Journal of Network and Computer Applications 173 (2021) 102852

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

When RSSI encounters deep learning: An area localization scheme for pervasive sensing systems

Zhishu Shen[a,*,1], Tiehua Zhang[b,1], Atsushi Tagami[a], Jiong Jin[b]

[a] KDDI Research, Inc., 2-1-15 Ohara, Fujimino-shi, Saitama, 356-8502, Japan
[b] School of Software and Electrical Engineering, Swinburne University of Technology, John Street, Hawthorn, Victoria, 3122, Australia

ARTICLE INFO

Keywords:
Wireless sensor network
RSSI-Based localization
Area localization
Machine learning

ABSTRACT

Localization has long been considered as a crucial research problem for pervasive sensing systems, especially with the arrival of big data era. Various techniques have been proposed to improve the localization accuracy by leveraging common wireless signals, such as radio signal strength indication (RSSI), collected from sensors placed in pervasive environments. However, the measured signal value can be easily affected by noise caused by physical obstacles in such sensing environment, which in turn compromises the localization performance. Hence, we present a novel RSSI-based area localization scheme using deep neural network (DNN) to explore the underlying correlation between the RSSI data and the respective sensor placement to achieve a superior localization performance. Moreover, to cope with the sensor data loss issue that commonly occurs during wireless sensor network (WSN) operation, an algorithm is designed to reconstruct the missing data for respective sensors in order to preserve the performance of DNN localization model. The effectiveness of the proposed scheme is verified with a real-world WSN testbed deployed inside an office building. The results demonstrate that the proposed scheme provides satisfactory prediction accuracy in area localization for pervasive sensing systems, regardless of the data loss issue that occurs with the respective sensors.

1. Introduction

Pervasive sensing systems, which incorporate pervasive computing and ubiquitous sensing, have attracted increasing attentions due to their indispensable role in our daily activities, such as offering abundant applications for environmental monitoring (Lombardo et al., 2018) and object tracking (Wang et al., 2016). Considering the immense amount of data generated by the numerous ubiquitous sensors offered, the applications involving wireless sensor networks (WSNs) are becoming increasingly data-centric. For many of these applications, knowledge of the sensor location is crucial for the meaningful interpretation of the collected big data. Moreover, accurate localization information assists in improving the routing among sensors, thus reducing power consumption generated during data transmission.

The localization of pervasive sensing systems has been actively studied due to its indispensable role in WSN setup and operation. Indeed, precise localization can be achieved by assembling sensors with global positioning system (GPS) functionality (Drawil et al., 2013). However, implementing GPS functions to a large number of sensors with truncated functionality and limited computational resources is not realistic, as well as the inapplicability of the GPS in indoor environments. For this reason, numerous studies have shifted focus to the localization of indoor pervasive sensing systems (Zafari et al., 2019). In general, these works calculated the geographic coordinates of sensors based on the *anchor nodes* whose positions are given in advance.

Indeed, for WSN applications such as vehicle traffic monitoring and pedestrian tracking, precise location information of sensors (users) is essential for achieving effective management over multiple moving targets. In contrast, for object/event-driven WSN applications, the main purpose is to confirm the areas where anomalous moving objects or events are detected. In other words, the sensor location is estimated within a certain area rather than as exact coordinates. For example, building energy management system (BEMS) constantly monitors the temperature and humidity inside a building and takes the particular interests in maintaining an appropriate comfort level in each area for which environmental sensor data are collected for analysis. Once a hotspot is detected, the BEMS will automatically send messages to air conditioners in the proximity of the hotspot area for moderating the

* Corresponding author.
E-mail address: shen@kddi-research.jp (Z. Shen).
[1] indicates the co-first authorship.

https://doi.org/10.1016/j.jnca.2020.102852
Received 23 March 2020; Received in revised form 3 September 2020; Accepted 26 September 2020
Available online 8 October 2020

temperature level (Shen et al., 2019a). More examples regarding area localization for WSN applications are summarized in Section 2. For WSN applications such as BEMSs, it is important to accurately identify areas in which a respective anomalous moving object or event is detected.

Another common issue that occurs in pervasive sensing systems is related to sensor failures. Specifically, some deployed sensors might need to be brought back by network administrators due to hardware failure or power drainage issues. To reduce the configuration efforts, the network operator returns the fixed sensors without placing them exactly at their initial positions. In other words, a sensor could be placed in the same area as long as comparable sensing performance can be guaranteed. In general, the definition of an area depends on the sensing/communication coverage of the sensor and the actual sensing environment where the pervasive sensing system operates. Additionally, the area cannot be simply determined by the sensor's coordinates without supplementary knowledge of the pervasive sensing system and its environment. Utilizing limited information obtained from the sensors to achieve accurate area localization is thus deemed crucial for object/event-driven WSN applications.

As mentioned above, by adding auxiliary functionality for localization, such as GPS, is not considered cost-effective for the majority of sensors. Therefore, recent studies have focused on developing localization techniques by using common wireless signal characteristics, such as the received signal strength indication (RSSI). Since RSSI-based localization only utilizes the inherent communication capability of sensors without the extra hardware installation, it is now the most widely used approach for indoor localization (Yang et al., 2013). However, it is worth pointing out that the measured RSSI signal contains a certain degree of noise caused by undesired radio waves or environmental obstacles, thus leading to unstable localization results. Even for a senor that receives data through end-to-end data communication from its nearby peers, the measured RSSI value is still plagued by recurring interference, which results in the deterioration of localization performance (Sadowski and Spachos, 2018). Therefore, a sophisticated technique that exploits the RSSI information considering the respective sensor location and alleviates the effects of various types of interference is highly needed.

In this paper, we propose a novel RSSI-based area localization scheme using deep neural network (DNN). In this scheme, DNN is trained to learn the correlation between the sensor location and RSSI signal in the initialization stage. More specifically, the model infers the sensor location by analyzing the collected RSSI signals from neighboring sensors in the data collection stage. This model is generic for all sensors, and it can predict their location simultaneously without the help of the anchor node. Our proposed scheme is also feasible for other cases, such as when data from some sensors are not available. By utilizing the RSSI signals collected from neighboring sensors alone, our scheme is expected to accurately perform sensor localization. Specifically, the main contributions of this paper are summarized as follows:

Scheme: A scheme that achieves RSSI-based area localization for pervasive sensing systems is proposed. This scheme is able to provide real-time localization in sensing environments in which only utilizes the RSSI signals received from other sensors. Our localization scheme does not require anchor nodes for assistance.
Algorithm: An algorithm is designed to achieve satisfactory area localization performance, where DNN is introduced to ingeniously learn the correlation between sensor locations and RSSI signals collected from multiple neighbor sensors. Moreover, a data reconstruction algorithm is proposed for filling the missing sensor data which enables the utilization of prediction model generated by DNN.
Implementation: A WSN testbed inside a real-world office building is deployed. This testbed is composed of a number of sensors to enable the evaluation of different common cases that occur during WSN operation. Based on this testbed, some primary factors that can affect the fidelity of the RSSI signal are also investigated.
Evaluation: The performance of our proposed localization scheme is compared to various state-of-the-art localization methods. Additionally, the proposed scheme is also tested on various cases, including those when some of the data are not available due to the inactive status of sensors.

The remainder of this paper is organized as follows: Section 2 reviews related works including the WSN applications requiring area location and state-of-the-art studies on localization for pervasive sensing systems. Section 3 scopes the problem regarding the area-based localization adopted for the WSN applications. Section 4 overviews our proposed scheme and explains the localization algorithm using DNN and the data processing algorithm in detail. Section 5 verifies the effectiveness of the proposed scheme by testing on the real-world WSN testbed. Finally, the work is concluded in Section 6 with providing our promising perspectives regarding future works.

2. Related work

2.1. Use case of area localization for pervasive sensing systems

We introduce three typical WSN applications, i.e., smart home, smart agriculture and wildlife monitoring, as examples to clarify *why* and *when* area localization is required, as well as *how* to define the area. The overview is summarized in Table 1. The details are as follows.

Table 1
Area localization for different WSN applications.

Application name	Definition of **area**	**Why** area localization is needed	**When** area localization is needed
ICCF (Shen et al., 2019a)	Physical environment	Anomaly detection	A hotspot is detected
Precooling for BEMS (Vishwanath et al., 2019)	Physical environment, sensing coverage	Anomaly detection	Temperature anomaly is detected
KRIPIS Smart Home (Tegou et al., 2019)	Sensing coverage	Monitoring the behaviour of users	User enters the room
Smart Syndesi (Zhao et al., 2017)	Physical environment, sensing coverage	(a) User tracking, (b) improve sensing accuracy	(a) User enters the room, (b) a sensor is moved
Strobe (Ding and Chandra, 2019)	Transmission coverage, sensing coverage	Improve sensing accuracy	A sensor is moved
Precision Agriculture (Abouzar et al., 2016)	Transmission coverage, physical environment	WSN maintenance	After sensor replacement
Badgers monitoring (Dyo et al., 2010)	Sensing coverage	WSN maintenance	After sensor replacement
Bird monitoring (Cerpa et al., 2001)	Transmission coverage	Bird tracking	A bird enters the sensing area
Rabbit tracking (Garcia-Sanchez et al., 2010)	Transmission coverage, sensing coverage	Improve sensing accuracy	A sensor is moved

2.1.1. *Smart home*

A typical smart home includes multiple sensors installed in rooms to detect anomalous events or objects. BEMS (Shen et al., 2019a; Vishwanath et al., 2019) is designed to detect anomalies in temperature or humidity and control the respective air conditioner closest to the anomaly area, such as in a given thermal zone (Vishwanath et al., 2019). The WSNs for smart home consist of several areas that depend on several factors, such as the room layout and the placement of variable air volume (VAV) boxes. For other location-aware smart home applications such as indoor people detection (Tegou et al., 2019; Zhao et al., 2017), multiple sensors are introduced to detect the people inside a given area for controlling electrical appliances, such as desk fans and lights. Another case for area localization involves repositioning sensors that have been moved from the original location for better sensing performance. Newly developed sensor technologies, which can facilitate convenient installation and removal (C develops wireless sen, 2019), will further enable sensors to be easily moved from one spot to another to achieve accurate monitoring.

As aforementioned, for smart home applications, GPS is often ineffective in indoor localization. Hence, an accurate area-based localization technique that can utilize the inherent communication data of sensors is needed.

2.1.2. *Smart agriculture*

WSN applications are designed to assist efficient agricultural activities based on different tasks, such as monitoring soil moisture (Ding and Chandra, 2019). A critical issue is that a large number of sensor boxes, which contain various components, including batteries and transceivers, are deployed for typical precision agriculture applications. To spread large-scale soil sensors to farmland, recently developed nanopaper-based sensors even allow administrators to reduce efforts to recover sensors since these devices can decompose in the soil after 40 days (Kasuga et al., 2019).

Unfortunately, the operation status of these sensors is not always stable, and thus, the WSN operators need to bring some of the sensors back for maintenance or even re-install some sensors (Abouzar et al., 2016). Accurate area localization will help operators avoid the need to mount sensors at their initial positions, which would save considerable time and human resources.

2.1.3. *Wildlife monitoring*

Pervasive sensing systems have also been applied to track and monitor wildlife animal behavior because of their potential for automatic data collection and processing without on-site human intervention. In general, a sensor is placed close to the spot where the target animal tends to appear. For example, in (Dyo et al., 2010), the sensors for monitoring the temperature and humidity were placed on known badger setts and latrines. Similar to smart agriculture applications, a critical issue in wildlife monitoring applications is that WSN administrators need to revisit the site to fetch sensors for battery replacement or hardware updates. Hence, area localization is expected to reduce the load of labor-intensive field work when relocating the multiple improved sensors at a local site.

To reduce the energy consumed when operating WSNs, some sensors can enter sleep modes when they are not needed. When a target is detected, the sensors located in a limited area that is close to the target are turned to an active state. Therefore, an area-based localization algorithm that can autonomously determine which sensors need to be awake or not is needed (Cerpa et al., 2001).

2.2. *Localization algorithm for wireless sensor networks*

Numerous studies of localization for pervasive sensing systems have focused on techniques for analyzing wireless signals that are directly measured from sensors with communication capability. In addition to RSSI, other potential wireless signal candidates for WSN localization include time of arrival (ToA) (Shen et al., 2014), time difference of arrival (TDoA) (Sun et al., 2019) and channel state information (CSI) (Wang et al., 2017). However, both ToA and TDoA require strict clock synchronization among communication sensors. At the same time, modification of the device driver is needed to obtain CSI from the network interface card (NIC), and these approaches are thus not realistic for WSN consists of multiple low-price sensors with limited configurability.

RSSI-based WSN localization is being actively studied since this approach does not demand additional hardware requirements for sensors, nor does it require users to provide the details of the physical surroundings where the WSN is deployed. A. Mackey et al. (Mackey and Spachos, 2017) used different types of beacons for RSSI-based distance measurement inside a lecture hall. In (Chen et al., 2017), in addition to RSSI, other types of sensor data, including motion and magnetometer data from multiple beacons, were calibrated for location and tracking estimation. Compared with the RSSI-based fingerprinting localization method in (Zhang et al., 2016) in which several access points (APs) served as anchor nodes to discover a target object's location, this work focuses on the usage for WSN applications, such as BEMS to automatically assessing sensor's locations without introducing anchor nodes for assistance.

Machine learning techniques are expected to improve localization accuracy due to their competence in analyzing the correlated features from the fluctuating datasets. M. Al Qathrady et al. (Al Qathrady and Helmy, 2017) estimated the distance between sensors by neural network (NN) and random forest, using RSSI and TxPower data collected from an end-to-end communication system assuming no obstacles were present between the nodes. A similar approach was presented in (Ibrahim et al., 2018) where convolutional neural network (CNN) was used to analyze time-series RSSI. Although these approaches can accurately estimate a single target object distance, they do not consider the fact that RSSI is vulnerable to physical obstacles. Consequently, these methods are not applicable in real multiple sensor environments in which the influences of obstacles cannot be neglected. W. Zhang et al. (2016) proposed an RSSI-based fingerprinting positioning algorithm using DNN. A similar fingerprinting algorithm was developed in (Wang et al., 2017), in which CSI data were analyzed for WSN localization. This algorithm was designed for cases in which the entire sensing area covered by WSN is equally divided into multiple grid areas. In each area, APs with a fixed location are used as the cornerstone to help support the positioning process. Apparently, this division scheme is not suitable for WSN application installed in complex sensing environments. Moreover, this approach cannot used for the case in which data loss occurs from one of the APs during the localization process. Our scheme aims to solve the aforementioned issues to achieve satisfactory localization performance.

3. Problem statement

3.1. Specification of WSN applications

Based on the WSN applications described in Section 2, the proposed area localization scheme is applied to WSN applications with the following features:

Sensor: Economical wireless sensors with basic sensing and communication capabilities are utilized in WSNs, i.e., when high-performance data processing capabilities, large data storage and GPS functions are not available. The adopted wireless communication system depends on the WSN application, e.g., for long-range communication, a low-power wide-area network (LPWAN) can be utilized, whereas bluetooth low energy (BLE) could be a solution for short-range indoor communication. The sensor might be replaced by others or moved to other positions for improving sensing performance.
Sensing environment: For the aforementioned WSN applications (i.e., Table 1), a large-scale physical change in the given sensing environment does not commonly occur, especially with running WSN applications. In this context, the divergence of the RSSI signal in a given environment should not exceed a certain threshold.

There is a broad range of cases for which the proposed localization scheme can be applied:

Case 1: all sensors have an active status in a WSN,
Case 2: some sensors are inactive in a WSN,
Case 3: sensors are used to replace the malfunctioning sensors, and
Case 4: sensors with positions that differ from the original positions are installed.

For **Case 2**, the *inactive* sensors refer to those that have been placed in sleep mode during WSN operation to reduce energy consumption. Hence, the *active* sensors cannot receive RSSI signals from those in inactive mode. It worth noting that adjusting a sensor's active/inactive mode can reduce the power consumption during WSNs operation since no data communication takes place for the inactive sensors. The selection of sensor for activation/inactivation can be determined by the communication quality (Ye and Zhang, 2018) or quality of sensor data processing (Shen et al., 2019b).

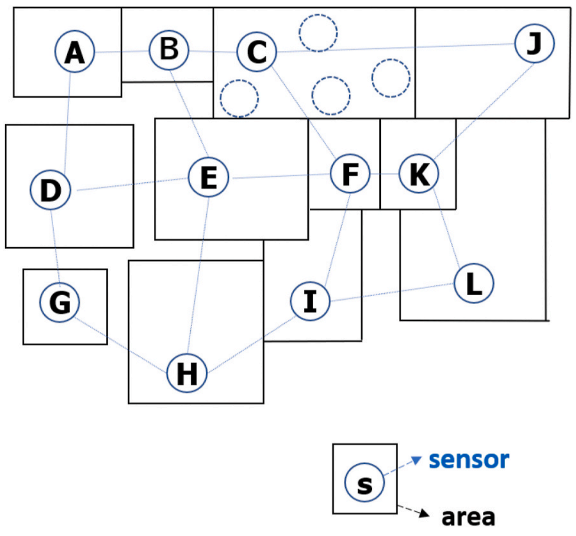

Fig. 1. An example of a real-world WSN area map.

Meanwhile, as mentioned in Section 2.1, the inactive status of sensors can also be caused by the sensor's malfunction since these low-cost sensors are often deployed in a harsh uncontrolled environment (Tang and Chow, 2016), such as those mentioned in Section 2.1. In general, the sensor faulty issues can be classified into hard faults and soft faults (Zhang et al., 2018): the hard faults are mainly due to the communication module faults or power depletion, while the soft faults refer to those caused by the issues such as sensors maliciously behavior or server delay in the sensor response to the service.

3.2. Notation

We define the *space* covered by the WSN is divided into several *areas*, where a sensor is deployed in each of these areas. Herein, an area is defined as the sensor coverage required to capture a moving object or a related event, as illustrated in Fig. 1.

Area: Areas may not have to be strictly adjacent to each other, or equally divided in a grid; instead, areas can be flexible depending on the relevant room layout, actuator positions, topography, etc. The structure is given in advance though.
Placement: Sensors are not strictly placed at the centroid of each area, and they naturally follow a practical deployment pattern, such as deployment at available positions on a shelf or a desk. Additionally, multiple positions in each area that could serve as sensor's locations might exist. For instance, sensor *C* from Fig. 1 could easily be redeployed at other positions, as denoted by dashed circles in the same area.

The sets of areas and sensors are represented as $A = \{a_1, a_2, \ldots, a_n\}$ and $S = \{s_1, s_2, \ldots, s_n\}$, respectively, where n is the number of areas. Since there is only one sensor in each area, the number of sensors is equal to the number of areas.[2] Each sensor s_j receives RSSI signal $x^{(t)}_{j,k} \in \mathbb{R}$ from a neighbor sensor k at time t. $x^{(t)}_{j,j}$, indicating the RSSI signal strength to itself, is set to zero. We define a set of all RSSI signals $\boldsymbol{x}^{(t)}_j$ received by sensor s_j at time t as follows:

$$\boldsymbol{x}^{(t)}_j = \left\{x^{(t)}_{j,1}, x^{(t)}_{j,2}, \ldots, x^{(t)}_{j,n}\right\} \tag{1}$$

When the sensor s_k is in inactive mode, $x^{(t)}_{j,k}$ is set to be $\varnothing$. Additionally, if a radio signal from a sensor cannot be received, then that sensor is considered inactive at the corresponding time.

The problem that our scheme intends to solve is determining the area a_i in which a sensor s_j is located based solely on the received RSSI signals $\boldsymbol{x}^{(t)}_j$. Specifically, the process is represented by the mapping function $\mathbf{m}$ (any localization method), which takes the RSSI signals $\boldsymbol{x}^{(t)}_j$ of s_j as the input and makes the localization predictions. This process can be formulated as:

$$\mathbf{m} : \left(s_j, \boldsymbol{x}^{(t)}_j\right) \longmapsto a_i. \tag{2}$$

If sensor data loss occurs (as described in **Case 2** when some sensors are inactive in communication), the scheme detects and compensates for the missing data before using the prediction model. Here, an unreceived

[2] Following the dataset construction method described later, our scheme also works for the case in which there is more than one sensor placed in an area; in this paper, one sensor is placed in each area based on the characteristics of the considered WSN applications.

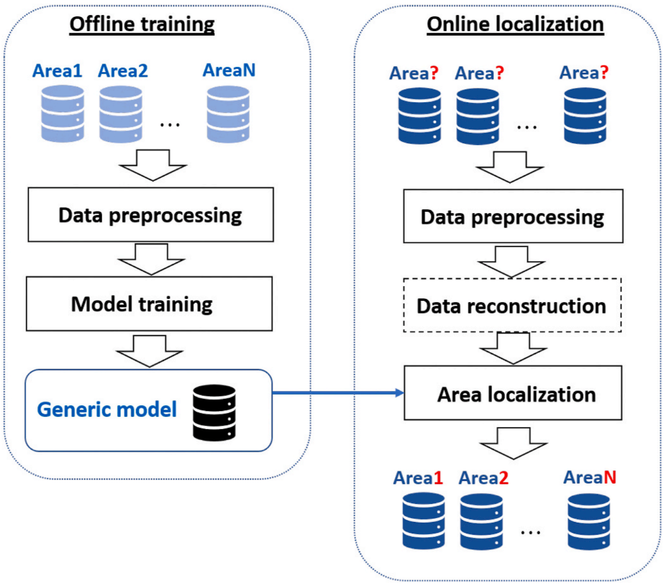

Fig. 2. Flowchart of proposed area localization scheme.

data reconstruction algorithm is applied to complement the lost data to make up the RSSI signals set $x_j^{(t)}$.

It is worth noting that the data training process is executed with a central node, e.g., a gateway node or cloud, and in localization, the central node distributes the generated prediction model to the place acting as data processing node for localization. Equation (2) uses only data measured by a certain sensor s_j, i.e., does not require data collected by other sensors. Thus, to reduce the data transmission volume, the location at which the localization process is preferred to be as close to the sensor as possible, such as at the sensor itself or a local fog node. Afterwards, only the results of the localization procedure are sent to the central node to notify the network administrators.

4. Proposed area localization scheme for pervasive sensing systems

4.1. Overview

Fig. 2 illustrates the flowchart of the proposed area localization scheme for pervasive sensing systems. The scheme consists of the **offline training** phase and the **online localization** phase. By using the pre-collected sensor data, the offline training phase trains the DNN model (Goodfellow et al., 2016) to generate a generic model for all sensors in the given WSN. Then, during the online localization phase, the generic model is utilized to localize the sensors for different cases. Further information regarding the processing of DNN can be found in (Sze et al., 2017).

4.2. Offline training phase

For the offline training phase, the scheme starts with collecting the historical RSSI data labeled with the respective sensor area. In the **data preprocessing** step, we first design the structure of training dataset that is fed into the DNN so that the DNN is able to learn the underlying correlation between RSSI signals and the area in which the sensor resides. As mentioned above, the objective of this scheme is to detect the area in which a sensor is located; hence, the area identification $a_i \in A$ is used as the label, where A is a set of all possible locations. Our scheme requires the training dataset X, which is defined as:

$$X = \left\{ \left(a_i, \overline{x}_{s_j}^{(t)} \right) \middle| \forall a_i \in A, s_j = p(a_i, t), 1 \le t \le T \right\} \tag{3}$$

where $p(a_i, t)$ is a sensor in the area a_i at time t. Note that the position of sensor s_j may change over time t. $\overline{x}_{s_j}^{(t)}$ is a set of RSSI signals with min-max normalization in the training dataset. During the data collection period, all sensors maintain an active status, i.e., $x_{j,k}^{(t)} \neq 0$, $\forall j, k, t$. The dataset derived from Equation (3) is then fed into DNN for training process.

When it comes to **model training**, we use $\mathcal{M}$ to represent the continually refined DNN model. Additionally, $\mathcal{L}$ represents the total number of layers of the DNN, and there are n_ℓ neurons at each layer ℓ, where $1 \le \ell \le \mathcal{L}$. In addition, f_ℓ is considered as the activation function at layer ℓ, in which $w_\ell \in \mathbb{R}^{n_{\ell-1} \times n_\ell}$ and $b_\ell \in \mathbb{R}^{n_\ell}$ are used to denote the weight and bias of this layer, respectively. Therefore, the following equation is used to represent each layer's output:

$$x_\ell = f_\ell(x_{\ell-1} \times w_\ell + b_\ell) \tag{4}$$

Note that $x_{\ell-1}$ is the input data $\overline{x}^{(t)}$ if ℓ is the first hidden layer. To obtain the likelihood of each output area, softmax function is applied at the output layer to convert the logits value y_{a_i} of the output layer into probability score over all area candidates. For example, the probabilities of all locations for sensor s_j can be calculated using the softmax function, from which the most likely area can be derived in the following:

$$\arg\max_{a_i} \frac{e^{y_{a_i}}}{\sum_{\forall a_k \in A} e^{y_{a_k}}} \tag{5}$$

Owing to the feedforward process supported by both Equation (4) and Equation (5), a preliminary model, which can be represented by the mapping function in Equation (2), designed specifically for the localization of the given sensors can be obtained. The model is then optimized by minimizing the batch cross-entropy loss calculated for predicted locations by comparison with the true locations, and the trainable parameters, including both w and b, are updated through a backpropagation process. More details of the model training algorithm are summarized in Algorithm 1.

4.3. Online localization

For the online localization phase, first **data preprocessing** is executed, as in the previous phase. Throughout the sensor operation period, some sensors enter the sleep mode to reduce the overall energy consumption in the WSN. Additionally, it is inevitable that factors such as temporary communication failure could lead to the unavailability of data. This situation is emphasized in **Case 2**, and **data reconstruction** is

```
Input: A deep neural network model $\mathcal{M}$ with total layers $L$, each layer
    $l_i \in L$ obtains a specific activation function $f_i$.
Phase 1: Feedforward
 1: Random initialization of weight $W$ and bias set $B$ in $\mathcal{M}$. Each $w_i \in W$
    represents the weight matrix for layer $i$, and each $b_i \in B$ denotes the bias
    vector at the same layer. $x_i$ and $a_i$ represent the input data and label
    for that layer, respectively.
 2: for $\forall l_i \in L$ do
 3:     $x_i \leftarrow f_i(w_i x_{i-1} + b_i)$
 4: end for
Phase 2: Back propagation for parameters tuning
 1: $n \leftarrow batch\ size$
 2: for each epoch $e$ do
 3:     for each batch $n$ at $e$ do
 4:         $x_{batch}, a_{batch} \leftarrow X[n], A[n]$
 5:         $\hat{a}_{batch} \leftarrow$ make predictions on the batch data $x_{batch}$ through model
    $\mathcal{M}$
 6:         calculate the batch cross-entropy loss as:
 7:         $J_{batch} \leftarrow -\frac{1}{n}\left(\sum_{i=1}^{n} a_i \cdot \log(\hat{a}_i)\right)$
 8:         for $\forall w_i \in W$ do
 9:             update $w_i \leftarrow w_i - \lambda \cdot \frac{\partial J_{batch}}{\partial w_i}$
10:         end for
11:     end for
12:     for $\forall b_i \in B$ do
13:         update $b_i \leftarrow b_i - \lambda \cdot \frac{\partial J_{batch}}{\partial b_i}$
14:     end for
15: end for
```

Algorithm 1. Model training algorithm.

required for area localization with the sensors that are active.

Once a sensor s_k is in inactive mode, other sensors s_j no longer receive the RSSI signal from s_k, i.e., $x_{j,k}^{(t)} = \emptyset$. In other words, at time t, sensor s_j might receive fewer signals than the number of areas n. In the case when the measured signals do not meet the input format requirement of the model $\mathcal{M}$, we introduce an unreceived data reconstruction algorithm to complement the inactive sensor data. The highlight of this algorithm is to search the previously collected data that are similar to the current measured data and then take the average of these values to complement the respective missing sensor data. This method is expected to be effective since the position of the sensor is unlikely to experience a significant change in a short period of time.

```
Input: $m$ : The number of missing data
Input: $\boldsymbol{x}_j^{(t)}$ : Measured signals data
Output: $\hat{\boldsymbol{x}}_j^{(t)}$ : Complemented signals data
 1: $S^{(t)} \leftarrow$ sending sensors in the set $\boldsymbol{x}_j^{(t)}$
 2: $\check{\boldsymbol{x}}_k = \emptyset, \forall k$
 3: $counter = 0, i = 1$
 4: while $i < T'$ and $counter < Th_c$ do
 5:     $S^{(t-i)} \leftarrow$ sending sensors in the set $\boldsymbol{x}_j^{(t-i)}$
 6:     $S' = S^{(t)} \cap S^{(t-i)}$, ▷ $S'$ indicates the data set in which data collected
    from $k$ is complete.
 7:     $d \leftarrow \sqrt{\sum_{k \in S'} \left( x_{j,k}^{(t-i)} - x_{j,k}^{(t)} \right)^2}$
 8:     if $d < Th_d$ then
 9:         for each $k$ which data are unreceived in $x_j^{(t)}$ do
10:             append $x_{j,k}^{(t-i)}$ into $\check{\boldsymbol{x}}_k$
11:             $counter \leftarrow counter + 1$
12:         end for
13:     end if
14:     $i \leftarrow i + 1$
15: end while
16: $\hat{\boldsymbol{x}}_j^{(t)} \leftarrow \boldsymbol{x}_j^{(t)}$
17: $\{\check{\boldsymbol{x}}'_k\} \leftarrow$ Select $m$ elements in descending order of $|\check{\boldsymbol{x}}_k|$
18: $\hat{x}_{j,k}^{(t)} \leftarrow$ average $(\check{\boldsymbol{x}}'_k)$, $\forall k$
```

Algorithm 2. Data reconstruction algorithm.

Algorithm 2 demonstrates the reconstruction process. This algorithm sequentially scans the previously collected data saved by each sensor (or fog nodes), and stores the unreceived sensor data if the Euclidean distance from the current measured data (d on line 7) is less than a threshold value Th_d. This scanning process continues until the number of retained values T' is exceeded or the number of stored values exceeds the threshold Th_c. Finally, the sensors with the most stored data are selected, and the average value is used as a complementary value (lines 15 to 16).

Finally, the completed datasets are forwarded to the **area localization** module to output the area locations of the given sensors. Note that this scheme can be also applied to **Case 3** and **Case 4** if the sensor maintenance is needed.

5. Experimental evaluation

We use FeasyBeacon FSC-BP103 as the experimental sensor, which is configured to send BLE advertisements with 1 s transmission interval. Sensors are placed inside an office building. The reasons to adopt FSC-BP103 include its compact size and one-year battery life which enables easy deployment in our real-world indoor environment for a long-term WSNs operation. Moreover, its superb configurability on various parameters related to wireless communication (Bluetooth 5.0) can further supports our experimental validation. We first take a closer look at the sensor RSSI fluctuation through an end-to-end sensor communication and then evaluate the effectiveness of the proposed area-based localization scheme with a WSN testbed. The details of them are described in the respective subsections.

5.1. Measurement of RSSI signals

5.1.1. Experimental setup

The measurements are obtained through end-to-end sensor communication. Specifically, sensors send BLE advertisements with different TxPower levels (= −12 dBm, 0 dBm, and 12 dBm), each of which has a transmission interval of 1000 ms. Here, various distances (= d ranges from 0.5 m to 10 m) between each node pair are tested. For each setup parameter (TxPower and d pair), we recorded the RSSI data at the receiver side (BUFFALO USB Bluetooth 4.0 Low Energy Micro Adapter inserted into a laptop) during a 1-hour data communication.

5.1.2. Measurement results

Theoretically, for the free space model, the strength of the RSSI signal is attenuated with the increase of the sensors distance. However, in a real sensing environment, the measured RSSI contains noise caused by undesired radio waves or physical obstacles in the environment. The simple moving average removes noise and helps to obtain feature quantities. Fig. 3 shows the variances of the RSSI by averaging the data over the previous W (window size) seconds.

This line plots the mean values with different TxPower levels, indicating that a simple moving average can effectively remove noise. Based on the observations, the value obtained by a simple moving average over 30 s is adopted in the evaluation phase.

Next, we explore the feasibility of using the RSSI data in distance-based localization. For simplicity, we conduct the localization for

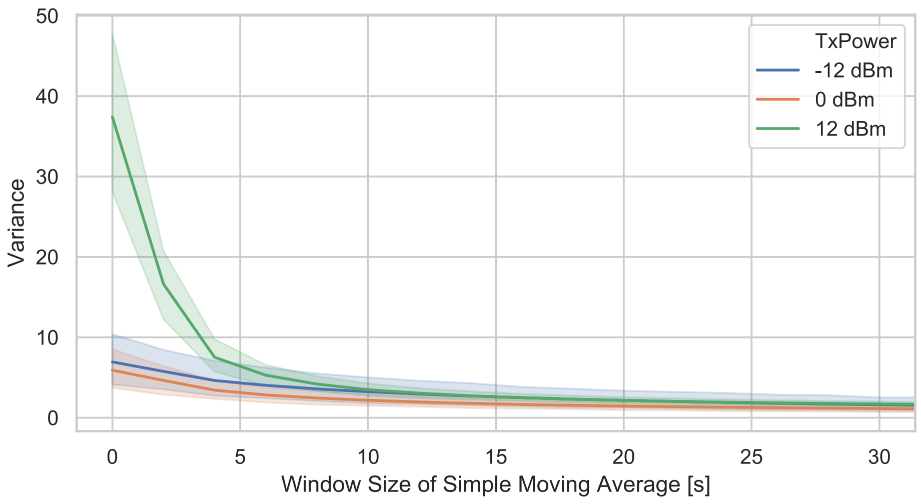

Fig. 3. Reduction of RSSI variance by simple moving average with 95% confidence interval.

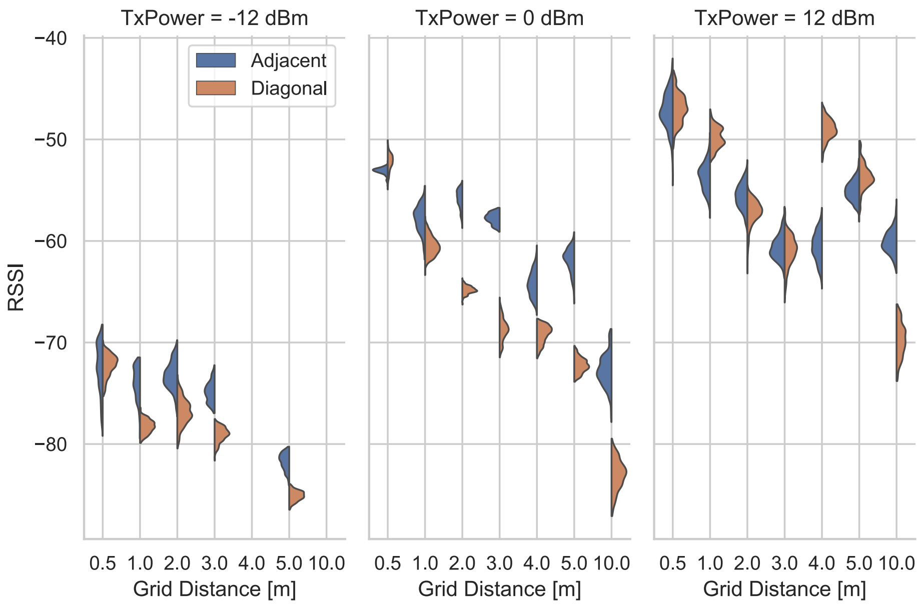

Fig. 4. The difference of RSSI distribution between the diagonal node and the adjacent node.

sensors placed in a regular grid topology with no obstacles between sensors. We define *adjacent* sensors as those within the grid distance d from the target sensor, and *diagonal* sensors are those with a distance of $\sqrt{2}d$. For a sensor, the direct way of distinguishing neighbor sensors from the other sensor is by comparing the collected RSSI value to find out the variance. Fig. 4 plots the RSSI distributions with different grid distances d, and *Adjacent* and *Diagonal* indicate the RSSI values between adjacent sensors and diagonal sensors, respectively. For example, when the grid distance is 2.0 m, *Adjacent* and *Diagonal* show the RSSI distributions at greater than 2.0 m and $2.0\sqrt{2}$ meters distances.

In general, when the grid distance is short, e.g., less than 1 m, it becomes difficult to differentiate between these two distributions due to the overlapping signal, and the difference in the RSSI values between the adjacent node and others is small. Additionally, the increase in sensor distance is accompanied by increasing variance, which could be used as a clear sign to distinguish these two correlated sensors once the mean values of the two distributions are sufficiently separated.

On the other hand, when TxPower is 12 dBm, the mean values of the RSSI vary as the grid distance increases due to the effects of factors such as reflected waves. Therefore, in the following experiments, we fix TxPower level at 0 dBm when the distance between each pair of sensors is greater than 3 m. Note that in most of the sensing environment, simply using a threshold RSSI value for localization is insufficient since it is difficult to ensure that the established WSN is designed in a perfect grid-based manner, as the potential noise in the collected RSSI signal may be considerable. In the following subsections, we evaluate the performance of the proposed scheme in a realistic sensing environment in which unpredictable radio interference is expected.

5.2. *Experiments on area localization*

5.2.1. *Experimental setup*

In this experiment, we established a WSN consisting of 9 sensors. As shown in Fig. 5, the topology here is not in a rigid polygrid, and the placement of sensors was determined by the actual space availability. Accordingly, we divided the WSN into 9 areas, namely, A1 to A9 in Fig. 5, and each sensor was located in one of these areas. Note that physical obstacles including office appliances and staffs, and interferences caused by Wi-Fi APs, smartphones and PCs, exist within the area covered by WSNs.

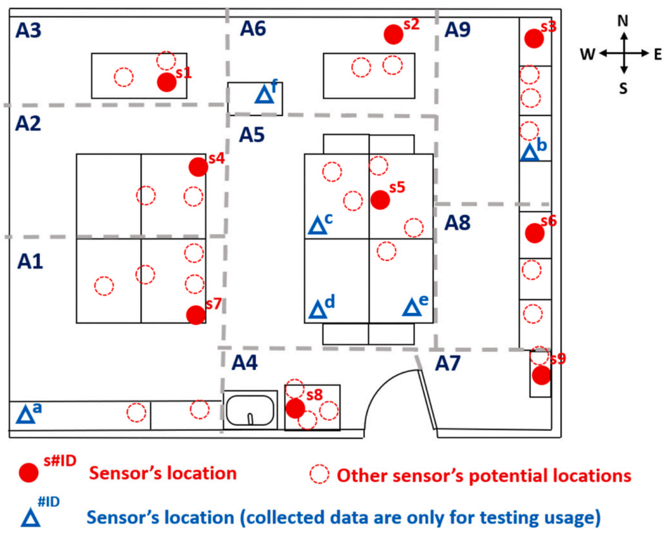
A1
A2
A3
A4
A5
A6
A7
A8
A9
s1
s2
s3
s4
s5
s6
s7
s8
s9
a
b
c
d
e
f
N
W
E
S
s#ID Sensor's location
Other sensor's potential locations
#ID Sensor's location (collected data are only for testing usage)

Fig. 5. Sensor placement information for the evaluation.

Table 2
Deep neural network architecture.

Parameter	Layer ID			
	Layer1	Layer2	Layer3	Layer4
Number of neurons	128	1024	128	64
Activation function	ReLU	sigmoid	ReLU	ReLU

We randomly assigned a sensor to one of the areas in the WSN. The RSSI data used to train the DNN model were collected through 72 different combinations of sensor assignments in the respective area. It is worth pointing out that the area was not directly divided based on physical boundaries such as walls or partitions, the division of area considered the availability of space and locations of the electrical appliances to be controlled in the given sensing environment. Each combination of sensor positions was tested for 1-hour with a 1-second communication interval. Here, sensors could be placed in several different spots in each area, as illustrated by the dashed circles in Fig. 5. For each combination, the position of a sensor was randomly selected from all the candidate positions in the corresponding area.

With regard to the DNN architecture built through PyTorch,[3] Table 2 details on the number of neurons and corresponding activation functions for each neural layer. The commonly used rectified linear unit (ReLU) and sigmoid are selected as the activation function since their simplicity and ability to enable fast training (Sze et al., 2017). In summary, the designated network encompasses four hidden layers, and there are a total of 273,408 trainable parameters.

Fig. 6 plots a sample of the RSSI signal values from neighboring sensors collected by sensor s_5, the location of which is depicted in Fig. 5. The result shows that the time series of the RSSI signal fluctuates by no more than 15 dBm due to various types of interference existing in the sensing environment. As aforementioned, these fluctuations cannot simply be removed by using a uniform filter considering the diversity of sensor locations with different influences from obstacles. Hence, DNN is expected to exploit the correlation between the sensor location and collected RSSI data for accurate localization, and the following process verifies the performance of the model from different perspectives.

The collected datasets are divided into a training dataset and a testing dataset at a ratio of 8 : 2. An adaptive learning rate optimization algorithm, namely, Adam (Kingma and BaAdam, 2015), is used to train the deep network throughout the experiment. The learning rate starts from 1.0×10^{-3} with a weight decay of 5.0×10^{-4}. More information regarding the method of selecting simulation parameter can be found in (Wu et al., 2019).

[3] https://pytorch.org.

5.2.2. *Accuracy of area localization*

We validate the accuracy of the area localization of the proposed scheme using DNN, and compare the performance of this scheme with that of 6 other comparative methods. These comparative methods can be classified into two categories: Vanilla-CNN (Schmidhuber, 2015), support vector machine (SVM) (Chang and Lin, 2011), and adaptive boosting (AdaBoost) (Ji et al., 2009) are typical supervised machine learning algorithms while GridGraph (Kubo et al., 2012) and DV-hop (Chen et al., 2008) are geometry-based algorithms in which each sensor does not need to be equipped with a ranging device.

The machine learning methods can learn the area in which a sensor is located by applying the proposed scheme using the respective learning model. The two geometry-based methods do not require historical data for model training; instead, they determine whether two sensors are neighbors based on the collected RSSI information. GridGraph extracts subgraphs with a grid topology that approximates a Euclidean space and then assigns coordinates to each sensor in a decentralized manner. For DV-hop, the coordinates of each node are determined based on the distance to the anchor nodes and the hop count. Since these two geometry-based methods are designed for calculating the precise coordinates of sensors, for fairness with other area-based localization methods, the coordinate results obtained from the geometry-based methods are then converted into area results. Note that unlike the proposed scheme, the geometry-based localization methods require anchor nodes as references.

Table 3 summarizes the area localization accuracy of different methods based on the network topology shown in Fig. 5. The same datasets are utilized for the model training of the machine learning methods, i.e., DNN, vanilla-CNN, SVM and AdaBoost. For GridGraph and DV-hop, we select the sensors located at corners as anchor nodes. When the anchor number (*#Anc*) is 3, the sensors in A1, A3, and A9 are the anchors, and $\#Anc = 4$ means the anchor sensors are those placed at all four corners.

In general, the machine learning methods outperform the geometry-based methods since the former methods can better capture the correlations between RSSI data and area location, whereas the accuracy of the latter depends on the precision of determining sensor adjacency relations through variations in the RSSI signals. Our testbed is deployed inside a real office building where obstacles like furniture, wall and even walking people exist. For this reason, the collected RSSI signals are fluctuating which deteriorates the performance of geometry-based algorithm.

When the number of sensors is limited, the volume of the reliable RSSI signals to be analyzed is only limited in the geometry-based methods. Taking DV-hop as an example, all sensors are simply assigned with the same coordinates, and the location of one of the sensors (A5 in this case) will always coincide with the assigned coordinates, hence resulting in 100% accuracy for A5. Overall, machine learning-based localization has yielded more promising results in achieving satisfactory localization accuracy. The results demonstrate that DNN achieves the best performance among these methods for sensor localization in all areas since its ability to extract the complex high-level features from the input data by utilizing multiple deep layer architecture to further explore the underlying correlations among RSSI signals. Moreover, the computational time required for area localization using machine learning methods is limited as shown in Table 4. The results indicate that real-time area localization can be achieved by all the tested machine learning techniques, including the proposed DNN approach. Note that the experiment is executed on a Raspberry Pi 2 Model B, a small single-board computer that is suitable for the IoT applications discussed in this paper.

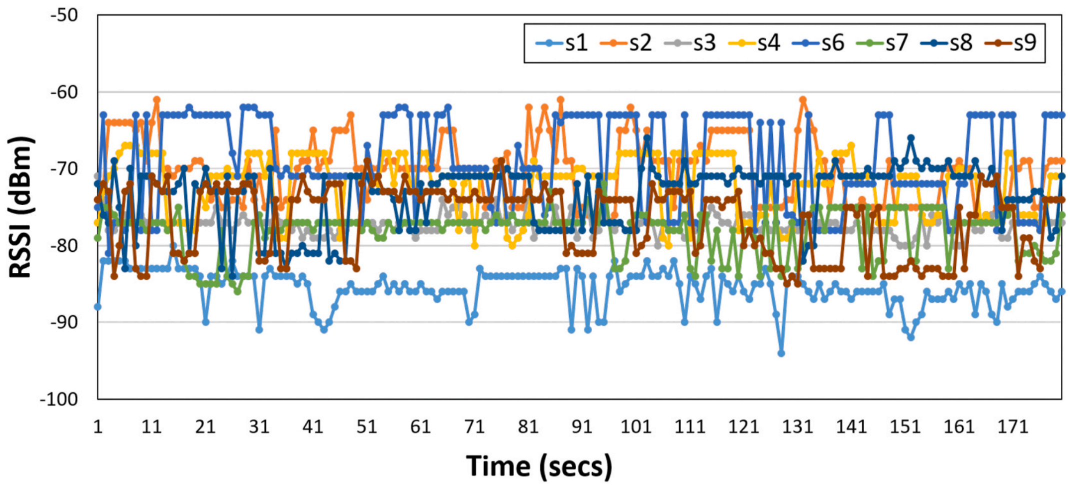

Fig. 6. A sample of collected neighbor sensor's RSSI.

Table 3
Localization accuracy of different methods.

Method	#Anc	Accuracy of area localization								
		A1	A2	A3	A4	A5	A6	A7	A8	A9
Machine learning										
DNN	0	0.97	0.97	0.97	0.91	0.94	0.96	0.89	0.95	0.96
Vanilla-CNN (Schmidhuber, 2015)	0	0.87	0.94	0.90	0.81	0.87	0.92	0.77	0.79	0.82
SVM (Chang and Lin, 2011)	0	0.82	0.83	0.84	0.87	0.94	0.82	0.83	0.64	0.79
AdaBoost (Ji et al., 2009)	0	0.39	0.27	0.32	0.22	0.42	0.52	0.34	0.11	0.54
Geometry-based										
GridGraph (Kubo et al., 2012)	3	–	0.07	–	0.00	0.41	0.32	0.00	0.13	–
DV-hop (Chen et al., 2008)	3	–	0.00	–	0.00	1.00	0.00	0.00	0.00	–
GridGraph (Kubo et al., 2012)	4	–	0.23	–	0.07	0.54	0.40	–	0.04	–
DV-hop (Chen et al., 2008)	4	–	0.00	–	0.00	1.00	0.00	–	0.00	–

#Anc: Number of anchor nodes.
-: Anchor node whose location is given in advance.

Since our proposed DNN-based scheme outperforms all other comparative methods in real-timely localizing all tested areas, we adopt DNN as the default learning model for the next simulations. We then evaluate the area localization performance from **Case 1** to **Case 4**.

5.2.3. *Results for **case 1***

Case 1 assumes that all the sensors are active for data analysis. In addition to the results shown in Table 3, we explore the performance of the model with different dataset sizes, as shown in Fig. 7. In particular, the horizontal axis represents the available dataset volume, e.g., when the normalized data volume r is 0.5, the available training data are half to those with a maximum volume of 1. It is apparent that a decline in the training data volume leads to a reduction in the test accuracy. In our experiment, if more than 67% of the data ($r = 0.67$) are available for data processing, our model can achieve at least 90% accuracy. In the following experiments, we use the maximum volume of available sensor data for verification.

Fig. 8 demonstrates the correlations between training loss and the prediction accuracy along with the increase of epochs. It is evident that the accuracy converges to approximately 96% after 50 epochs, with 99% as the maximum accuracy achieved, and the loss stabilizes after 80 epochs.

Table 4
Localization time for machine learning methods.

Method	Computation time (seconds)
DNN	3.9×10^{-4}
Vanilla-CNN (Schmidhuber, 2015)	1.9×10^{-4}
SVM (Chang and Lin, 2011)	9.2×10^{-3}
AdaBoost (Ji et al., 2009)	8.2×10^{-4}

5.2.4. *Results for **case 2***

Fig. 9 summarizes the localization accuracy for each area with different numbers of randomly selected inactive sensors. We used different sets of data reconstruction parameters Th_d and Th_c and discovered that the accuracy is highest when $Th_d = 15$ and $Th_c = 60$. The results clearly demonstrate that the proposed scheme achieves superior localization performance with an accuracy of greater than 90% overall, even when 4 of 9 sensor data are unavailable (Fig. 9d). Notably, compensation data are obtained by utilizing the historical data with high similarity, which is then beneficial to improving the overall quality of data being fed into the DNN prediction model. Note that even if the on-site sensors are replaced with new ones, our scheme can still produce a satisfactory result by replacing the previous sensor data with the new data in the input datasets of the prediction model without requiring any other updates.

5.2.5. *Results for **case 3** and **case 4***

These two cases are related to the situation in which a new sensor is placed in the existing sensor networks for maintenance operation, such as sensor replacement. Fig. 10 illustrates the localization accuracy for the positions that are not previously trained. These locations are plotted as the blue triangles in Fig. 5. The sensors identified as a to c are those located close to the pretrained sensor's positions, and the rest of the three sensors (d, e and f) are further from the pretrained positions. Even

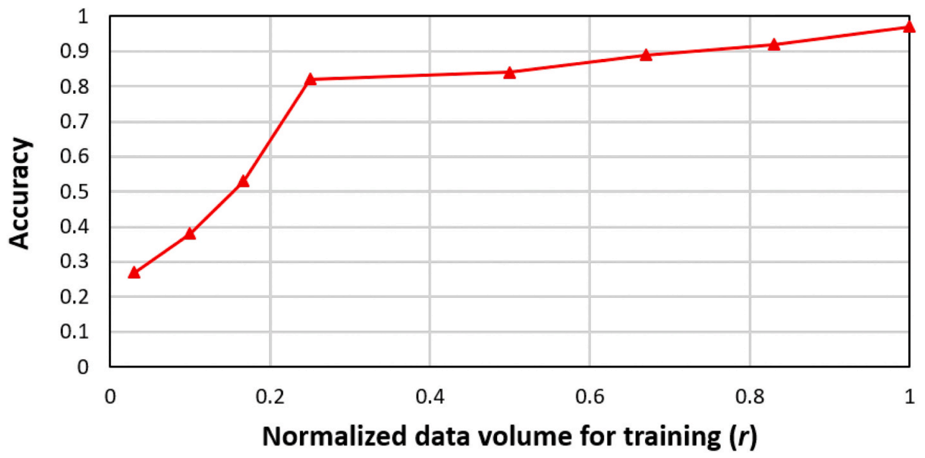

Fig. 7. Prediction accuracy of the model on different volumes of datasets.

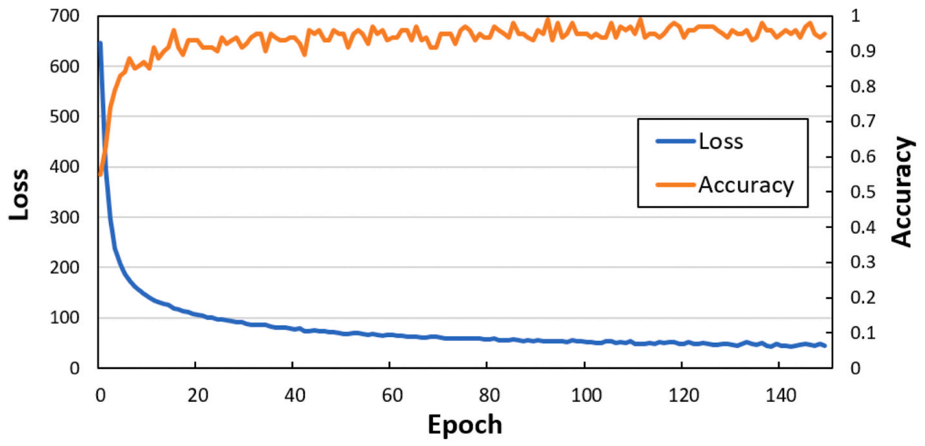

Fig. 8. Correlations between training loss and prediction accuracy.

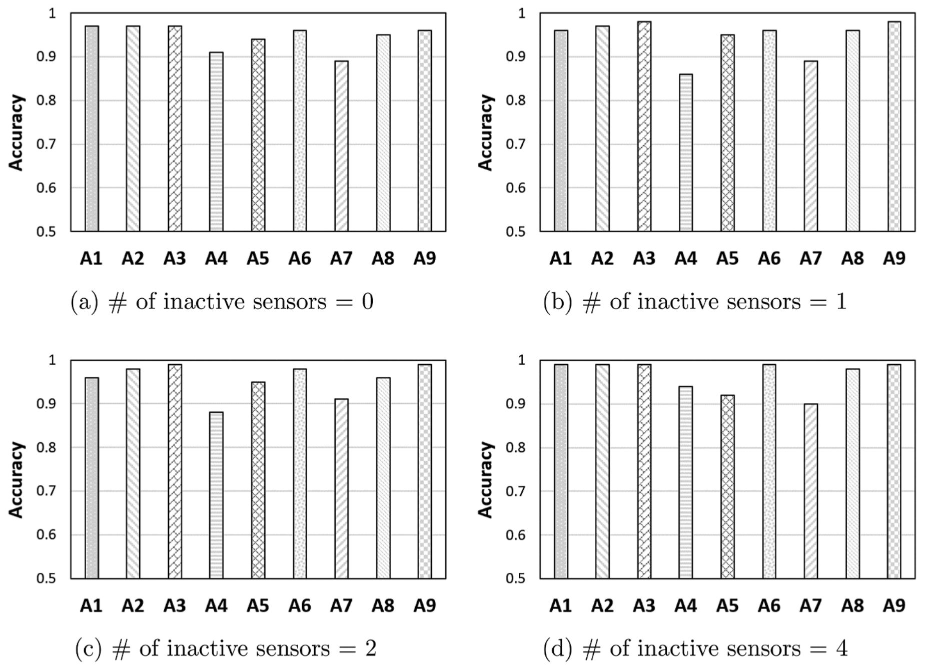

Fig. 9. Localization performance with various number of inactive sensors.

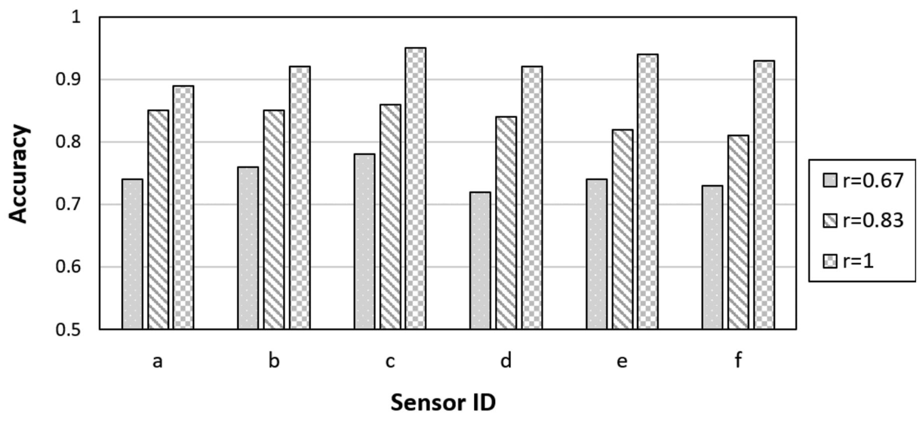

Fig. 10. Localization for sensors in different untrained spots.

for these newly added sensor positions, our scheme is still capable of making an accurate prediction at the area level. For each sensor, we test the impact of the dataset volume on the localization accuracy. Specifically, the adopted *r* values are based on the normalized data volume for data processing equal with 0.67, 0.83, and 1, corresponding to the horizontal axis of Fig. 7. The result confirms that this scheme is also applicable for locating sensors, even when the prediction model has no previous knowledge of the exact positions of these sensors. The greater the amount of data available for training is, the better the localization performance of the proposed scheme. Note that since the border of each area is based on the actual obstacle conditions of the sensing environment such as room space divisions or furniture placement, for sensors located in the same area, similar trends in the RSSI data can be expected, thereby not deteriorating the performance of the DNN model. Therefore, a suitable machine learning method with a feasible data processing scheme is necessary for obtaining satisfactory accuracy in area localization.

6. Conclusion

In this paper, we propose a novel RSSI-based area localization scheme for pervasive sensing systems using DNN. Our scheme utilizes DNN to exploit the correlation between the time series of the RSSI signal and sensor locations. The prediction model generated by DNN can be utilized for area localization in many real-world scenarios. In addition, a data reconstruction algorithm is employed for the case in which data loss occurs in the obtained datasets. Extensive experiments performed with real-world WSN testbed verify the superiority of our scheme. The DNN-based scheme can outperform all other comparative methods in terms of area localization accuracy while its computation time is acceptable for real-time localization. Moreover, our scheme can maintain a relatively high localization accuracy even when (1) data from some of the inactive sensors are unavailable, (2) sensors are not placed at the exact pretrained position during the sensor placement. Our future works will include studying the approach to efficiently processing data collected from large-scale pervasive sensing systems and applying our scheme in other WSN applications.

CRediT authorship contribution statement

Zhishu Shen: Conceptualization, Methodology, Validation, Writing - original draft, Writing - review & editing. **Tiehua Zhang:** Conceptualization, Methodology, Validation, Writing - original draft, Writing - original draft, Writing - review & editing. **Atsushi Tagami:** Supervision, Validation, Writing - original draft, Writing - review & editing. **Jiong Jin:** Supervision, Writing - review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work was supported in part by “Research and Development of Innovative Network Technologies to Create the Future (No. 19103)”, the Commissioned Research of National Institute of Information and Communications Technology (NICT), Japan, and in part by the Australian Research Training Program Scholarship.

References

Abouzar, P., Michelson, D.G., Hamdi, M., 2016. RSSI-based distributed self-localization for wireless sensor networks used in precision agriculture. IEEE Trans. Wireless Commun. 15, 6638–6650.
Al Qathrady, M., Helmy, A., 2017. Improving BLE distance estimation and classification using TX power and machine learning: a comparative analysis. In: Proceedings of the ACM International Conference on Modelling, Analysis and Simulation of Wireless and Mobile Systems, pp. 79–83.
DIC Develops Wireless Sensor Which Realizes Soft Electronics, 2019. http://www.dic-global.com/en/release/2019/20190201_01.html.
Cerpa, A., Elson, J., Estrin, D., Girod, L., Hamilton, M., Zhao, J., 2001. Habitat monitoring: application driver for wireless communications technology. Comput. Commun. Rev. 31, 20–41.
Chang, C.-C., Lin, C.-J., 2011. LIBSVM: a library for support vector machines. ACM Transactions on Intelligent Systems and Technology 2, 27:1–27:27.
Chen, H., Sezaki, K., Deng, P., So, H., 2008. An improved DV-Hop localization algorithm for wireless sensor networks. In: Proceedings of the IEEE Conference on Industrial Electronics and Applications, pp. 1557–1561.
Chen, D., Shin, K.G., Jiang, Y., Kim, K.-H., 2017. Locating and tracking BLE beacons with smartphones. In: Proceedings of the International Conference on Emerging Networking EXperiments and Technologies, pp. 263–275.
Ding, J., Chandra, R., 2019. Towards low cost soil sensing using Wi-Fi. In: Proceedings of the ACM International Conference on Mobile Computing and Networking, pp. 39:1–39:16.
Drawil, N.M., Amar, H.M., Basir, O.A., 2013. GPS localization accuracy classification: a context-based approach. IEEE Trans. Intell. Transport. Syst. 14, 262–273.
Dyo, V., Ellwood, S.A., Macdonald, D.W., Markham, A., Mascolo, C., Pásztor, B., Scellato, S., Trigoni, N., Wohlers, R., Yousef, K., 2010. Evolution and sustainability of a wildlife monitoring sensor network. In: Proceedings of the ACM Conference on Embedded Networked Sensor Systems, pp. 127–140.
Garcia-Sanchez, A.-J., Garcia-Sanchez, F., Losilla, F., Kulakowski, P., Garcia-Haro, J., Rodríguez, A., López-Bao, J.-V., Palomares, F., 2010. Wireless sensor network deployment for monitoring wildlife passages. Sensors 10, 7236–7262.
Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. The MIT Press.
Ibrahim, M., Torki, M., ElNainay, M., 2018. CNN based indoor localization using RSS time-series. In: Proceedings of the IEEE Symposium on Computers and Communications, pp. 1044–1049.
Ji, Z., Hui, Z., Saharon, R., Trevor, H., 2009. Multi-class AdaBoost. Stat. Interface 2, 349–360.
Kasuga, T., Yagyu, H., Uetani, K., Koga, H., Nogi, M., 2019. Return to the soil: nanopaper sensor device for hyperdense sensor networks. ACS Appl. Mater. Interfaces 11, 43488–43493.

Kingma, D.P., Ba, J., Adam, 2015. A method for stochastic optimization. In: Proceedings of the International Conference for Learning Representations, pp. 1269–1272.
Kubo, T., Tagami, A., Hasegawa, T., Hasegawa, T., Walrand, J., 2012. Range-free localization using grid graph extraction. In: Proceedings of the IEEE International Conference on Network Protocols, pp. 1–11.
Lombardo, L., Corbellini, S., Parvis, M., Elsayed, A., Angelini, E., Grassini, S., 2018. Wireless sensor network for distributed environmental monitoring. IEEE Trans. Instrum. Measur. 67, 1214–1222.
Mackey, A., Spachos, P., 2017. Performance evaluation of beacons for indoor localization in smart buildings. In: Proceedings of the IEEE Global Conference on Signal and Information Processing, pp. 823–827.
Sadowski, S., Spachos, P., 2018. RSSI-based indoor localization with the Internet of Things. IEEE Access 6, 30149–30161.
Schmidhuber, J., 2015. Deep learning in neural networks: an overview. Neural Network. 61, 85–117.
Shen, H., Ding, Z., Dasgupta, S., Zhao, C., 2014. Multiple source localization in wireless sensor networks based on time of arrival measurement. IEEE Trans. Signal Process. 62, 1938–1949.
Shen, Z., Zhang, T., Jin, J., Yokota, K., Tagami, A., Higashino, T., 2019a. ICCF: an information-centric collaborative fog platform for building energy management systems. IEEE Access 7, 40402–40415.
Shen, Z., Yokota, K., Tagami, A., Higashino, T., 2019b. Energy-efficient activation/inactivation strategy for long-term iot network operation. In: Proceedings of the IEEE Conference on Ubiquitous Intelligence Computing, pp. 747–752.
Sun, Y., Ho, K.C., Wan, Q., 2019. Solution and analysis of TDOA localization of a near or distant source in closed form. IEEE Trans. Signal Process. 67, 320–335.
Sze, V., Chen, Y., Yang, T., Emer, J.S., 2017. Efficient processing of deep neural networks: a tutorial and survey. Proc. IEEE 105, 2295–2329.
Tang, P., Chow, T.W.S., 2016. Wireless sensor-networks conditions monitoring and fault diagnosis using neighborhood hidden conditional random field. IEEE Trans. Indust. Inform. 12, 933–940.
Tegou, T., Kalamaras, I., Tsipouras, M., Giannakeas, N., Votis, K., Tzovaras, D., 2019. A low-cost indoor activity monitoring system for detecting frailty in older adults. Sensors 19.
Vishwanath, A., Chandan, V., Saurav, K., 2019. An IoT-based data driven precooling solution for electricity cost savings in commercial buildings. IEEE Internet of Things J. 6, 7337–7347.
Wang, T., Peng, Z., Liang, J., Wen, S., Bhuiyan, M.Z.A., Cai, Y., Cao, J., 2016. Following targets for mobile tracking in wireless sensor networks. ACM Trans. Sens. Netw. 12, 31:1–31:24.
Wang, X., Gao, L., Mao, S., Pandey, S., 2017. CSI-based fingerprinting for indoor localization: a deep learning approach. IEEE Trans. Veh. Technol. 66, 763–776.
Wu, Y., Liu, L., Bae, J., Chow, K., Iyengar, A., Pu, C., Wei, W., Yu, L., Zhang, Q., 2019. Demystifying learning rate policies for high accuracy training of deep neural networks. In: Proceedings of the IEEE International Conference on Big Data, pp. 1971–1980.
Yang, Z., Zhou, Z., Liu, Y., 2013. From RSSI to CSI: indoor localization via channel response. ACM Comput. Surv. 46, 25:1–25:32.
Ye, D., Zhang, M., 2018. A self-adaptive sleep/wake-up scheduling approach for wireless sensor networks. IEEE Trans. Cybern. 48, 979–992.
Zafari, F., Gkelias, A., Leung, K.K., 2019. A survey of indoor localization systems and technologies. IEEE Commun. Surv. Tutor. 21, 2568–2599.
Zhang, W., Liu, K., Zhang, W., Zhang, Y., Gu, J., 2016. Deep neural networks for wireless localization in indoor and outdoor environments. Neurocomputing 194, 279–287.
Zhang, Z., Mehmood, A., Shu, L., Huo, Z., Zhang, Y., Mukherjee, M., 2018. A survey on fault diagnosis in wireless sensor networks. IEEE Access 6, 11349–11364.
Zhao, Z., Kuendig, S., Carrera, J., Carron, B., Braun, T., Rolim, J., 2017. Indoor location for smart environments with wireless sensor and actuator networks. In: Proceedings of the IEEE Conference on Local Computer Networks, pp. 535–538.

Zhishu Shen received the B.E. degree from the School of Information Engineering at the Wuhan University of Technology, Wuhan, China, in 2009, and the M.E. and Ph.D. degrees in Electrical and Electronic Engineering and Computer Science from Nagoya University, Japan, in 2012 and 2015, respectively. He joined KDDI Corp., Japan in 2015. He is currently a research engineer at Future Communication System Laboratory, KDDI Research, Inc., Japan. His major interests include network design and optimization, data learning, and Internet of Things.

Tiehua Zhang received the B.S. degree in Computer Science from Jilin University, China in 2013, and M.E. in Information Technology from University of Melbourne, Australia in 2015. Between 2015 and 2017, he was a Software Engineer focusing on industrial projects and solutions. He is currently pursuing the Ph.D. degree in the School of Software and Electrical Engineering, Swinburne University of Technology, Australia. His research interests include Internet of Things and Edge Intelligence.

Atsushi Tagami received the M.E. and Ph.D degrees in Computer Science from Kyushu University, Japan in 1997 and 2010, respectively. He joined KDDI R&D Laboratories Inc. in 1997, where he has been engaged in research and development on performance measurement of communication networks and overlay networking. He is currently a senior manager at Future Communication System Laboratory of KDDI Research, Inc., Japan.

Jiong Jin received the B.E. degree with First Class Honours in Computer Engineering from Nanyang Technological University, Singapore, in 2006, and the Ph.D. degree in Electrical and Electronic Engineering from the University of Melbourne, Australia, in 2011. He is currently a Senior Lecturer in the School of Software and Electrical Engineering, Faculty of Science, Engineering and Technology, Swinburne University of Technology, Melbourne, Australia. Prior to it, he was a Research Fellow in the Department of Electrical and Electronic Engineering at the University of Melbourne from 2011 to 2013. His research interests include network design and optimization, fog and edge computing, robotics and automation, Internet of Things and cyber-physical systems as well as their applications in smart manufacturing, smart transportation and smart cities.