A Fingerprint Technique for Indoor Localization using Autoencoder based Semi-supervised Deep Extreme Learning Machine

Zahra Ezzati Khatab[1], Amirhosein Hajihoseini Gazestani[1], Seyed Ali Ghorashi[1,2], Mohammad Ghavami[3]

[1] *Cognitive Telecommunication Research Group, Department of Electrical Engineering, Shahid Beheshti University G. C. Tehran, Iran*
[2] *School of Architecture, Computing and Engineering, University of East London, London, United Kingdom*
[3] *School of Engineering London South Bank University, London, United Kingdom*

Abstract

In recent years, because of the growing demand for location based services in indoor environment and development of Wi-Fi, fingerprint-based indoor localization has attracted many researchers' interest. In Wireless Sensor Networks (WSNs), fingerprint based localization methods estimate the target location by using a pattern matching model for the measurements of the Received Signal Strength (RSS) from the available transmitter sensors, which are collected by a smartphone with internal sensors. Due to the dynamic nature of the environment, the fingerprint database needs to be updated, periodically. Hence, it is better to add new fingerprint data to the primary database in order to update them. However, collecting the labeled data is time consuming and labor intensive. In this paper, we propose a novel algorithm, which uses high level extracted features by an autoencoder to improve the localization performance in the classification process. Furthermore, to update the fingerprint data base, we also add crowd-sourced labeled and unlabeled data in order to improve the localization performance, gradually. Simulation results indicate that the proposed method provides a significant improvement in localization performance,

[☆] Corresponding author: Seyed Ali Ghorashi (Email address: s.a.ghorashi@uel.ac.uk)

using high level extracted features by the autoencoder, and by increasing the number of unlabeled training data.

Keywords: `Indoor localization, fingerprint, wireless sensor network, semi-supervised, autoencoder, deep extreme learning machine`

1. Introduction

Due to the increasing request for location-based services inside homes and offices, indoor localization has undergone a lot of investigations. In outdoor areas, Global Positioning System (GPS) and Global Navigation Satellite Systems (GNSS) have been identified as the common positioning technologies, because of the accurate location information they provide [1]. However, these technologies are not applicable indoor, because of the unpredictability of the radio propagation, very weak GPS signals sent from satellites to devices while penetrating through buildings and the low visibility of satellite in indoor areas. Therefore, alternative signals such as Wi-Fi, Bluetooth, FM radio, radio-frequency identification (RFID), light, and magnetic field have been used for localization [2, 3, 4, 5]. Because of the wide usage of Wireless Local Area Networks (WLANs) as indoor access systems and the popularity of wireless communication equipment, indoor location-based services such as indoor object search, indoor object localization, navigation and tracking have been greatly expanded. However, complex indoor areas and high equipment costs, have prevented the achievement of satisfactory location accuracy in practice [6]. Therefore, it is important to obtain real-time and accurate location information using mobile devices without extra hardware installation or modification. Also, Channel State Information (CSI) can be used for localization [7, 8, 9, 10], which may achieve even better accuracy than RSS based methods [8, 9]. However, the use of RSS of Wi-Fi signals is more popular as the measurement of the RSS can be done by most of the present smart phones, while CSI measurement needs special instruments [8, 9, 10].

Recently, fingerprint based indoor localization methods have been used in

Wireless Sensor Networks (WSNs), due to their advantages of lower cost and flexible applications [11, 12, 13, 14]. WSNs are made of spatially distributed sensors to monitor physical or environmental conditions [15]. Fingerprint based localization methods in WSNs use Received Signal Strength (RSS) measurements from the available transmitters in the area which are collected by smartphones with internal sensors. These measured RSSs and corresponding positions are used for target location estimation [16].

Fingerprint methods usually consist of two phases. During the offline phase, samples of RSSs for each transmitter sensor are taken at different Reference Points (RPs) and are stored in a database. In this phase, machine learning methods are used to train fingerprints and extract reliable features according to a certain rule to find a relationship between features and RPs' locations. During the online phase, the server compares the target RSS with offline database, using some pattern matching algorithms and estimates the target location. Such pattern matching algorithms include deterministic and probabilistic methods [3]. Training probabilistic methods such as Horus systems may require largest datasets than that of deterministic methods. In deterministic methods such as k nearest neighbors (k-NN), Support Vector Machine (SVM) and linear discriminant analysis, a similarity metric such as Euclidean distance is used to differentiate between different measured RSS of the target. Then, the target location is estimated based on the closest stored RSS in the fingerprint database. Deterministic methods are most commonly used because of their simplicity [3, 17, 18].

In recent years, several fingerprinting methods have been proposed such as k-NN and SVM [19]. Neural networks are also used for fingerprinting [20]. Deep Learning (DL) methods based on neural network, have been proposed since 2006, which simulate hierarchical structure of human brain [21] and can explore the features by a multilayer feature representation framework and get the optimal weights [22]. The authors of [23] introduce a Deep Neural Network (DNN) which is pre-trained by stacked de-noising autoencoder. In [24] SAE method, a deep learning method based on stacked autoencoder using Wi-Fi fingerprinting, is introduced that allows to efficiently reduce the feature space in order to achieve

a robust and precise classification. In [8] a deep learning network with 4 hidden layers is proposed for indoor localization, which uses channel state information. Also this network uses hidden Markov model based fine localizer to smooth the initial position estimation obtained by DNN based coarse localizer. In [25], a sparse deep autoencoder network is proposed to realize location, activity and gesture recognition simultaneously.

For indoor localization, labeled data are required, which both RSSs and corresponding locations of them are known. Theoretically, more labeled data results in better localization performance [26]. However, obtaining the labeled data is time consuming and costly. Thus, the number of labeled data is limited.

Indoor environments are dynamic because of factors such as people, doors, elevators, escalators, the change of temperature and humidity. Thus, fingerprint data need to be updated. Moreover, high level feature extraction of signals is an important problem for efficient localization. One of the common methods to extract shallow level information is the mean value method that collects RSS vectors at the same location and extract the features by averaging those vectors [27]. The other method is Gaussian model that is applied when the RSS at the same location has a Gaussian distribution [28]. To deal with the dynamic nature of indoor environments, an algorithm is proposed in [29] which uses the average of a number of maximum RSS observations by analyzing the spatial resolution of the signal strength.

The process of collecting fingerprint data requires a considerable time and labor cost, which makes the number of labeled data limited. In order to reduce the required costs and time for collecting labeled fingerprint data which is needed in dynamic environments frequently, another localization method is employed that uses unlabeled data with just a limited number of labeled data, and is called semi-supervised machine learning method [30]. In contrast to collection of labeled data, the collection of unlabeled data can be done easily. One method for unlabeled data collection is utilizing handsets. This method prompts volunteer users to report their collected data, in order to participate in data updating [31]. The procedure of collecting unlabeled data in order to update fingerprint

database, in addition to save cost and time, keeps the privacy of participants. Utilizing unlabeled data also preserve participants' privacy, who may be concerned about the disclosure of their location information during crowd-sensing precess. Since in collecting unlabeled data, the location of participant is not needed, the volunteer users who don't like others to be aware of their location in the area, can also easily help in data collection by sending their RSS from the access points. Since unlabeled data collection does not require users location, it keeps the privacy of the users.

The authors of [32] introduce an approach that uses manifold learning for localization. It builds a manifold based model for labeled and unlabeled data and then uses weighted k Nearest Neighbors (WKNN) to localize the target. In [33] semi-supervised manifold alignment is used to localize targets. Pulkkinen et al present another semi-supervised manifold learning technique in [34]. This technique constructs a nonlinear projection to map high dimensional RSS data onto a two-dimensional manifold. The authors of [5] use graph based radio map generation for semi-supervised indoor localization. They use graph-based signal processing schemes for polishing the input data, determining the outliers and radio-map generation and achieve to about 2.6 meter localization error in $18 \times 36m^2$ area.

To improve localization performance using unlabeled data, manifold learning based and semi-supervised learning based works, were proposed in [35]. After data collection, the challenge in indoor localization is feature extraction. In traditional localization methods, feature extraction is carried out by shallow networks, which is not efficient for complex environment because of limited modeling and representational power when dealing with such big and noisy data problem [23]. Deep learning (DL) automatically learns high level features and can represent original data better than shallow networks. Although DL is better than traditional neural networks, its training time is high. Moreover, as mentioned above, by increasing the size of fingerprint dataset, the localization process is time consuming. In order to address these problems, Extreme Learning Machine (ELM) is utilized, whose learning speed is higher than DL [36]. The

authors of [37] introduce semi-supervised ELM, which shows that with the same number of labeled data, Semi-supervised Extreme Learning Machine (SELM) which uses more unlabeled data has a better performance than traditional ELM. Semi-supervised Deep Extreme Learning Machine (SDELM) is the other ELM based method, which has the advantages of semi-supervised learning and deep learning [26]. The authors of [26] show that SDELM has 25% success rate by using 500 labeled data and 3000 unlabeled data. However, because of the randomness of input weight matrix and bias of ELM, it cannot extract high level features from the input data. In [38], a semi-supervised ELM based method is introduced which uses Wi-Fi and Bluetooth Low Energy (BLE) signals. By using 3000 labeled data, they can achieve to 95.5% localization performance for 3 m error distance threshold.

In this paper, considering the aforementioned challenges, we propose an Autoencoder based Semi-supervised Deep Extreme Learning Machine (ASDELM) for classification of users based on their location in indoor areas. It uses an autoencoder which learns high level feature extraction from RSS samples and merges these features into the deep extreme learning machine (DELM) in order to improve the localization performance. This paper is an extension to our previous work [39] in the following aspects:

- Using unlabeled data to encourage more participant in crowd-sensing process: semi-supervised method is used, which decreases the time and labor cost of collecting labeled data. Also, by crowdsensing technique, the volunteers who are in different places of indoor environment can participate in collecting unlabeled without disclosure of their location. Since the participants only send the collected received signal strength from the transmitter, and do not send their locations (the room number which they are located), their locations is kept confidential. This privacy can be very encouraging for the possible participants in the crowd-sensing process, and our proposed method helps to have much more participants with much cheaper expenses.

- Decreasing the complexity and improving the accuracy: since in this work, all of the outputs of the hidden layers are calculated by an autoencoder network, the time required for the calculation of the extracted feature matrix for last hidden layer is decreased. Moreover, the localization accuracy is improved in comparison with our previous method for the same semi-supervised scenario.

- Real test in addition to computer simulations: in this work, we have examined the capabilities of our proposed method by using real data and we have shown that our proposed algorithm has appropriate localization performance in real scenarios.

The rest of this paper is organized as follows: In section 2 the basic knowledge of ELM and autoencoder are introduced. The proposed localization algorithm (ASDELM) is presented in section 3. Section 4 shows the experimental results and confirms the advantages of the proposed algorithm. Finally, the conclusion is drawn in section 5.

2. PRELIMINARIES

Graph based semi-supervised learning which uses deep autoencoder network to extract discriminative features is one of the approaches to improve localization performance. Due to the long training time for deep learning, ELM is used and increases the learning speed. This section introduces the basic knowledge of graph based semi-supervised learning, deep learning, autoencoder networks and ELM. Then the ASDELM will be presented in detail.

2.1. Graph based semi-supervised learning

Traditionally, we may categorize the learning processes into two types: learning with a teacher and learning without a teacher. The former is also referred to as supervised learning whose goal is to learn a mapping from $\mathbf{X}$ to $\mathbf{Y}$, given the training set made of pairs $(\mathbf{x}_i, \mathbf{y}_i)$. The latter may be subcategorized into unsupervised learning and reinforcement learning and as the name implies, there is

no teacher to oversee the learning process and relies only on unlabeled examples [40]. In semi-supervised learning, a training sample that consists of labeled and unlabeled data is utilized. This can reduce the time and labor needed for data collection. In addition to unlabeled data, this method is provided with some supervision information [40, 41].

For semi-supervised learning to work, certain assumptions have to hold, such as smoothness assumption, cluster assumption, and manifold assumption. Semi-supervised smoothness assumption states that if two points $\mathbf{x}_1$ and $\mathbf{x}_2$ are close in high density region, then the corresponding outputs $\mathbf{y}_1$ and $\mathbf{y}_2$ should be close. The cluster assumption implies that if points are in the same cluster, they are likely to be of the same class. The manifold assumption that forms the basis of several semi-supervised learning methods states that the high dimensional data lie roughly on a low dimensional manifold. The curse of dimensionality is a problem of many learning algorithms, which is a term to describe the problem caused by the exponential increase in volume related to adding extra dimensions to Euclidean space. If data lie on a low dimensional manifold, the learning algorithm can operate in a space of corresponding dimension [45]. Semi-supervised learning should satisfy at least one of the aforementioned assumptions [41, 42].

Graphs are general forms of data representation which are utilized for explaining the geometric structure of data domains in multiple applications, such as energy, transportation, sensor and neural networks. The weight of each edge in the graph mostly represents the similarity between two vertices that connect them. The connections and corresponding edge weights are allocated by the physics of problem or data. For example, the edge weights may have an inverse relationship with the physical distance between nodes of the network [43]. In graph based semi-supervised learning, both labeled and unlabeled data are the vertices of the graph and this provides a paradigm for modeling the manifold structures. The main goal of these methods is to find an optimal classification

function f^* by solving the following optimization problem [44]:

$$\begin{aligned} &\underset{f}{\operatorname{argmin}} [\tfrac{1}{l} \sum_{i=1}^{l} \left(f\left(\mathbf{x}_i\right) - \mathbf{y}_i\right)^2 + \lambda_A ||f||^2 + \\ &\lambda_I \sum_{i,j=1}^{n} \left(f\left(\mathbf{x}_i\right) - f\left(\mathbf{x}_j\right)\right)^2 \mathbf{W}_{ij}] \end{aligned} \tag{1}$$

where $\{(\mathbf{x}_i, \mathbf{y}_i)\}; i = 1, ..., l$ are labeled samples, $\{\mathbf{x}_i\}; i = l+1, ..., n$ are unlabeled samples, $\mathbf{x}_i$ is RSS sample and y_i is its corresponding label. λ_A and λ_I are the control parameters of smoothness in the ambient and inherent spaces, l is the number of labeled samples and n is the sum of the number of labeled and unlabeled data. $\mathbf{W}$ is an affinity matrix whose element $\mathbf{W}_{ij}$ denotes the similarity weight of samples $\mathbf{x}_i$ and $\mathbf{x}_i$. To use the geometrical structure of feature space, most researchers adopt a Laplacian graph, which makes (1) be expressed as:

$$\begin{aligned} &\underset{f}{\operatorname{argmin}} [\tfrac{1}{l} \sum_{i=1}^{l} \left(f\left(\mathbf{x}_i\right) - \mathbf{y}_i\right)^2 + \\ &\lambda_A ||f||^2 + \lambda_I \mathbf{f}^T \mathbf{L} \mathbf{f}] \end{aligned} \tag{2}$$

where $\mathbf{L}$ is the Laplacian matrix [44], and will be explained in section 3.

2.2. Autoencoder

An autoencoder is a neural network that is trained to replicate its input to output. Training an autoencoder is unsupervised in the sense that no labeled data is needed. Each layer of autoencoder consists of encoding and decoding procedure. Hidden layer of autoencoder, h, describes a code for representing the input data and the decoder produces a reconstruction. The encoder and decoder can have multiple layers, but for simplicity consider that each of them has only one layer. If the input data is $\mathbf{x} \in \Re^{D_i}$, then the encoder maps the input to $\mathbf{r} \in \Re^{D_e}$ as Eq. (3), where D_i and D_e are input dimension and encoded representation dimension, respectively.

$$\mathbf{r}^{(1)} = h^{(1)} \left(\mathbf{\Psi}^{(1)} \mathbf{x} + \mathbf{b}^{(1)}\right) \tag{3}$$

where $h^{(1)}$ is activation function for the encoder, and $\mathbf{\Psi}^{(1)}$ is encoding weight matrix and $\mathbf{b}^{(1)}$ is the encoding bias. The decoder maps the encoded represen-

tation $\mathbf{r}$ back into an estimate of the original input vector:

$$\hat{\mathbf{x}} = h^{(2)}\left(\mathbf{\Psi}^{(2)}\mathbf{r}^{(1)} + \mathbf{b}^{(2)}\right) \tag{4}$$

where $h^{(2)}$ is the decoder activation function and, $\mathbf{\Psi}^{(2)}$ and $\mathbf{b}^{(2)}$ are decoding weight matrix and decoder bias, respectively [45, 46, 47].

The learning process of autoencoder is defined as minimizing a loss function that measures the error between the input and its reconstruction [25]:

$$\begin{array}{l} J(\mathbf{\Psi}, \mathbf{b}) = \left[\frac{1}{N}\sum\limits_{i=1}^{N}\left(\frac{1}{2}||\mathbf{r}_i^{(l-1)} - \hat{\mathbf{x}}_i^{(l)}||^2\right)\right] \\ +\frac{\vartheta}{2}\sum\limits_{j=1}^{n_{l-1}}\sum\limits_{i=1}^{n_l}\left(\mathbf{\Psi}_{ij}^{(l-1)}\right)^2 \end{array} \tag{5}$$

where N is the total number of training samples, n_l indicates the number of units in layer l, and we define $\mathbf{r}_i^1 = \mathbf{x}_i$, and second term denotes the weight decay which is utilized to avoid overfitting, and ϑ is the weight decay parameter.

Traditionally, autoencoders were utilized for feature learning or dimension reduction. If the capacities of encoder and decoder are too much or if the hidden code has dimension equal to or greater than the input, this autoencoder is called overcomplete which fails to learn anything useful. The cost function which is used in regularized autoencoder leads to further properties in capacity model, including the sparsity of representation, the small size of the derivative of the representation, and robustness to noise or to missing inputs [45].

A sparse autoencoder is simply an autoencoder that adds a regularizer to the cost function [48]. This regularizer is a function of the average output activation value of a neuron. Particularly, if a sparsity constraint is imposed on hidden neurons, the autoencoder can extract useful features even if the number of hidden neurons is large.

The cost function for training a sparse autoencoder is defined as:

$$\hat{J}\left(\mathbf{\Psi}, b\right) = J\left(\mathbf{\Psi}, b\right) + \gamma.\Omega_{sparsity} \tag{6}$$

where the second term is sparsity penalty term and γ controls the weight of sparsity term. We can learn the parameters $\mathbf{\Psi}$ and b by minimizing the overall cost

function [25]. Finding of researches show that deep autoencoders' efficiency is much better than that of shallow or linear autoencoders [49]. In a deep autoencoder network, each layer is trained as a one layer autoencoder by minimizing the error in reconstructing [45].

2.3. Extreme Learning Machine

We consider N arbitrary distinct samples $(\mathbf{x}_i, \mathbf{y}_i)$ where $\mathbf{x}_i = [x_{i1}, x_{i2}, ..., x_{in}]^T \in \Re^n$ are RSS samples and $\mathbf{y}_i = [y_{i1}, y_{i2}, ..., y_{im}]^T \in \Re^m$ are corresponding labels, which n and m are the numbers of transmitter sensors and classes, respectively. The output of a neural network with one layer including $\tilde{N}$ neurons is:

$$\begin{aligned} &f_{\tilde{N}}(\mathbf{x}_j) = \sum_{i=1}^{\tilde{N}} \boldsymbol{\varphi}_i G(\boldsymbol{\rho}_i, b_i, \mathbf{x}_j), \\ &\boldsymbol{\rho}_i \in \Re^n, \mathrm{b}_i \in \Re, \boldsymbol{\varphi}_i \in \Re^m, j = 1, 2, ..., N \end{aligned} \tag{7}$$

where $\boldsymbol{\varphi}_i = [\varphi_{i1}, \varphi_{i2}, ..., \varphi_{im}]^T$ is the weight vector connecting the i^{th} hidden neuron and output neurons, and $\boldsymbol{\rho}_i = [\rho_{i1}, \rho_{i2}, ..., \rho_{in}]^T$ is the weight vector connecting the input neurons and the i^{th} hidden neuron. b_i is the threshold of the i^{th} hidden neuron, and $G(\boldsymbol{\rho}_i, b_i, \mathbf{x}_j)$ is the output of the i^{th} hidden neuron. If the activation function of hidden neuron is $g(.)$, the output of the i^{th} hidden neuron is given by Eq. (8):

$$G(\boldsymbol{\rho}_i, b_i, \mathbf{x}_j) = g(\boldsymbol{\rho}_i.\mathbf{x}_j + b_i) \tag{8}$$

The goal of this one layer network with $\tilde{N}$ hidden neurons and activation function $g(.)$ is to approximate these N samples by tending the mean error to zero according to $\sum_{j=1}^{\tilde{N}} ||f_{\tilde{N}}(\mathbf{x}_j) - \mathbf{y}_j||_2^2 = 0$, where $||.||_2$ represents the 2-norm operation. Therefore, there exist $\boldsymbol{\varphi}_i$, $\boldsymbol{\rho}_i$ and b_i such that:

$$\sum_{i=1}^{\tilde{N}} \boldsymbol{\varphi}_i G(\boldsymbol{\rho}_i, b_i, \mathbf{x}_j) = \mathbf{y}_j, j = 1, 2, ..., N. \tag{9}$$

Eq. (9) for the whole network can be rewritten as [26]:

$$\mathbf{H\Phi} = \mathbf{Y};$$

$$\begin{aligned} \mathbf{H} &= \begin{bmatrix} G(\boldsymbol{\rho}_1, b_1, \mathbf{x}_1) & \cdots & G(\boldsymbol{\rho}_{\tilde{N}}, b_{\tilde{N}}, \mathbf{x}_1) \\ \vdots & \ddots & \vdots \\ G(\boldsymbol{\rho}_1, b_1, \mathbf{x}_N) & \cdots & G(\boldsymbol{\rho}_{\tilde{N}}, b_{\tilde{N}}, \mathbf{x}_N) \end{bmatrix}_{N \times \tilde{N}}, \\ \mathbf{\Phi} &= \begin{bmatrix} \boldsymbol{\varphi}_1^T \\ \vdots \\ \boldsymbol{\varphi}_{\tilde{N}}^T \end{bmatrix}_{\tilde{N} \times m}, \mathbf{Y} = \begin{bmatrix} \mathbf{y}_1^T \\ \vdots \\ \mathbf{y}_N^T \end{bmatrix}_{N \times m}. \end{aligned} \tag{10}$$

where $\mathbf{H}$ is the feature matrix. The goal of ELM is to minimize the error between real output and expected one and its solution can be represented as [26]:

$$\mathbf{\Phi} = \arg\min_{\mathbf{\Phi}} ||\mathbf{H\Phi} - \mathbf{Y}|| \rightarrow \mathbf{\Phi} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{HY} \tag{11}$$

3. PROPOSED METHOD

As claimed above, due to the dynamic nature of indoor environment, training data need to be updated. Also, since the labeled data collecting is time consuming and labor intensive, the number of them is limited. To address this problem, and to preserve the privacy of the participants of crowd-sensing, we propose a semi-supervised deep autoencoder based algorithm, which uses both labeled and unlabeled data for training.

Since we use both labeled and unlabeled data, to study their characteristics, this problem is changed to [26]:

$$\mathbf{H\Phi} = \mathbf{X} \tag{12}$$

where $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N] \in \Re^{n \times N}$ represents N data including N_1 labeled data $(\mathbf{x}_i, \mathbf{y}_i)$; $\mathbf{x}_i = [x_{i1}, x_{i2}, ..., x_{in}]^T \in \Re^n$, $\mathbf{y}_i = [y_{i1}, y_{i2}, ..., y_{im}]^T \in \Re^m$ and N_2 unlabeled data $\mathbf{x}_j = [x_{j1}, x_{j2}, ..., x_{jn}]^T \in \Re^n$. If $\mathbf{x}_j$ is a labeled data and belongs to class k, the k^{th} value of $\mathbf{t}_j$ is set to be 1 and the rest $(m-1)$ is set to be -1. The label vector of unlabeled data is set with its all elements to be

0. Therefore, the label matrix is $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, .., \mathbf{y}_{N_1}, \mathbf{0}, ..., \mathbf{0}] \in \Re^{m \times N}$, where $\mathbf{0} \in \Re^m$. In order to extract discriminative features from training data, we use deep autoencoder network. The feature matrix, $\mathbf{H}$, is obtained by training a deep autoencoder network, rather than using $\mathbf{H}$ in (10), that is generated by random weights and biases.

After feature extraction by neurons in hidden layers of deep autoencoder network, these features are put into the classifier. Assuming that $\mathbf{F} = \mathbf{H}\mathbf{\Phi}$, the training aims to find an optimal classification function $\mathbf{\Phi}$ optimizing as:

$$\min_{\varphi} \frac{1}{2}||\mathbf{F} - \mathbf{Y}||^2 + \frac{\gamma}{2}||\mathbf{\Phi}||^2 \tag{13}$$

The second term $\frac{\gamma}{2}||\mathbf{\Phi}||^2$ is the ℓ_2 norm regularization which is added to guarantee the generalization ability, and γ is a balance factor that controls the influence of the manifold regularization. Besides ℓ_2 norm regularization, Laplacian regularization is brought into (13) to provide a closed form solution to an optimization problem:

$$\min_{\varphi} \frac{1}{2}||\mathbf{F} - \mathbf{Y}||^2 + \frac{\gamma}{2}||\mathbf{\Phi}||^2 + \lambda Tr\left(\mathbf{F}^T\mathbf{L}\mathbf{F}\right) \tag{14}$$

where λ is a balance factor to control the influence of the manifold regularization. Here $Tr(.)$ is the trace of the matrix and $\mathbf{L}$ is Laplacian matrix. The graph Laplacian is defined as $\mathbf{L} = \mathbf{D} - \mathbf{W}$, where the degree matrix $\mathbf{D}$ is a diagonal matrix and each diagonal element is given by $\mathbf{D}_{ii} = \sum_{i=1}^{N} \mathbf{W}_{ij}$. $\mathbf{W}$ is the weight edge matrix. When the edge weights are not naturally defined, one common way is via Gaussian kernel [43]:

$$\mathbf{W}(i,j) = \exp\left(-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2\varsigma^2}\right) \tag{15}$$

where ς is the kernel parameter for location coordinates.

By considering $\mathbf{F} = \mathbf{H}\mathbf{\Phi}$, the final goal of the semi-supervised classifier becomes [26]:

$$\begin{aligned} l_2 = \min_{\varphi}\{&||\mathbf{J}\mathbf{H}\mathbf{\Phi} - \mathbf{Y}||^2 + \gamma||\mathbf{\Phi}||^2 + \\ &\lambda Tr\left((\mathbf{H}\mathbf{\Phi})^T\mathbf{L}\left(\mathbf{H}\mathbf{\Phi}\right)\right)\} \end{aligned} \tag{16}$$

where $\mathbf{J} = diag\,(1, 1, ..., 1, 0, 0, ..., 0) \in \Re^{N \times N}$ with the first N_1 diagonal elements set to 1, and $\mathbf{Y} \in \Re^{N \times m}$ is the corresponding label matrix. The derivative of $\mathbf{\Phi}$ with respect to l_2 is:

$$\frac{\partial l_2}{\partial \mathbf{\Phi}} = (\mathbf{JH\Phi} - \mathbf{Y})^T \mathbf{JH} + \gamma \mathbf{\Phi}^T + \lambda (\mathbf{H\Phi})^T \mathbf{LH} \tag{17}$$

The optimum solution of $\mathbf{\Phi}$ is given by [26]:

$$\frac{\partial l_2}{\partial \mathbf{\Phi}} = 0 \rightarrow \mathbf{\Phi} = \left(\gamma \mathbf{I} + \mathbf{H}^T \left(\mathbf{J} + \lambda \mathbf{L}\right) \mathbf{H}\right)^{-1} \mathbf{H}^T \mathbf{JY} \tag{18}$$

Similar to ELM, the corresponding locations for an input data matrix as $\mathbf{X}_D$ are calculated by $\mathbf{H\Phi} = \mathbf{Y}_D$, in that $\mathbf{H}$ is the feature matrix obtained by training a deep autoencoder network, $\mathbf{\Phi}$ is the optimum classification function in (18), and $\mathbf{Y}_D$ is the corresponding label matrix of $\mathbf{X}_D$.

The classifier parameters in Eq. (18), (γ and λ), affect the overall localization performance, so they need to be tuned. Besides, to study the influence of deep autoencoder network depth on the performance, we increase the depth by 1 and adopt the same procedure for parameter optimization, to calculate the localization performance.

4. PERFORMANCE EVALUATION

In this section, we describe the simulation setup to test our purposed algorithm for fingerprint data adopted from transmitter sensors. In order to generate the RSS values at different RPs to construct the fingerprint database, the large scale path-loss model with lognormal shadowing is used [51]:

$$PL(d) = PL(d_0) - 10n\log_{10}\frac{d}{d_0} - \sum_{p=1}^{P} WAF(p) + X(\sigma) \tag{19}$$

where d_0 is the reference distance and d is the distance between transmitter sensors and the RP. $PL(d_0)$ is the received signal in the reference distance and its value for different transmitter sensors are different. Moreover, n is the path-loss exponent and $X(\sigma)$ is a zero mean Gaussian random variable with the standard deviation σ, which models the shadowing effect. Also WAF is the

wall attenuation factor and P is the number of walls. In this simulation we consider $10dB$ power loss for each wall. All simulations are carried out using MATLAB 2016(a) software. These simulations are performed on a platform with an Intel(R) Core i3-4160 CPU @3.60GHz, and 8 GB RAM.

4.1. MODEL GENERATION

Experiments are conducted in a rectangular shape laboratory of $40 \times 29m^2$, including 19 rooms. The allocated transmitter sensors in this environment are shown in Fig. 1. We simulate such an environment to generate 5000 fingerprint data for localization. 1200 of them were randomly chosen as testing data and the rest became training data. As mentioned above, the number of training data

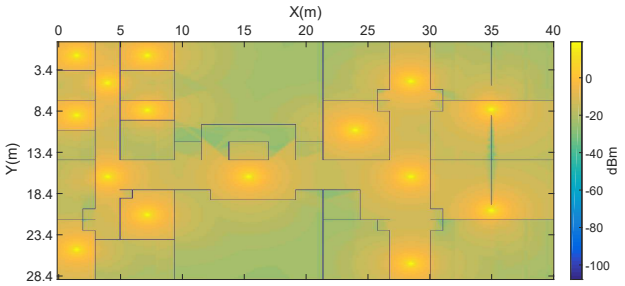

Figure 1: Simulation area layout with transmitters' coverage.

affects the localization performance. We use a limited number of labeled data and a large number of unlabeled data for training. We increase the number of training data gradually, and show how the labeled and unlabeled training data can affect the proposed fingerprint based localization performance.

To verify the performance of the proposed algorithm, at first we set up an effective model. We construct the ASDELM network with 3 hidden layers in which the numbers of neurons are 300,500 and 700, respectively. In the first step, we set a limited number of data from the training database to learn the network and calculate the localization performance of the test data. Then, in order to show how the increasing training data improves the localization performance, we add extra training data and calculate the localization performance. In this

paper, we have considered some rooms in the area, and we want to determine that each user is located in which room (classification of users in terms of rooms). Therefore, if the room number which we estimate for a special user is correct, we say that the localization is successful, and if the estimated room is not correct, we say that the localization is not successful. So, in this work the correct localization means the correct estimate of room number for a user. The localization success rate is defined as $N_{correct}/N_{total}$, where $N_{correct}$ and N_{total} are the number of correct localizations and the number of total localization, respectively.

4.2. LOCALIZATION PERFORMANCE

In this subsection, we evaluate the localization performance of our proposed method. The parameters γ and λ are optimized by hierarchical optimization mechanism. For this purpose, firstly we assign empirical values to γ, and then we tune the other parameter. After optimizing parameters in the λ, we adjust the γ. By this hierarchical adjustment mechanism, the optimal values for classifier parameters are $\gamma = 0.01$, and $\lambda = 0.01$. After the parameters are optimized, we test the localization performance. For this purpose, in the first step, we use 158 labeled data from the training set to train the network and measure the localization accuracy for the test data. In the remainder of this article, wherever we use the accuracy, we mean the success rate. In order to investigate the effect of increasing the number of the training data on the localization performance, we add 300 extra data in each step to training data and measure the localization accuracy, again. At each step, 30% of the added training data are labeled data and the rest are unlabeled. As shown in Fig.2, in the first stage, our proposed algorithm leads to 77.95% success in localization. By increasing the number of training data, including unlabeled data and a limited number of labeled data in the next stages, it can be seen that the localization performance is improved. In the last stage, the localization success exceeds 89.42%. As can be seen, our proposed algorithm can improve the localization success rate by about 24% than our previous Autoencoder based Deep Extreme Learning Machine

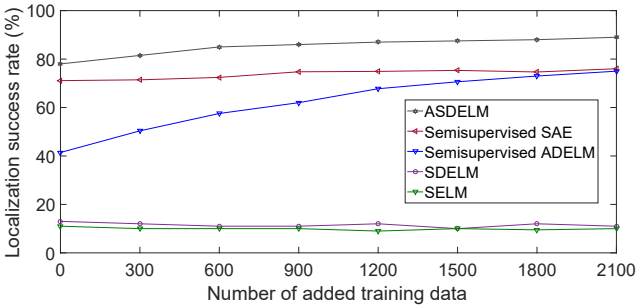

Figure 2: Localization performance with 158 primary labeled training data and adding extra training data for semi-supervised localization methods, where in each step 30% of added data are labeled and the rest of them are unlabeled.

Table 1: Localization success rate for different semi supervised methods.

Method name	ASDELM	semi-supervised ADELM	semi-supervised SAE	SDELM	SELM
Success rate(%)	85.90	61.73	73.82	11.40	9.88

(ADELM) method in semi-supervised way and more than 70% compared with the traditional SELM and SDELM.

Moreover, to compare our proposed algorithm localization performance in a supervised way with other supervised algorithms, we apply our proposed algorithm and some conventional supervised localization methods in the same scenario. For this purpose, in the first step we use 158 labled training data and calculate the localization performance. In order to update the training data, we have increased the number of training data by adding labeled data in several steps. As shown in Fig.3, our proposed method in supervised way, has better localization performance compared with other conventional supervised localization methods.

In Table I, the mean of localization success rate for the proposed method (ASDELM) is compared with the performance of some different semi-supervised methods. Also, in Table II, the mean of localization success rate for the supervised ASDELM is compared with the performance of some different supervised

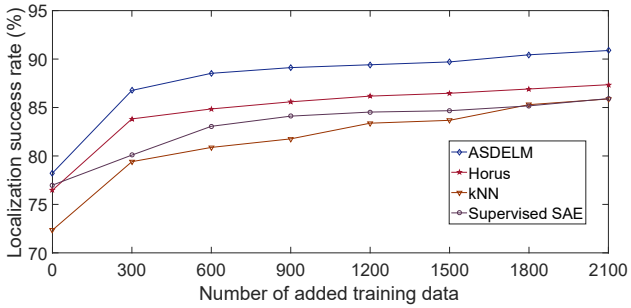

Figure 3: Localization performance with 158 primary labeled training data and adding extra labeled training data for supervised localization methods.

Table 2: Localization success rate for different supervised methods.

Method name	supervised ASDELM	Horus	supervised SAE	k-NN
Success rate(%)	87.73	84.58	83.16	81.58

methods.

Moreover, we have compared the required time of ASDELM with other methods in Table III. For this purpose, we have considered the same scenario that is considered in paper. In the first step, we have 158 primary labeled data and in each next step, 300 extra data is added, which 30% of them are labeled for semi-supervised methods and for supervised methods all of added training data in each step are labeled. Although the measured time for each method is for total training and testing phases, but most of this time is for training phase. In other words, after learning the network and extracting the feature matrix, testing phase is done quickly. Moreover, it should be noted that, though the training time for our proposed algorithm is more than SELM and SDELM, our proposed method achieves about 86% localization accuracy, but SELM and SDELM localization accuracy for the same scenario is less than 12%. In comparison with k-NN and Horus, though ASDELM training time is more, but this method is semi-supervised which saves the time and cost, while k-NN and Horus are supervised and they need labeled data, which collecting of them is time

Table 3: Total training and testing time (localization time) for different localization methods.

Method name	ASDELM	ADELM[39]	SELM	SDELM	Horus	k-NN
Localization time(sec)	3031	3804	50	60	957	7

consuming and labor intensive.

In order to investigate the effect of the number of labeled training data on the localization performance, another scenario is considered. In the first step, 67 training data are used and the localization performance is calculated. In order to update the fingerprint database, we add extra training data, step by step; in each step 90 extra data are added. As shown in Fig.4, with the increase of labeled data ratio, the localization performance increases. The case in which all of added training data are labeled, is a supervised algorithm. As depicted in Fig.4, with a slight loss in localization performance compared with the supervised case, we can save the time and cost of collecting labeled data by adding unlabeled data for training purposes.

Moreover, in order to compare the accuracy of our proposed method in supervised and semi supervised cases with the Horus, k-NN and supervised and semi-supervised SAE, for different RSS measurement noises, we have calculated the success rates of these methods for the case with 158 training data. The RSS noise is generated based on factors such as human movements, variable temperature, and transmitter/ receiver noise. In order to model the effects of mentioned factors, we consider RSS noise as a Gaussian model. In each step we have added the same noise to the data. In supervised methods, all of the training data are labeled. However, in semi-supervised method 28% of 158 data (45 data) are labeled and the rest are unlabeled. As shown in Fig. 5, with a 15% drop in accuracy, we have saved the time and cost by using just 28% labeled data. As shown in Fig. 5, by increasing the σ of the Gaussian noise, our proposed method in supervised and semi-supervised cases, have good resistance to RSS noise in the environment compare with the Horus method.

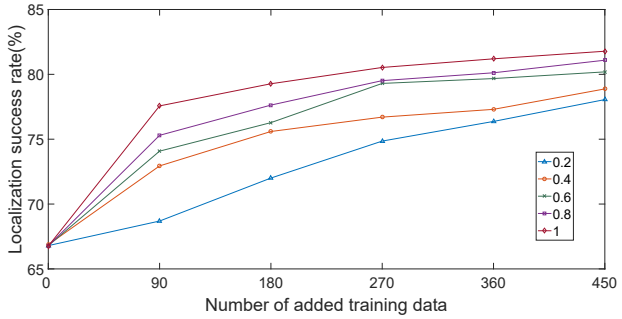

Figure 4: ASDELM based localization performance with 67 primary training data and adding extra training data with different fraction of labeled data.

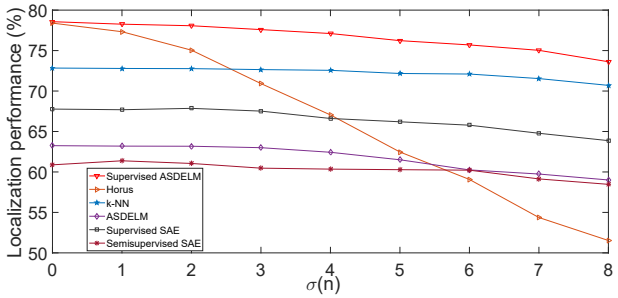

Figure 5: Accuracy of the different methods for different RSS noise levels.

4.3. LOCALIZATION PRACTICAL PERFORMANCE

In this subsection, localization accuracy of a real-scenario experiments is calculated. For this purpose, we collected RSS data from available transmitters in the area. By using an android application which was installed on a Sony Xperia ZR C5502 smart phone, the power of received signals from available transmitters were collected, according to Fig. 6.

We consider a laboratory area of $24 \times 16m^2$, including 5 rooms and 6 transmitter sensors on the second floor of the Cyberspace Research Institute of Shahid Beheshti University. The rooms and locations of the transmitter sensors are shown in Fig. 7. For data collection, in each point we have collected the received powers for 30 seconds (totally about 10 samples from each transmitters in the area in this 30 seconds) in 2 different days. After collecting data, we have

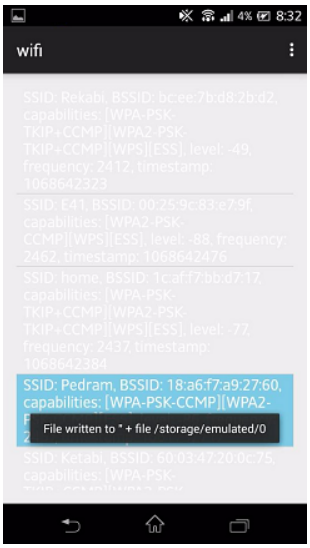

Figure 6: Screenshot of data collection application.

calculated the average of these data taken in 30 seconds and have considered it as the fingerprint data of that point. We have used the collected data of the first day as training data and we have tested our method by 47 randomly selected data from collected data of the second day.

To verify the practical performance of our algorithm, we have used the collected data for training and testing. For this purpose, we constructed the same ASDELM network with 3 hidden layers in which the number of neurons are 300-500-700, respectively. In the first step, we used 29 labeled data to train the ASDELM network. Then in each of the next steps, we added 25 extra data to update the training data, from which 8 were labeled and 17 were unlabeled. Fig. 8 shows the location of labeled, unlabeled and test data in the laboratory environment. As expected, our proposed approach was applicable for practical environments. In order to compare the performance of our proposed method with another semisupervised method, we apply our ASDELM and SAE method in the same scenario. As shown in Fig. 9, in first step using only 29 labeled training, the localization success rate of ASDELM was 77%, and with the increase of the number of training data, including labeled and unlabeled data,

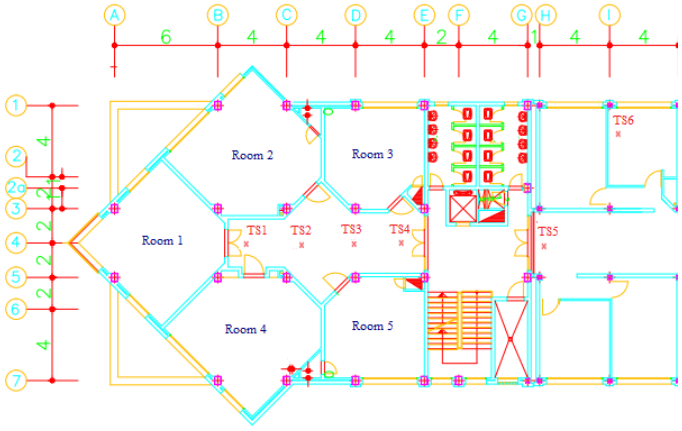

Figure 7: Layout of the environment and transmitter sensors.

the localization performance was improved. By updating training data with the new fingerprint data, in the last step, the localization performance was about 83.1%. In all steps, ASDELM has better localization performance compared with semisupervised SAE.

Also, in order to compare the performance of our proposed method with the traditional supervised method with using practical data, we apply our ASDELM, Horus, k-NN and supervised SAE methods in the same scenario. For this purpose, in the first step we use 29 labeled training data. In order to update the training data, we add 100 extra labeled training data, step by step. As shown in Fig. 10, our proposed method in supervised way has better localization performance compared with the Horus, k-NN and based localization methods.

5. Conclusion

In this paper, we have proposed an Autoencoder-based Semi-supervised Deep Extreme Learning Machine (ASDELM) indoor localization method, which uses the high level extracted features by autoencoder from the collected RSS measurements using a smartphone with internal sensors. In order to improve the localization performance, we showed that using more training data, including

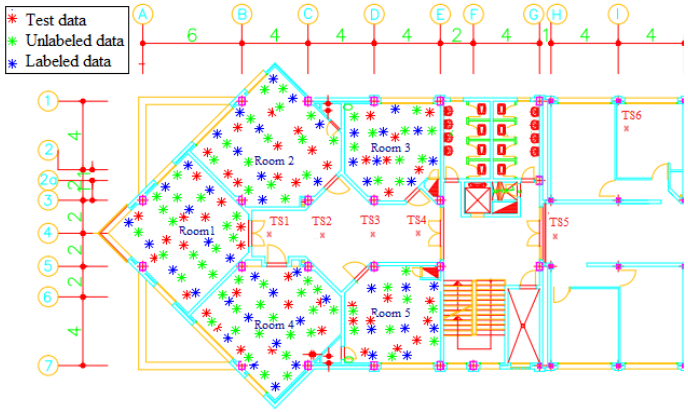

Figure 8: The location of labeled, unlabeled and test data in the laboratory environment.

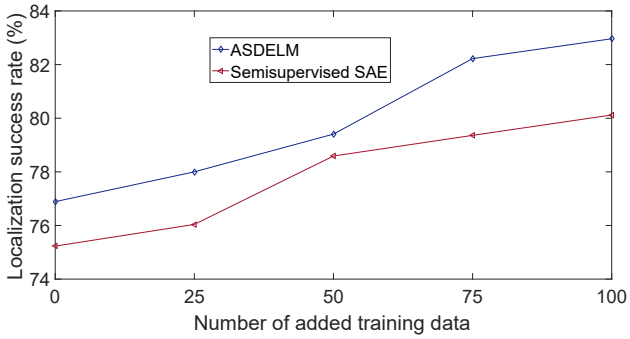

Figure 9: ASDELM based localization performance with 29 primary training data and adding extra training data by utilizing collected data.

unlabeled and a limited number of labeled data, can lead to an improved localization performance. Also using more unlabeled data, can save the time and the cost of collecting labeled data. We compared our proposed algorithm performance with the traditional semi-supervised ELM and semi-supervised DELM, which use random input weights and bias. Simulation results show that our ASDELM method improves the performance, and using more training data including unlabeled data and a limited number of labeled data, further improves the localization performance. This last achievement means that by using smartphones to collect labeled and unlabeled data, we can improve the localization

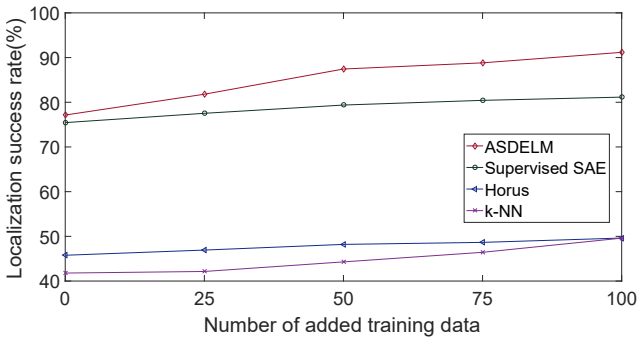

Figure 10: Localization performance with 29 primary labeled training data and adding extra collected labeled training data for supervised localization methods.

performance by updating the training data set.

6. References

[1] V. Moghtadaiee and A. G. Dempster, "Indoor Location Fingerprinting Using FM Radio Signals," *IEEE Trans. Broadcasting*, vol. 60, no. 2, pp. 336-346, Jun. 2014.

[2] Kh. Majieed, S. Sorour, T. Y. Al-Naffouri, and Sh. Valaee, "Indoor Localization and Radio Map Estimation using Unsupervised Manifold Alignment with Geometry Perturbation," *IEEE Trans. Mobile Computing*, vol. 15, no. 11, pp. 2794-2808, Nov. 2016.

[3] S. He and S.-H. Gary Chan, "Wi-Fi fingerprint-based indoor positioning: recent advances and comparisons," *IEEE Communication Survey and Tutorials*, vol. 18, no. 1, pp. 466-490, Jan. 2016.

[4] S. Yiu, M. Dashti, H. Claussen, and F. Perez-Cruz, "Wireless RSSI fingerprinting localization," *Signal Processing*, vol. 131, pp. 235-244, Feb. 2017.

[5] M. Fallah, and V. Pourahmadi, "Graph-based iterative measurement denoising and radio-map generation for semisupervised indoor localisation," *IET Communications*, vol. 12, no. 7, pp. 848-853, Mar. 2018.

[6] G. Deak, K. Curran, and J. Condell, "A survey of active and passive indoor localization systems," *Computer Communications*, vol. 35, no. 16, pp. 1939-1954, Sep. 2012.

[7] X. Huanga, S. Guoa, Y. Wua, and Y. Yang, "A fine-grained indoor fingerprinting localization based on magnetic field strength and channel state information, " *Pervasive and Mobile Computing*, vol. 41, pp. 150-165. Oct. 2017.

[8] X. Wang, L. Gao, Sh. Mao, and S. Pandey, "CSI-based fingerprinting for indoor localization: A deep learning approach," *IEEE Trans. Vehicular Technology*, vol.66, no. 1, pp. 763-776, Jan. 2017.

[9] Y. Chapre, A. Ignjatovic, A. Seneviratne, and S. Jha, "CSI-MIMO: An effcient WiFi fngerprinting using Channel State Information with MIMO," *Pervasive and Mobile Computing*, http://dx.doi.org/10.1016/j.pmcj.2015.07.002

[10] Q. Song, S. Guo, X. Liu, and Y. Yang, "CSI Amplitude Fingerprinting Based NB-IoT Indoor Localization," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 1494 1504, Dec. 2017.

[11] X. Fang, L. Nan, Z. Jiang, and L. Chen, "Fingerprint localisation algorithm for noisy wireless sensor network based on multiobjective evolutionary model," *IET Communications*, vol. 11, no. 8, pp. 1297-1304, Feb. 2017.

[12] M. Bshara, U. Orguner, F. Gustafsson, and L. V. Biesen, "Fingerprinting localization in wireless networks based on received-signal-strength measurements: A case study on WiMAX networks," *IEEE Trans. Vehicular Technology*, vol. 59, no. 1, pp. 283-294, Jan. 2010.

[13] O. M. Elfadil, Y. M. Alkasim, and E. B. Abbas, "Indoor navigation algorithm for mobile robot using wireless sensor networks," in *International Conf. on Communication, Control, Computing and Electronics Engineering, ICCCCEE*, Jan. 2017. pp. 1-5.

[14] H. Teng, K. Ota, A. Liu, T. Wang, and Sh. Zhang, " Vehicles joint UAVs to acquire and analyze data for topology discovery in large-scale IoT systems," *Peer-to-Peer Networking and Applications*, vol. 13, pp. 1720-1743, Jan. 2020.

[15] A. Hajihoseini Gazestani, R. Shahbazian, and S. A.Ghorashi, "Decentralized consensus based target localization in wireless sensor networks," *Wireless Personal Communication*, vol. 97, no. 3, pp 35873599, Dec. 2017.

[16] A. Zhang, Y. Yuan, Q. Wu, Sh. Zhu, and Jian Deng, "Wireless Localization Based on RSSI Fingerprint Feature Vector," *International Journal of Distributed Sensor Networks*, vol. 11, no. 11, pp. 1-7, Nov. 2015.

[17] M. Youssef and A. Agrawala, "Handling samples correlation in the Horus system," in *IEEE INFOCOM 2004*, Hong Kong, Nov. 2004.

[18] Z. Ezzati Khatab, V. Moghtadaiee, and S. A. Ghorashi, "A fingerprint-based technique for indoor localization using fuzzy Least Squares Support Vector Machine," in *Iranian Conf. on Electrical Engineering, ICEE*, May. 2017, pp. 1944-1949.

[19] D. Li, B. Zhang, and Ch. Li, "A feature scaling based k-nearest neighbors algorithm for indoor positioning system," *IEEE Internet of Things Journal*, vol. 3, no. 4, pp. 590-597, Aug. 2016.

[20] H. Dai, W. Ying, and J. Xu, "Multi-layer neural network for received signal strengthbased indoor localisation," *IET Communications*, vol. 10, no. 6, pp. 717-723, Apr. 2016.

[21] J. Patterson and A. Gibson *Deep learning: A Practitioner's approach*, O'Reilly, 2017.

[22] M. Eslami, A. H. Gazestani, and S. A. Ghorashi, "Introduction and Patent Analysis of Signal Processing for Big Data, " *Advances in Parallel Computing*, vol. 33, no. Big Data and HPC: Ecosystem and Convergence, pp. 101-119, 2018.

[23] 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, no. 19, pp. 279-287, Jun. 2016.

[24] S. BelMannoubi, H. Touati, and H. Snoussi, "Stacked auto-encoder for scalable indoor localization in wireless sensor networks," in *International Wireless Communications & Mobile Computing Conference (IWCMC)*, Jun. 2019, pp. 1245-1250.

[25] J, Wang, X. Zhang, Q. Gao, H. Yue, and H. Wang, "Device-free wireless localization and activity recognition: A deep learning approach," *IEEE Trans. Vehicular Technology*, vol. 66, no.7, pp. 6258-6267, Jul. 2017.

[26] Y. Gu, Y. Chen, J. Liu, and X. Jiang, "Semi-supervised deep extreme learning machine for Wi-Fi based localization," *Neurocomputing*, vol. 166, pp. 282-293, Oct. 2015.

[27] Y. Chen, J. Chiang, H. Chu, P. Huang, and A. Tsui, "Sensor-assisted Wi-Fi indoor location system for adapting to environmental dynamics," in Proc. *ACM MSWiM*, 2005, pp. 118125.

[28] J. Zhu, H. Zhao, P. Sun, and Y. Bi, "Equilateral triangle localization algorithm based on average RSSI," *Journal of Northeast University (Natural Science)*, vol. 28, no. 8, pp. 1094-1097, 2007.

[29] W. Xue, W. Qiu, X. Hua, and K. Yu, "Improved Wi-Fi RSSI measurement for indoor localization," *IEEE Sensor Journal*, vol. 17, no. 7, pp. 2224-2230, Apr. 2017.

[30] Y. Xia, L. Ma, Zh. Zhang, and Y. Wang, "Semi-supervised positioning algorithm in indoor WLAN environment," in *Vehicular Technology Conference (VTC Spring)*, May. 2015.

[31] Q. Jiang, Y. Ma, K. Liu, and Zh. Dou, "A probabilistic radio map construction scheme for crowdsourcing-based fingerprinting localization," *IEEE Sensor Journal*, vol. 16, no. 10, pp. 3764-3774, May. 2016.

[32] J. J. Pan, S. J. Pan, J. Yin, L. M. Ni, and Q. Yang, "Tracking mobile users in wireless networks via semi supervised colocalization," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 34, no. 3, pp. 587-600, Mar. 2012.

[33] J. Krumm and J. Platt, "Minimizing calibration efforts for an indoor 802.11 device location measurement system," *Microsoft Research*, Nov 2003.

[34] T. Pulkkinen, T. Roos, and P. Myllymaki, "Semi-supervised learning for wlan positioning," in *International Conference on Artificial Neural Networks*, 2011, pp. 355-362.

[35] B. Yang, J. Xu, J. Yang, and M. Li, "Localization algorithm in wireless sensor networks based on semi-supervised manifold learning and its application," *Cluster Computing*, vol. 13, no. 4, pp. 435-446, Feb. 2010.

[36] J. Tang, Ch. Deng, and G. Huang, "Extreme Learning Machine for multilayer perceptron," *IEEE Trans. Neural Network and Learning Systems*, vol. 27, no. 4, pp. 809-821, Apr. 2016.

[37] J. Liu, Y. Chen, M. Liu, and Zh. Zhao, "SELM: Semi-supervised ELM with application in sparse calibrated location estimation," *Neurocomputing*, vol. 74, no. 16, pp. 2566-2573, Sep. 2011.

[38] X. Jiang, Y. Chen, J. Liu, Y. Gu, and L. Hu. "FSELM: fusion semi-supervised extreme learning machine for indoor localization with Wi-Fi and Bluetooth fingerprints," *Soft Computing*, vol. 22, no. 11, pp. 3621-3635, Jun. 2018).

[39] Z. Ezzati 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, Mar. 2018.

[40] S. Haykin, *Neural network and learning machines*, Pearson Education, 2009.

[41] O. Chapelle, B. Scholkopf, and A. Zien, *Semi-supervised learning*, The MIT Press, 2006.

[42] M. Belkin, P. Niyogi, and V. Sindhwani, "Manifold regularization: A geometric framework for learning from labeled and unlabeled examples," *Journal of Machine Learning Research*, vol. 7, pp. 23992434, Nov. 2006.

[43] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging fileld od signal processing on graphs," *IEEE Signal Processing Magazine*, vol. 30, no. 3, pp. 83-98, May. 2013.

[44] H. Hu, B. Ma, J. Shen, and L. Shao, "Manifold regularized correlation object tracking," *IEEE Trans. Neural Networks and Learning Systems*, no. 99, pp. 1-10, Apr. 2017.

[45] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning,* MIT Press, 2016.

[46] S. Haykin, *Neural network and learning machines,* Pearson Education, 2009.

[47] E. Hosseini-Asl, J. M. Zurada, and O. Nasraoui, " Deep learning of part-based representation of data using sparse autoencoders with nonnegativity constraints," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 12, pp. 2486-2498, Oct. 2015.

[48] B. A. Olshausen and D. J. Field, "Sparse coding with an overcomplete basis set: A strategy employed by V1?," *Vision Research*, vol. 37, no.23, pp. 3311-3325, Dec. 1997.

[49] G. E. Hinton and R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504- 507, Jul. 2006.

[50] M. Zhou, Y. Tang, W. Nie, L. Xie, and X. Yang, "GrassMA: Graph-based semi-supervised manifold alignment for indoor WLAN localization," *IEEE Sensors Journal*, vol. 17, no. 21, pp. 7086-7095, Nov. 2017.

[51] G. Sun and W. Guo, "Robust mobile geo-location algorithm based on LS-SVM," *IEEE Trans. Vehicular Technology*, vol. 54, no. 3, pp. 1037-1041, May. 2005.