a window, we calculate the density gradient of $p(\mathbf{x}_j)$, and we set $g(s) = -k'(s)$.

$$\begin{aligned}
\nabla p(\mathbf{x}_j) &= \frac{2c}{h^{M+2}} \sum_{i=1}^{n^{(j)}} (\mathbf{x}_i - \mathbf{x}_j)\, g\left(\left\|\frac{\mathbf{x}_j - \mathbf{x}_i}{h}\right\|^2\right) \\
&= \frac{2c}{h^{M+2}} \left[\sum_{i=1}^{n^{(j)}} g\left(\left\|\frac{\mathbf{x}_j - \mathbf{x}_i}{h}\right\|^2\right)\right] \times \\
&\quad \left[\frac{\sum_{i=1}^{n^{(j)}} \mathbf{x}_i g\left(\left\|\frac{\mathbf{x}_j - \mathbf{x}_i}{h}\right\|^2\right)}{\sum_{i=1}^{n^{(j)}} g\left(\left\|\frac{\mathbf{x}_j - \mathbf{x}_i}{h}\right\|^2\right)} - \mathbf{x}_j\right]
\end{aligned} \tag{4}$$

The second term of (4) is mean-shift vector $\mathbf{m}(\mathbf{x}_j)$ in $win(\mathbf{x}_j, h)$.

$$\mathbf{m}(\mathbf{x}_j) = \frac{\sum_{i=1}^{n^{(j)}} g\left(\left\|\frac{\mathbf{x}_j - \mathbf{x}_i}{h}\right\|^2\right) \mathbf{x}_i}{\sum_{i=1}^{n^{(j)}} g\left(\left\|\frac{\mathbf{x}_j - \mathbf{x}_i}{h}\right\|^2\right)} - \mathbf{x}_j \tag{5}$$

The mean-shift vector always points in the direction of the increasing maximum density as shown in Fig. 2. Since $\mathbf{x}_j$ and the mean-shift vector are known, the next candidate center point of a window is calculated as follows:

$$\begin{aligned}
\mathbf{x}_{j+1} &= \mathbf{m}(\mathbf{x}_j) + \mathbf{x}_j \\
&= \frac{\sum_{i=1}^{n^{(j)}} g\left(\left\|\frac{\mathbf{x}_j - \mathbf{x}_i}{h}\right\|^2\right) \mathbf{x}_i}{\sum_{i=1}^{n^{(j)}} g\left(\left\|\frac{\mathbf{x}_j - \mathbf{x}_i}{h}\right\|^2\right)}
\end{aligned} \tag{6}$$

Hence, the next window is $win(\mathbf{x}_{j+1}, h)$. Moreover, according to Cheng [25], no matter from which data point the calculation starts, the final result is convergent at the mode of probability density of the observed data.

IV. Local Outlier Detection Method

In this section, we introduce our local outlier detection method. We assume that the WSN in our algorithm is a standard class[1] -based WSN. In accordance with the similar environment, the sensor nodes and class head (CH) are distributed into different classes. Sensor nodes communicate with their CH, which transmits the gathered sensing data points to the base station.

Supposing a WSN contains P classes and one class has $W^{(p)}, (p \in [1, \cdots, P])$ sensor nodes, each sensor node transmits G data points to CH in period t. Hence, each CH receives a set of data points, whose size is $N^{(p)} = W^{(p)} \times G$. One data point $\mathbf{x}_i$ contains M features, $\mathbf{x}_i = (x_{i1}, \ldots, x_{iM}), i = (1, \ldots, N^{(p)})$.

The goal of the method is to cluster collected sensing data points of CH into different clusters and then find which cluster is an outlier in the sensing dataset. We add two main features to accompany the mean-shift algorithm: 1) anchor data points to fix the density of sensing dataset for each period to efficiently utilize the mean-shift algorithm; and 2) a labeling technique to classify the properties of cluster as "normal" or "outliers" in an unsupervised manner. The algorithm is divided into three steps.

A. Step 1: Fixing Density of Sensing Data and Detecting Global Outliers

We define the density of a collected sensing dataset at period t as follows:

$$dens^{(p)} = \frac{N^{(p)}}{\Pi_{m=1}^{M} R_m^{(p)}}, \tag{7}$$

where, $R_m^{(p)}$ is the difference between the maximum and minimum value of the data points' feature m of class p at period t. The value range of feature m of the sensing dataset is different in different periods because the environment is different in different periods. Thus, the density changes along with the period.

Moreover, when the density is changing, it is not appropriate to use the mean-shift algorithm because mean-shift is sensitive to the density of a dataset, and variable density of the sensing dataset reduces the accuracy of the clustering result of the mean-shift algorithm. Furthermore, an incorrect clustering result will reduce the accuracy of outlier detection. To avoid the density changes in such a situation, we define the anchor data points, low anchor $L_m^{(p)}$, and high anchor $H_m^{(p)}$ for each feature m of class p. The low anchor $L_m^{(p)}$ is calculate by the minimum of normal feature m subtract δ_m and the high anchor $H_m^{(p)}$ is calculated by the maximum of normal feature m plus δ_m. The normal range of feature m and the value of δ_m is decided by users. Thus, a fixed density uses anchor data points as follows:

$$\hat{dens}^{(p)} = \frac{N^{(p)}}{\Pi_{m=1}^{M}(H_m^{(p)} - L_m^{(p)})}, \tag{8}$$

These anchor data points can also remove global, e.g., if a data point is lower than $L_m^{(p)}$ or larger than $H_m^{(p)}$. For example, in an office, the normal temperature range is from 20 °C to 30 °C. We set two anchor data points to 15°C and 35°C. A measurement of 10°C would be a global outlier.

B. Step 2: Clustering with Mean-Shift Algorithm

The purpose of this step is to cluster the collected sensing data of class p at period t into different clusters by mean-shift algorithm. Moreover, we have to update radius h_t at every period to guarantee the accuracy of the clustering result. Algorithm 1 shows the procedure.

Algorithm 1: Mean-Shift based Clustering	
01	for sensing dataset at each t
02	calculating radius h_t at period t
03	for data point $\mathbf{x}_i, i \in (1, \cdots, N^{(p)})$, execute the mean-shift algorithm by moving $win(\mathbf{x}_i, h_t)$ to $win(\mathbf{c}_l^{(p)}, h_t)$
	data swept by $win(\mathbf{c}_l^{(p)}, h_t)$ is defined as cluster $C_l^{(p)}$
05	if some windows share the same $\mathbf{c}^{(p)}$,
06	merge the clusters generated by those windows

As explained in Section 3(B), the mean-shift algorithm can find the mode of a cluster. First, CH calculates radius h_t which is the standard deviation of all the data points in period t. Then, the mean-shift algorithm clusters the sensing dataset by moving $win(\mathbf{x}_i, h_t), i \in (1, \cdots, N^{(p)})$ to $win(\mathbf{c}_l^{(p)}, h_t)$, where l indicates the number of clusters. If window $win(\mathbf{x}_j, h_t)$ finally stops at $\mathbf{c}_l^{(p)}$, the data points that are swept by the window are considered as cluster $C_l^{(p)}$. Moreover, if the distance between some modes of clusters is very small, we consider that these clusters share the same mode and merge those clusters. The new mode of merged cluster is the average of mode of each cluster before merging.

C. Step 3: Local Outlier Labeling Technique

We define two distances with the mode of each cluster and the median value of the collected sensing dataset, respectively. WSNs use these two distances to detect outliers. The detail of the two distances and how to detect outliers are as follows.

We define a Euclidean distance of cluster l that is the average distance from the mode $\mathbf{c}_l^{(p)}$ of cluster l to every data point in the collected sensing dataset of class p. We write this Euclidean distance as

$$Dis_l^{(p)} = \frac{\sum_{i=1}^{N^{(p)}} \left\| (\mathbf{x}_i^{(p)} - \mathbf{c}_l^{(p)}) \right\|}{N^{(p)}} \tag{9}$$

$\mathbf{M}_t^{(p)}$ is the median value of the collected sensing dataset of class p at period t. We define another Euclidean distance that is the average distance from $\mathbf{M}_t^{(p)}$ to every data point in the collected sensing dataset of class p. We write it as

[1]In WSNs, a group of sensor nodes is called a 'cluster'. In this paper, we call it a 'class' to distinguish it from 'cluster' in the mean-shift algorithm.

$$DIS^{(p)} = \frac{\sum_{i=1}^{N^{(p)}} \left\| \mathbf{x}_i^{(p)} - \mathbf{M}_t^{(p)} \right\|}{N^{(p)}} \tag{10}$$

We also find that $Dis_l^{(p)}$ is always larger or equal to $DIS^{(p)}$. The proof is as follows. The sensing dataset contains two parts. $\mathbf{x}_i : i = 1, \cdots, N$ is the normal part of the dataset, and $\mathbf{y}_j : j = 1, \cdots, n$ is the outlier part of the dataset. $\mathbf{M}_t$ is the median value of the dataset, and $N \gg n$. For the normal part, $\hat{\rho} = E(|\mathbf{x}_i - \mathbf{M}_t|)$ is the average deviation of the normal data points, and $\rho = max\{|\mathbf{x}_i - \mathbf{M}_t|\}$. For the outlier part, $\hat{R} = E(|\mathbf{y}_j - \mathbf{M}_t|)$ is the average deviation of outliers, and $R = min\{|\mathbf{y}_j - \mathbf{M}_t|\}$. $\mathbf{c}^{(l)}$ is the mode of cluster l, and the distance from every data point to $\mathbf{c}^{(l)}$ is:

$$\begin{aligned} Dis_l^{(p)} &= \sum_{i=1}^{N} \left| \mathbf{x}_i - \mathbf{c}^{(l)} \right| + \sum_{j=1}^{n} \left| \mathbf{y}_j - \mathbf{c}^{(l)} \right| \\ &\geq \sum_{i=1}^{N} \left(\left| \mathbf{c}^{(l)} - \mathbf{M}_t \right| - |\mathbf{x}_i - \mathbf{M}_t| \right) \\ &\geq N(R - \hat{\rho}) \end{aligned} \tag{11}$$

On the other hand, the distance from every data point to $\mathbf{M}_t$ is:

$$\begin{aligned} DIS^{(p)} &= \sum_{i=1}^{N} |\mathbf{x}_i - \mathbf{M}_t| + \sum_{j=1}^{n} |\mathbf{y}_j - \mathbf{M}_t| \\ &= N\hat{\rho} + n\hat{R} \end{aligned} \tag{12}$$

Then, the difference between $Dis_l^{(p)}$ and $DIS^{(p)}$ satisfies:

$$Dis_l^{(p)} - DIS^{(p)} \geq N(R - 2\hat{\rho}) - n\hat{R} \tag{13}$$

We suppose $N(R - 2\hat{\rho}) - n\hat{R} \geq 0$, then:

$$\frac{R - 2\hat{\rho}}{\hat{R}} \geq \frac{n}{N} \tag{14}$$

Since $R \gg \hat{\rho}$ and $N \gg n$, then $\frac{R}{\hat{R}} - 2\frac{\hat{\rho}}{\hat{R}} \gg 0$ and $\frac{R}{\hat{R}} - 2\frac{\hat{\rho}}{\hat{R}} \geq \frac{n}{N}$. Thus, our assumption that $\frac{R-2\hat{\rho}}{\hat{R}} \geq \frac{n}{N}$ is true. We get $Dis_l^{(p)} \geq DIS^{(p)}$.

According to our assumption that data from a similar environment is generated by the same PDF, the sensing data of every sensor node in the same class has the same PDF because sensor nodes in similar environments are classified into the same class. Hence, the center of every cluster (the mode of each cluster) is similar to the center of the entire sensing dataset (the median value of the entire dataset) of the class. Thus, if cluster l is normal, $Dis_l^{(p)}$ should be close to $DIS^{(p)}$. In other words, the ratio of $Dis_l^{(p)}$ to $DIS^{(p)}$ should be close to 1. Moreover, because $Dis_l^{(p)} \geq DIS^{(p)}$, we set threshold ϵ, which is a very small empirical value, and use discrimination $\frac{Dis_l^{(p)}}{DIS^{(p)}} - 1 \leq \epsilon$ to detect outliers. The algorithm for detecting outliers is as follows.

Algorithm: Outlier detection of cluster	
01	**for** each cluster $Dis_l^{(p)}$
02	**if** $\frac{Dis_l^{(p)}}{DIS^{(p)}} - 1 \leq \epsilon$
03	cluster l is labeled as normal
04	**else**
05	cluster l is labeled as outlier

V. Simulations

In this section, we show our simulation results based on a real dataset from the Intel Berkeley Research Laboratory and a synthetic dataset. We also compare our simulation results with those of Z. Yang et al. [5] and A. Fawzy et al. [6]. Both of them detected outliers on the basis of unsupervised method, since they used the same real dataset as we did, we compare our method with theirs by using the synthetic dataset generated in the same way for the sake of testing the generality of our method. Moreover, we compare simulation results with and without setting the anchor data since this is an important characteristic of our method.

A. Simulation Results of Real Dataset

In this subsection, we simulate our method on the real dataset from Intel Berkeley Research Laboratory[2] as shown in Fig. 3. Each sensor node in the WSN records temperature, humidity, light, and voltage once every 31 seconds. We choose the sensor nodes $1, 2, 33, 34, 35, 36$, and 37 inside the circle (35 is the CH), and we use two features, the temperature and humidity of $5th$ March 2004. The normal data ranges and the averages of temperature and humidity are shown in Table I. According to the settings of Table I, we set four types of outliers and anchor data for the real dataset, which are shown in Tables II and III. The four types of outlier cover the cases that outliers are close to or far away from the normal data range, and they are generated by different uniform distributions. For instance, the temperature values of the $outlier1$ are generated by uniform distribution in interval $[26 \sim 30]$. Moreover, we randomly insert outliers into the dataset to respectively generate datasets containing 5%, 10%, 15%, 20%, and 25% outliers for every type of outliers. The anchor data points are set by the rule that the minimum value of normal feature m subtract δ_m and the maximum value of normal feature m plus δ_m, where the δ_m is set to 6 units of a feature, such as $6°C$.

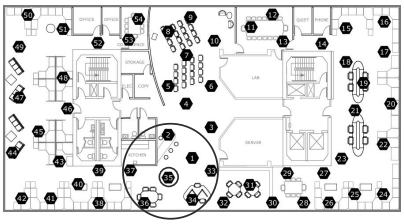

Fig. 3. Sensor nodes deployed in Intel Berkeley Research Laboratory.

- Outlier1 is near the normal data, some outliers are even inside the normal range.

[2]The dataset can be downloaded from http://db.csail.mit.edu/labdata/labdata.html, 2016

TABLE I. NORMAL DATA SETTING

	Range	Average
Temperature ($^{\circ}C$)	21.32∼28.14	23.14
Humidity (%)	26.39∼44.02	37.69

TABLE II. OUTLIER DATA SETTING

Type of Outlier	Outlier1	Outlier2	Outlier3	Oulier4
Temperature ($^{\circ}C$)	26∼30	31∼35	22∼28	31∼35
Humidity (%)	42∼46	47∼52	47∼52	27∼44

- Outlier2 is far from the normal data; however, they cannot be removed by anchor data.
- Outlier3 is such that the value of temperature is normal; however, the value of the humidity is abnormal.
- Outlier4 is the opposite setting of Outlier3.

The following terms are used to access our method:

- True Positives (TPs) are true outliers that were detected as outliers by our method.
- False Positives (FPs) are true normal samples that are wrongly detected as outliers.
- True Negatives (TNs) are true normal samples that were detected as outliers.
- False Negatives (FNs) are true outliers that are detected as normal samples.

The false positive rate (FPR) is the ratio of the normal data detected as outliers to the total true normal data, which is $\frac{FP}{FP+TN}$, and it estimates the ability of the algorithm to distinguish outliers and normal data. The FPR of our method is shown in Fig. 4. Moreover, we compare the FPR with Yang's method [5] and Fawzy's method [6]; the result is shown in Table IV. It shows that the performance of our method is acceptable, because the FPR of each case is relatively low comparing with other two related works in Table IV.

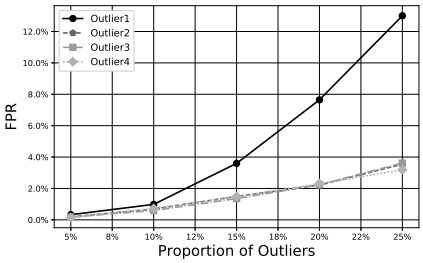

Fig. 4. Simulation results using real dataset of Intel Berkeley Research Laboratory.

In Fig. 4, $outlier2$ and $outlier3$ have similar curves so that $outlier2$ is blocked by $outlier3$. The FPR of $outlier2$, $outlier3$, and $outlier4$ kept below 3.3% when the outliers' percentage was less than or equal to 20%. Even in extreme conditions where a dataset contains 25% outliers, the worst case ($outliers1$) in our simulation has an FPR of about 12.8%.

Moreover, $outlier2$, $outlier3$, and $outlier4$ have similar results. $Outlier2$ can easily be detected as outliers because its temperature and humidity are both abnormal. Although features of $outlier3$ and $outlier4$ are partially normal, we can imagine that the distributions of $outlier3$ and $outlier4$ deviated from the normal range in two-dimension. The results of $outlier2$, $outlier3$, and $outlier4$ prove that our method can easily be adapted to different types of outliers.

TABLE III. ANCHOR DATA SETTING

Type of Anchor Data	Low Anchor	High Anchor
Temperature ($^{\circ}C$)	15.32	34.14
Humidity (%)	20.39	50.02

TABLE IV. COMPARISON OF FPR (%) ON REAL DATASET

Proportion of outlier	5%	10%	15%	20%	25%
Our method	0.20	0.74	1.98	3.60	5.83
Yang's method	1.37	7.12	11.21	18.32	19.10
Fawzy's method	0.31	2.76	4.11	8.54	11.66

Another fact (Fig. 4) is that more outliers significantly affect the FPR of our method. In $outlier1$, with the proportion of outliers increasing, more and more outliers appear in the normal range because some part of $outlier1$ overlaps the normal range. Hence, a lot of normal data points are easily detected as outliers. Similar results also appear in $outlier2$, $outlier3$, and $outlier4$ because with the proportion of outliers increasing, a great many outliers appear near to the normal range. The FPR of our method decreases when the proportion of outliers increases because normal data points are incorrectly detected as outliers. However, comparing with the other two related works according to Table IV, our method can correctly detect outlier when proportion of outliers increases.

Recall is equal to $\frac{TP}{FN+TP}$ and acts as one estimator that evaluates how many true outliers are correctly detected. The recall of our simulation is shown in Fig. 5.

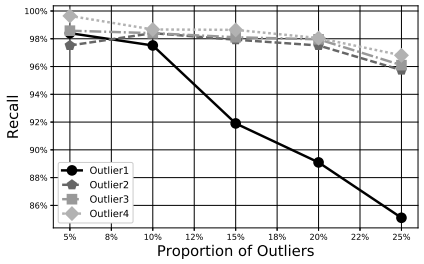

Fig. 5. Simulation results of recall.

This figure shows that all types of outliers have recall near 98% when the proportion of outliers is 5%. The recalls of $outlier2$, $outlier3$, and $outlier4$ are around 96% with increasing proportion of outliers. Even the worst case with $outlier1$ with 25% outliers, the recall is near 85%. The simulation results of every type show that our method can correctly detect outlier.

B. Simulation Results of Synthetic Datasets

Synthetic sensing data are generated by mixing three Gaussian distributions. The mean μ is randomly selected from

$(0.3, 0.35, and\ 0.45)$, and the standard deviation is $\sigma = 0.03$. Outliers are generated by uniform distribution, which is distributed in an interval of $[0.5, 1]$. According to the empirical rules of Gaussian, the value range of Gaussian distributions is $\mu \pm 3\sigma$, and the normal range of the synthetic data is $[0.21, 0.54]$. The anchor data is $[0.11, 0.64]$ which is calculated by the normal range of synthetic data $\pm$ 0.1. This synthetic dataset blends all the conditions we discussed in real data, which are outliers overlapping normal data, outliers near to normal data, and partial feature values are normal.

Fig. 6 is the result of FPR of our method. Because the synthetic data blends all types of outliers and the outliers were randomly generated, sometimes more outliers fall into or near the normal range. Thus, we can only control the quantity of outliers; however, we cannot decide where the outliers falls. This leads to the FPR of our method being higher than that of the real data, and this is the reason that the FPR is higher when the proportion of outliers is 15%.

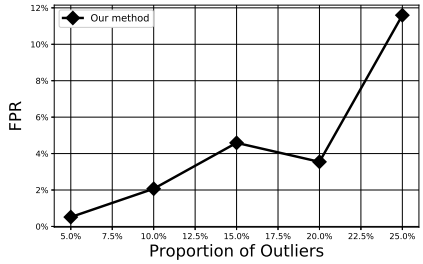

Fig. 6. Simulation results for FPR of proposed algorithm.

The comparison result between our method and Yang's method and Fawzy's method is shown in Table V. According to Table V, Yang's method and Fawzy's method tends to break down with the number of outlier increasing. Meanwhile, our method keeps a relatively low FPR, so that it can detect the outlier correctly.

TABLE V. COMPARISON OF FPR (%) ON SYNTHETIC DATASET

Method	5%	10%	15%	20%	25%
Our method	0.51	2.06	4.59	3.54	11.59
Yang's method	2.41	8.51	13.53	19.61	25.01
Fawzy's method	1.33	5.06	10.79	16.54	21.96

We also calculate the recall of our method performed on the synthetic data to confirm the effect of outliers, which is shown in Fig. 7. The result shows that the recall of our method fluctuates because the randomly generated outliers sometimes fall inside the normal range. When outliers fall inside the normal range, they significantly affect our results. However, the recall of synthetic data has a similar trend, which is decreasing with increasing outliers, with the recall of real data. Moreover, because the probability that outliers occur is low, a dataset that contains 25% outliers is an extreme case. Even in the extreme case, the recall keeps close to around 80% (Fig. 7). Hence, we conclude that our proposed method also has an acceptable performance in the more general cases.

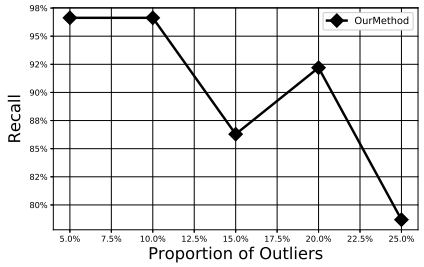

Fig. 7. Simulation results for recall on synthetic datasets.

C. Simulation Results Affected by Anchor Data

As mentioned in Section 4(A), the mean-shift algorithm may cluster the normal data into several clusters because the density of the dataset is changing with time, which leads to normal data being detected as outliers. Since using anchor data points is a feature of this work, to evaluate this aspect, we performed the following simulation where an outlier-free dataset is distributed in a 2-D Gaussian distribution. As shown in Fig. 8(a), two anchor data points were inserted at each point L and H (Fig. 8(b)). The simulation results in Fig. 8(a) show that, without setting anchor data points, the dataset were clustered into four classes, and two of them were determined as outliers. On the other hand, the simulation results in Fig. 8(b) show that, taking advantage of the anchor data points, the normal data were clustered as one class and were correctly determined as "normal."

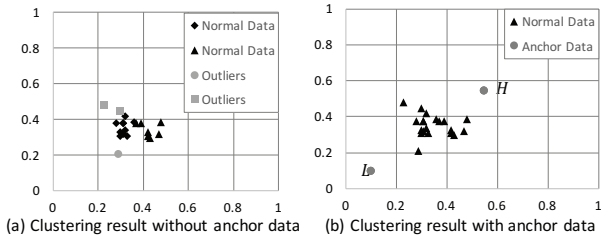

Fig. 8. Clustering results with and without anchor data.

VI. CONCLUSION

In this paper, we described the necessity for detecting outliers in WSNs and presented an unsupervised learning based outlier detection method to solve this problem. In our method, we first fixed the density of the dataset to utilize the mean-shift algorithm efficiently by using anchor data. Then, the mean-shift algorithm was used to cluster the collected sensing dataset into clusters. Finally, we proposed a labeling technique to label those clusters as "normal" or "outliers"; hence, outliers in the sensing dataset can be detected. In the simulations, we showed the performance of our proposed method and compared our work with related work [5], [6]. The results showed that our method has a lower FPR than that of the related work, and when outliers are far away from the normal data, our method obtained an FPR below 3.3%, which is quite low. Moreover, even in datasets where the distributions of outliers are close to the normal data or a substantial number of outliers are in the dataset, our method can still keep FPR at a low rate. The

simulations on synthetic dataset also showed the generality of our method.

From the QoS perspective of WSNs, to keep the WSN working properly, when outliers in the sensing data are discovered, approaches such as how to tolerate the outliers or how to detect outliers on the sensor node side should be considered. Therefore, part of our future work is methods for tolerating outliers and distributed outlier detection in sensor nodes. Moreover, our method can be used for event detection because outliers are an event in the dataset. Based on the current method, we want to improve it, for example, how to reduce computing and using less dataset, which can prolong the life of sensor nodes.

REFERENCES

[1] A. De Paola, S. Gaglio, G. L. Re, F. Milazzo, and M. Ortolani, "Adaptive distributed outlier detection for wsns," *IEEE transactions on cybernetics*, vol. 45, no. 5, pp. 902–913, 2015.

[2] E. W. Dereszynski and T. G. Dietterich, "Spatiotemporal models for data-anomaly detection in dynamic environmental monitoring campaigns," *ACM Transactions on Sensor Networks (TOSN)*, vol. 8, no. 1, p. 3, 2011.

[3] S. Mascaro, A. E. Nicholso, and K. B. Korb, "Anomaly detection in vessel tracks using bayesian networks," *International Journal of Approximate Reasoning*, vol. 55, no. 1, pp. 84–98, 2014.

[4] X. Li, J. Han, S. Kim, and H. Gonzalez, "Roam: Rule-and motif-based anomaly detection in massive moving object data sets," in *Proceedings of the 2007 SIAM International Conference on Data Mining*. SIAM, 2007, pp. 273–284.

[5] Z. Yang, N. Meratnia, and P. Havinga, "An online outlier detection technique for wireless sensor networks using unsupervised quarter-sphere support vector machine," in *in Proc. IEEE International Conference on Intelligent Sensors, Sensor Networks and Information Processing*. IEEE, 2008, pp. 151–156.

[6] A. Fawzy, H. M. Mokhtar, and O. Hegazy, "Outliers detection and classification in wireless sensor networks," *Egyptian Informatics Journal*, vol. 14, no. 2, pp. 157–164, 2013.

[7] Y. Zhang, N. Meratnia, and P. Havinga, "Outlier detection techniques for wireless sensor networks: A survey," *IEEE Communications Surveys & Tutorials*, vol. 12, no. 2, pp. 159–170, 2010.

[8] M. A. Pimentel, D. A. Clifton, L. Clifton, and L. Tarassenko, "A review of novelty detection," *Signal Processing*, vol. 99, pp. 215–249, 2014.

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

[10] M. Xie, S. Han, B. Tian, and S. Parvin, "Anomaly detection in wireless sensor networks: A survey," *Journal of Network and Computer Applications*, vol. 34, no. 4, pp. 1302–1325, 2011.

[11] M. Gupta, J. Gao, C. C. Aggarwal, and J. Han, "Outlier detection for temporal data: A survey," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 9, pp. 2250–2267, 2014.

[12] W. Wu, X. Cheng, M. Ding, K. Xing, F. Liu, and P. Deng, "Localized outlying and boundary data detection in sensor networks," *IEEE transactions on knowledge and data engineering*, vol. 19, no. 8, pp. 1145–1157, 2007.

[13] 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.

[14] T. Palpanas, D. Papadopoulos, V. Kalogeraki, and D. Gunopulos, "Distributed deviation detection in sensor networks," *ACM SIGMOD Record*, vol. 32, no. 4, pp. 77–82, 2003.

[15] S. Rajasegarar, C. Leckie, M. Palaniswami, and J. C. Bezdek, "Quarter sphere based distributed anomaly detection in wireless sensor networks." in *in Proc. IEEE International Conference on Communications*, vol. 7, 2007, pp. 3864–3869.

[16] S. Subramaniam, T. Palpanas, D. Papadopoulos, V. Kalogeraki, and D. Gunopulos, "Online outlier detection in sensor data using non-parametric models," in *in Proc. International Conference on Very Large Data Bases*. VLDB Endowment, 2006, pp. 187–198.

[17] B. Sheng, Q. Li, W. Mao, and W. Jin, "Outlier detection in sensor networks," in *Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing*. ACM, 2007, pp. 219–228.

[18] I. Kiss, B. Genge, and P. Haller, "A clustering-based approach to detect cyber attacks in process control systems," in *Industrial Informatics (INDIN), 2015 IEEE 13th International Conference on*. IEEE, 2015, pp. 142–148.

[19] S. Rajasegarar, C. Leckie, M. Palaniswami, and J. C. Bezdek, "Distributed anomaly detection in wireless sensor networks," in *in Proc. IEEE Singapore International Conference on Communication Systems*. IEEE, 2006, pp. 1–5.

[20] M. A. Livani and M. Abadi, "Distributed pca-based anomaly detection in wireless sensor networks," in *in Proc. IEEE International Conference for IEEE Internet Technology and Secured Transactions*. IEEE, 2010, pp. 1–8.

[21] J. Han, J. Pei, and M. Kamber, *Data mining: concepts and techniques*. Elsevier, 2011.

[22] K. Publishers, *Anomaly Detection with Data Mining, Kyoritsu Publishers*. Elsevier, 2009.

[23] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 5, pp. 603–619, 2002.

[24] K. Fukunaga and L. Hostetler, "The estimation of the gradient of a density function, with applications in pattern recognition," *IEEE Transactions on information theory*, vol. 21, no. 1, pp. 32–40, 1975.

[25] Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE transactions on pattern analysis and machine intelligence*, vol. 17, no. 8, pp. 790–799, 1995.

[26] D. Comaniciu, V. Ramesh, and P. Meer, "The variable bandwidth mean shift and data-driven scale selection," in *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, vol. 1. IEEE, 2001, pp. 438–445.

[27] C. M. Bishop, *Pattern recognition and machine learning*. Springer, 2006.