Detecting Energy Depriving Malicious Nodes by Unsupervised Learning in Energy Harvesting Cooperative Wireless Sensor Networks

Boqi Gao¹,a) Daichi Amagata¹,b) Takuya Maekawa¹,c) Takahiro Hara¹,d)

Abstract: This paper presents an unsupervised learning-based method for detecting energy depriving malicious nodes in an energy harvesting cooperative wireless sensor network (EHC-WSN). In EHC-WSNs, nodes wirelessly transfer a portion of their energy to their neighboring nodes if their neighboring nodes lack energy. A malicious energy depriving mode may falsely show that it has little energy and in this way obtains energy from neighboring nodes, thus depriving them of energy. To detect these malicious nodes, we utilize a clustering method. In our method, each node first observes the energy of its neighboring nodes, then it utilizes this information to obtain data points for the clustering. After clusters are formed, each node judges on a cluster of data points from malicious nodes and makes a malicious node determination. We investigate the performance of our method and confirm that our method outperforms the baseline method in terms of detection accuracy and false detection rate.

Keywords: machine learning, security, wireless sensor networks, energy harvesting, energy cooperative systems, malicious node detection

1. Introduction

With the rapid development of IoT and edge computing technologies [37], wireless sensor networks (WSNs) have been the focus of greater interest due to their real-time monitoring and computing capability. The recent breakthrough in energy harvesting cooperation (EHC) technology [29] relieves the bottleneck of energy limitation in WSNs. Therefore, the concept of energy harvesting cooperative wireless sensor networks (EHC-WSNs) has arisen and is attracting more and more attention [10]. An EHC-WSN is a WSN where nodes can harvest energy from ambient environments (e.g., harvesting from solar energy [42] and vibration [5]) and transfer energy to other nodes.

Much effort has been devoted to extending the lifetime of EHC-WSNs. Most studies focus on optimizing the energy transferring [10] and designing energy-aware routing protocols [1] to enable a WSN to have a longer lifetime. Meanwhile, few works focused on security issues of EHC-WSNs. For example, in the energy cooperative architecture, a node can obtain energy from its neighboring nodes if it lacks energy. A malicious node can therefore claim that it lacks energy even if it already has sufficient energy. By taking this energy, it can deprive neighboring nodes of their energy. This type of energy depriving attack can damage network reliability and functionality which triggers event losses. Even worse, such an attack may lead to severe risks, particularly for real-time and safety-critical applications, such as extreme weather monitoring [20], water quality monitoring [7], and forest fire alarming [36]. Valuable insight is therefore needed for dealing with attacks and security issues in EHC-WSNs. In this paper, we focus on detecting energy depriving malicious nodes in EHC-WSNs. This is the first known study that focuses on detecting energy depriving malicious nodes.

1.1 Motivation

Numerous studies have figured out various attack models for malicious nodes and proposed classifiers, rules, and encryption-based methods for detecting malicious nodes in a WSN environment [28], [32], [35]. However, these techniques cannot provide security for EHC-WSNs because they do not consider malicious nodes that harm energy harvesting and energy cooperation.

When developing a countermeasure for the energy depriving attack, we face two challenges. (i) The information about energy of a node is private data, and it cannot be made known by other nodes. This is an inherent problem because a malicious node can easily claim that it has little energy with no fear of discovery. Some energy-aware routing protocols demand that nodes periodically report their current energy storage status or add the energy storage status to the header [17], [41]. However, malicious nodes can still ignore these settings and pretend to have little energy. We may be able to design a rule-based method that

---

¹ Graduate School of Information Science and Technology Osaka University, Suita, Osaka 565–0871, Japan
a) gao.boqi@ist.osaka-u.ac.jp
b) amagata.daichi@ist.osaka-u.ac.jp
c) maekawa@ist.osaka-u.ac.jp
d) hara@ist.osaka-u.ac.jp

The preliminary version of this paper was published at Distributed Processing System Workshop (DPSWS 2019), November 2019. The paper was recommended to be submitted to Journal of Information Processing (JIP) by the chief examiner of SIGDPS.
identifies those nodes always claiming to have low energy as malicious node. However, this is impractical because of the second challenge that (ii) the energy harvesting efficiency of each node in EHC-WSN is different. For example, in forest fire alarming EHC-WSN where nodes harvest solar energy, the movements of the sun and clouds will result in shadows over some nodes. Therefore, the harvesting efficiencies would be low. A rule-based method that simply identifies nodes with low energy as malicious nodes therefore cannot deal well with this problem. Moreover, no classifier-based method are suitable for this situation because not enough training data can be prepared. This is because the malicious nodes have different falsified amount of energy storages, and different EHC-WSNs have different network environments. Therefore, it is impossible to prepare a robust classifier based on insufficient training data. Consequently, a well-designed energy depriving nodes detection method for EHC-WSNs is required.

1.2 Contribution

In this paper, we design a malicious node detection method based on unsupervised learning for EHC-WSNs, where energy depriving malicious nodes exist. Specifically, we make the following contributions.

- We tackle the problem of energy depriving nodes detection in EHC-WSNs online. To the best of our knowledge, we are the first to address this problem.
- We propose a deep neural network-based clustering method to detect energy depriving malicious nodes in EHC-WSNs.
- We propose a method to obtain data for the clustering, and propose inherent features to represent the energy depriving attack model.
- We conduct extensive experiments to investigate the performance of our method. Our experimental results demonstrate that our method outperforms the baseline method.

This paper is organized in the following way. Section 2 gives a brief overview of related works. Section 3 introduces our assumption in this paper. Section 4 presents our proposed method, and experimental results are reported in Section 5. Finally, Section 6 gives a conclusion to this paper.

2. Related Works

Energy harvesting and energy cooperation are promising methods for extending the lifetime of WSNs. In Ref. [34], Raghu-nathan et al. were the first to design a solar energy harvesting wireless system. They proposed that wireless nodes can harvest energy from solar energy to relieve energy constraints. Inspired by their work, many studies suggested that nodes in WSNs can harvest from various ambient environments [5], [39]. Later on, with the rapid development of wireless transfer technology, the lifetime of WSN has been further extended. In Ref. [15], Huang et al. were the first to enable the wireless power transfer (WPT) in cellular networks. They designed the architecture, model, and deployment for WPT in cellular networks. As an application of wireless energy transfer in WSNs, Shi et al. [38] studied a scenario where nodes in a WSN receive energy wirelessly from a mobile charging vehicle. Their experimental results proved that the lifetime of WSN can be extensively extended by wireless energy transfer. Then, Gurakan et al. [10] proposed a method that combines energy harvesting and wireless energy transfer to create wireless energy harvesting cooperation (EHC) systems (EHC-WSNs). In Ref. [18], Janhunen et al. designed a particular sensor node that is capable of energy transferring. They evaluated their sensor nodes and confirmed their efficiency of energy transferring. The work of Minasian et al. [29] further improved EHC systems by optimizing energy allocation. These studies proved that EHC-WSNs are promising; however, no work have addressed security issues of EHC-WSNs.

A large number of WSN studies have investigated various security issues. For a single kind of attack detection, Li et al. [25] developed a method that detects jamming attacks by enabling normal nodes to appropriately respond to the jammers. For replica node attacks, Ho et al. [12] utilized a sequential analysis to identify abnormal nodes in a WSN. To address the security issues of EHC networks, Kang et al. [21] proposed 4 kinds of attack models in EHC networks, and they proposed a naïve approach to mitigate the effect of malicious nodes. A few years later, in view of security issues in wireless rechargeable sensor networks, Lin et al. [26] proposed a novel attack model, which is called cooperative denial of charging attack to demonstrate that security in wireless rechargeable sensor networks needs to be further emphasized. However, they do not concern about the energy depriving attack.

Besides countermeasures for a single type of attack, many studies also aim to provide a secure environment for WSNs. In Ref. [27], Liu et al. were the first to propose an application-friendly method to detect insider attackers in WSNs by monitoring many aspects of sensor networking behaviors. Hu et al. [14] presented an attack-tolerant time-synchronization for secure data aggregation in WSNs. However, these studies consider only a WSN that is unaware of energy issues. When energy issues are concerned, their methods become inapplicable. Hence, an energy depriving node detection method needs to be devised in EHC-WSN.

3. Assumption

3.1 Network Model

In this paper, we assume an energy harvesting cooperative wireless sensor network (EHC-WSN) consisting of static wireless nodes with unique identifiers. These nodes can directly communicate with other nodes if they are within the communication range. We assume that all nodes have the same communication range, and if a given node is within the communication range of other nodes, it is a neighboring node of those other nodes. As a routing protocol, AODV, which is a standard routing protocol in WSNs, is employed. That is, when a node $s$ wants to send a data packet to another node $d$ and it does not know the route, $s$ broadcasts a route request (RReq) to create a packet transmission route, and this message is transmitted by some intermediate nodes. When node $d$ receives the RReq, it sends a route reply (RRep) toward $s$, then $s$ sends the data packet and it is transmitted through the route (see Ref. [33] for detail). Note that, nodes must add their current energy storage status to the headers of all packets sent for energy cooperation.

© 2020 Information Processing Society of Japan
3.2 Energy Harvesting and Energy Cooperation

We assume that all nodes are capable of harvesting energy from ambient environments. Due to the random nature of ambient sources (e.g., shadows over solar energy harvesting panels), we assume that each node harvests energy with different efficiency. This is the same as described existing studies [6], [8]. No node can harvest additional energy if the current energy storage is already in the maximum energy storage state.

Following the groundbreaking work of Zhang et al. [45], we assume that nodes transfer information and energy simultaneously when they send packets. The amount of transferred energy is based on the water-filling algorithm [10]. In other words, when a node transfers energy to a node, it aims to balance their energy storage. We also assume that when energy is transferred between nodes, a particular amount of the energy is lost because of the power loss [31]. Let \( E_{s} \) and \( E_{d} \) denote the status of energy storage of node \( s \) and node \( d \), respectively. Assume that \( E_{s} \) is larger than \( E_{d} \), and let \( E_{tr} \) and \( E_{rev} \) denote the energy transferred from node \( s \) and received by node \( d \), respectively. Let \( \lambda \) denote the energy transferring efficiency:

\[
E_{tr} = \lambda \cdot E_{s}.
\]

Consequently, in order to maintain the balance of status of energy storage after energy transferring, \( E_{tr} \) is calculated as:

\[
E_{tr} = \frac{E_{s} - E_{d}}{1 + \lambda}.
\]

3.3 Attack Model

In the energy deprivation attack, the malicious node pretends to have a lower energy level than its actual energy level by adding noise to its falsified amount of energy before sending a packet. Malicious nodes, in particular, shift small amounts in different values from their actual energy storage to avoid being detected. Also remember that the actual energy storage status should be included in the header.

4. Proposed Method

In this section, we describe our proposed method for detecting energy depriving nodes in EHC-WSNs. We utilize a clustering method because malicious nodes may have different falsified energy levels, and a classifier-based detection method may not work well. A rule-based method that simply decides nodes with low energy as malicious nodes also cannot deal well with this problem because different nodes have different energy consumption and harvesting rates. Moreover, the energy consumption of nodes near and far from the sink is different (a node near the sink clearly consumes more energy because of frequent packet transmissions). Hence, it is difficult to prepare a classifier-based solution to recognize such differences. On the other hand, in a clustering-based approach, a node only focuses on data from neighboring nodes and does not suffer from a different position. Therefore, we utilize a clustering method to detect malicious neighboring nodes.

A large number of studies, e.g., Refs. [3], [16], [43], have demonstrated that compared with other clustering methods, deep neural network-based clustering methods have better performance due to the theoretical function approximation properties [13] and their feature learning capabilities [2]. Therefore, we utilize a deep neural network-based clustering method. Note that in general, the task of clustering is to divide a set of data points into some clusters. In our method, each normal node, playing the role of an observing node, first prepares data points for the clustering by observing its neighboring nodes. Then, observing nodes utilize the data points to form clusters. Two clusters, which are normal and malicious clusters are formed. After the clustering, observing nodes utilize the clustering results to identify malicious nodes. For a neighboring node \( a \), if the data points from node \( a \) in the malicious cluster are more than data points from node \( a \) in the normal cluster, node \( a \) is identified as a malicious node.

4.1 Preparation of Data Points for Clustering

In this section, we describe how normal nodes prepare data points for the clustering. We assume that a node clusters its neighboring nodes at time slot \( T \), and we assume the node holds the observed data from its neighboring nodes. In our method, we split the observed data to create more data points for clustering because a large amount of data can elevate the performance of clustering the method.

As mentioned, nodes have to add their current energy storage status in the headers. Each normal node thus can observe the status of energy storage of its neighboring nodes by overhearing their packets. At each time slot, normal nodes create features from the observed energy. These features are hereinafter called energy features. At each time slot, an energy feature vector of each neighboring node is created. Therefore, at time slot \( T \), a normal node has \( T \) energy feature vectors for each of its neighboring nodes.

Figure 1 illustrates the energy feature vector set extraction procedure of neighboring nodes by node \( a \). In this figure, node \( a \) extracts the energy feature vector set of its neighboring nodes. Hereinafter, we use the term original feature vector set to denote the set of \( T \) vectors of a neighboring node. It is important to note that the energy feature vector set of each neighboring node is time-series data because it is obtained along with time.

Recall that our approach is to cluster the neighboring nodes of each normal node. We have already obtained the original feature vector set of each neighboring node by the above procedure. However, because we utilize a deep neural network-based clustering method, it is impractical to treat an original feature vector set as a data point for the clustering. The reason is that the number of energy feature vector sets is usually small (the number of energy feature vector sets is equal to the number of neighboring nodes). It is clear that machine learning approaches are not functional or easy to overfit when less information is available [4]. We therefore need more data points to enable the clustering method functional.

To deal with this problem, we propose a method that creates more data points for the clustering. Instead of using the original feature vector set as a data point for the clustering, we use subsets of it. Let \( k \) denote the size of a subset. Our method extracts vectors between \((sk + 1)\)-th vector to \((s + 1)k\)-th vector from each
An example of energy feature vector set creation. (1) At each time slot, node $a$ creates an energy feature vector of each neighboring node. (2) At time $T$, for each neighboring node, node $a$ holds an energy vector set which contains $T$ energy feature vectors.

In each subset, the vectors are still time-series data. Our method thus maintains the properties of time-series in each subset. Figure 2 shows an example of our method. In this example, we create 2 subsets with size 5 from the original feature vector set with size 10. Then, we treat these subsets as data points for the clustering. That is, a subset is a data point for future clustering.

### 4.2 Clustering Method

As mentioned above, we utilize a deep neural network-based clustering method to cluster the above-mentioned subsets. In particular, we reorganize the method proposed in Ref. [43], which is a standard deep neural network-based clustering method, called deep embedding clustering (DEC), to obtain our clustering method.

Here, we offer a brief introduction to DEC. DEC is a method that simultaneously learns feature representations and cluster assignments using deep neural networks. DEC learns a mapping from the data space to a lower-dimensional feature space in which it iteratively optimizes the clustering objective. In general, the task of clustering is to cluster a set $X$ of $x$ points into $y$ clusters. Instead of clustering directly in the data space $X$, DEC aims to transform the data with a non-linear mapping $f_{\theta}: X \rightarrow Y$, where $\theta$ is a learnable parameter, which is parametrized by deep neural networks, and $Y$ is the latent cluster space. Finally, DEC can learn the parameters of the deep neural networks, and the deep neural networks represent a clustering model of $X \rightarrow Y$ (see more details in Ref. [43]).

The original model of DEC only employs fully connected dense layers, which are not good at processing time-series data [44]. Recall that our subsets for clustering are time-series data. We thus re-organize the structure of deep neural networks of DEC by adding 1D-convolutional layers and pooling layers to process time-series data better. The 1D-convolutional layers have been proved to be able to recognize the patterns in short time-series data with fixed length [23], [44]. Therefore, the 1D-convolutional layers work well in our data because the subsets are short time-series data with fixed length. To avoid the overfitting, we discard some fully connected layers of the original DEC structure. Figure 3 shows our re-organization to original DEC.

Note that our clustering method can set the number of clusters as a hyper-parameter. We thus set the number of clusters as two because we assume two categories of nodes (normal and malicious nodes).
4.3 Energy Features

Here we describe the energy features which are used in malicious node detection. We assume that each node observes the status of energy storage of its neighboring node. Before we describe the energy features, we present the information about energy related to neighboring nodes observed by each node (Table 1). This is used to compute the energy features. To enable normal nodes to have a better understanding of neighboring nodes, we design short-term and long-term features. Short-term features enable normal nodes to recognize instant status of energy storage of themselves and their neighboring nodes. Long-term features let normal nodes understand the historical behaviors about the energy storage of their neighboring nodes. In particular, we directly employ the current status of energy storage of the observing node and the neighboring node as two short-term features. They are utilized as energy features because (i) they both have maximum and minimum values which are suitable as inputs for deep neural networks, because they can be normalized smoothly.

Table 2 shows the long-term energy features used in our method, and they are designed based on the inference of the energy depriving attack. These long-term energy features are all ratios calculated from the information observed by nodes. Such ratios are more robust against the difference in node density than numbers, because it is clear that the number of neighboring nodes can easily influence features based solely on numbers.

Here, we take TransferRatio (energy transferred ratio) and StorageRatio (energy storage ratio) as two examples to illustrate the reason why our energy features can describe the influence of the energy depriving attack. Assume that a normal node a observes its neighboring node b. (i) TransferRatio: this feature of b can help node a to measure how much energy is transferred to node b compared with all its transferred energy. If node b is a malicious node, then node a transfers comparably more energy to node b than the other neighboring nodes because it claims to have lower energy. (ii) StorageRatio: this feature measures the ratio of the current status of energy storage of nodes b and a. If node b is a malicious node, this feature value would be comparably lower than the feature values of the other neighboring nodes. This is because node b falsifies its status of energy storage to a lower value. Therefore, node a can utilize these two features to cluster b as a malicious node. Other long-term energy features are also designed based on the same approach.

It is important to note that the energy features are obtained from messages sent by neighboring nodes. In other words, each normal node can obtain the energy features by overhearing messages, so our method incurs no additional communication cost.

| Table 1 | Information about energy observed by each node. |
|--------|------------------------------------------------|
| **Information** | **Definition** |
| NTRA | Energy transferred to one neighbor |
| NREC | Energy received from one neighbor |
| TTRA | Total energy transferred |
| TREC | Total energy received |
| SS | Current status of energy storage of observing node |
| SN | Current status of energy storage of one neighbor |
| TH | Total energy harvested |

| Table 2 | Long-term energy features. |
|--------|---------------------------|
| **Behavioral features** | **Definition** |
| TRANSFER RATIO | NTRA / TTRA |
| RECEIVE RATIO | NREC / TREC |
| COMPENSATE RATIO | NTRA / TTRA |
| INCOME RATIO | NREC / TTRA |
| GIVE RATIO | NTRA / TH |
| VOLUNTARY RATIO | TTRA / TH |
| STORAGE RATIO | SS / SN |
| DELIVERY RATIO | TTRA / TH |

neighbors. After two clusters are formed, we calculate the average of feature SN (see Table 1) for both the clusters:

\[
SN_{ave} = \frac{\sum_{i=1}^{k} S N}{N - k},
\]

where \(N\) denotes the number of subsets from a cluster.

Then, the cluster with lower \(SN_{ave}\) is judged as a cluster that contains subsets from malicious nodes. This is because of the inference of the energy depriving malicious nodes, i.e., they falsely show that they have little energy.

**Malicious node decision.** After the categories of two clusters are obtained, node a identifies the malicious neighboring nodes. Assume that node a holds \(n\) subsets of each neighboring node, and \(m\) subsets of a particular neighboring node are in the cluster with malicious subsets. Let \(\hat{y}\) denote the result of malicious node decision determined by node a for this neighboring node. Here, \(\hat{y}\) is obtained as follows:

\[
\hat{y} = \begin{cases} 
\text{normal} & (m < \frac{n}{2}) \\
\text{malicious} & \text{(otherwise)}
\end{cases}
\]

We set \(\frac{n}{2}\) as a decision threshold because during the clustering procedure, all the subsets are obtained from the same node (node a). That is, from the point of view of an observing node (in this example, node a), the subsets of each node should have the same weight when deciding malicious nodes, and no bias is incurred. As a result, in the malicious node decision procedure, there is no difference of the weights of each subset. When \(m = \frac{n}{2}\), this neighboring node is identified as a malicious node because detecting malicious nodes is considered as more important than misdetecting normal nodes.

**Figure 4** shows an example of node a assessing its neighboring nodes. Node a first creates data points of its neighboring nodes. In this figure, we utilize different marks to represent data points from different neighboring nodes. Utilizing the methods stated above, node a forms two clusters of these data points. Then, because the number of data points of node b in malicious cluster is larger than that in normal cluster, node b is decided as a malicious node.

Note that our detection method detects malicious nodes online.
5.1 Setting

The performance of our proposed method was evaluated through experiments similarly to previous studies [6], [10], [26], [31]. In particular, we investigated the performance of our proposed method with a different number of malicious nodes and total nodes. To investigate the influence of a different number of malicious nodes, we set the number of nodes as 500, while the number of malicious nodes (m) were 50, 75, 100, 125, and 150. We conducted parameter analysis to investigate the effectiveness of creating subsets.

Malicious nodes. Our assumption is that malicious nodes falsely show they have little energy (Section 3.3). However, if malicious nodes always falsify their status of energy storage to an extreme low value (e.g., 0), it is clear that these malicious nodes can be detected easily. We thus added noises to their amount of falsified energies. In particular, let \( E_{\text{real}} \) and \( E_{\text{falsified}} \) denote the real status of energy storage of a malicious node and the falsified status of energy storage, respectively. To obtain \( E_{\text{falsified}} \), this malicious node deducts \( E_{\text{real}} \) by a Gaussian white noise [24] denoted by \( E_{\text{gw}} \) as:

\[
E_{\text{falsified}} = E_{\text{real}} - c \cdot E_{\text{gw}},
\]

where \( c \) is randomly selected between 0.5 to 2.

Then, the malicious node will add \( E_{\text{falsified}} \) into the header when it sends a packet. The mean of \( E_{\text{gw}} \) is 0.1, which is the same energy consumption when transmitting a packet, and the standard deviation is 0.05.

Evaluation methods. To investigate the effectiveness of our proposed method, we prepared the following methods.

- **K-means**: This method clusters the original energy vector set into two clusters by K-means [11], which is a standard method for clustering. After two clusters are formed, each node judges the cluster with lower average energy as a cluster that contains malicious energy vector set. Then, the neighboring nodes hold the original energy vector set of the malicious cluster are judged as malicious nodes. We prepare this method to investigate the effectiveness of creating subsets.

- **WSK-means**: This method clusters subsets into two clusters by K-means. We prepare this method to investigate the effectiveness of our deep neural network-based clustering method.

- **Proposed**: This is the proposed method in this paper.

- **WSDEC**: This method clusters subsets into clusters by the original DEC [43]. Therefore, this method does not have convolutional and pooling layers in the deep neural network model. We prepare this method to investigate the effectiveness of our method to handle time-series data.

5. Experiment

This section summarizes our experiments that evaluate the performance of our proposed method.

5.1 Setting

We used the Qualnet 7.4 network simulator\(^1\), and we set our experiments similarly to previous studies [6], [10], [26], [31].

We randomly deployed 500 nodes in a 100 m × 100 m square field, and 100 nodes were malicious nodes. Each node transmitted messages and data packets with a payload sizes of 256 bytes, using an IEEE 802.11b device. The communication range of each node was adjusted to roughly 3 meters and the network bandwidth was 11 Mbps. The maximum energy of each node was 100 mJ. We decided a time slot as 4 seconds, and a node harvested energy per time slot randomly from 0.01 mJ to 1 mJ [6]. For example, the Silicon Labs’ ENERGY-HARVESTING-RD\(^2\) can harvest energy from 0.01 mJ to 1 mJ per second. The energy consumption of transmitting and receiving a packet were 0.1 mJ and 0.08 mJ, respectively. We considered a random destination node scenario [22] because the development of edge computing enables each sensor node to process data [37].

We randomly chose a pair of source node and destination node every 1 second. If the source node has an active route to the destination node, the source node sends a data packet to the destination node directly. Otherwise, the source node broadcasts an RReq to find a route to the destination node. When a node forwards a data packet through a neighboring node, if the status of energy storage of this neighboring node is less than the this node, this node transfers energy to this neighboring node simultaneously (see Section 3.2). The simulation time was 4,000 seconds, which means a simulation consisted of 1,000 time slots, and the subset size \( k \) is 10. Similarly to Ref. [31], where Park et al. have proposed a method for wirelessly transferring power with 3 meters, we set \( \lambda \) as 0.29.

In addition, we conducted parameter analysis to investigate the robustness of our proposed method over different parameters. In particular, we investigated the performance of our proposed method with a different number of malicious nodes and total nodes. To investigate the influence of a different number of malicious nodes, we set the number of nodes as 500, while the number of malicious nodes (m) were 50, 75, 100, 125, and 150.

The parameter configuration of the EHC-WSN in experiments is summarized in Table 3.

### Table 3: Parameter Configuration of EHC-WSN

| Number of Malicious Nodes | Number of Total Nodes |
|---------------------------|-----------------------|
| 50                        | 500                   |
| 75                        | 750                   |
| 100                       | 1000                  |
| 125                       | 1250                  |
| 150                       | 1500                  |

Therefore, the malicious node detection procedure does not require labeled data from a certain network environment.
Implementation. We implemented our deep neural network set on Keras 2.2.4 \(^{3}\) with TensorFlow \(^{4}\) as backend. Determining hyper-parameters by cross-validation on a validation set is not an option in unsupervised clustering because we do not have labeled data. Thus we use commonly used parameters for deep neural networks. In particular, inspired by Ref. [40], we set network dimensions of WSDEC to \(d=25-25-100-10\), where \(d\) is the original data-space dimension determined by the subset size \(k\) and the number of time slots. All layers are densely (fully) connected. For our proposed method, we set network dimensions to \(d-\text{Conv1D}(25, 10)-\text{Conv1D}(25, 10)-\text{MaxPooling1D}(4)-\text{Conv1D}(50, 10)-\text{Conv1D}(50, 10)-\text{GlobalAveragePooling}()-100-10,\) where \(\text{Conv1D}\) denotes a 1-dimension convolutional layer. The number of clusters are set as 2, because we have two categories of nodes.

5.2 Results

Criteria. As mentioned earlier, our method incurs no additional communication costs. We therefore focus on the following criteria to measure the performance of the above methods.

- **Accuracy:** This is represented by \(\frac{O_{\text{nort}}-O_{\text{mal}}}{O}\), where \(O_{\text{nort}}\) and \(O_{\text{mal}}\) are respectively the set of correctly judged neighboring nodes of all normal nodes and the set of all neighboring nodes of all normal nodes.
- **Detection rate:** This is represented by \(\frac{O_{\text{mal}}}{O_{\text{mal}}-O_{\text{mal}}\cup O_{\text{nort}}}\), where \(O_{\text{nort}}\) and \(O_{\text{mal}}\) are respectively the set of correctly judged malicious neighboring nodes of all normal nodes and the set of all malicious neighboring nodes of all normal nodes.
- **Mis-Detection rate:** This is represented by \(\frac{O_{\text{nort}}}{O_{\text{mal}}\cup O_{\text{nort}}\cup O_{\text{mal}}\cup O}\), where \(O_{\text{nort}}\) and \(O_{\text{mal}}\) are respectively the set of wrongly judged normal neighboring nodes of all normal nodes and the set of all normal neighboring nodes of all normal nodes.

Comparison with K-means and WSK-means. Figures 5 (a)–5 (c) show the performances of our method (Proposed), and the performances of K-means and WSK-means. In particular, K-means is about averagely 25% worse than that of Proposed with regard to the three criteria. This is because (i) K-means uses the original vector set, (ii) compared with K-means, deep neural network-based clustering method has better feature learning capabilities, and (iii) our data points are not balanced (the data points from normal nodes is much more than those of malicious nodes), and K-means is not good for handling unbalanced data [19]. Compared with K-means, WSK-means is about averagely 5% better with regard to the three criteria. This is because WSK-means clusters subsets, and more data points provide better performance.

Comparison with WSDEC. Figures 5 (a)–5 (c) show the performances of our method (Proposed), and the performances of WSDEC. We can see that Proposed is averagely 4% better than that of WSDEC with regard to the three criteria. This is because subsets are time-series data, and Proposed utilizes convolutional layers to better process time-series data. Note that a WSN is usually composed of a large number of sensor nodes. Proposed detected 4% more malicious nodes than WSDEC, which means that dozens of malicious nodes are further detected by Proposed. The neighboring nodes of these malicious nodes are even larger number, and they are no longer influenced by the malicious nodes after they are detected and ignored. Therefore, Proposed can adequately protect the EHIC-WSN security.

Influence of simulation time. From Figs. 5 (a)–5 (c), we can see that all methods will provide better performance over a longer time. This is because longer time can generate a larger amount of data points for clustering. It is clear that clustering methods clearly provide better results when more data points are available.

Influence of number of malicious nodes. Figures 6 (a)–6 (c) show the performances of Proposed with different number of malicious nodes. From these figures, we can notice that the performance of Proposed becomes worse as the number of malicious nodes increases. This is because when the number of malicious nodes increases, the data points of malicious nodes increase. Therefore, the accuracy of clustering method decreases, since more data points have random property (recall that the malicious nodes randomly decide their falsified energy storage). This may be a limitation of our method, but if the majority is malicious, such a WSN is already broken, so we do not assume this environment. However, the Proposed still can detect 70% of malicious nodes even the ratio of malicious nodes is high, which confirms the effectiveness of Proposed.

Influence of number of nodes. Figures 7 (a)–7 (c) show the per-
formances of Proposed with different number of nodes. The performance of Proposed becomes better as the number of nodes grows. This is because when the number of nodes grows, data points of normal nodes increase. Therefore, the cluster of normal nodes is more accurately formed due to the similarity of data points of normal nodes. However, when the density of node decreases, if the number of malicious nodes remains the same, the performance of Proposed becomes worse, which is a limitation.

Information gain of energy features. Table 4 shows the information gain of energy features. The information gain is used to find distinguishing features of feature vectors. The more the information gain of a feature increases, the better the feature classifies the vector. We obtained the information gain by adding a ground truth label to each energy feature vector. For example, if a neighboring node is malicious, the labels of energy vectors observed from it are malicious. From this table, we can find that the information gain of our designed features are all over 0.009. Compared with other classifier based studies for detecting malicious nodes in WSNs [9], [30], our energy features have competitive or better information gain. We set the top-3 information gain in **bold** font. This result shows that our designed features are effective and can be employed for detecting energy depriving attacks in EHC-WSNs.

6. Conclusion

This paper presents an energy depriving malicious node detection method for EHC-WSNs. We design inherent features of energy depriving attack. We propose a method for obtaining more data points for clustering, as well as a deep neural network based clustering method. The experiments reveal that the proposed method outperforms the comparison methods. However, our work is only designed for detecting energy depriving malicious nodes in EHC-WSNs. Other attack models, e.g., collusion and packet dropping attacks cannot be detected by our methods currently. Therefore, as a part of our future work, we would like to extend our methods to detect other attacks. Moreover, when the number of malicious nodes is large, the performance of our method degrades a bit. We plan to ease this limitation in our future work.

References

[1] Anisi, M.H., Abdul-Salaam, G., Idris, M.Y.I., Wahab, A.W.A. and Ahmedy, I.: Energy harvesting and battery power based routing in wireless sensor networks, *Wireless Networks*, vol.23, no.1, pp.249–266 (2017).

[2] Bengio, Y., Courville, A. and Vincent, P.: Representation learning: A review and new perspectives, *IEEE Trans. Pattern Anal. Mach. Intell.*, vol.35, no.8, pp.1798–1828 (2013).

[3] Caron, M., Bojanowski, P., Joulin, A. and Douze, M.: Deep clustering for unsupervised learning of visual features, *ECCV*, pp.132–149 (2018).

[4] Caruana, R., Lawrence, S. and Giles, C.L.: Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping, *NIPS*, pp.402–408 (2001).
Editor’s Recommendation

This paper proposes a new unsupervised learning-based method for detecting energy depriving malicious nodes in an energy harvesting cooperative wireless sensor network. The cooperative energy harvesting could be a solution for the energy consumptions of the devices in the future IoT. The proposed method outperforms the existing method in terms of detection accuracy and false detection rate. The results are significant enough and have the potential to enhance future IoT developments.
Daichi Amagata received his B.E., M.Sc., and Ph.D. degrees from Osaka University, Osaka, Japan, in 2012, 2014, and 2015, respectively. He is currently an Assistant Professor with the Department of Multimedia Engineering, Graduate School of Information Science and Technology, Osaka University. His research interests include distributed and parallel query processing, data monitoring over stream environments, and mobile computing. He is a member of IPSJ.

Takuya Maekawa was born in 1980. He received his B.E., M.E., and Ph.D. degrees from Osaka University, Osaka, Japan in 2002, 2003, and 2005, respectively. He is currently an Associate Professor at Osaka University, Japan. His research interests include ubiquitous computing and mobile sensing. He is a member of IPSJ.

Takahiro Hara received his B.E., M.E., and Dr.Eng. degrees in information systems engineering from Osaka University, Osaka, Japan, in 1995, 1997, and 2000, respectively. He is currently a Full Professor with the Department of Multimedia Engineering, Osaka University. His research interests include distributed databases, peer-to-peer systems, mobile networks, and mobile computing systems. He is a member of IEICE and a senior member of IEEE. He is a member of three other learned societies. He is a Distinguished Scientist of ACM. He is a senior member of IPSJ.