Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 6660990, 12 pages
https://doi.org/10.1155/2021/6660990

Research Article

Wi-Fi Fingerprint-Based Indoor Mobile User Localization Using Deep Learning

Junhang Bai,[1] **Yongliang Sun**,[1] **Weixiao Meng**,[2] **and Cheng Li**[3]

[1]*School of Computer Science and Technology, Nanjing Tech University, Nanjing 211816, China*
[2]*School of Electronics and Information Engineering, Harbin Institute of Technology, Harbin 150001, China*
[3]*Faculty of Engineering and Applied Science, Memorial University, St. Johns, NL, Canada A1B 3X5*

Correspondence should be addressed to Yongliang Sun; syl_peter@163.com

Received 12 November 2020; Revised 11 December 2020; Accepted 18 December 2020; Published 8 January 2021

Academic Editor: Xin Liu

In recent years, deep learning has been used for Wi-Fi fingerprint-based localization to achieve a remarkable performance, which is expected to satisfy the increasing requirements of indoor location-based service (LBS). In this paper, we propose a Wi-Fi fingerprint-based indoor mobile user localization method that integrates a stacked improved sparse autoencoder (SISAE) and a recurrent neural network (RNN). We improve the sparse autoencoder by adding an activity penalty term in its loss function to control the neuron outputs in the hidden layer. The encoders of three improved sparse autoencoders are stacked to obtain high-level feature representations of received signal strength (RSS) vectors, and an SISAE is constructed for localization by adding a logistic regression layer as the output layer to the stacked encoders. Meanwhile, using the previous location coordinates computed by the trained SISAE as extra inputs, an RNN is employed to compute more accurate current location coordinates for mobile users. The experimental results demonstrate that the mean error of the proposed SISAE-RNN for mobile user localization can be reduced to 1.60 m.

1. Introduction

As the development of the fifth-generation (5G) networks, the key technologies of 5G are paving the way for applications of Internet of Things (IoT) [1–3], for they can greatly improve the network connectivity [1], spectrum efficiency [3], and so on. Since location-based service (LBS) is essential to the IoT and people's demand for LBS has increased rapidly, LBS has attracted extensive attention [4, 5]. Although satellite-based navigation and positioning systems like global positioning system (GPS) and BeiDou navigation satellite system (BDS) can meet most LBS requirements outdoors, these outdoor localization and navigation systems are not suitable for indoor applications owing to the signal attenuations caused by the blockage of the buildings [6, 7]. Meanwhile, people daily spend most of their time indoors. Therefore, indoor localization has been extensively researched in the past years because of its application and commercial potentials [8].

So far, people have developed various indoor localization methods using infrared, ultrasound, radio frequency identification (RFID), ZigBee, Bluetooth, ultra wideband (UWB), and Wi-Fi [9–12]. Among them, the indoor localization method using Wi-Fi has become a research hotspot because of its low cost, widely deployed infrastructure, and so on. Compared with the Wi-Fi localization using time of arrival (TOA), time difference of arrival (TDOA), and angle of arrival (AOA) [12–14], the Wi-Fi fingerprinting localization using received signal strength (RSS) has been favored because it does not require extra hardware and has a satisfactory performance under nonline-of-sight (NLOS) environments.

Generally, a basic Wi-Fi fingerprint-based localization method can be divided into two phases: the offline phase and online phase [15]. In the offline phase, RSS vectors from deployed Wi-Fi access points (APs) are recorded as fingerprints at a number of location-known reference points (RPs) to establish a fingerprint database, namely, radio map. In the online phase, after an online RSS vector is measured by a user's terminal device, similarities between the online RSS vector and the RSS vectors in the radio map can be calculated to select RPs for localization, or the online

RSS vector is input into a machine learning-based localization algorithm trained with the radio map in the offline phase to calculate the user's location coordinates. However, indoor radio propagation is time-varying and easily affected by multipath effect, shadowing effect, and environmental dynamics, which could degrade the performance of Wi-Fi fingerprinting localization, let alone the localization for mobile users with only limited RSS data available. Therefore, it will be a serious challenge for the Wi-Fi fingerprint-based indoor mobile user localization to achieve high localization accuracy.

Recently, various deep learning algorithms have been proposed, and some of them have been applied in Wi-Fi fingerprinting localization for performance improvement like deep belief network (DBN) [16], convolutional neural network (CNN) [17], and stacked autoencoder (SAE) [18]. As an unsupervised deep learning algorithm, a stacked sparse autoencoder (SSAE) can learn high-level feature representations of data efficiently [19]. Because the dimensions of the hidden layers of the SSAE could be greater than the dimension of its input layer, it can be applied to learn RSS features in the scenarios with a few deployed APs. Moreover, as a supervised deep learning algorithm, a recurrent neural network (RNN) is powerful for processing sequential correlation data and is able to improve the localization accuracy for mobile users. Therefore, we propose a Wi-Fi fingerprint-based indoor mobile user localization method that integrates a stacked improved sparse autoencoder (SISAE) and an RNN to achieve high localization accuracy. The main contributions of this study can be summarized as follows:

(1) We propose a Wi-Fi fingerprint-based indoor mobile user localization method using deep learning. We use the stacked encoders of three improved sparse autoencoders to obtain high-level feature representations of RSS vectors. Then, we add an output layer to the stacked encoders to construct an SISAE for localization. In addition, to process the spatial correlation locations of mobile users, we integrate the SISAE with an RNN to calculate more accurate location coordinates

(2) We propose an SISAE fingerprinting algorithm. We first improve the sparse autoencoder by adding an activity penalty term in its loss function to control the neuron outputs in the hidden layer and stack the encoders of three improved sparse autoencoders. Then, we add a logistic regression layer as the output layer to the stacked encoders to construct an SISAE, which takes the locations of RPs as its expected outputs for supervised fine-tuning. The proposed SISAE fingerprinting algorithm has a superior localization performance

(3) We propose an RNN tracking algorithm to improve the localization accuracy for mobile users. We construct the RNN by using the same network structure as well as the weights and biases of the trained SISAE and also by adding the previous ground-truth location coordinates of a trajectory as extra inputs for the RNN training. In the online phase, the previous location coordinates computed by the SISAE are used as the extra inputs of the RNN for current localization, and therefore, the localization performance can be improved

The remainder of this paper is organized as follows. Section 2 summarizes the related works of fingerprinting localization using deep learning. Section 3 describes our proposed localization method that integrates the SISAE and RNN. Section 4 demonstrates the experimental setup, parameter settings, and experimental results and analyses. Finally, we conclude this work in Section 5.

2. Related Works

In the last two decades, many famous Wi-Fi fingerprint-based indoor localization systems have been developed, such as RADAR [20], Nibble [21], and Horus [22]. At the same time, many fingerprinting localization algorithms have been presented, for example, K-nearest neighbors (KNN) [23], maximum likelihood probability [24], and neural network [25]. Because deep learning algorithms outperform many traditional machine learning algorithms, researchers have focused their attention on the deep learning-based fingerprinting and tracking algorithms as follows.

2.1. Deep Learning-Based Fingerprinting Algorithms. Some researchers used deep reinforcement learning (DRL) algorithms for fingerprinting localization. Li et al. [26] proposed an unsupervised wireless localization method based on DRL. The method extracted landmark data from unlabeled RSS automatically and eased the demand for retraining the DRL. Mohammadi et al. [27] proposed a semisupervised model using a variational autoencoder and DRL and applied the model to indoor localization in a smart city scenario. Meanwhile, some researchers utilized deep learning approaches for RSS feature extraction. Le et al. [16] proposed a DBN constructed by stacked restricted Boltzmann machines for fingerprint feature extraction. The network could reduce the workload of fingerprint collection and maintain the localization accuracy. Shao et al. [17] combined Wi-Fi and magnetic fingerprints to generate a new image and utilized the convolutional windows to extract the Wi-Fi features automatically. Hsieh et al. [28] presented a deep neural network (DNN) implemented with one-dimensional CNN, which was used to extract hierarchical features and to reduce the network complexity. Moreover, some researchers preferred autoencoders (AEs) for the RSS feature extraction. Khatab et al. [29] combined an AE and a deep extreme learning machine to improve the localization performance by using the AE to extract high-level feature representations of RSS data. Song et al. [18] employed an SAE for dimension reduction of RSS data and combined it with a one-dimensional CNN to improve the localization performance. In [30], a time-reversal fingerprinting localization method with RSS calibration was proposed. The method used the amplitude-AE and phase-AE to calibrate the online RSS measurements. Zhang et al. [31] first utilized a DNN structure

trained by a stacked denoising autoencoder for coarse localization. After the DNN, a hidden Markov model was used for fine localization. However, these algorithms based on undercomplete AE were applied for dimension reduction of RSS features [29] and may not be suitable for the scenarios with a few deployed APs.

2.2. Deep Learning-Based Tracking Algorithms. Bai et al. [32] proposed a real-time localization system that consisted of two RNNs. The first RNN computed the locations using the historical observation information, and the second RNN refined the locations with the historical computed locations. In [33], an RNN solution for localization using the correlation among the RSS measurements in a trajectory was proposed. Different RNN structures were also compared and analyzed by the authors. Reference [34] proposed a novel RNN for indoor tracking. The RNN estimated the orientation and velocity using inertial measurement units of terminal devices as well as learned and predicted a human movement model. Zhang et al. [35] embedded a particle filter algorithm into an RNN to realize real-time map matching. They also employed a CNN to learn map information for particle weight computation. Chen et al. [36] proposed a deep long short-term memory (LSTM) algorithm to learn high-level representations of the extracted RSS features and conducted the experiments in two different scenarios to verify the effectiveness of the algorithm. Tariq et al. [37] tested and analyzed several neural networks for localization and concluded that the LSTM could have a comparable performance with the lowest processing effort and fewest network parameters. Among these deep learning-based tracking algorithms, researchers concentrated on the spatial correlation locations of mobile users to improve the localization performance.

In this study, we propose an SISAE fingerprinting algorithm whose dimensions of the hidden layers can be freely set and not be limited by its input dimension. Also, based on the trained SISAE, we add the previous location coordinates of mobile users as extra inputs to construct an RNN. We integrate the SISAE with the RNN to improve the mobile user localization accuracy. To the best of our knowledge, so far, neither have the SSAE and SISAE been used for fingerprinting localization nor has the integration of the SISAE and RNN been proposed for mobile user localization.

3. Proposed Localization Method

Similar to the basic Wi-Fi fingerprinting localization method, our proposed localization method also has two phases: the offline phase and online phase shown in Figure 1. In the offline phase, we first collect the RSS vectors at selected RPs to establish the fingerprint database 1 that is the radio map. We train three improved sparse autoencoders with the RSS vectors of RPs and stack their encoders to obtain high-level feature representations of RSS vectors. Also, we add a logistic regression layer with two neurons that takes the location coordinates of RPs as its expected outputs to the stacked encoders to construct an SISAE for fingerprinting localization, and then we fine-tune the SISAE by using the back propagation (BP) algorithm. Meanwhile, we also consecutively collect extra RSS vectors along a trajectory with known locations to establish the fingerprint database 2 for the RNN training. In this process, we construct the RNN by using the same network structure as well as the weights and biases of the trained SISAE and also by adding the previous ground-truth location coordinates of the trajectory as extra inputs. Then, we train the weights of the extra inputs with the fingerprint database 2. In the online phase, after a mobile user's terminal device measures an RSS vector, the RSS vector is input into the SISAE to compute the current location coordinates of the mobile user, which can be used as the extra inputs of the RNN to calculate the following location coordinates of the mobile user. Also, with the RSS vector and the previous location coordinates calculated by the SISAE, more accurate current location coordinates of the mobile user can be calculated by the RNN.

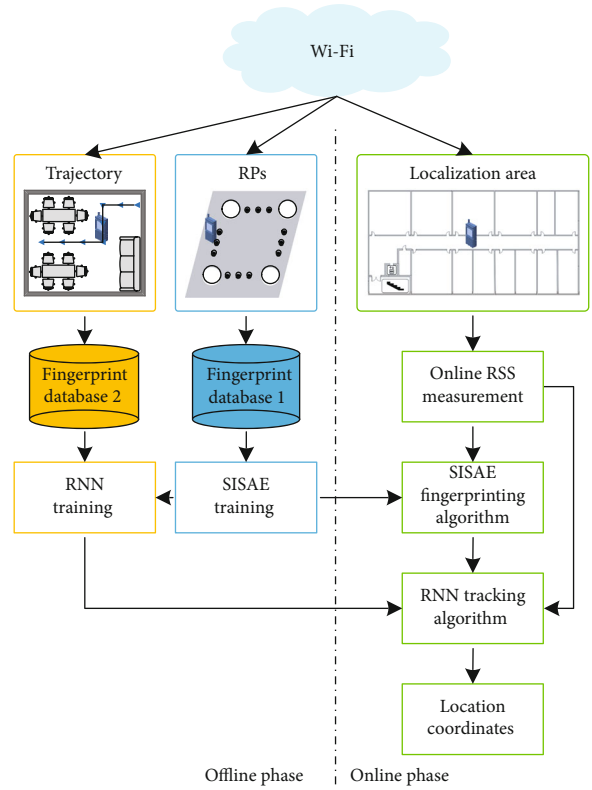


Figure 1: The framework of the proposed localization method.

3.1. SISAE Fingerprinting Algorithm

3.1.1. Unsupervised Feature Learning. In order to improve the localization accuracy, traditional AEs are usually used for feature extraction and dimension reduction of RSS data, which forces the network to learn compressed representations of the inputs [19]. However, when only a few APs are deployed in a localization area, it means the dimension of RSS vectors is limited. Thus, the dimension reduction of the traditional AEs may directly affect the learning ability and network stability. To solve this problem and improve the network performance, we use an SISAE to increase the number of neurons

in each hidden layer, which is not limited by the input dimension.

Regarding the principle of a traditional AE, it is a neural network with a single hidden layer and can be divided into two parts: an encoder function $f(x)$ and a decoder function $g(x)$. The outputs of the AE are expected to be approximate to its inputs through training, which can be denoted by $g[f(x)] \approx x$. The jth neuron output h_j in the hidden layer and the ith neuron output $\hat{x}_i$ in the output layer of the AE can be computed by:

$$\begin{cases} h_j = f\left(\sum_{i=1}^{l} w_{ij}x_i + b_j^{(1)}\right), j = 1, 2, \cdots, m, \\ \hat{x}_i = g\left(\sum_{j=1}^{m} w_{ij}^{\mathrm{T}}h_j + b_i^{(2)}\right), i = 1, 2, \cdots, l, \end{cases} \tag{1}$$

where x_i is the ith input of the AE, w_{ij} is the weight between the ith input of the AE and the jth neuron in the hidden layer, w_{ij}^{T} that is the transposition of w_{ij} is the weight between the jth neuron in the hidden layer and the ith output of the AE, $b_j^{(1)}$ is the bias of the jth neuron in the hidden layer, $b_i^{(2)}$ is the bias of the ith output of the AE, l is the number of the neurons in the input or output layer, and m is the number of the neurons in the hidden layer.

BP algorithm is employed to update the weights and biases of the AE to decrease the loss function J_{AE} between the inputs and outputs. The weights and biases of the AE are optimized as the loss function J_{AE} decreases, which is calculated by:

$$J_{\mathrm{AE}} = \frac{1}{n}\sum_{k=1}^{n} \|\hat{\boldsymbol{x}}_k - \boldsymbol{x}_k\|_2, \tag{2}$$

where $\hat{\boldsymbol{x}}_k$ is the kth output vector, $\boldsymbol{x}_k$ is the kth expected output vector that is also the kth input vector, n is the number of input vectors or output vectors, and $\|\cdot\|_2$ is the l_2 -norm.

A sparse autoencoder that has a sparsity constraint in its hidden layer is an improved AE. The sparse autoencoder is able to learn sparse representations of the RSS data, and the dimension of the hidden layer can be increased even greater than the dimension of the input layer. The structure of a sparse autoencoder is shown in Figure 2. Usually, we define a neuron in the hidden layer as "activated" if its output is approximate to 1 and as "inactivated" if its output is approximate to 0. The sparsity constraint can be interpreted as the neurons in the hidden layer are "inactivated" most of the time.

Firstly, a sparsity penalty term is added to the loss function of the AE to constrain the sparsity. We calculate the average activation p_j' of all the n input vectors of the jth neuron in the hidden layer of a sparse autoencoder by:

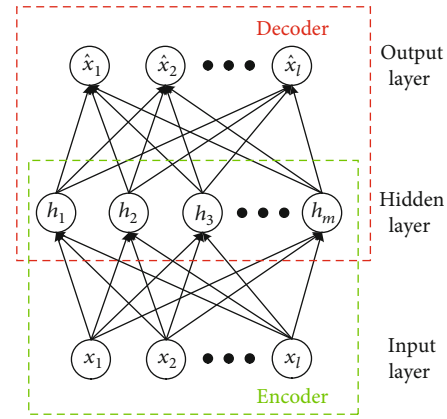


Figure 2: The structure of a sparse autoencoder.

$$p_j' = \frac{1}{n}\sum_{k=1}^{n} h_{kj}, \tag{3}$$

where h_{kj} is the jth neuron output in the hidden layer of the kth input vector.

A sparsity constant p approximate to 0 is used as a criterion for the sparsity constraint. The sparsity penalty term J_{KL} based on the Kullback-Leibler (KL) divergence is calculated by:

$$J_{\mathrm{KL}} = \sum_{j=1}^{m}\left(p\log\frac{p}{p_j'} + (1-p)\log\frac{1-p}{1-p_j'}\right). \tag{4}$$

In addition, a weight decay term W is added to the loss function to prevent overfitting, which can be calculated by:

$$W = \sum_{i=1}^{l}\sum_{j=1}^{m} w_{ij}^2. \tag{5}$$

So the loss function J_{LOSS} of the sparse autoencoder is calculated by:

$$J_{\mathrm{LOSS}} = J_{\mathrm{AE}} + \alpha J_{\mathrm{KL}} + \beta W, \tag{6}$$

where α is a weighted factor to control the contribution of the sparsity penalty term and β is a weighted factor to control the contribution of the weight decay term.

In this paper, we improve the sparse autoencoder by adding an activity penalty term A to the loss function J_{LOSS} to control the neuron outputs in the hidden layer. The activity penalty term A is calculated by:

$$A = \frac{1}{n}\sum_{k=1}^{n}\sum_{j=1}^{m} h_{kj}^2. \tag{7}$$

Therefore, the loss function J'_{LOSS} of the improved sparse autoencoder is calculated by:

$$J'_{\mathrm{LOSS}} = J_{\mathrm{LOSS}} + \lambda A, \tag{8}$$

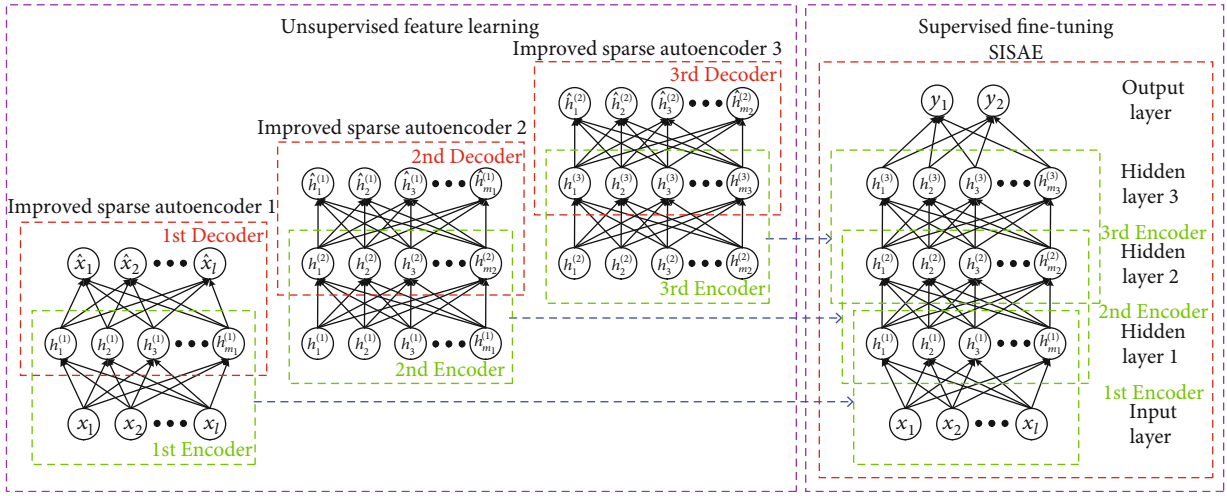


FIGURE 3: The construction of the proposed SISAE structure.

where λ is a weighted factor to control the contribution of the activity penalty term.

An SISAE can be composed of the encoders of several improved sparse autoencoders and an output layer. The encoder outputs of the first trained improved sparse autoencoder can be taken as the encoder inputs of the second improved sparse autoencoder for its training, so are the trainings of the other improved sparse autoencoders. Then, the encoders of these improved sparse autoencoders are stacked to learn high-level feature representations of data efficiently.

In this study, we normalize the RSS data of each AP to [-1, 1] to improve the network performance. As shown in Figure 3, we construct an SISAE with three improved sparse autoencoders: improved sparse autoencoder 1, 2, and 3. We first train the improved sparse autoencoder 1 by taking the normalized RSS vectors of RPs as its inputs. Then, we train the improved sparse autoencoder 2 by taking the neuron outputs in the hidden layer of the trained improved sparse autoencoder 1 as the inputs of the improved sparse autoencoder 2, so is the training of the improved sparse autoencoder 3. Finally, the SISAE can be constructed by using the weights and biases of the encoders of the trained improved sparse autoencoder 1, 2, and 3 as the weights and biases of its hidden layer 1, 2, and 3, respectively, and also by adding a logic regression layer with two neurons as the output layer to the stacked encoders for the supervised fine-tuning.

3.1.2. Supervised Fine-Tuning. We take the RSS vectors of RPs as the inputs and the location coordinates of RPs as the expected outputs of the SISAE for its supervised fine-tuning. The oth neuron output y_o in the output layer of the SISAE is calculated by:

$$y_o = \sum_{j=1}^{m_3} w_{jo}^{(4)} h_j^{(3)} + b_o^{(4)}, o = 1, 2, \tag{9}$$

where $h_j^{(3)}$ is the jth neuron output in the hidden layer 3, $w_{jo}^{(4)}$ is the weight between the jth neuron in the hidden layer 3 and the oth neuron in the output layer, $b_o^{(4)}$ is the bias of the oth neuron in the output layer, and m_3 is the number of neurons in the hidden layer 3.

Then, we use the BP algorithm to fine-tune the weights and biases of the SISAE to decrease the loss function J_{SISAE}, and the loss function J_{SISAE} is calculated by:

$$J_{\text{SISAE}} = \frac{1}{n} \sum_{k=1}^{n} \|\boldsymbol{y}_k - \boldsymbol{L}_k\|_2, \tag{10}$$

where $\boldsymbol{y}_k$ is the estimated location vector in the output layer computed by the SISAE using the kth input RSS vector, and $\boldsymbol{L}_k$ is the ground-truth location vector of the kth input RSS vector. In addition, we add a dropout layer after the hidden layer 3 to prevent overfitting for the supervised fine-tuning of the SISAE.

3.2. RNN Tracking Algorithm. Considering the spatial correlation of the mobile user locations in a short time slot, we propose an RNN structure for mobile user tracking, which uses the same network structure and parameters of the trained SISAE. To construct the RNN, we also add an extra input vector $\boldsymbol{s}_{t-1}$ that represents the previous location coordinates at time $t-1$ to the hidden layer 3 at time t and initialize a weight matrix $\mathbf{u}$ for the extra input vector. In the offline phase, the ground-truth location vector at time $t-1$ is used as $\boldsymbol{s}_{t-1}$ for the RNN training. In the online phase, the location vector of a mobile user calculated by the SISAE at time $t-1$ is used as $\boldsymbol{s}_{t-1}$ for the mobile user tracking at time t. Particularly, when a mobile user enters the localization area for the first time, because there are no previous location coordinates of the user, the location coordinates computed by the SISAE are taken as the final location coordinates. After adding the extra input vector $\boldsymbol{s}_{t-1}$, the jth neuron output $h_{t,j}^{(3)}$ in the hidden layer 3 at time t can be calculated by:

$$h_{t,j}^{(3)} = f\left(\sum_{i=1}^{m_2} w_{ij}^{(3)} h_{t,i}^{(2)} + \sum_{o=1}^{2} u_{oj} s_{t-1,o} + b_j^{(3)}\right), \tag{11}$$

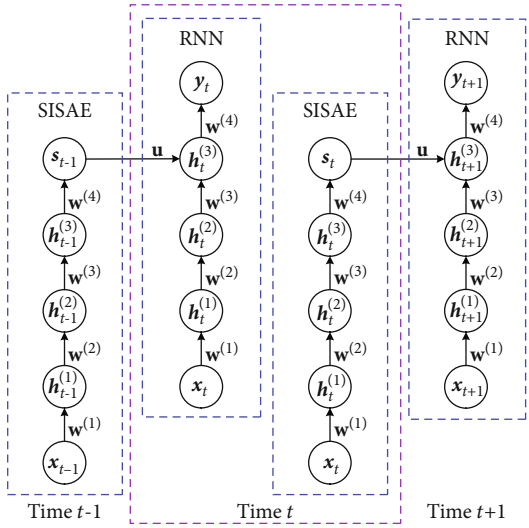


Figure 4: The proposed SISAE-RNN structure.

where $h_{t,i}^{(2)}$ is the ith neuron output in the hidden layer 2 at time t, $w_{ij}^{(3)}$ is the weight between the ith neuron in the hidden layer 2 and the jth neuron in the hidden layer 3, $s_{t-1,o}$ is the oth element of the extra input vector $\boldsymbol{s}_{t-1}$, u_{oj} is the weight between the oth element of the extra input vector $\boldsymbol{s}_{t-1}$ and the jth neuron in the hidden layer 3, $b_j^{(3)}$ is the bias of the jth neuron in the hidden layer 3, and m_2 is the number of neurons in the hidden layer 2.

The oth neuron output in the output layer of the RNN at time t is defined as $y_{t,o}$, which can be calculated using (9). The weights and biases of the trained SISAE in the RNN structure are invariant during the RNN training. In addition, we also use the BP algorithm to optimize the weight matrix $\mathbf{u}$ for the extra input vector of the RNN and add a dropout layer after the hidden layer 3 to prevent overfitting in the RNN training phase. The proposed SISAE-RNN structure is shown in Figure 4.

4. Experiment and Results

4.1. Experimental Setup. We collected all the experimental data in a real office environment. As shown in Figure 5, the experimental environment is a rectangular area with a size of 51.6 m × 20.4 m. We deployed 7 TP-LINK TL-WR845N APs shown in Figure 6(a) in the experimental environment at a height of 2.2 m. We selected 116 RPs in the corridor, Room 616, and Room 620 and collected 120 RSS vectors at each RP to establish the fingerprint database 1 that is the radio map. Meanwhile, in Room 616, we selected 19 locations in a training trajectory at an interval of 0.6 m and collected 60 RSS vectors at each location to establish the fingerprint database 2. Similarly, from one location near the elevator in the corridor to one location in Room 620, we selected 90 testing points (TPs) in a testing trajectory at an interval of 0.6 m and collected the RSS vectors at a sampling rate of 1 RSS vector per second to simulate a scenario that a mobile user moves along the trajectory at a speed of 0.6 m/s. We also collected the RSS vectors for 60 times at each TP to test the proposed localization method. We measured all the RSS vectors using a Meizu M2 smart phone shown in Figure 6(b), which was placed on a tripod at a height of 1.2 m. We installed an Android application developed by us in the smart phone, and the sampling rate of the Android application was 1 RSS vector per second as mentioned above.

4.2. Parameter Settings. Considering that the parameter settings could directly affect the network performance, we tested the proposed method multiple times with different parameters and obtained the optimized parameters. In the unsupervised feature learning of the SISAE, we set the sparsity constants of the three improved sparse autoencoders equal to 0.04, 0.01, and 0.02, respectively. We set the weighted factor α of the sparsity penalty term equal to 10, β of the weight decay term equal to 0.001, and λ of the activity penalty term equal to 0.1. Meanwhile, we initialized the weight matrix $\mathbf{w}^{(i)}$, $i = 1, 2, 3$ of each improved sparse autoencoder using a normal distribution with a mean of 0 and a standard deviation of 0.1 and set each element of the bias vector $\boldsymbol{b}^{(i)}$, $i = 1, 2, 3$ of each improved sparse autoencoder equal to 0.1. We set the numbers of neurons in the hidden layer 1, 2, and 3 equal to 200, 400, and 800, respectively. In addition, we set the learning rate of each improved sparse autoencoder and number of iterative epochs equal to 0.001 and 1000, respectively.

In the supervised fine-tuning of the SISAE, we initialized the weight matrix $\mathbf{w}^{(4)}$ of the output layer of the SISAE using a normal distribution with a mean of 0 and standard deviation of 1, and set each element of the bias vector $\boldsymbol{b}^{(4)}$ of the output layer of the SISAE equal to 0.1. We set the learning rate, dropout rate, and number of iterative epochs equal to 0.005, 0.2, and 200, respectively.

In the supervised training of the RNN, we initialized the weight matrix $\mathbf{u}$ for the extra inputs of the RNN using a normal distribution with a mean of -0.01 and a standard deviation of 0.01. We set the learning rate, dropout rate, and number of iterative epochs equal to 0.001, 0.2, and 200, respectively.

Besides, we used the rectified linear units (ReLU) as the activation function for the encoder and the Tanh as the activation function for the decoder of each improved sparse autoencoder in the unsupervised feature learning phase of the SISAE. We also used the ReLU as the activation function for each hidden layer in the supervised training phases of the SISAE and RNN because the ReLU activation function could have a faster training speed and better performance than the Sigmoid activation function. At the same time, we selected Adam optimizer for the BP algorithm to optimize the weights and biases of each improved sparse autoencoder and also to optimize the weights and biases of the SISAE and RNN.

4.3. Experimental Results and Analyses

4.3.1. Localization Results of the SISAE. We compare our proposed SISAE with some other fingerprinting algorithms including KNN, weighted K-nearest neighbors (WKNN), multilayer perceptron (MLP), DNN, SAE, and SSAE. We

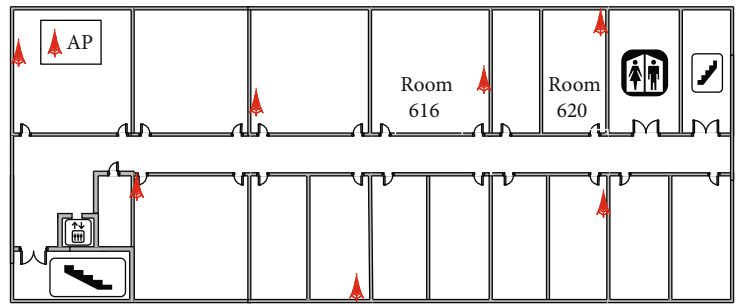


Figure 5: Experimental floor plan.

(a) (b)

Figure 6: Experimental scenario of Wi-Fi fingerprint-based localization: (a) TP-LINK TL-WR845N AP (b) Meizu M2 smart phone on a tripod.

select 7 nearest RPs for the KNN and WKNN. The MLP is a basic neural network that has a single hidden layer with 50 neurons. The DNN, SAE, and SSAE have the same network structure and parameter settings as the SISAE as mentioned in Section 4.2. With all the 5400 testing RSS vectors, the localization results computed by these fingerprinting algorithms are shown in Table 1. The mean errors of the KNN, WKNN, MLP, DNN, SAE, SSAE, and SISAE are 4.22 m, 4.20 m, 3.06 m, 2.19 m, 2.14 m, 2.01 m, and 1.93 m, respectively. The standard deviations of the localization errors are 4.00 m, 4.00 m, 1.91 m, 1.43 m, 1.66 m, 1.52 m, and 1.34 m. The SISAE has the minimum mean error and standard deviation among these algorithms, and the deep learning fingerprinting algorithm DNN, SAE, SSAE, and SISAE outperform the other three traditional fingerprinting algorithms. As shown in Figure 7, the cumulative probabilities of these algorithms within a localization error of 1 m are all relatively low. The cumulative probabilities of the SAE, SSAE, and SISAE with feature learning within a localization error of 1 m are obviously higher than that of the DNN, which proves the effectiveness of the RSS feature learning. The cumulative probabilities of the SISAE within a localization error of 2 m and 3 m are 61.4 % and 82.5 %, respectively, which are the highest among these algorithms. These experimental results verify that the SISAE fingerprinting algorithm outperforms the other fingerprinting algorithms.

Table 1: Performance comparison of fingerprinting algorithms.

Algorithm	Mean error (m)	Standard deviation (m)	Cumulative probability (%) Within 1 m error	Within 2 m error	Within 3 m error
KNN	4.22	4.00	20.0	37.4	54.6
WKNN	4.20	4.00	20.0	37.9	54.4
MLP	3.06	1.91	8.80	34.5	55.2
DNN	2.19	1.43	16.6	56.3	78.2
SAE	2.14	1.66	24.4	58.4	77.5
SSAE	2.01	1.52	29.1	60.0	81.1
SISAE	1.93	1.34	27.2	61.4	82.5

4.3.2. Tracking Results of the RNN. Because the DNN, SAE, SSAE, and SISAE localization algorithms have the same network structure, we integrate each of these algorithms with an RNN for localization performance improvement. With all the 5400 testing RSS vectors, the localization results computed by the DNN-RNN, SAE-RNN, SSAE-RNN, and SISAE-RNN are shown in Table 2. After adding the RNN, all the mean errors of these algorithms are reduced. The mean errors of the DNN-RNN, SAE-RNN, SSAE-RNN, and SISAE-RNN are 1.97 m, 1.97 m, 1.81 m, and 1.60 m, respectively. The standard deviations of the localization

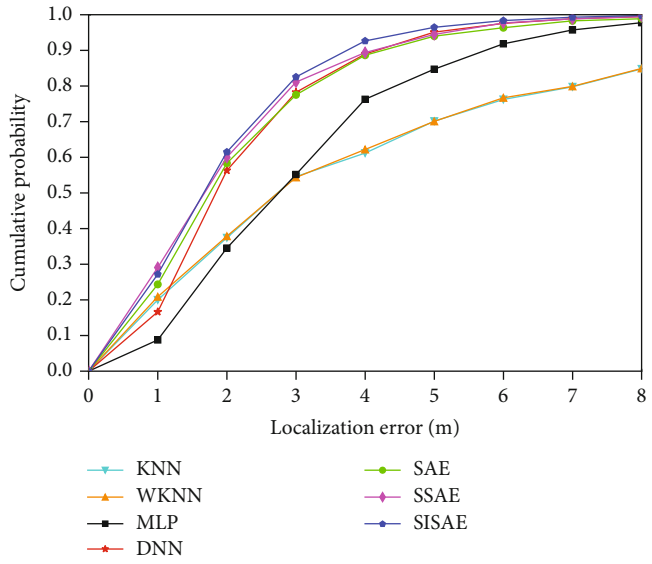


Figure 7: Cumulative probabilities of fingerprinting algorithms.

Table 2: Performance comparison of fingerprinting algorithms with RNN.

Algorithm	Mean error (m)	Standard deviation (m)	Cumulative probability (%)		
			Within 1 m error	Within 2 m error	Within 3 m error
DNN-RNN	1.97	1.12	19.3	56.2	84.1
SAE-RNN	1.97	1.44	25.8	60.5	82.6
SSAE-RNN	1.81	1.24	29.7	63.1	86.6
SISAE-RNN	1.60	1.06	33.4	70.2	89.7

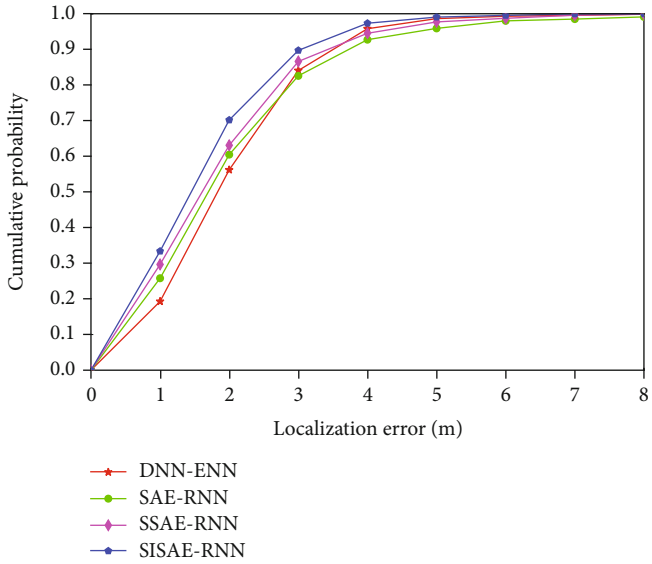
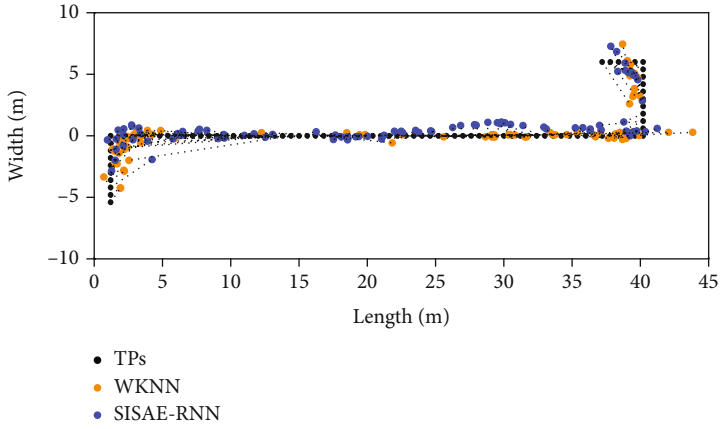


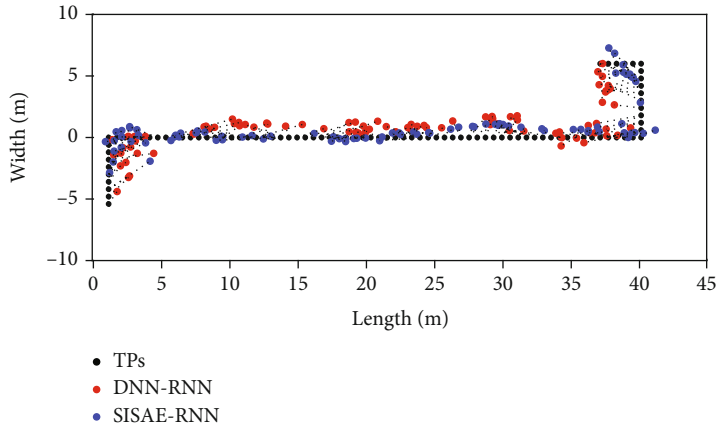
Figure 8: Cumulative probabilities of fingerprinting algorithms with RNN.

errors are 1.12 m, 1.44 m, 1.24 m, and 1.06 m. As shown in Figure 8, the cumulative probabilities of these algorithms within a localization error of 1 m are all increased after adding the RNN. The cumulative probabilities of the DNN-RNN, SAE-RNN, SSAE-RNN, and SISAE-RNN within a localization error of 2 m are 56.2 %, 60.5 %, 63.1 %, and 70.2 %, respectively, and the cumulative probabilities of the DNN-RNN, SAE-RNN, SSAE-RNN, and SISAE-RNN within a localization error of 3 m are able to reach 84.1 %, 82.6 %, 86.6 %, and 89.7 %, respectively. These experimental results prove that our proposed SISAE-RNN has a superior localization performance.

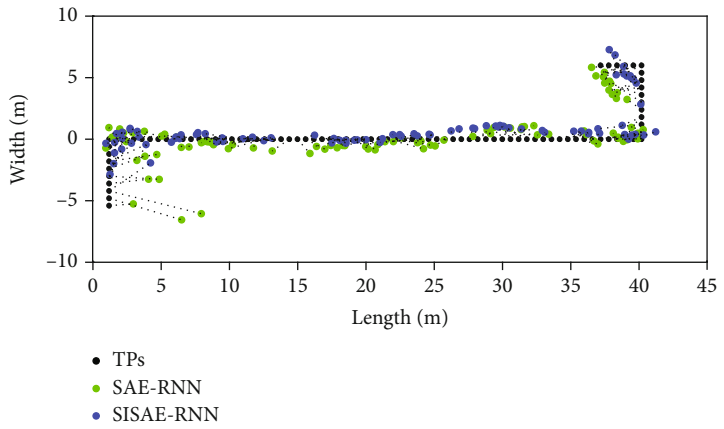
We also take one trajectory as an example and compare the SISAE-RNN with the WKNN, DNN-RNN, SAE-RNN,



(a)

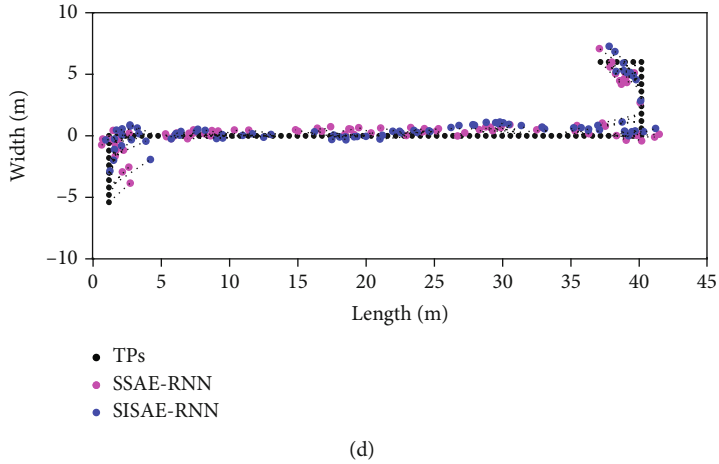


(b)



(c)

Figure 9: Continued.



(d)

Figure 9: Localization result comparisons of one trajectory: (a) WKNN and SISAE-RNN, (b) DNN-RNN and SISAE-RNN, (c) SAE-RNN and SISAE-RNN, and (d) SSAE-RNN and SISAE-RNN.

and SSAE-RNN. With 90 testing RSS vectors of the trajectory, the mean errors of the WKNN, DNN-RNN, SAE-RNN, SSAE-RNN, and SISAE-RNN are 4.07 m, 2.00 m, 1.83 m, 1.67 m, and 1.47 m, respectively. The comparison results are shown in Figure 9. In Figure 9(a), although the localization results of the WKNN are generally near the ground-truth trajectory, the localization errors of the WKNN in the horizontal axis are much larger than the SISAE-RNN. Both Figures 9(b) and 9(c) show that the localization results computed by the SISAE-RNN are more accurate and less fluctuant than those computed by the DNN-RNN and SAE-RNN. Although the difference of localization results between the SISAE-RNN and SSAE-RNN shown in Figure 9(d) is not obvious, the mean error of the SISAE-RNN is 1.47 m, which is the minimum one among these algorithms. These experimental results of the single trajectory with 90 testing RSS vectors are in accordance with the results computed with all the 5400 testing RSS vectors.

5. Conclusions and Future Works

In this study, we propose a Wi-Fi fingerprint-based indoor mobile user localization method that integrates an SISAE and an RNN. We first propose an SISAE fingerprinting algorithm whose dimensions of the hidden layers can be freely set. We improve the sparse autoencoder by adding an activity penalty term in its loss function to control the neuron outputs in the hidden layer. We stack the encoders of three improved sparse autoencoders and add a logistic regression layer as the output layer to the stacked encoders to construct the SISAE for fingerprinting localization. Moreover, we also construct an RNN by using the same network structure and parameters of the trained SISAE and also by adding the previous location coordinates computed by the SISAE as the extra inputs of the RNN to compute more accurate current location coordinates. The experimental results show that our proposed SISAE-RNN is able to reduce the mean error to 1.60 m. The SISAE-RNN has a superior localization performance compared with some other Wi-Fi fingerprint-based localization algorithms and also can be used in the scenarios with a few deployed APs, which proves the validity and generality of our proposed method.

In the future, we intend to continue the research work on the improvement of the deep learning-based fingerprinting and tracking algorithms, such as the acceleration of unsupervised feature learning, the optimization of deep network structure, and the integration of fingerprinting and tracking algorithms.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant no. 61701223 and the Natural Science Foundation of Jiangsu Province under Grant no. BK20171023.

References

[1] D. Minoli and B. Occhiogrosso, "Practical aspects for the integration of 5G networks and IoT applications in smart cities environments," *Wireless Communications and Mobile Computing*, vol. 2019, Article ID 5710834, 30 pages, 2019.

[2] X. Liu and X. Zhang, "NOMA-based resource allocation for cluster-based cognitive industrial internet of things," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 8, pp. 5379–5388, 2020.

[3] X. Liu, M. Jia, X. Zhang, and W. Lu, "A novel multichannel internet of things based on dynamic spectrum sharing in 5G communication," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 5962–5970, 2019.

[4] X. Guo, N. Ansari, L. Li, and L. Duan, "A hybrid positioning system for location-based services: design and implementation," *IEEE Communications Magazine*, vol. 58, no. 5, pp. 90–96, 2020.

[5] D. Zou, W. Meng, S. Han, K. He, and Z. Zhang, "Toward ubiquitous LBS: multi-radio localization and seamless positioning," *IEEE Wireless Communications*, vol. 23, no. 6, pp. 107–113, 2016.

[6] M. Zhou, Y. Wang, Z. Tian, and Q. Zhang, "Indoor pedestrian motion detection via spatial clustering and mapping," *IEEE Sensors Letters*, vol. 2, no. 1, pp. 1–4, 2018.

[7] Z. Deng, W. Si, Z. Qu, X. Liu, and Z. Na, "Heading estimation fusing inertial sensors and landmarks for indoor navigation using a smartphone in the pocket," *EURASIP Journal on Wireless Communications and Networking*, vol. 2017, no. 1, p. 12, 2017.

[8] S. He, G. Chan, L. Yu, and N. Liu, "Maxlifd: joint maximum likelihood localization fusing fingerprints and mutual distances," *IEEE Transactions on Mobile Computing*, vol. 18, no. 3, pp. 602–617, 2019.

[9] Y. Sun, X. Zhang, X. Wang, and X. Zhang, "Device-free wireless localization using artificial neural networks in wireless sensor networks," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 4201367, 8 pages, 2018.

[10] L. Song, T. Zhang, X. Yu, C. Qin, and Q. Zhang, "Scheduling in cooperative UWB localization networks using round trip measurements," *IEEE Communications Letters*, vol. 20, no. 7, pp. 1409–1412, 2016.

[11] Y. Sun, W. Meng, C. Li, N. Zhao, K. Zhao, and N. Zhang, "Human localization using multi-source heterogeneous data in indoor environments," *IEEE Access*, vol. 5, pp. 812–822, 2017.

[12] A. Yassin, Y. Nasser, M. Awad et al., "Recent advances in indoor localization: a survey on theoretical approaches and applications," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 2, pp. 1327–1346, 2017.

[13] J. Zhang, L. Ding, Y. Wang, and L. Hu, "Measurement-based indoor NLoS ToA/RSS range error modelling," *Electronics Letters*, vol. 52, no. 2, pp. 165–167, 2016.

[14] M. Sheng, Y. Zheng, J. Liu, S. Valaee, and J. Li, "Accurate indoor localization assisted with optimizing array orientations and receiver positions," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 1, pp. 509–521, 2020.

[15] X. Tian, W. Li, Y. Yang, Z. Zhang, and X. Wang, "Optimization of fingerprints reporting strategy for WLAN indoor localization," *IEEE Transactions on Mobile Computing*, vol. 17, no. 2, pp. 390–403, 2018.

[16] D. V. Le, N. Meratnia, and P. J. M. Havinga, "Unsupervised deep feature learning to reduce the collection of fingerprints for indoor localization using deep belief networks," in *2018 International conference on indoor positioning and indoor navigation (IPIN)*, pp. 1–7, Nantes, France, September 2018.

[17] W. Shao, H. Luo, F. Zhao, Y. Ma, Z. Zhao, and A. Crivello, "Indoor positioning based on fingerprint-image and deep learning," *IEEE Access*, vol. 6, pp. 74699–74712, 2018.

[18] X. Song, X. Fan, C. Xiang et al., "A novel convolutional neural network based indoor localization framework with WiFi fingerprinting," *IEEE Access*, vol. 7, pp. 110698–110709, 2019.

[19] B. Yan and G. Han, "Effective feature extraction via stacked sparse autoencoder to improve intrusion detection system," *IEEE Access*, vol. 6, pp. 41238–41248, 2018.

[20] P. Bahl and V. N. Padmanabhan, "RADAR: an in-building RF-based user location and tracking system," in *Proceedings IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (Cat. No.00CH37064)*, pp. 775–784, Tel Aviv, Israel, Israel, March 2000.

[21] P. Castro, P. Chiu, T. Kremenek, and R. Muntz, "A probabilistic room location service for wireless networked environments," in *International conference on ubiquitous computing*, pp. 18–34, Springer, Berlin, Heidelberg, 2001.

[22] M. Youssef and A. Agrawala, "The Horus WLAN location determination system," in *Proceedings of the 3rd international conference on Mobile systems, applications, and services - MobiSys '05*, pp. 205–218, Seattle, Washington, 2005.

[23] D. Li, B. Zhang, and C. Li, "A feature-scaling-based *k*-nearest neighbor algorithm for indoor positioning systems," *IEEE Internet of Things Journal*, vol. 3, no. 4, pp. 590–597, 2016.

[24] X. Wang, L. Gao, S. Mao, and S. Pandey, "CSI-based fingerprinting for indoor localization: a deep learning approach," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 1, pp. 763–776, 2017.

[25] S. Fang and T. Lin, "Indoor location system based on discriminant-adaptive neural network in IEEE 802.11 environments," *IEEE Transactions on Neural Networks*, vol. 19, no. 11, pp. 1973–1978, 2008.

[26] Y. Li, X. Hu, Y. Zhuang, Z. Gao, P. Zhang, and N. El-Sheimy, "Deep reinforcement learning (DRL): another perspective for unsupervised wireless localization," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6279–6287, 2020.

[27] M. Mohammadi, A. al-Fuqaha, M. Guizani, and J. Oh, "Semisupervised deep reinforcement learning in support of IoT and smart city services," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 624–635, 2018.

[28] C. Hsieh, J. Chen, and B. Nien, "Deep learning-based indoor localization using received signal strength and channel state information," *IEEE Access*, vol. 7, pp. 33256–33267, 2019.

[29] Z. E. Khatab, A. Hajihoseini, and S. A. Ghorashi, "A fingerprint method for indoor localization using autoencoder based deep extreme learning machine," *IEEE Sensors Letters*, vol. 2, no. 1, pp. 1–4, 2018.

[30] L. Zheng, B. Hu, J. Qiu, and M. Cui, "A deep-learning-based self-calibration time-reversal fingerprinting localization approach on Wi-Fi platform," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 7072–7083, 2020.

[31] W. Zhang, K. Liu, W. Zhang, Y. Zhang, and J. Gu, "Deep neural networks for wireless localization in indoor and outdoor environments," *Neurocomputing*, vol. 194, pp. 279–287, 2016.

[32] S. Bai, M. Yan, Q. Wan, L. He, X. Wang, and J. Li, "DL-RNN: an accurate indoor localization method via double RNNs," *IEEE Sensors Journal*, vol. 20, no. 1, pp. 286–295, 2020.

[33] M. T. Hoang, B. Yuen, X. Dong, T. Lu, R. Westendorp, and K. Reddy, "Recurrent neural networks for accurate RSSI indoor localization," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 10639–10651, 2019.

[34] A. Belmonte-Hernandez, G. Hernandez-Penaloza, D. Martin Gutierrez, and F. Alvarez, "Recurrent model for wireless indoor tracking and positioning recovering using generative networks," *IEEE Sensors Journal*, vol. 20, no. 6, pp. 3356–3365, 2020.

[35] L. Zhang, M. Cheng, Z. Xiao, L. Zhou, and J. Zhou, "Adaptable map matching using PF-net for pedestrian indoor

localization," *IEEE Communications Letters*, vol. 24, no. 7, pp. 1437–1440, 2020.

[36] Z. Chen, H. Zou, J. Yang, H. Jiang, and L. Xie, "WiFi fingerprinting indoor localization using local feature-based deep LSTM," *IEEE Systems Journal*, vol. 14, no. 2, pp. 3001–3010, 2020.

[37] O. B. Tariq, M. T. Lazarescu, and L. Lavagno, "Neural networks for indoor human activity reconstructions," *IEEE Sensors Journal*, vol. 20, no. 22, pp. 13571–13584, 2020.