

# **Classification for Device-free Localization based on Deep Neural Networks**

Lingjun Zhao

A DISSERTATION  
SUBMITTED IN FULFILLMENT OF THE REQUIREMENTS  
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY  
IN COMPUTER SCIENCE AND ENGINEERING

Graduate Department of Computer and Information Systems

The University of Aizu

2019



© Copyright by Lingjun Zhao

All Rights Reserved.

The thesis titled

*Classification for Device-free Localization based on Deep Neural Networks*

by


Lingjun Zhao

is reviewed and approved by:

**Chief referee**

*Professor*

Chunhua Su

Su Chunhua 

*Professor*

Incheon Paik

Incheon Paik 

*Professor*

Yoichi Tomioka

Yoichi Tomioka



*Professor*

Xiang Li

Xiang Li



*Professor*

Shuxue Ding

丁数学



THE UNIVERSITY OF AIZU

2019





# Contents

<b>List of Abbreviations</b>	<b>vii</b>
<b>List of Symbols</b>	<b>ix</b>
<b>List of Figures</b>	<b>xiv</b>
<b>List of Tables</b>	<b>xv</b>
<b>Abstract</b>	<b>xvi</b>
<b>Chapter 1 Introduction</b>	<b>1</b>
1.1 Wireless Localization . . . . .	1
1.1.1 Overview of wireless localization . . . . .	2
1.1.2 Device-free localization . . . . .	4
1.2 Deep Neural Network . . . . .	6
1.2.1 Restricted Boltzmann machine . . . . .	7
1.2.2 Autoencoder . . . . .	8
1.2.3 Convolutional neural network . . . . .	9
1.3 Thesis Structure . . . . .	10
1.4 Motivation and Main Contributions . . . . .	11
Chapter 2 An accurate and efficient DFL approach based on GBRBMs . . . . .	12
Chapter 3 DFL with CNN by image processing method for in- door and outdoor environments . . . . .	12
Chapter 4 An accurate and robust approach of DFL with CAE .	12
1.5 Publication . . . . .	12
Major Journal paper . . . . .	12
Major Conference Paper . . . . .	13
<b>Chapter 2 An Accurate and Efficient Device-Free Localization Approach Based on Gaussian Bernoulli Restricted Boltzmann Machine</b>	<b>14</b>
2.1 Introduction . . . . .	14
2.2 Problem Statement . . . . .	15
2.2.1 Description of the DFL problem . . . . .	15
2.2.2 Formulation of the DFL as a classification problem . . . . .	16
2.3 Proposed Method . . . . .	17
2.4 Performance Evaluation . . . . .	21
2.4.1 Experiment configurations . . . . .	22
2.4.2 Experiment results and discussion . . . . .	22
2.5 Summary . . . . .	25

<b>Chapter 3</b>	<b>Device-Free Localization with Convolutional Neural Network by Image Processing Method for Indoor and Outdoor Environments</b>	<b>26</b>
3.1	Introduction . . . . .	26
3.2	Related Works . . . . .	29
3.3	Problem Statement . . . . .	30
3.3.1	Description of the DFL problem . . . . .	30
3.3.2	Formulation of the DFL as an image-classification problem . . .	31
3.3.3	Dataset construction . . . . .	32
3.4	Proposed Method . . . . .	32
3.5	Performance Evaluation . . . . .	34
3.5.1	Experiment configurations . . . . .	34
	Description of Indoor Experiments . . . . .	34
	Description of Outdoor Experiments . . . . .	35
3.5.2	Data pre-processing . . . . .	36
3.5.3	Performance of the BE-CNN for indoor DFL . . . . .	38
3.5.4	Performance of the BE-CNN for outdoor DFL . . . . .	40
3.6	Summary . . . . .	43
<b>Chapter 4</b>	<b>An Accurate and Robust Approach of Device-Free Localization with Convolutional Autoencoder</b>	<b>44</b>
4.1	Introduction . . . . .	44
4.2	Related Works . . . . .	47
4.2.1	Device-free localization . . . . .	47
4.2.2	DFL algorithms . . . . .	48
4.3	Problem Statement . . . . .	49
4.3.1	Description of the DFL problem . . . . .	49
4.3.2	Formulation of the DFL as an image-classification problem . . .	50
4.4	Proposed Methods . . . . .	51
4.4.1	Background . . . . .	51
	Autoencoder (AE) . . . . .	51
	Convolutional Neural Network (CNN) . . . . .	52
4.4.2	Convolutional autoencoder (CAE) - the proposed architecture .	53
4.5	Performance Evaluation . . . . .	54
4.5.1	Experiment configurations . . . . .	55
4.5.2	Data pre-processing . . . . .	55
4.5.3	Performance of the CAE in DFL . . . . .	57
	Convolutional Layer Number . . . . .	57
	Convolutional Filter Size . . . . .	58
	Number of Convolutional Filters in Each Layer . . . . .	59
	Max Pooling Layer . . . . .	59
	Learning Rate . . . . .	60
	Epoch Number . . . . .	60
4.5.4	Localization performance and comparison of CAE, CNN and AE	61
	Add Noise into the Training Data . . . . .	62
	Add Noise into the Testing Data . . . . .	63
	Add Noise into Both Training and Testing Data . . . . .	64
	Conclusion of Localization Performance . . . . .	65
4.5.5	Analysis of computational complexity . . . . .	67

4.5.6	Disadvantages and improvements of the CAE . . . . .	68
4.5.7	Effect of different system parameters for the CAE . . . . .	69
	Impact of the Number of Wireless Sensor Nodes in DFL System	69
	Impact of the Number of Training Samples on Localization . . .	70
4.6	Summary . . . . .	70
<b>Chapter 5</b>	<b>Conclusion</b>	<b>72</b>
	<b>Acknowledgment</b>	<b>74</b>
	<b>References</b>	<b>88</b>



# List of Abbreviations

AI	Artificial Intelligence
AE	Autoencoder
ANN	Artificial Neural Network
AP	Anchor Point
BE	Background Elimination
CAE	Convolutional Autoencoder
CD	Contrastive Divergence
CDF	Cumulative Distribution Function
CNN	Convolutional Neural Network
CSI	Channel State Information
DFL	Device-Free Localization
DNN	Deep Neural Network
DV-Hop	Distance Vector-Hop
FP	Fingerprinting
GBRBM	Gaussian Bernoulli Restricted Boltzmann Machine
GPS	Global Positioning System
IoT	Internet Things of Journal
ISTA	Iterative Shrinkage-Thresholding Algorithm
KNN	K-Nearest Neighbor

NLP	Natural Language Processing
RBM	Restricted Boltzmann Machine
ReLU	Rectified Linear Unit
RFID	Radio Frequency Identification
RP	Reference Position
RSSI	Received Signal Strength Indicator
RSS	Received Signal Strength
RTI	Radio Tomographic Imaging
SC	Sparse Coding
SNR	Signal to Noise Ratio
SRC	Sparse Representation Classification
SVM	Support Vector Machine
TDOA	Time Difference of Arrival
TOA	Time of Arrival
UWB	Ultra-wideband
WSN	Wireless Sensor Network

# List of Symbols

$\mathbf{A}$	Matrix of output after convolutional encoder.
$\mathbf{A}_s$	The $s$ -th sub-matrix of $\mathbf{A}$ .
$a$	The entry of output after encoder.
$\mathbf{b}$	Vector of bias terms for visible layers in RBMs.
$b_i$	The $i$ -th entry of $\mathbf{b}$ .
$\mathbf{c}$	Vector of bias terms for encoder layers.
$c_j$	The $j$ -th entry of $\mathbf{c}$ .
$\mathbf{c}'$	Vector of bias terms for decoder layers.
$c'_j$	The $j$ -th entry of $\mathbf{c}'$ .
$d$	The entry of output after decoder.
$D$	The number of anchor points.
$E$	The value of reconstruction error.
$Er$	The value of classification error.
$f$	The non-linear activation function (ReLU).
$F_i^q$	The entry of output after max-pooling associated with $\mathbf{U}_i^q$ .
$\mathbf{h}$	Vector of the hidden layer.
$h_j$	The $j$ -th entry of $\mathbf{h}$ .
$K$	The total number of hidden layers.
$L$	The number of classes of the dataset.

$P$	The number of trials of the DFL experiment.
$\mathbf{R}^{\text{vacant}}$	Matrix of RSS collected when the monitoring area is vacant.
$\mathbf{R}^{\text{target}}$	Matrix of RSS collected with the target in the monitoring area.
$\Delta \mathbf{R}$	Variation matrix of RSS between $\mathbf{R}^{\text{vacant}}$ and $\mathbf{R}^{\text{target}}$ .
$R_{m,n}^{\text{vacant}}$	The entry in the $m$ -th row and $n$ -th column of $\mathbf{R}^{\text{vacant}}$ .
$R_{m,n}^{\text{target}}$	The entry in the $m$ -th row and $n$ -th column of $\mathbf{R}^{\text{target}}$ .
$\Delta R_{m,n}$	The entry in the $m$ -th row and $n$ -th column of $\Delta \mathbf{R}$ .
$R_{m,n}^{\text{vacant}}$	The entry in the $m$ -th row and $n$ -th column of $\mathbf{R}^{\text{vacant}}$ .
$R_{m,n}^{\text{target}}$	The entry in the $m$ -th row and $n$ -th column of $\mathbf{R}^{\text{target}}$ .
$\Delta R_{m,n}$	The entry in the $m$ -th row and $n$ -th column of $\Delta \mathbf{R}$ .
$S$	The number of samples of the entire dataset.
$\mathbf{U}$	Matrix of output after convolution.
$\mathbf{U}_i^q$	Sub-matrix associated with the $q$ -th receptive field of $i$ -th $\mathbf{U}$ .
$\mathbf{V}$	Matrix of the entire dataset.
$\mathbf{V}_{lp}$	Sub-matrix associated with the $l$ -th RP and $p$ -th trial of $\mathbf{V}$ .
$\mathbf{V}_s$	The $s$ -th sub-matrix of $\mathbf{V}$ .
$\mathbf{v}$	Vector of $\mathbf{V}$ .
$\mathbf{v}_{lp}$	The vector associated with the $l$ -th RP and $p$ -th trial of $\mathbf{V}$ .
$\mathbf{v}_s$	Vector indexed for some purpose such as the $s$ -th row or column of $\mathbf{V}$ .
$v_i$	The $i$ -th entry of $\mathbf{v}$ .
$\mathbf{W}$	Matrix of weights between every two encoder layers.
$\mathbf{W}'$	Matrix of weights between every two decoder layers.
$w_{ij}$	The entry in the $i$ -th row and $j$ -th of $\mathbf{W}$ .
$\mathbf{Y}$	Matrix of labels of the dataset.



$\mathbf{Y}'$	Matrix of the predicted results.
$\mathbf{y}$	Vector of $\mathbf{Y}$ .
$\mathbf{y}_s$	Vector indexed for some purpose such as the $s$ -th row or column of $\mathbf{Y}$ .
$\mathbf{y}'$	Vector of $\mathbf{Y}'$ .
$\mathbf{y}'_s$	Vector indexed for some purpose such as the $s$ -th row or column of $\mathbf{Y}'$ .
$\mathbf{Z}$	Matrix of the output after convolutional decoder.
$\mathbf{Z}_s$	The $s$ -th sub-matrix of $\mathbf{Z}$ .
$\boldsymbol{\theta}$	Matrix of parameters in the RBMs.
$\eta$	Value of learning rate for training.



# List of Figures

Figure 1.1	Illustration of wireless localization applications. . . . .	1
Figure 1.2	Illustration of device-free localization with wireless sensors. . . .	5
Figure 1.3	Structure of a simple Deep neural network (DNN). . . . .	7
Figure 1.4	Structure of a restricted Boltzmann machine (RBM) and illustration of the stacked RBMs. V1 and V2 are short for visible layer 1 and 2; H1 and H2 are short for hidden layer 1 and 2. . . . .	8
Figure 1.5	Structure of a simple autoencoder (AE). . . . .	9
Figure 1.6	Structure of a simple convolutional neural network (CNN). . . . .	10
Figure 1.7	Structure of this dissertation. . . . .	10
Figure 2.1	Illustration of a device-free localization (DFL) system model. . .	16
Figure 2.2	Illustration of system architecture and working flow for device-free localization (DFL) via GBRBM-based autoencoder (GBRBM-AE). . . . .	17
Figure 2.3	The architecture of encoder network. . . . .	20
Figure 2.4	Illustration of real DFL experiment setup. . . . .	21
Figure 2.5	Classification accuracy of GBRBM-based autoencoder (GBRBM-AE) in training stage by using data with different dimensions of 10, 15, 20, 35 and 200-dims, respectively. . . . .	22
Figure 2.6	Localization performance of GBRBM-based autoencoder (GBRBM-AE) in testing stage with dimensional data from 2-dims to 200-dims. . . . .	23
Figure 2.7	Cumulative distribution function of localization accuracy of autoencoder and GBRBM-based autoencoder (GBRBM-AE) with low dimensional data of 15-dims. . . . .	24
Figure 2.8	Localization performance of GBRBM-based autoencoder (GBRBM-AE) in the testing stage on noisy data with different SNR. . . . .	25
Figure 3.1	Internet-of-Things (IoT) fundamental blocks with device-free localization (DFL) technique for the indoor scenario of intrusion detection and monitoring in a smart city. . . . .	27
Figure 3.2	Illustration of the framework of the proposed background elimination pre-processing based convolutional neural network (BE-CNN). . . . .	28
Figure 3.3	Illustration of a device-free localization (DFL) system model and description of the background elimination (BE) and imaging process of the RSS matrix when a target is at a certain reference position (RP). . . . .	31
Figure 3.4	Schematic architecture of the convolutional neural network (CNN). . . . .	33

Figure 3.5	Schematic diagram of the convolutional filter concatenation. . . .	33
Figure 3.6	Experimental layout of the indoor DFL scenarios in the living room and the corridor. There are totally 9 reference positions (RPs) in the experimental scenarios. . . . .	35
Figure 3.7	Experimental layout of the outdoor DFL scenario. . . . .	36
Figure 3.8	Comparisons of raw RSS matrices (Top row) and background eliminated RSS matrices (Bottom row). Here, the data of reference position 2 (RP2) and reference position 4 (RP4) are taken as the examples. . . . .	37
Figure 3.9	Featured images of noiseless signal and noisy signal with different SNR. Here, the outdoor data are taken as the examples, corresponding to the reference position 4 (RP4). . . . .	37
Figure 3.10	Indoor localization performance of BE-CNN, BE-KNN and BE-SVM on the noiseless and noisy data. . . . .	39
Figure 3.11	Localization performance comparisons by using CNN, KNN, and SVM. Here, all data are with noise. BE is short for background elimination. . . . .	41
Figure 3.12	Comparisons of localization accuracy by employing different numbers of sensors. Here, the distance between the adjacent sensors is 3 feet, 9 feet, and 12 feet for sensor numbers of 28, 8, and 7, respectively. . . . .	42
Figure 4.1	Fundamental blocks of IoT with respect to the device-free localization (DFL) model. Here, WSN is short for wireless sensor network. . . . .	45
Figure 4.2	Illustrations of data patterns and the flowchart of the system. . . .	46
Figure 4.3	Illustration of a device-free localization (DFL) system model. . . .	50
Figure 4.4	Schematic of a convolutional autoencoder (CAE) architecture. All dashed circles in the fully connected layer indicate randomly dropped neurons. . . . .	52
Figure 4.5	Imaging process of RSS matrices from original signals to variation signals at two different reference positions (RPs). Note that different colors in the RSS images reveal the values of the corresponding RSS matrices. The unit for each value of the color bar is dBm. . . . .	56
Figure 4.6	Imaging of noisy data collected at the reference position 4 (RP4) with different SNRs. . . . .	56
Figure 4.7	Cumulative distribution function of localization accuracy on noisy testing data by using CAE with different convolutional layers. . . .	57
Figure 4.8	Localization performance of the convolutional autoencoder (CAE) with different optimization parameters. For (a) and (b), testing signals are noisy. . . . .	58
Figure 4.9	Localization accuracy of convolutional autoencoder (CAE), convolutional neural network (CNN) and autoencoder (AE) on noisy training data with SNR= 0 dB. . . . .	62
Figure 4.10	Reconstruction performance of convolutional autoencoder (CAE) compared with autoencoder (AE). Here, the signals of reference position 4, 16 and 24 are taken as examples. . . . .	63

Figure 4.11	Comparison of the average localization accuracy among the convolutional autoencoder (CAE), convolutional neural network (CNN) and autoencoder (AE) on noisy testing data. . . . .	64
Figure 4.12	Comparison of the average localization accuracy among the convolutional autoencoder (CAE), convolutional neural network (CNN) and autoencoder (AE) on the noisy dataset . . . . .	65
Figure 4.13	Confusion matrix of different reference positions (RPs) for the convolutional autoencoder (CAE). The sample shows the experiment result of the localization performance on the noisy dataset in the testing stage with the SNR equal to 0 dB. The overall accuracy is 98.9%. . . . .	66



# List of Tables

Table 2.1	Localization accuracy compared with other methods. . . . .	25
Table 3.1	Optimal parameters of the BE-CNN for indoor DFL system. . . .	38
Table 3.2	Impact of the number of training samples on localization accuracy.	38
Table 3.3	Optimal parameters of the BE-CNN for outdoor DFL system. . . .	40
Table 3.4	Localization accuracy compared with other methods. . . . .	41
Table 4.1	Optimal parameters of the CAE neural network . . . . .	61
Table 4.2	Comparison of Localization performance . . . . .	67
Table 4.3	Computational Complexity for All Methods . . . . .	68
Table 4.4	Impact of the number of sensors for the CAE . . . . .	69
Table 4.5	Impact of the number of packets for the CAE . . . . .	70





# Abstract

Wireless sensor networks (WSNs) has spawned a variety of emerging applications in recent years, such as location-based services in smart city, patient or aging healthcare-monitoring in smart home, mobile robot localization in smart factory and so on. Among the conventional device-based wireless localization techniques, e.g., global positioning system (GPS), the target must be attached with wireless devices or tags. However, this may not be applicable to some emerging scenarios. For example, in the security safeguard, one cannot usually pre-equip a traceable device on the intruder for monitoring the locations of the targets. To address this kind of problems, a new kind of wireless localization technology, named device-free localization (DFL), is proposed.

Contrasting with the conventional localization technology, DFL does not need the targets carry any electronic devices. Therefore, it has recently attracted the extensive attentions. The focus of this thesis presents contributions to the field of DFL based on deep neural networks (DNNs). The organization and main contributions of this dissertation are briefly summarized in the following manner.

Chapter 1 first offers an introduction and background of the DFL problem. Then gives an overview of DNNs and some other architectures that are used to solve the DFL problems in this thesis.

Chapter 2 describes an outdoor DFL experiment and an accurate approach based on a pre-trained autoencoder (AE) to address the DFL problem. In this algorithm, multiple Gaussian Bernoulli restricted Boltzmann machines (GBRBMs), a variant of RBMs, are utilized for the pre-training of the AE. The GBRBMs-based AE (GBRBMs-AE) extracts discriminative features from the received signal strength indicator (RSSI) measurements automatically to characterize the target's location. In addition, this algorithm is also used for dimensionality reduction, which reduce the effects of noise as well as the time cost for locating.

Chapter 3 describes two designed real testbeds of indoor DFL scenarios and an outdoor DFL experiment. In addition, this chapter also presents a method, called background elimination (BE), to pre-process the collected RSSI measurements. In this chapter, the DFL problem is formulated as an image classification problem; moreover, Chapter 3 presents an BE pre-processing based convolutional neural network (BE-CNN) to perform target locating for both indoor and outdoor DFL.

In order to improve the accuracy of DFL, Chapter 4 presents an accurate and robust approach of DFL based on convolutional autoencoder (CAE). The proposed approach performs unsupervised feature extraction from raw signals followed by supervised fine-tuning for classification. The CAE combines the advantages of an CNN and an AE in the feature learning and signals reconstruction.

Finally, Chapter 5 summarizes our contributions in this thesis and presents some directions for the further research.



# Chapter 1

## Introduction

### 1.1 Wireless Localization

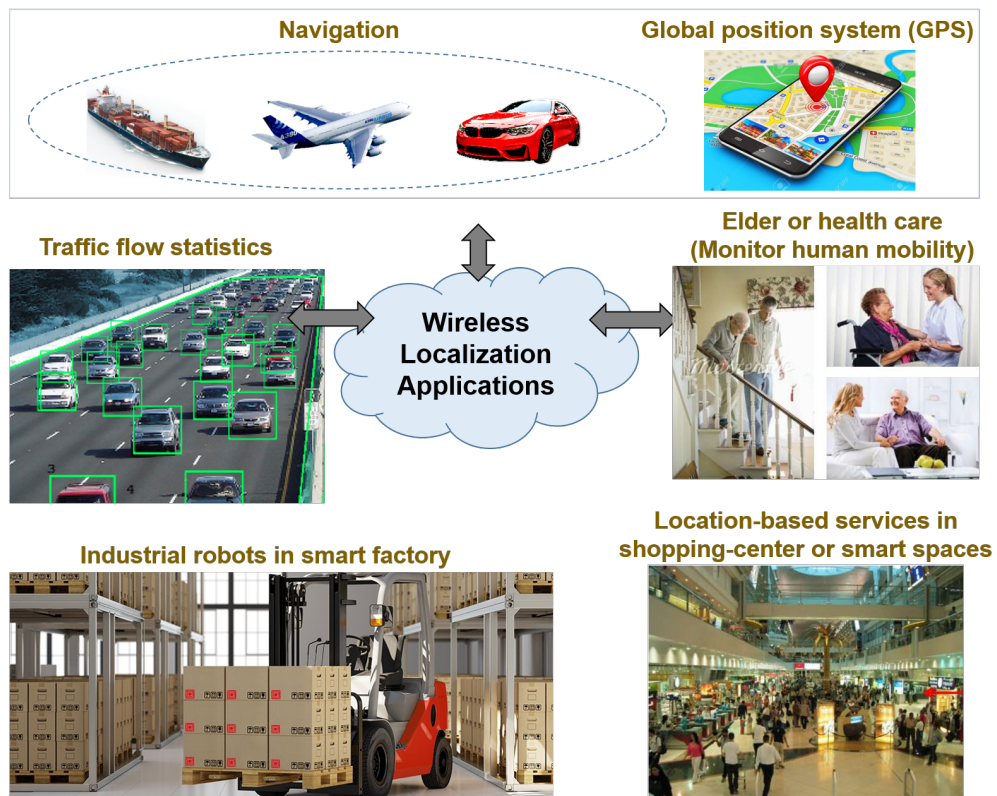


Figure 1.1: Illustration of wireless localization applications.

Localization means obtaining the location information of the targets in a coordinate system by measuring a series of necessary parameters. Wireless localization employs wireless sensors to locate the objects by measuring the wireless links among the sensors. Due to the advances in wireless communications and electronics, the wireless sensors have been developed to low-cost, low-power, small size and multi-function. The recent advances of wireless sensors enable to employ the wireless localization technology in

a large number of location-based services, such as navigating for people, guiding autonomous vehicles, tracking patients in the health-care center, etc. Because of the location information is being more and more important in modern life, wireless localization technologies have draw a great deal of attention.

### **1.1.1 Overview of wireless localization**

Wireless localization technologies are including global positioning system (GPS) [1], radio frequency identification (RFID) localization technology [2], ZigBee localization technology [3], ultra-wideband (UWB) localization technology [4], Wi-Fi localization technology [5], etc.

GPS is a satellite-based global navigation system which is freely accessible to anyone with a GPS receiver around the world. The GPS provides reliable continuous geolocation, speed, course and time information to military, civil, and commercial users especially in outdoor environment. However, in some conditions that there are many mountains or buildings, GPS signals are weak which result in less helpful for indoor localization.

RFID is a technique which retrieves and stores data by electromagnetic transmission to a radio frequency compatible integrated circuit. An basic RFID system consists of a series of RFID readers, RFID tags, and the communication between them. The RFID reader can read data emitted from RFID tags. Moreover, RFID readers and tags use a defined radio frequency and protocol to transmit and receive data. SpotON system and LANDMARC system are two well-known RFID localization system, who realize localization by analyzing signals strength and using reference labels, respectively.

ZigBee is a low-power, low-rate, and close proximity wireless ad hoc network which is developed based on IEEE 802.15.4 protocol. This technique is typically applicable in low data rate applications that require long battery life and secure networking, such as home automation and medical device data collection. UWB is a radio technology that can make short-range and high-bandwidth communications with very low power. Compared with other radio frequency technologies, UWB is high-accuracy, high-security and high interference immunity.

For recent years, the Wi-Fi localization technology has more advantages than some other wireless localization technologies. It is because that large numbers of intelligent mobile devices, such as personal computers, smart phones, printers, MP3 players, video game consoles and laptops, have been equipped with Wi-Fi transceivers which indeed cut the cost of Wi-Fi-based localization. Although the Wi-Fi technique can transmit signals with high speed, it is power consuming compared with other techniques, such as ZigBee.

With the rapid development of wireless communication technology, related infor-

mation processing technology and hardware technology, many kinds of wireless communication mechanisms are emerging, such as ZigBee, UWB, near field communication, etc. Based on different wireless communication mechanisms, wireless localization algorithms can be divided into range-based localization algorithms and range-free localization algorithms.

The range-based localization algorithms locate the targets by measuring and calculating the distances or directions between every adjacency wireless sensor nodes. The commonly used ranging-based localization algorithms include the time of arrival (TOA) localization algorithm [6], the time difference of arrival (TDOA) localization algorithm [7] and the received signal strength indicator (RSSI) localization algorithm [8].

In detail, the TOA localization algorithm mainly uses the transmitting time of a radio signal to predict the distance from the predicted sensor node to the reference sensor node. The distance can be directly calculated from the time of arrival as signals travel with a known velocity. However, the TOA has high requirements on the hardware and functions of sensor nodes. Compared with the TOA, the TDOA technique uses the measured time difference between two kinds of signals, i.e., Ultrasonic wave and electromagnetic wave, traveling from a transmitting node to a receiving node. Similar with the TOA, the distance can be calculated from the time difference with the known velocities of the two signals. Note that the TDOA can achieve higher localization accuracy than the TOA. The RSSI localization algorithm is realized based on the collection of RSSI measurements derived by wireless sensor nodes in a monitoring area. RSSI measurements changes with the changing of the communication distance between two sensor nodes, i.e., longer the range is, smaller of the measurement of the RSSI. The power cost can be calculated by measuring the power of the transmitter and the receiver. In addition, the power cost can be converted into distance by radio signal propagation models. Although the RSSI performs well in experimental environments, it is easily affected by temperature, obstacles, etc., in practical environments.

Although the range-based localization algorithms can precisely locate the targets, they require costly devices and consume high energy. Therefore, the range-free algorithms have attracted significant interest. Well known techniques include the centroid localization algorithm [9], the distance vector-hop (DV-Hop) localization algorithm [10], the approximate point-in-triangulation test (APIT) localization algorithm [11], the fingerprinting (FP) localization algorithm [12], etc.

In detail, the centroid localization algorithm was proposed by professor Bulusu in the university of California, which has nothing to do with the distance and connectivity only relevant outdoor localization algorithm [13]. According to the location information of all the connected sensor nodes, the sensor node estimates its location as the the centroid of the polygon composed by the multiple sensor nodes. Although this localization algorithm is easy to perform, it can not locate the targets precisely. With regard

to the DV-Hop localization algorithm, the basic idea behind it is to represent the distance between an unknown target node and an anchor as a product of the average hop size and hop count. Based on the distance, the location of the target sensor node can be calculated by maximum likelihood estimation. Although this localization algorithm has low requirement on the hardware of sensor nodes, it can not realize precise localization. As to the APIT localization algorithm, the basic theory of this algorithm is perfect point-in-triangulation test. It is a method to narrow down the localization area in which a target node resides. In this algorithm, the target node exchanges RSSI with three random selected neighboring nodes, and the target node is judged in the testing triangular pyramid. Repeat testing until all the combination of nodes are employed or the localization accuracy is sufficient high. Compared with the DV-Hop localization algorithm, the APIT algorithm can achieve high localization accuracy. However, it needs the sensor network has strong connectivity.

The FP localization algorithm [14] in essence uses a model to infer physical locations from radio map data. The models are usually either probabilistic or neural networks consisting of one or more layers. Generally, the existing FP localization algorithm including the k-nearest neighbors algorithm (KNN), the probabilistic model, the support vector machine (SVM) and the neural network. The FP algorithm generally includes two phases [15]: the offline phase for fingerprint calibration and the online phase for location estimation. In detail, it collect a set of features (called fingerprint) that estimates location by matching test measurements against previously collected fingerprints. The features collected generally are based on signal analysis, such as RSSI. The FP localization algorithm has better localization performance against the other methods described before because it collects features from the environment.

### **1.1.2 Device-free localization**

Among the wireless localization technologies, some of them, such as GPS, RFID, etc., need the targets to carry wireless electronic devices, e.g., smart phones, RFID tags, which is named device-carried localization technology in this thesis. Although these technologies are popular for tracking and navigation purposes, they have limitations as well. For example, in indoor environments or in cities with tallest buildings, the GPS is no functionality because of the requirement of line-of-sight to communicate satellites. Moreover, the requirement of attached devices with the targets all the times or pre-installing the electronic devices on the targets is also a limitation in some scenarios, such as in the sensitive areas and the emergency rescue. To address this kind of problems, another kind of wireless localization technology, which is called device-free localization (DFL) technology, is proposed.

Contrary to the device-carried localization technology, the DFL technology does

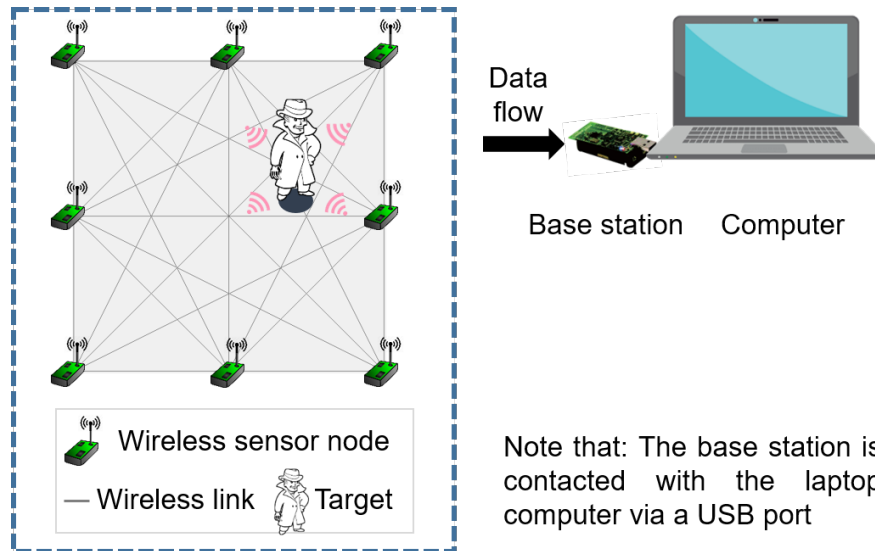


Figure 1.2: Illustration of device-free localization with wireless sensors.

not need the targets carry any electronic devices. Therefore, the DFL technology has recently attracted the attention of the research community [16]. Fig. 1.2 shows a simple example of DFL system with two targets in the monitoring area. As illustrated in this figure, the DFL technology is employed in a pre-designed DFL system. In this system, a number of wireless devices (or sensor nodes) are pre-installed in an environment, which is usually called the monitoring area. Considering the communication among these wireless sensor nodes, the targets in the monitoring can be located by analyzing the signals derived by the targets.

Note that the information transmitted among the sensor nodes are radio signals, such as RSSI. As mentioned in Section 1.1.1, the RSSI, sometimes referred as received signal strength (RSS), is a measurement of the power present in a received radio signal, which is easily acquired by users. In details, the RSSI values are measured in dBm and have typical negative values ranging between 0 dBm and -110 dBm. In addition, the RSSI is an indication of the power level being received by the receive radio after the antenna and possible cable loss. Therefore, the higher the RSSI number, the stronger the signal. Thus, when an RSSI value is represented in a negative form (e.g. 100), the closer the value is to 0, the stronger the received signal has been. In addition, the RSSI can vary and be affected in wireless networks. In other words, it can be affected differently by different target's locations. In a DFL system, sensor nodes are used to sense the target by collaboratively transmitting and receiving radio signals with each other. If the target change its location, it will surely affect some of the wireless signals. That is to say, the target in different locations will derive correspondingly specific RSSI measurements. Therefore, the DFL system can estimate the target's location by the RSSI variations produced by the targets.

Furthermore, because the DFL system is pre-designed, where the positions of the

sensor nodes are unchangeable, the target in a same location will derive similar RSSI measurements. In other words, the RSSI measurements collected in the same position can be seen as the data with a common class; moreover, the RSSI measurements collected in all the potential positions can be seen as the data with classes of all the positions. In this respect the DFL problem can be regarded as a classification problem. To figure out the classification problem, we employ the deep neural network (DNN), an algorithm of the FP localization algorithm, in this thesis as described in Section 1.1.1, which has better localization performance against the other aforementioned methods. The more details about the DNN are described in Section 1.2.

## 1.2 Deep Neural Network

Artificial intelligence (AI) was coined as a term in 1955, by John McCarthy who proposed that the study of AI is “to proceed on the basis of the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it.” [17] Furthermore, the last two decades have witnessed major advances in AI, which is relevant to any intellectual task, such as robotics, natural language processing and computer vision (such as image recognition). Machine learning is a component of AI, and it can be used to solve problems in the field of AI. In addition, machine learning algorithms have been used in a wide variety of applications, such as data mining, detection of network intruders and search computing, where it is infeasible to develop an algorithm of specific instructions for performing the task.

The DNN (or deep learning) is a part of a broader subset of machine learning. In recent years, it has achieved outstanding performance in almost every machine learning task, such as image recognition, automatic machine translation, brain cancer detection, etc. Different with most traditional machine learning algorithms, DNNs can extract features automatically without human intervention [18]. Considering the rapid development of the DNN, many different kinds of neural network architectures exist and are being proposed all the time. Therefore, it is impossible to use one architecture to represent all the DNNs. Although different DNNs have their own structures, there are commonalities among them. In other words, every neural network consists of individual, interconnected neurons, which compose an input layer, an output layer and several hidden layers, as shown in Fig. 1.3. These neurons are also called nodes or units. The neurons in the input layer receive data and then pass through every hidden layer to learn multiple levels of representation and abstraction that help to make sense of data, such as images and text. The calculation of each layer basically includes linear transformation



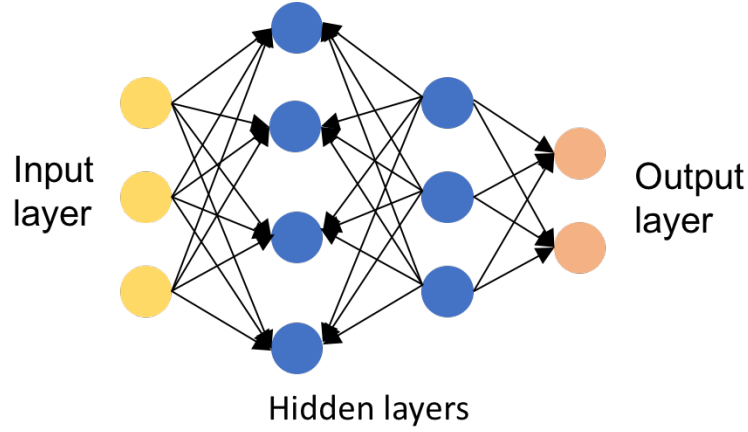


Figure 1.3: Structure of a simple Deep neural network (DNN).

followed by a non-linear activation function, shown as follows:

$$\mathbf{y} = f(W \cdot \mathbf{x} + b) \quad (1.1)$$

where  $\mathbf{x}$  denotes the input of this layer,  $W$  denotes the weight matrix,  $b$  denotes the bias,  $f(\cdot)$  denotes the activation function and  $\mathbf{y}$  denotes the output of the layer. Note that the output of the previous layer will be the input of the next layer. Therefore, a multi-layer DNN is built by chaining together multiple layers.

The toy DNN shown in Fig. 1.3 is a four-layer DNN, containing one input layer, two hidden layers and one output layer. Moreover, it can also be seen as an simple example of the feed-forward neural network, which is the first and simplest type of artificial neural network devised [18]. For a feed-forward neural network, the information passes in only the forward direction, from the input neurons, through the hidden nodes and to the output nodes. The connections between the neurons do not form cycles or loops. Note that an essential factor of DNNs is the mathematics of the hidden layer, which means the architecture of an DNN have a great effect on the types of problems can be solved and the corresponding performance of the neural network. In this thesis, we mainly focus on three kinds of DNNs, the restricted Boltzmann machine (RBM), the Autoencoder (AE) and the Convolutional neural network (CNN).

### 1.2.1 Restricted Boltzmann machine

An RBM [19] is a generative stochastic artificial neural network that can learn a probability distribution over its set of inputs. It is one of the foundational technologies of deep learning. Fig. 1.4 shows the probabilistic graphical model corresponding to an RBM. As their name implies, an RBM is a special type of the more general class of Boltzmann machines, with the restriction that their neurons must form a bipartite graph: a pair of neurons from each of the two groups of neurons may have a symmet-

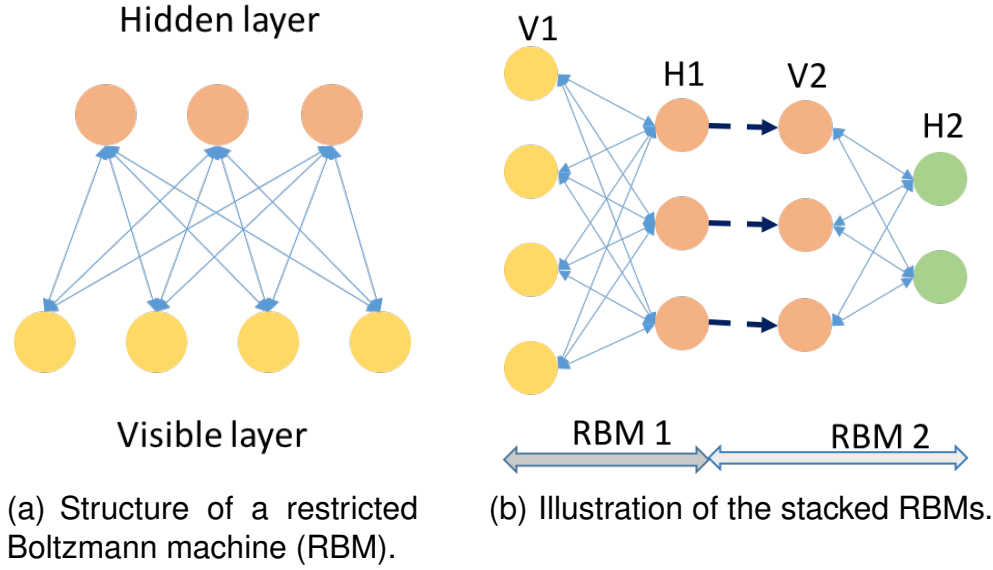


Figure 1.4: Structure of a restricted Boltzmann machine (RBM) and illustration of the stacked RBMs. V1 and V2 are short for visible layer 1 and 2; H1 and H2 are short for hidden layer 1 and 2.

ric connection between them; and there are no connections between neurons within a group. Moreover, the upper level comprised neurons corresponding to hidden variables and the lower level consists of visible neurons. This restriction allows for more efficient training algorithms than are available for the general class of Boltzmann machines, in particular the gradient-based contrastive divergence algorithm [20].

RBMs have found applications in dimensionality reduction [19], classification [21, 22], collaborative filtering [23, 24], feature learning [25], recommendation system [26] and body quantum mechanics [27]. They can be trained in either supervised or unsupervised ways, depending on the task. Furthermore, RBMs can also be used in deep learning networks. In particular, two or more RBMs can be stacked into a DNN. It can be seen from the Fig. 1.4(b) that the visible neurons in the RBM1 receive inputs and the hidden neurons pass their output directly to the visible neurons of the RBM2. There are no weights between the two RBMs. This layer-by-layer training of RBMs is called unsupervised pre-training for the DNN, which can extract features from the data itself without labels. This pre-training procedure is very important to provide better initialized parameters for the fine-tuning of the resulting DNN with gradient descent and backpropagation.

### 1.2.2 Autoencoder

An AE is a type of DNNs, which is typically used for dimensionality reduction and feature extraction in an unsupervised manner. Architecturally, the simplest form of an AE is a feed-forward, non-recurrent neural network very similar to the regular DNN,

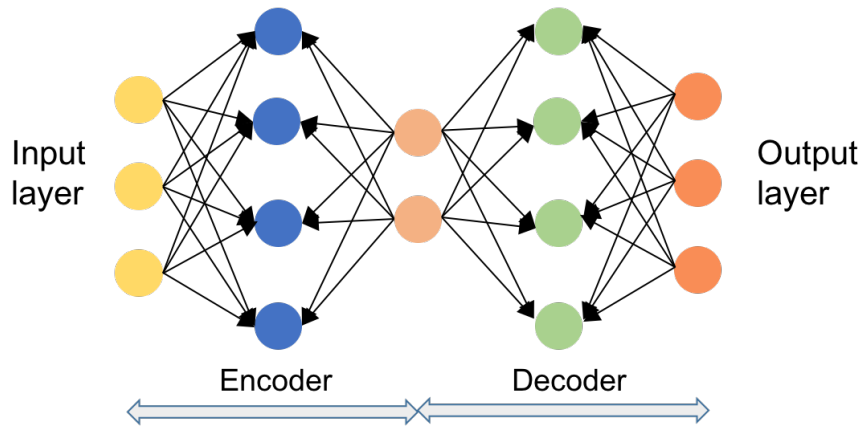


Figure 1.5: Structure of a simple autoencoder (AE).

having an input layer, an output layer and one or more hidden layers connecting them. The different is that the number of neurons of the output layer is exactly same with the input layer, as shown in Fig. 1.5. In detail, an AE consists of two parts, the encoder and the decoder. The output of the encoder is the reduced representation of the input pattern and the decoder aims to reconstruct the representation as close as possible to its original input.

To train an AE we do not need to do anything fancy, just throw the raw input data at it. The AEs are considered an unsupervised learning technique since they don't need explicit labels to train on. But to be more precise they are self-supervised because they generate their own labels from the training data. In particular, the output of encoder is not only a lower dimensional representation but also a representation of input with less redundant information. That is to say, the output of the decoder will not be exactly the same as the input, which makes it possible to learn a representation of input with less noise. Although the AE is an unsupervised learning model, which learn features from the input itself without labels, it can be employed in either supervised or unsupervised manners depending on the tasks. For example, some architectures successfully employ the AEs for data (such as images and texts) recognition and classification [28], image reconstruction [29] and speech generation [30].

### 1.2.3 Convolutional neural network

The CNN [31] is a special kind of DNN, which is most commonly applied to analyzing visual imagery. Like most neural networks, they are trained with a version of the backpropagation algorithm. Where they differ is in the architecture, whose connectivity pattern between neurons is inspired by the organization of the animal visual cortex [32]. Except for the input layer and the output layer, as shown in the Fig. 1.6, the hidden layers of a CNN typically consist of convolutional layers, pooling layers and fully connected layers [31]. In a regular DNN, all the neurons are in a fully-connected

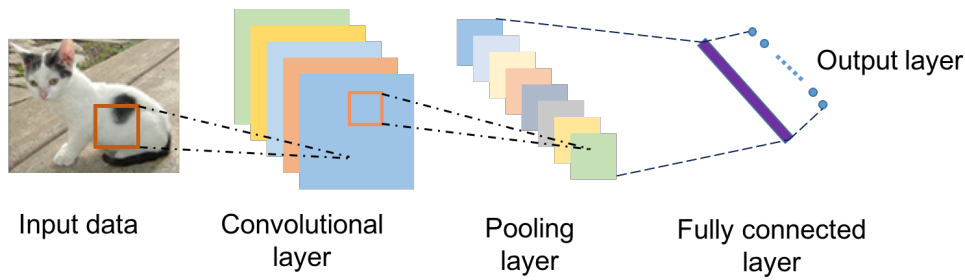


Figure 1.6: Structure of a simple convolutional neural network (CNN).

manner. However, in the CNN, individual neurons in a layer respond to stimuli only in a restricted region of the visual field, known as the receptive field, in the layer before it. A collection of such fields overlap to cover the entire visual area. In addition, the elements involved in carrying out the convolution operation with every receptive field in the convolutional layers is called the kernel (or filter).

The CNN is designed to recognize visual patterns directly from pixel images with minimal pre-processing [33]. In particular, the network can detect features such as edges, lines, blobs of color, and other visual elements. This means that the CNN learns the filters that in traditional algorithms were hand-engineered. This independence from prior knowledge and human effort in feature design is a major advantage. Furthermore, it can recognize patterns with extreme variability (such as handwritten characters), and with robustness to distortions and simple geometric transformations. Benefiting from its architecture, the CNN performs well on learning a hierarchy of visual features and can be trained to understand the sophistication of the image data better. Moreover, it has outstanding performance on image and video recognition [34], image classification [35], medical image analysis [36], and natural language processing [37].

### 1.3 Thesis Structure

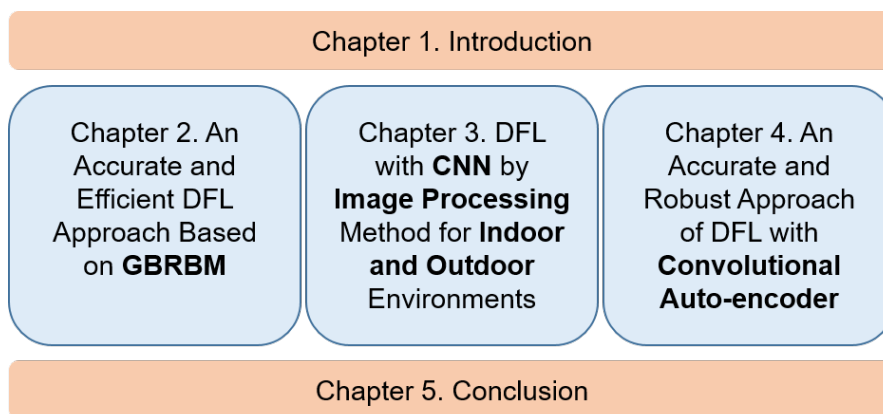


Figure 1.7: Structure of this dissertation.

The focus of the work presented in this thesis is to present a series of contributions to the field of DFL based on DNNs. The structure of this thesis is illustrated in the Fig. 1.7.

Chapter 1 first offers an introduction and background of the DFL problem. It starts from wireless localization technologies and summarizes some popular algorithms to address the localization problems. It then gives some description on DFL technology and a kind of radio signal, RSSI, usually employed in the DFL. Chapter 1 then gives an overview of DNNs and some architectures that are main approaches used to solve the DFL problems in this thesis.

Chapter 2 describes an outdoor DFL experiment and an accurate approach based on a pre-trained AE to address the DFL problem. In this algorithm, multiple Gaussian Bernoulli RBMs (GBRBMs), a variant of RBMs, are utilized for the pre-training of the AE. The GBRBMs-based AE (GBRBMs-AE) extract discriminative features from the RSSI measurements automatically to characterize the target's location. In addition, this algorithm is also used for dimensionality reduction, which reduce the effects of noise as well as the time cost for locating.

Chapter 3 describes two designed real testbeds of indoor DFL scenarios and an outdoor DFL experiment. In addition, this chapter also presents a method, called background elimination (BE), to pre-process the collected RSSI measurements. In this chapter, the DFL problem is formulated as an image classification problem; moreover, Chapter 2 presents an BE pre-processing based convolutional neural network (BE-CNN) to perform target locating for both indoor and outdoor DFL.

In order to improve the accuracy of DFL, Chapter 4 presents an accurate and robust approach of DFL based on convolutional autoencoder (CAE). The proposed approach performs unsupervised feature extraction from raw signals followed by supervised fine-tuning for classification. The CAE combines the advantages of an CNN and an AE in the feature learning and signals reconstruction.

Chapter 5 summarizes our contributions in this thesis and presents some directions for the further research.

## **1.4 Motivation and Main Contributions**

This thesis is a report about three years of research on DFL technology and deep learning. The content described in Chapter 2, Chapter 3 and Chapter 4 are the results of this work. The main contributions of this thesis are summarized as followings.

As an emerging technology, the DFL, using radio frequency sensor networks to detect targets who do not carry any attached devices, has spawned extensive applications in the Internet of Things field. Considering the complex condition of the environment, for DFL, a key problem is how to locate the targets with high accuracy and low time

cost.

## **Chapter 2 An accurate and efficient DFL approach based on GBRBMs**

In this chapter, the DFL problem is formulated as a classification problem, present an GBRBM-AE approach to extract discriminative features automatically from wireless signals. In addition, the proposed approach can obtain high location accuracy with high-dimensional DFL data and is robust to noise, which outperforms the conventional AE on the real-world dataset in both location accuracy and robustness.

## **Chapter 3 DFL with CNN by image processing method for indoor and outdoor environments**

In this chapter, the collected RSSI matrices are regarded as image matrices and pre-processed by eliminating the background influences; moreover, the DFL problem is formulated as an image classification problem. Two kinds of indoor testbeds, one outdoor experimental scenario are performed, and the CNN architectures for each scenario are designed. In the locating stage, the BE-CNN can output the location of the target automatically according to the input signal. The localization performance of the BE-CNN is verified that outperforms SVM and kNN on the real-world datasets in both indoor and outdoor DFL.

## **Chapter 4 An accurate and robust approach of DFL with CAE**

To achieve better performance (localization accuracy and robust ability on noise), in this chapter, a three-layer CAE architecture and an effective algorithm to learn the parameters of CAE are designed to perform unsupervised feature extraction from raw signals followed by supervised fine-tuning for classification. The localization performance of the CAE is verified as superior to other compared state-of-the-art approaches on a real-world dataset.

## **1.5 Publication**

The following papers have been published in major conference and journals. The corresponding results of these works are presented in Chapters 2, 3 and 4.

### **Major Journal paper**

1. **Lingjun Zhao**, Huakun Huang, Xiang Li, Shuxue Ding, et al. An Accurate and Robust Approach of Device-Free Localization with Convolutional Autoencoder [J]. IEEE Internet of Things Journal ,2019, 6: 5825-5840.

2. **Lingjun Zhao**, Chunhua Su, Huakun Huang, et al. Intrusion Detection Based on Device-Free Localization in the Era of IoT [J]. Symmetry, 2019, 11(5): 630.

**Major Conference Paper**

3. **Lingjun Zhao**, Huakun Huang, Shuxue Ding, et al. An Accurate and Efficient Device-Free Localization Approach Based on Gaussian Bernoulli Restricted Boltzmann Machine. In 2018 IEEE International Conference on Systems Man, and Cybernetics (SMC), Oct 2018, pp. 2319-2324.

## Chapter 2

# An Accurate and Efficient Device-Free Localization Approach Based on Gaussian Bernoulli Restricted Boltzmann Machine

### 2.1 Introduction

With the repaid advance in wireless network technologies, the DFL, using radio frequency sensor networks to detect, track and locate targets who do not carry any attached devices, has attracted tremendous interest recently [38, 39].

In a DFL system, wireless sensor nodes are used to sense target by transmitting and receiving wireless signals with each other collaboratively. When a target stays in different locations, it will cause different influence on the wireless signals around it. Therefore, the DFL system can estimate the target's location by the wireless sensor signal variations produced by the target. However, in practice, the wireless signal variations caused by a target is tremendously weak and the complex noise in the environment make the signal quality worse. In order to address the problems, many pioneer works have been conducted including FP approaches [40], radio tomographic imaging approach [41], compressive sensing approach [42], and deep learning [43], etc.

Among these existing methods, deep learning is tremendous attractive because of its ability to process a large amount of data, extracting feature automatically and high accuracy in various applications. Wang et al. [44] proposed a deep learning approach for indoor localization and obtained higher accuracy compared with three existing methods in two representative indoor environments. Wang et al. [43] presented a sparse autoencoder network to learn discriminative features from the wireless signals and merge the learned features into a softmax-regression-based machine learning framework to realize



location. Zhang et al. [45] designed a deep autoencoder network to learn discriminative features automatically from the raw wireless signals. For conventional autoencoder, the training performance relies on the initialization of weights, which restricts its application range. To improve its performance, the RBM was proposed to initialize the parameters of neural networks layer by layer [46–48]. And its variant GBRBM used as the feature extractor to initialize the DNN and dimension reduction method has achieved significant performance in more broad fields [49, 50].

Based on its notable success, in this thesis, we explore the method of performing DFL with deep learning based on GBRBM approach. Specifically, we design a series of GBRBM blocks to extract features automatically from the wireless signals, then fine-tune the parameters via the AE, and finally merge the learned features into the encoder neural network to train the parameters and perform location prediction. The major contributions of our study can be summarized as follows:

- We formulate DFL as a classification problem, present a GBRBM-AE approach to learn features automatically from wireless signals.
- We propose a DFL algorithm that can obtain high location accuracy with high-dimensional DFL data and is robust to noise.
- This method outperforms the conventional AE on the real-world dataset in both location accuracy and robustness.

The remainder of this thesis is organized as follows. Section 2.2 introduces the DFL as a classification problem. The proposed algorithm is presented in Section 2.3. Section 2.4 demonstrates the results of performance evaluation. Finally, Section 2.5 concludes this work.

## 2.2 Problem Statement

### 2.2.1 Description of the DFL problem

As shown in Fig. 2.1, the DFL area is deployed by several wireless devices, which are normally named anchor points (APs). All APs in a DFL system can communicate with each other, called wireless links. Assuming that  $D$  APs are employed in DFL system, where  $D$  denotes the number of APs. In this thesis, let  $R_{m,n}^{\text{vacant}}$  and  $R_{m,n}^{\text{target}}$  denote the measurements of RSS that are transmitted from the  $n$ -th AP to the  $m$ -th AP without a target and with an existing target in the monitoring area, respectively. Then, when there exists a target, let  $\Delta R_{m,n}$  denote the variation of the measurement as

$$\Delta R_{m,n} = R_{m,n}^{\text{target}} - R_{m,n}^{\text{vacant}} \quad (2.1)$$

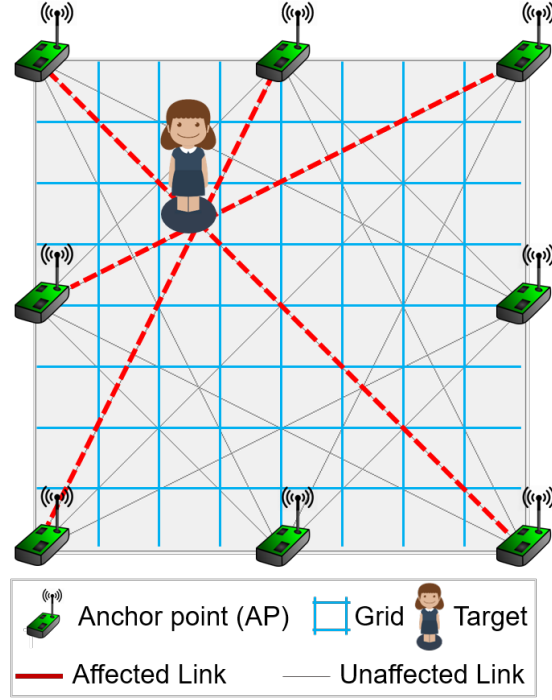


Figure 2.1: Illustration of a device-free localization (DFL) system model.

Then an RSS matrix  $\Delta \mathbf{R}$ , considering all  $D \times D$  pairs, can be established as follows:

$$\Delta \mathbf{R} = \begin{bmatrix} \Delta R_{1,1} & \Delta R_{1,2} & \cdots & \Delta R_{1,D} \\ \Delta R_{2,1} & \Delta R_{2,2} & \cdots & \Delta R_{2,D} \\ \vdots & \vdots & \ddots & \vdots \\ \Delta R_{D,1} & \Delta R_{D,2} & \cdots & \Delta R_{D,D} \end{bmatrix} \quad (2.2)$$

Since the target will absorb and scatter some of the transmitted signals as shown in Fig. 2.1, if the target location changes, RSS measurements of the affected links will be changed as well. Therefore, the measurement variation could be used to estimate the target location.

### 2.2.2 Formulation of the DFL as a classification problem

As described above, the target in different positions will derive different RSS matrix. Hence, different RSS matrix represents a different pattern of DFL which reflects corresponding target location. From this point of view, the DFL problem could be formulated as classification problem which could be well solved by deep learning [51,52].

Assume that the entire monitoring area is discretized into  $L$  grids as shown in Fig. 2.1, and each grid is acted as one class. Thus, all the potential locations are divided into  $L$  classes in this DFL problem. For each location  $l = 1, 2, \dots, L$ , we perform  $p = 1, \dots, P$  trails. By vectorizing the two-dimensional shape of  $D \times D$  into an

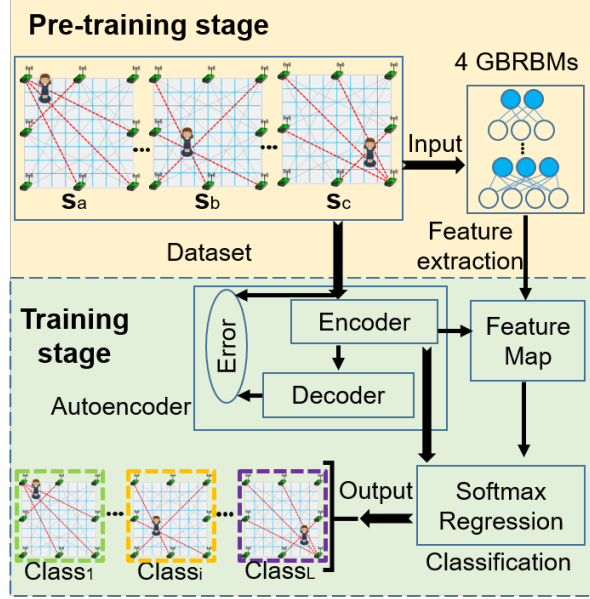


Figure 2.2: Illustration of system architecture and working flow for device-free localization (DFL) via GBRBM-based autoencoder (GBRBM-AE).

one-dimension shape, the atom vector  $v_{lp} \in \mathbf{R}^{D \times D}$ ). We can obtain the input data with location information for grids and trails as  $\mathbf{V} = [v_{11} \ v_{12} \ \cdots \ v_{1P} \ \cdots \ v_{lp} \ \cdots \ v_{LP}]$ . Here, we let  $S$  denote the total sample-number of  $\mathbf{V}$  (for  $S = L \times P$ ) and let  $\mathbf{V}_s$  ( $s = 1, 2, \dots, S$ ) denote the  $s$ -th sample of the entire dataset.

Fig. 2.2 illustrates the working principle and system architecture for solving the DFL problem via GBRBM-AE. Firstly, the DFL dataset is collected, and then inputted into deep learning architecture. In this model, the deep neural network uses GBRBM as feature extractor for pre-training and dimension reduction. And then, the pre-trained parameters are merged to the autoencoder for fine-tuning. Finally, the extracted features are used for classification via softmax regression. It is worth noting that the conventional RBM assumes that the visible unit is of binary data, which is a limitation to the various applications. GBRBM was proposed to apply real-valued input data by assuming the input data as Gaussian distribution and has achieved effective performance on many previous studies [53, 54]. Based on its notable success, GBRBM-AE algorithm can be a promising classification method for this target position localization problem.

## 2.3 Proposed Method

GBRBM is an undirected probabilistic graphical model also known as Markov random fields [55]. It has  $I$  visible real-value units  $v_i$ , ( $i = 1, \dots, I$ ) and  $J$  hidden binary units  $h_j$ , ( $j = 1, \dots, J$ ) with fully connection between them. However, there are no connections among nodes of the same layer. The visible layer represents the observed data and the hidden layer is used to model a distribution of observed variables.

---

**Algorithm 1** Unsupervised pre-training of GBRBM

---

**Input:**  $\mathbf{v}$ , epoch number, RBM number  $K$ .

**Output:** Updated parameters of GBRBMs:  $\mathbf{W}^k, \mathbf{b}^k, \mathbf{c}^k$

- 1: Initialization:  $\mathbf{v}_{data}^1 \leftarrow \mathbf{v}$ , initialize  $\mathbf{w}^k, \mathbf{b}^k, \mathbf{c}^k$  randomly
  - 2: **for** each epoch **do**
  - 3:   **for**  $k = 1$  to  $K$  **do**
  - 4:     Sample  $\mathbf{h}_{data}^k$  according to (2.6).
  - 5:     Sample  $\mathbf{v}_{model}^k$  with  $\mathbf{h}_{data}^k$  according to (2.7).
  - 6:     Sample  $\mathbf{h}_{model}^k$  according to (2.6).
  - 7:     Update  $\mathbf{W}^k, \mathbf{b}^k, \mathbf{c}^k$  via CD learning rule according to (2.9), (2.10) and (2.11).
  - 8:   **end for**
  - 9:   Until the convergence criterion is met.
  - 10: **end for**
- 

The energy function of a certain GBRBM is defined as

$$E(\mathbf{v}, \mathbf{h}) = - \sum_{i=1}^I \sum_{j=1}^J w_{ij} h_j \frac{v_i}{\sigma_i} - \sum_{i=1}^I \frac{(v_i - b_i)^2}{2\sigma_i^2} - \sum_{j=1}^J c_j h_j \quad (2.3)$$

where  $b_i$  and  $c_j$  denote the bias terms for visible and hidden units respectively,  $w_{ij}$  denotes the connection weights between visible and hidden units and  $\sigma_i$  is the standard deviation of visible units  $v_i$ .

Based on the property of MRFs, the joint probability distribution of this model can be expressed by the following formula,

$$P(\mathbf{v}, \mathbf{h}) = \frac{1}{Z} e^{-E(\mathbf{v}, \mathbf{h})} \quad (2.4)$$

where the  $Z = \sum_{\mathbf{v}} \sum_{\mathbf{h}} \exp(-E(\mathbf{v}, \mathbf{h}))$  is called the partition function. Then according to (2.4), the marginal distribution of  $\mathbf{v}$  is given by

$$P(\mathbf{v}) = \frac{1}{Z} \sum_{\mathbf{h}} e^{-E(\mathbf{v}, \mathbf{h})} \quad (2.5)$$

Because the visible and hidden units are both conditionally independent, the condition probability of  $\mathbf{v}$  and  $\mathbf{h}$  are expressed in the following forms, respectively.

$$P(h_j = 1 | \mathbf{v}) = f(c_j + \sum_i w_{ij} \frac{v_i}{\sigma_i^2}) \quad (2.6)$$

$$P(v_i = v | \mathbf{h}) = N(v | b_i + \sum_j h_j w_{ij}, \sigma_i^2) \quad (2.7)$$

where  $N(\cdot | \mu, \sigma^2)$  denotes the Gaussian probability density function with mean  $\mu$  and variance  $\sigma^2$  and  $f(\cdot)$  is the logistic sigmoid function.

**Algorithm 2** Unsupervised fine-tuning of autoencoder**Input:**  $W_e^k, b_e^k, c_e^k$ , epoch number, layer numbers K.**Output:**  $W_e^k, b_e^k, c_e^k$ 

- 1: Initialization:  $W_e^k \leftarrow (w^k), b_e^k = b^k, c_e^k = c^k$ , initialize  $b_d^k$  and  $c_d^k$  randomly.
- 2: **for** each epoch **do**
- 3:   **for**  $k = 1$  to K **do**
- 4:     Calculate decoded output  $d(v^k)$  with (2.13).
- 5:   **end for**
- 6:   Calculate MSE with cost function (2.14).
- 7:   Fine-tune  $W_e^k, b_e^k, c_e^k, W_d^k, b_d^k, c_d^k$  via backpropagation.
- 8:   Until the convergence criterion is met.
- 9: **end for**

**Algorithm 3** Supervised fine-tuning with data labels**Input:**  $W_e^k, b_e^k, c_e^k$ , epoch number, labels  $y$ **Output:**  $W_o^k, b_o^k, c_o^k$ 

- 1: Initialization:  $W_o^k \leftarrow W_e^k, b_o^k \leftarrow b_e^k, c_o^k \leftarrow c_e^k$
- 2: **for** each epoch **do**
- 3:   Calculate classification error and fine-tune  $W_o, b_o, c_o$  via backpropagation.
- 4:   Until the convergence criterion is met.
- 5: **end for**

The learning algorithms of GBRBM are based on maximizing the log-likelihood  $\ln \mathcal{L}(\theta|v)$  by gradient ascent. The log-likelihood gradient given a single training example  $v$  is as follows,

$$\begin{aligned}
\frac{\partial \ln \mathcal{L}(\theta|v)}{\partial \theta} &= \frac{\partial}{\partial \theta} (\ln \sum_{\mathbf{h}} e^{-E(\mathbf{v}, \mathbf{h})}) - \frac{\partial}{\partial \theta} (\sum_{\mathbf{v}, \mathbf{h}} e^{-E(\mathbf{v}, \mathbf{h})}) \\
&= - \sum_{\mathbf{h}} p(\mathbf{h}|v) \frac{\partial E(\mathbf{v}, \mathbf{h})}{\partial \theta} + \sum_{\mathbf{v}, \mathbf{h}} p(\mathbf{v}, \mathbf{h}) \frac{\partial E(\mathbf{v}, \mathbf{h})}{\partial \theta}
\end{aligned} \tag{2.8}$$

where  $\theta$  are parameters of GBRBM.

Note that the direct calculation of this gradient is computational cost. To avoid this problem, there exit a proposed contrastive divergence (CD) learning that approximate the log-likelihood gradient for GBRBM [56], which has been proved as an efficient learning algorithm to train GBRBM [57–59]. Instead of calculating the second term in the gradient, CD learning method only samples a few iterations (usually set to 1) from the model distribution by Gibbs sampling.

The updating rules for parameters in GBRBM are as follows:

$$w_{ij} \leftarrow w_{ij} + \eta (\langle \frac{1}{\sigma_i^2} h_j v_i \rangle_{data} - \langle \frac{1}{\sigma_i^2} h_j v_i \rangle_{model}) \tag{2.9}$$

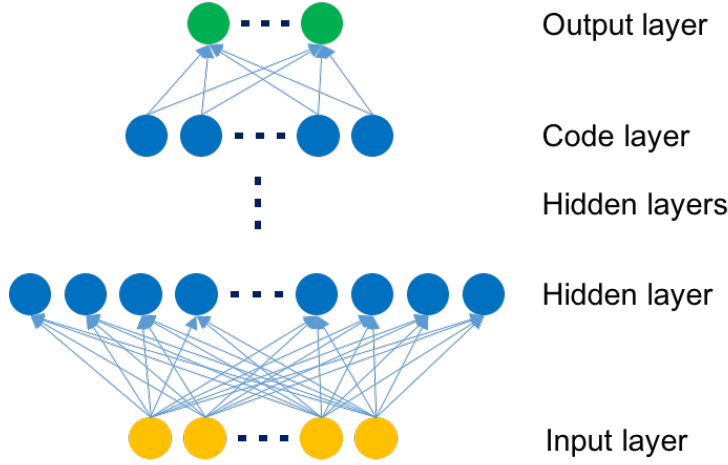


Figure 2.3: The architecture of encoder network.

$$b_i \leftarrow b_i + \eta(\langle \frac{1}{\sigma_i^2} v_i \rangle_{data} - \langle \frac{1}{\sigma_i^2} v_i \rangle_{model}) \quad (2.10)$$

$$c_j \leftarrow c_j + \eta(\langle \frac{1}{\sigma_j^2} h_j \rangle_{data} - \langle \frac{1}{\sigma_j^2} h_j \rangle_{model}) \quad (2.11)$$

where  $\eta$  is the learning rate,  $\langle \cdot \rangle_{data}$  and  $\langle \cdot \rangle_{model}$  denote the expectation over the data and model distribution respectively.

By calling the updating rules in (2.9), (2.10) and (2.11), a GBRBM can be trained effectively. The units in the hidden layer of lower GBRBM is used as input units of the upper layer. The above-mentioned layer by layer pre-training of GBRBM parameters is summarized in Algorithm 1.

We assume there are  $K$  hidden layers and each hidden layer  $\mathbf{h}^k$  has  $j$  units, for  $k = 1, \dots, K; j = 1, \dots, J$ . The pairs of GBRBM is  $(\mathbf{v}^k, \mathbf{h}^k)$ , and the output  $\mathbf{h}^k$  of the lower GBRBM is the input  $\mathbf{v}^{k+1}$  of the upper one.

After training a series of GBRBMs, they can be unrolled as an AE, which contains encoder and decoder phases. According to formulation (2.5), the encoder output of an AE can be written as

$$a(\mathbf{v}) = f(c_j + \sum_i w_{ij} \frac{v_i}{\sigma_i^2}) \quad (2.12)$$

After unrolling, the decoded output of the hidden layer can be reconstructed as

$$d(\mathbf{v}) = f(c'_j + \sum_i \sigma_i^2 w_{ij}^T a(\mathbf{v})) \quad (2.13)$$

Then it will be fine-tuned for optimal reconstruction via back propagation of mean square error (MSE) cost function,

$$E(\mathbf{v}) = \frac{1}{S} \sum_{s=1}^S (d(\mathbf{v}_s) - \mathbf{v}_s)^2 \quad (2.14)$$

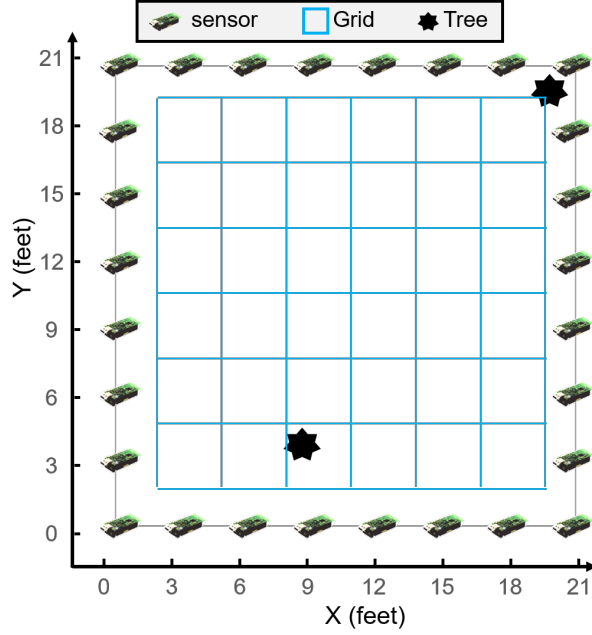


Figure 2.4: Illustration of real DFL experiment setup.

where  $S$  is the sample number of input dataset  $V$ .

It should be noticed that the training of autoencoder aims to reconstruct the input data does not use any information of the class labels. The aforementioned unsupervised training of autoencoder is summarized in Algorithm 2.

After extracting features from autoencoder, a softmax regression function [60] is added as output layer to the code layer to estimate optimal parameters. This neural network is called the encoder network, as shown in Fig. 2.3, which could make predictions of classification using labeled data. For the labeled training data  $(V, Y)$ , we employ cross-entropy error function as the cost function, defined as Eq. (2.15), which utilizes the prediction error between real labels  $y$  and predicted ones  $y'$  to fine-tune the parameters.

$$Er(W, c) = \frac{1}{S} \sum_{s=1}^S y_s \times \log(y'_s(W, c)) \quad (2.15)$$

where  $W$  and  $c$  denote the weights and biases of this encoder network. The procedure of the abovementioned supervised fine-tuning is summarized in Algorithm 3.

## 2.4 Performance Evaluation

In this section, we evaluate the performance of our proposal by using the real experimental dataset from the SPAN Lab of the University of Utah [41]. All the algorithms are implemented in tensorflow 1.2.0 on a system with GeForce GTX 1080 GPU and 32 GB memory.

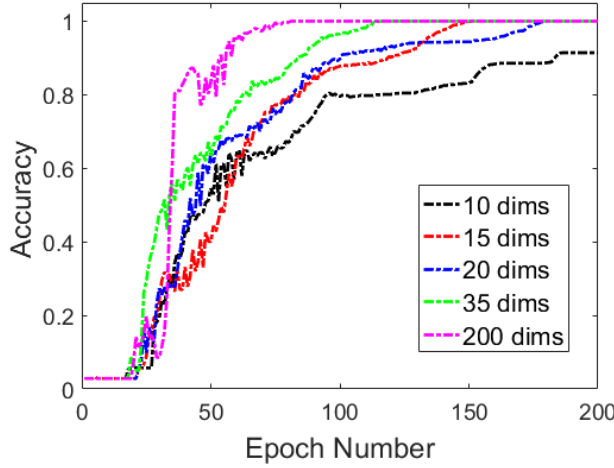


Figure 2.5: Classification accuracy of GBRBM-based autoencoder (GBRBM-AE) in training stage by using data with different dimensions of 10, 15, 20, 35 and 200-dims, respectively.

### 2.4.1 Experiment configurations

The layout of the wireless network setup is illustrated in Fig. 2.4. This network employs 28 wireless Crossbow TelosB sensor nodes. All the nodes use the IEEE 802.15.4 protocol and work in 2.4 GHz frequency band [41]. A base station node receives and then feeds the collected signals to a computer by USB. Each node is placed 3 feet apart along the  $21 \times 21$  feet perimeter square area and 3 feet away from the ground. As shown in Fig. 2.4, the monitoring area is discretized into 36 grids.

For data collection, the RSSs have been measured by 30 trials in a short time interval at different grids. Because there are no data in the location of coordinate (9, 6), we select the rest all 35 grids as test points in our performance evaluation. The total RSS sample matrices of each grid are split into two sets, where 25 trials are used as training data and the remaining 5 ones are for testing. We use the percentage of the number of correctly predicted locations to the total number of tested locations as accuracy to perform the evaluation of GBRBM.

For our proposed GBRBM-AE, the number of GBRBM blocks is 4. For the first three GBRBM blocks, the number of units in the hidden layer are 1000, 500 and 250, respectively. As to the last block, the unit number is set as 200, 150, 100, 50, 35, 30, 25, 20, 15 and 10, respectively, according to the data dimensional reduction experiments. In addition, the parameter  $\lambda$  of weight decay is set as  $\lambda = \eta \times 2 \times 10^{-4}$ .

### 2.4.2 Experiment results and discussion

In this section, we present numerical experiment results to evaluate the performance of the proposed GBRBM-AE, which is a feature extraction and dimensional reduction



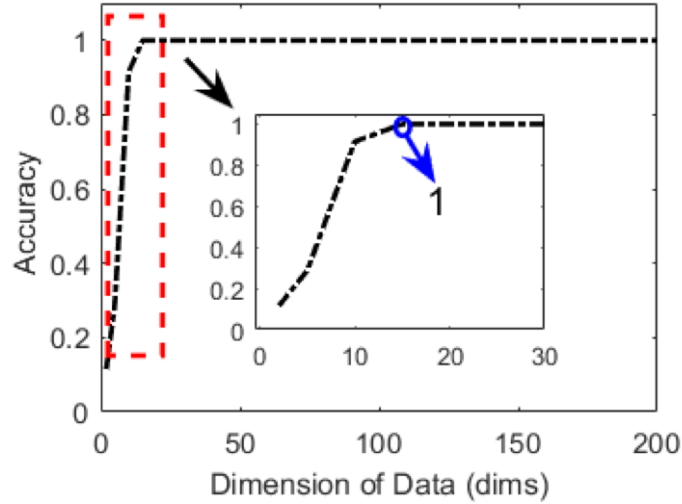


Figure 2.6: Localization performance of GBRBM-based autoencoder (GBRBM-AE) in testing stage with dimensional data from 2-dims to 200-dims.

method on dealing with DFL problems.

Fig. 2.5 shows the training classification accuracy by reducing the dimension of data from 784-dims to 200-dims, 35-dims, 20-dims, 15-dims, and 10-dims, respectively. From Fig. 2.5, it shows that our proposal can always achieve the classification accuracy of 100% within 200 epochs by employing data with dimension higher than 20-dims. In details, when performing on the data of 200-dims and 35-dims, the GBRBM-AE can achieve the highest 100% at within 100 epochs and remains stable with the increase of epoch number. Even when the data dimensions are reduced to 15-dims, the GBRBM-AE still can achieve the localization accuracy of 100% within 150 epochs. The result validates the good performance of the GBRBM-AE method in performing feature extraction together with data dimension reduction.

Whereas when the data dimensions reduced to 10-dims, the classification accuracy decreased sharply to 91.4%. This is because in this DFL problem, when the data dimension reduced lower than 15-dims, for some location data, the GBRBM-AE method cannot learn enough information to distinguish from others. Therefore, it can be argued that 15-dims data is the lowest and suitable dimension data to get the highest accuracy for the testing stage.

After doing training procedure on training DFL data, we use the GBRBM-AE to test on the testing data. Fig. 2.6 shows the testing localization accuracy of GBRBM-AE on dimension reduced data from 2-dims to 200-dims. It can be seen that the accuracy in 10-dims is 91.4%, and it increase to 100% in 15-dims. For the data dimension over 15-dims, the accuracy maintains the highest 100%. It indicates that with the data dimension decreasing, the localization accuracy goes down gradually, and 15-dims is the suitable dimension for this localization task, which is coincided with the prediction of

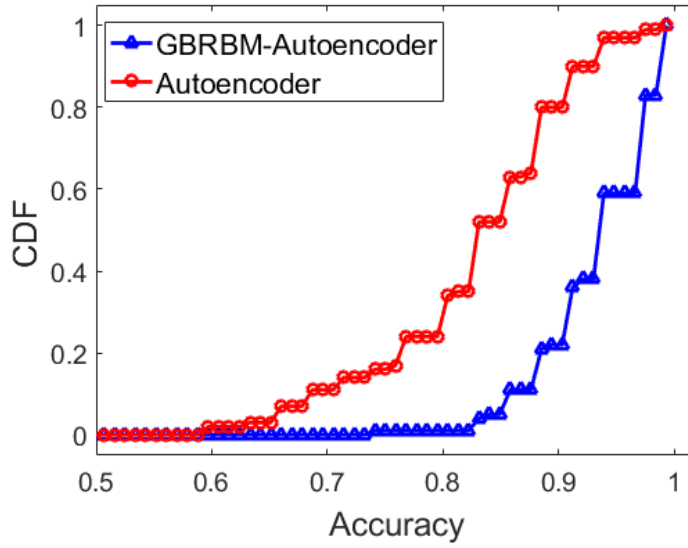


Figure 2.7: Cumulative distribution function of localization accuracy of autoencoder and GBRBM-based autoencoder (GBRBM-AE) with low dimensional data of 15-dims.

training stage.

To reveal the achievement of pre-training by GBRBM on the location accuracy, we explored the performance of AE without GBRBM. Fig. 2.7 shows the cumulative distribution function (CDF) of localization accuracy with the 15-dims data performed by GBRBM-AE and AE. The experiments are performed 100 times for each approach. It can be seen that the probability of the accuracies over 90%, obtained by GBRBM-AE, is 0.79, while probability of the same accuracy range obtained by AE is only 0.15. It is obvious that the range of accuracy acquired by AE is wider than GBRBM-AE, which indicate that the GBRBM-AE has better robust ability than AE.

Fig. 2.8 shows the testing performance of GBRBM-AE on noisy data with the different signal to noise ratio (SNR) and dimensions. It can be seen that the data with higher dimensions has better anti-noise ability. When the SNR equals 5 dB, the obtained accuracy is 89.7% by employing data with 20-dims and only the data with 150-dims can get the accuracy of 100%. When the SNR is increased to 10 dB, the data with dimension higher than 20-dims can achieve the highest 100%, which means it is robust to SNR = 10 dB.

Except the abovementioned AE, to demonstrate the effectiveness of our proposal, we compare it with two other methods. The first one is one dimensional convolutional neural network (CNN-1D) whose filter size is  $1 \times 9$  in this thesis. The other one is called the sparse representation classification method with a CVX tool (SRC-CVX) [61], where the CVX is a commonly-used convex-optimization toolbox. All these experiments are based on the noisy testing data with SNR equal to 10 dB and the dimension of data in both GBRBM-AE and AE is 20-dims. The comparison results are

summarized in the Table 2.1.

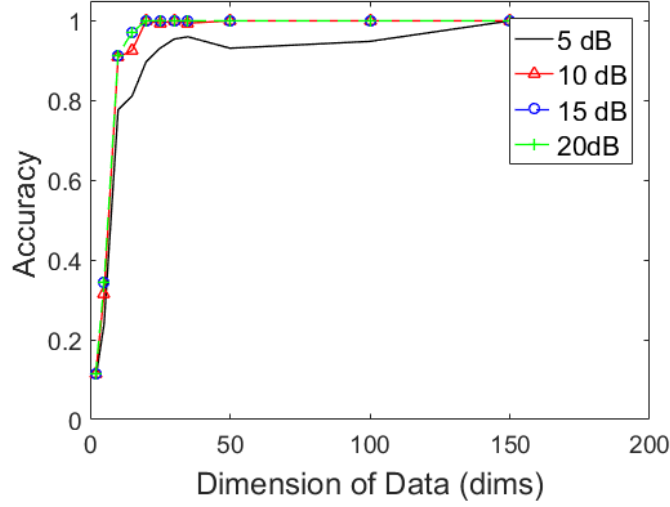


Figure 2.8: Localization performance of GBRBM-based autoencoder (GBRBM-AE) in the testing stage on noisy data with different SNR.

Table 2.1: Localization accuracy compared with other methods.

	GBRBM-AE	Autoencoder	CNN-1D	SRC-CVX [61]
Accuracy	<b>99.5%</b>	98.4%	91.3%	58.9%

Note: Experiments were performed on the noisy testing data with SNR = 10 dB.

## 2.5 Summary

In order to achieve the DFL problem accurately and efficiently, we formulate DFL as a classification problem, present a GBRBM-AE method, and conduct softmax regression function for target localization.

Experiment results show that in the testing stage with noiseless data, the our proposal can get location accuracy of 100% with data higher than 15-dims and has a sharp decline to 91.4% with 10-dims data, which evidences the GBRBM-AE method has good performance on dimension reduction data. When the noise appears in the testing stage, GBRBM-AE with 20-dims data is able to maintain high accuracy of 100% with SNR = 10 dB and is robust to SNR = 5 dB with 150-dims data. It indicates that the GBRBM-AE method can perform well on DFL problems with the real-world dataset.

Finally, we also perform experiment via autoencoder and two other comparison methods to confirm the improvement of performance. Experimental results indicate that this method of GBRBM-AE outperforms the compared methods on the real-world dataset in both location accuracy and anti-noise ability.

## Chapter 3

# Device-Free Localization with Convolutional Neural Network by Image Processing Method for Indoor and Outdoor Environments

### 3.1 Introduction

Wireless technologies have attracted tremendous attention in intrusion detection in the past few years [62–65]. The localization technique based on the wireless sensor network (WSN) has spawned a wide set of the Internet-of-Things (IoT) [66] applications, e.g., for intrusion detection and in security, for monitoring the statuses or positions of aging people in elderly health-care in a smart home, for locating the survivors in some emergency scenarios, for location-information-based services in some smart shopping centers, and so on [67].

To support these above-mentioned applications, many wireless localization technologies [68, 69] have been developed. For some wireless localization technologies, the target must be equipped with or carry an attached device, e.g., a smart phone or a tag; for example, the techniques of radio-frequency identification (RFID) [70] and GPS [71]. However, in some emergency scenarios, especially for intrusion monitoring or detection, it may be impossible to pre-equip any attached devices on the target.

Fortunately, device-free localization (DFL) [72, 73], as an emerging technology, is proposed to tackle this problem. DFL has been developed for target localization that need not equip the target with any extra tags or devices. As shown in Fig. 3.1, DFL systems are deployed to collect data on the target's locations based on WSNs, and then, the sensed data are sent to an edge server for processing. Through the edge server, some important locations or target-related information can be dug out for the user to

access. As with the description of Figure 3.1, the user can be the security personnel or a property administrator of buildings, a guard for the military, etc. Therefore, the DFL model is applicable to the various IoT applications [74] as described at the beginning of the Introduction, e.g., intrusion detection and monitoring for security.

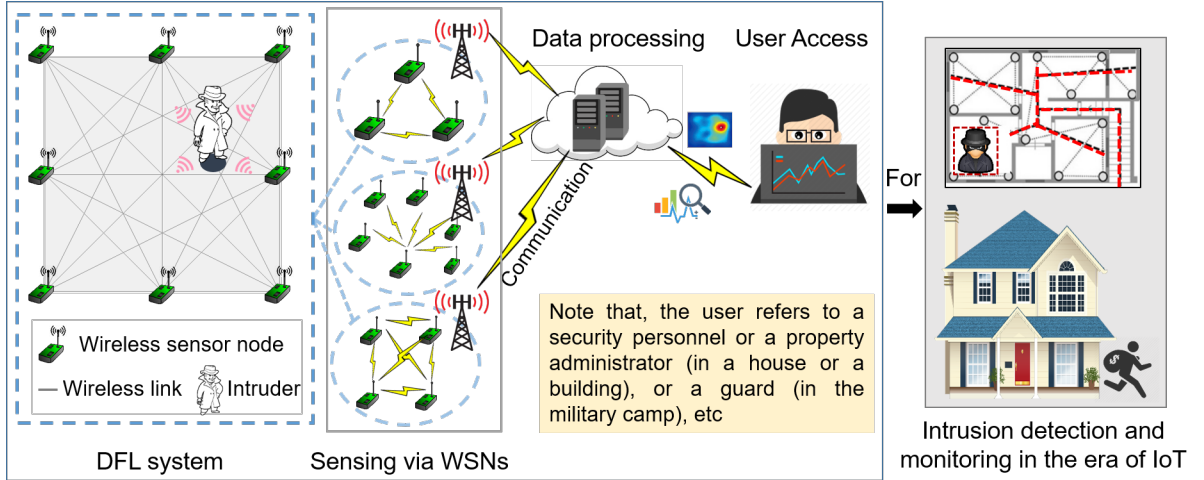


Figure 3.1: Internet-of-Things (IoT) fundamental blocks with device-free localization (DFL) technique for the indoor scenario of intrusion detection and monitoring in a smart city.

In the DFL system, a number of wireless sensors are deployed and communicate with each other. Every sensor node is employed to transmit wireless signals turn-by-turn, and the other nodes receive signals according to a time-schedule. Research [44, 75] has shown that the received-signal-strength (RSS) or the channel-state-information (CSI) can be used in the DFL problem, as it is affected differently by human movements and easily acquired. This means that if the targets enter the monitoring area of the DFL system or change their locations, they will derive specific wireless signals, i.e., RSS matrices (here, we take the RSS signal as an example). From this point of view, the locations of the targets can be estimated by analyzing the RSS matrices.

However, the corresponding relationship of the location-RSS is not accessible directly. To solve this problem, many previous research works regarded the DFL problem as a classification problem, arranged the collected wireless signals into vectors, and then employed the machine learning methods [76, 77] to extract features for classification. The commonly-used algorithms [78, 79] include support vector machines (SVM), K-nearest-neighbor (KNN), sparse coding, and deep neural networks. For these existing classification methods, deep learning is especially attractive because of its ability to process a large amount of data, extract complex features, and obtain outstanding performance in various fields. Nonetheless, since the variation of the signal derived by the target is exceedingly weak and easily affected by the environment, such as the surrounding noise, there still exist the problems of low accuracy and low robustness in

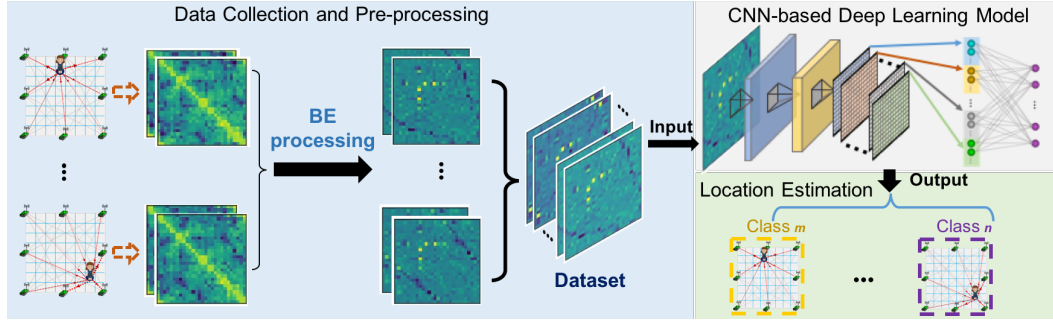


Figure 3.2: Illustration of the framework of the proposed background elimination pre-processing based convolutional neural network (BE-CNN).

the DFL approaches. Especially for indoor DFL, the environment is complex, such as blockage, scattering, etc., which makes the quality of the RSS signal worse. The low quality of RSS matrices makes it hard to recognize their features, which results in an adverse influence on the feature extraction.

In this chapter, aiming at improving the localization accuracy and robustness for DFL, we make full use of features underlying the collected RSS matrices and propose a background elimination (BE)-based convolutional neural network (BE-CNN) scheme. CNN has promoted the significant progress in many applications, e.g., object detection and recognition, intelligent systems for monitoring skin diseases, and so on [80–82], because of its excellent performance in feature extraction. Taking advantage of CNN, in this chapter, we first convert the RSS signal into an RSS-image matrix and then conduct a process of eliminating the background to dig out the variation components with distinguished features. Furthermore, the image matrices have specific patterns associated with different reference points (RPs) and have similar patterns at the same RP. Therefore, we make use of these feature-rich images by transforming the DFL problem into the image classification problem, where each RP is regarded as one class. Besides, we estimate the target's location as that of the RP with the most similar features of the collected matrix. Finally, a deep CNN is designed to extract features automatically to perform classification. It is expected that we will achieve an accurate and robust localization process.

To be more specific, we present the framework of the BE-CNN scheme for DFL in Fig. 3.2. From left to right, it first collects the wireless signals, i.e., RSS matrices in this chapter, and then pre-process all the raw RSS matrices by BE processing. In addition, all the processed RSS matrices with various classes of labels are input into the CNN for training. After the training procedure, the trained CNN can be employed to estimate the target's positions in the testing stage.

We summarize the three major contributions of this chapter as follows:

- We devise the RSS signal as the image matrix and then transform the DFL prob-

lem into the image classification problem.

- We perform two kinds of indoor testbeds, one outdoor experimental scenario and propose a scheme of BE-CNN for these localization scenarios, extracting features from the RSS-image automatically.
- The performance of the proposed BE-CNN scheme is validated on real-world datasets of both indoor and outdoor DFL by comparing with other baseline and state-of-the-art DFL methods.

The rest of this chapter is organized as follows. Section 3.2 presents the previous related works. In Section 3.3, we formulate the DFL problem as a classification problem and also devise the BE scheme. Section 3.4 presents the algorithm. Section 3.5 evaluates the performance of our proposal. Finally, we conclude the whole work in Section 3.6.

## 3.2 Related Works

We summarize the development of DFL and the related works about the methods of solving the DFL problem in this section.

Youssef et al. [83] firstly introduced the concept of device-free passive localization. In their work, the DFL was realized by using the RSS values. Moussa et al. [84] treated the DFL problem as a fingerprint-matching problem and then employed Wi-Fi equipment for intrusion detection and monitoring. Wilson et al. [85] firstly proposed the radio-tomographic-imaging (RTI) technology, which used RSS measurements to get images of the moving targets. Zhang et al. [86] eliminated the effects of noise in DFL by increasing the monitoring area and employed more sensor nodes. Inspired by these pioneering related works, many important research works [87,88] have been conducted, which indeed promoted the development of DFL. For example, Seifeldin et al. [89] designed the large-scale DFL system, which tracked entities in real environments. Wang et al. [44] and Gao et al. [90] proposed to realize the target localization by Wi-Fi equipment based on radio maps, which were constructed by CSI matrices. Liang et al. [91] focused on algorithms to eliminate the influences of outliers.

Based on the above works of DFL, in order to improve the performance, such as localization accuracy and detecting efficiency, many different approaches were proposed. Hong et al. [92] developed a novel localization system and employed the SVMs based on the combination of the spatial and temporal signals to evaluate the localization performance. Zhou et al. [93] compared the localization algorithm based on the principle components analysis with SVM classification and SVM regression. Tran et al. [94] represented fingerprints as a dissimilarity measurement between a pair of locations and

employed the KNN algorithm to realize the DFL. Zheng et al. [95] proposed an energy-efficient localization system and made use of adaptive weighted KNN to track targets with high accuracy. The above-mentioned methods belong to the traditional method, which have limits in exploiting the collected data and learning complex features. The current attractive deep neural networks, which perform feature extraction automatically, had achieved good performance in DFL. Wang et al. [43] designed DFL experiments and proposed a deep neural network (DNN) with the sparse autoencoder to locate the target's position. Zhao et al. [79] designed a four-layer neural network and employed restricted Boltzmann machines as a pre-training method to improve the accuracy and anti-noise capacity for outdoor localization. Zhou et al. [96] designed two neural networks for two DFL experiments and made use of them to analyze the CSI fingerprint patterns. Huang et al. [97] designed data argumentation based on the raw collected data to enlarge the dataset and employed a deep neural network to solve the localization problem.

In contrast with the above works, which extended the original RSS matrices into vectors, we first assumed each matrix as an image matrix as described at the end of Section 3.1. Then, based on analyzing the RSS matrices derived by the target in different locations, the BE-CNN scheme was proposed for improving the robustness and localization accuracy for both indoor and outdoor DFL.

### 3.3 Problem Statement

#### 3.3.1 Description of the DFL problem

Fig. 3.3(a) illustrates a model of the DFL system, which is similar with the DFL system described in subsection 2.2.1. From this figure, the monitoring area is surrounded by a series of wireless sensor nodes, which are normally called APs. All these APs can communicate with each other by transmitting or receiving wireless signals. The wireless links shown in Fig. 3.3(a) are the communication between every pair of APs.

As illustrated in subsection 2.2.1, when there is no target in the DFL system, i.e., vacant, let  $R_{m,n}^{\text{vacant}}$  denote the RSS measurement of the link from the  $n$ -th node to the  $m$ -th node. In details, each AP transmits signals in turns, and other APs receive signals according to a time-schedule. For example, as shown in Fig. 3.3(a), AP1 performs as a transmitter, while the rest of the APs, from AP2–AP8, receive signals. That is to say, if there is a total of  $D$  sensors deployed in the DFL system, e.g.,  $D = 8$  in Fig. 3.3(a), there will be a total number of  $D \times D$  RSS measurements collected. Here, let  $\mathbf{R}^{\text{vacant}}$  represent the RSS matrix consisting of the measurements collected from all the APs. Similar to  $\mathbf{R}^{\text{vacant}}$ , when the target enters the detection area,  $\mathbf{R}^{\text{target}}$  means an RSS matrix of all the wireless links.



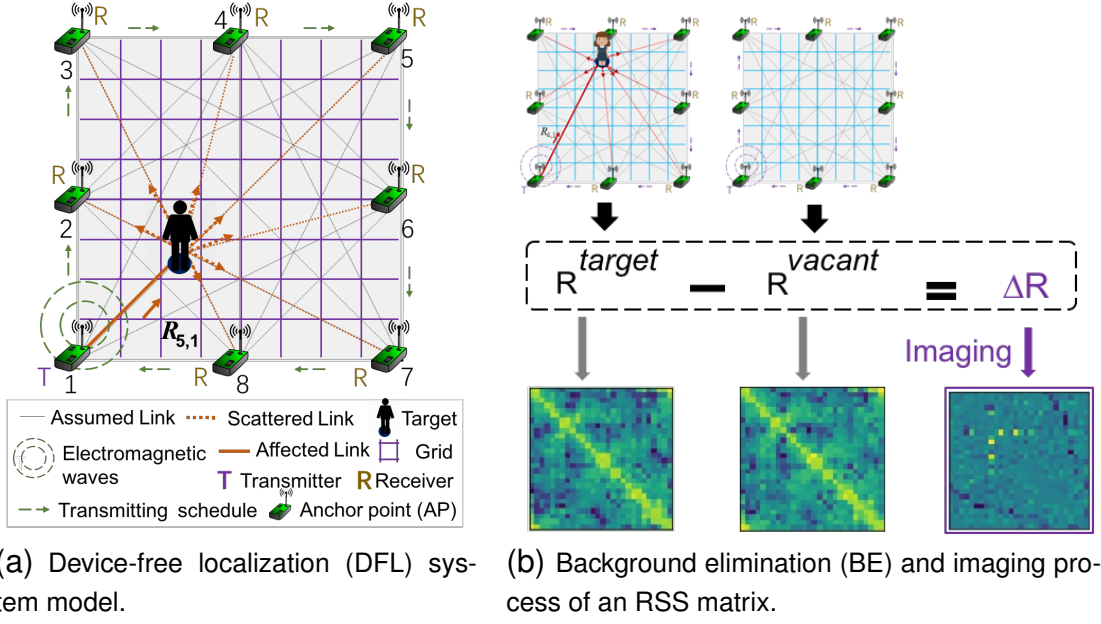


Figure 3.3: Illustration of a device-free localization (DFL) system model and description of the background elimination (BE) and imaging process of the RSS matrix when a target is at a certain reference position (RP).

There is no doubt that when the target exits the detection area, it will affect a series of the wireless links (or RSS measurements). For example, in Fig. 3.3(a), the target will scatter or absorb the signals from AP1, which results in changing the corresponding RSS measurements, such as  $R_{5,1}$ . Since the RSS signals are easily measured and they are changed differently in association with the target's positions, the RSS measurements derived by the object could be exploited to estimate the target's position.

Because the variation derived by the target is tiny when the target changes its position, to dig out the useful variation components, we performed a BE scheme to eliminate the effect of the background. Therefore, all the collected RSS matrices were pre-processed by subtracting the signal collected in the vacant detection area. Here,  $\Delta R_{m,n}$  denotes the measurement of the signal variation, as shown in Eq. (2.1), and then the variation RSS matrix  $\Delta \mathbf{R}$  can be established as Eq. (2.2) as described in subsection 2.2.1.

### 3.3.2 Formulation of the DFL as an image-classification problem

As illustrated in subsection 2.2.2, the target in different positions will derive RSS matrices with different patterns. Therefore, the DFL problem could be formulated into classification problem. In this chapter, each RSS matrix can be converted into an image matrix by regarding RSS measurements as pixels of the image. Moreover, each image matrix contains its specific patterns corresponding to the target's position. From this point of view, the DFL problem can be formulated as an image classification problem.

Fig. 3.3(b) shows the imaging process of the matrix  $\mathbf{R}^{\text{target}}$ ,  $\mathbf{R}^{\text{vacant}}$  and  $\Delta\mathbf{R}$ , which convert the RSS matrices into an image matrices. Note that different colors in the image matrices represent the RSS measurements in their associated RSS matrices. It is obvious that after BE process, the image of  $\Delta\mathbf{R}$  has more clear features than the original image of  $\mathbf{R}^{\text{target}}$ .

Since each image matrix reflects a specific pattern associated with the target's position, the image matrices derived from the same arrangement of the target contain similar features, i.e., patterns; whereas, if the target shifts positions from one grid to another, the corresponding image will be obtained with different patterns. Hence, these different matrices include the specific features that are associated with the corresponding locations of the target. Therefore, each RP of the monitoring area can be treated as one class of locations and the localization problem can be transformed into the image classification problem that could be effectively addressed with the CNN.

### 3.3.3 Dataset construction

The procedure for dataset construction is similar with the one described in subsection 2.2.2. However, instead of vectorizing the collected matrices into vectors, we directly collect the signal matrices as the dataset. In details, suppose that the total number of grids is  $L$ . Thus, all the potential locations of this detection area compose  $L$  classes in this localization problem. For each class, i.e.,  $\text{RP}_l$  (for  $l = 1, 2, \dots, L$ ), we conducted experiments with  $P$  trials. Regarding each trial, an RSS matrix,  $\mathbf{V}_{lp} \in \mathbf{R}^{D \times D}$  ( $p = 1, \dots, P$ ), can be organized. Therefore, we can get the dataset with location information for all grids and trials shown as  $\mathbf{V} = [\mathbf{V}_{11} \ \mathbf{V}_{12} \ \dots \ \mathbf{V}_{lp} \ \dots \ \mathbf{V}_{LP}]$ . Here, we let  $S$  denote the total sample-number of  $\mathbf{V}$  (for  $S = L \times P$ ) and let  $\mathbf{V}_s$  ( $s = 1, 2, \dots, S$ ) denote the  $s$ -th sample of the entire dataset.

## 3.4 Proposed Method

The CNN [98], one kind of deep neural network, has attracted tremendous attraction because of its notable ability to deal with data with two or three dimensions, e.g., the signals of videos or images. As shown in Fig. 3.4, illustrating a schematic diagram of a CNN architecture, the image is input for feature extraction, and then, the final layer outputs probabilities for each class.

In the convolutional layer, a series of two-dimensional kernels was employed to convolve with the receptive field of the feature maps from the former layer to extract data-specific feature maps. Note that the output from the previous layer was employed as the input for the next layer. For a given feature map  $\mathbf{U}$  from the former layer, the

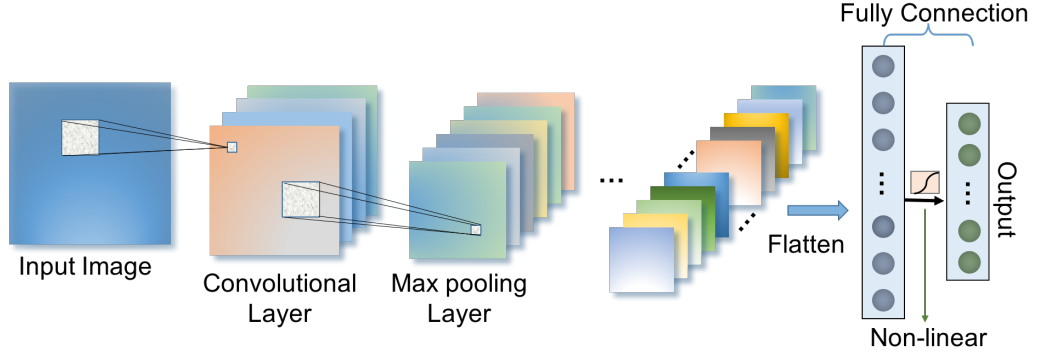


Figure 3.4: Schematic architecture of the convolutional neural network (CNN).

output after convolution is shown in:

$$\mathbf{y}_s = f(\mathbf{c} + \mathbf{W} * \mathbf{U}) \quad (3.1)$$

where  $*$  denotes the operation of convolution,  $\mathbf{W}$  and  $\mathbf{c}$  are the parameter terms connecting the convolutional layer, and  $f(\cdot)$  denotes the rectified linear unit (ReLU). The ReLU is the non-linear activation function, which is defined as  $f(z) = \max(0, z)$ , where  $z$  indicates the linear output of each layer.

Furthermore, a filter concatenation technology is exploited in this chapter. By employing this technique, features of different resolutions from the data can be captured [99]. Fig. 3.5 illustrates the details of filter concatenation in this chapter. The larger filter, i.e.,  $9 \times 9$  in this chapter, can capture the more general features, and the smaller filter, i.e.,  $3 \times 3$  in this chapter, finds details. Although filter concatenation increases computational cost, it can improve the performance of classification.

Generally, the subsampling is performed by pooling, i.e., max pooling in this chapter, which reduces the computational complexity of the neural network prone to over-fitting. However, this subsampling operation also loses some information of the data. Springenberg et al. [100] proposed that a convolutional layer with an increased stride can also achieve high accuracy with a shallower architecture, especially on some image

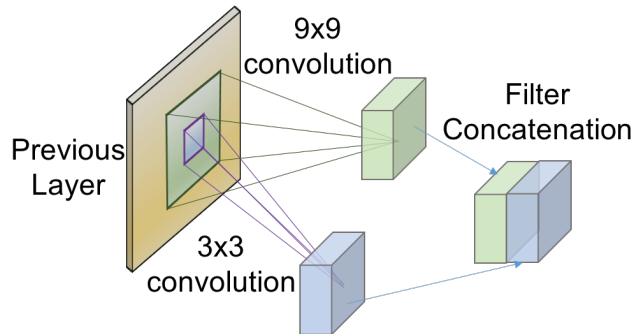


Figure 3.5: Schematic diagram of the convolutional filter concatenation.

recognition tasks. We compare these two subsampling operations in this work.

Additionally, the “dropout” operation in fully-connected layers means randomly making some units inactive in every epoch. This forces the neural network to extract different combinations of features from preceding layers. This operation can also avoid over-fitting at the same time. Finally, an output layer is followed to predict the probabilities of each class by employing the softmax regression function [60].

In the training stage, the labeled training data  $(\mathbf{V}, \mathbf{Y})$  were employed. To update the parameters in the neural network, the cross-entropy error function is employed to compute the error between the real labels  $\mathbf{y}_s$  and the predictions  $\mathbf{y}'_s$ , which is defined in Eq. (2.15). In the following experiments, the cost function is optimized by the Adam optimizer via backpropagation.

## 3.5 Performance Evaluation

In this section, we evaluate the performance of the BE-CNN by employing two kinds of real-world experimental datasets, including indoor and outdoor scenarios. All the experiments are implemented in TensorFlow 1.2.0 open source software on a system with a GeForce GTX 1080 GPU and 32 GB of memory.

### 3.5.1 Experiment configurations

In both of the indoor and outdoor DFL experiments, Crossbow TelosB nodes are employed for the wireless sensor network. All the nodes use the IEEE 802.15.4 protocol and work in 2.4 GHz frequency band. A base station node collects the signals and then transmits them to a computer by USB.

#### Description of Indoor Experiments

We perform a series of DFL experiments in an apartment including a living room and a corridor, which are with two promising application scenarios. Both of the two DFL systems employ four wireless sensor nodes together with a base station node. The photo of the deployed experimental scenario in the corridor and layout of the DFL system are shown in Fig. 3.6(a) and Fig. 3.6(b). In this DFL system, each of the four sensor nodes is placed at one corner of the corridor and 1.0 m away from the ground. In addition, the monitoring area is 3.0 m×0.7 m square area, and we take 5 positions distributed in the area as the RPs, as shown in Fig. 3.6(b). In this experiment, first, 250 trials of the RSSs were measured in a short time interval with one target at each RP. Then, 200 trials were measured in another short time interval. Therefore, as described in subsection 3.3.3, the corresponding number of RSS sample matrices is 2250. Among them, 1250 samples of these RSS matrices are employed as training data, 250 samples

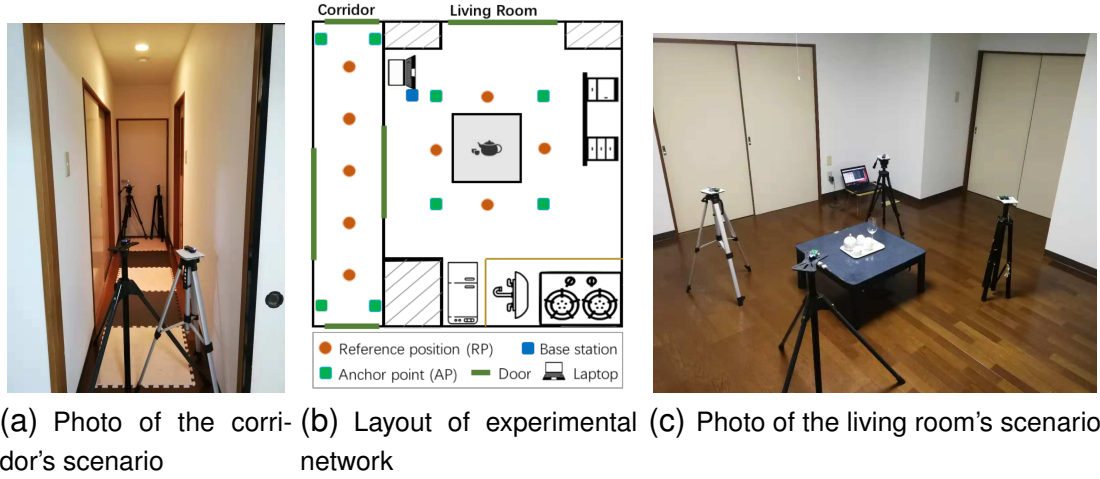


Figure 3.6: Experimental layout of the indoor DFL scenarios in the living room and the corridor. There are totally 9 reference positions (RPs) in the experimental scenarios.

are employed as validation data and 750 samples are employed as testing data. Note that in different trials, the target may be at different places, although in the same RP.

The layout of the wireless sensor network performed in the living room and the photo of the deployed experimental scenario are shown in Fig. 3.6(b) and 3.6(c). In this system, each node is placed at the corner of the  $2.5 \text{ m} \times 2.5 \text{ m}$  square area and 1.0 m away from the ground. In addition, there is a table in the center of the monitoring area to simulate the real scenario. Because of the limited number of wireless sensor nodes, we take 4 positions at each side of the table as the RPs in this DFL system, as shown in the Fig. 3.6(b). Similar with the experiments performed in the corridor, the RSSs were also measured in two short time interval with one target at each RP, including 125 trials of the RSSs were measured in the first time interval and then 100 trials were measured. Among the collected samples, 500 samples are for training, 100 samples are for validation and 300 samples are for testing.

Combining the two dataset collected in the two scenarios will create the indoor DFL dataset, which includes 1750 samples for training, 350 samples for validation and 1050 for testing.

### Description of Outdoor Experiments

For the outdoor experiment, we adopted the same dataset, the Outdoor RTI dataset [41], as described in Chapter 2. In details, RSSs were measured for 30 trials in a short time interval with a person in the each RP. The total RSS samples were split into two sets for each RP, where 25 samples were used for training, and the remaining five ones were used as the testing data. Different with the dataset described in Chapter 2, we selected all the 36 RPs as testing locations for performance evaluation, as shown in Figure 3.7. Therefore, in all the following outdoor experiments, there were 900 samples for training

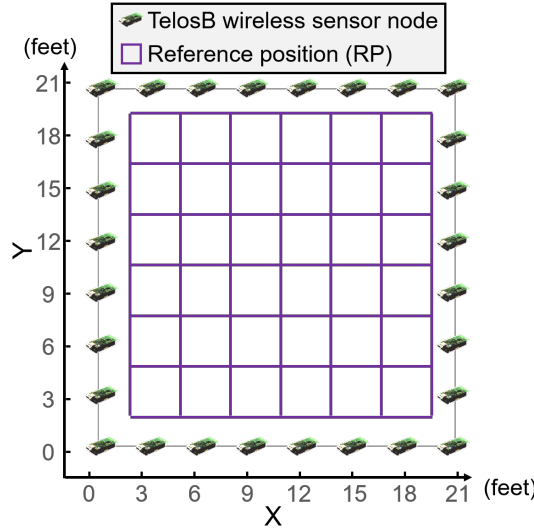


Figure 3.7: Experimental layout of the outdoor DFL scenario.

The grids in the top row correspond to reference positions (RPs) from RP31 to RP36; the grids in the second row correspond to RPs from RP25 to RP30; the grids in the third row correspond to RPs from RP19 to RP24; the grids in the fourth row correspond to RPs from RP13 to RP18; the grids in the fifth row correspond to RPs from RP7 to RP12; the grids in the bottom row correspond to RPs from RP1 to RP6.

and 180 samples for testing.

In this chapter, we employed localization accuracy, defined in Eq. (3.2), as the metric to perform the evaluation of BE-CNN. Suppose that  $C_{\text{total}}$  is the total number of testing samples and  $C_{\text{correct}}$  is the number of testing samples that were correctly estimated. Then, the localization accuracy is defined by:

$$\text{Accuracy} = \frac{C_{\text{correct}}}{C_{\text{total}}} \times 100\% \quad (3.2)$$

### 3.5.2 Data pre-processing

To show the features in the RSS images of the datasets, we take the signal data collected in two RPs, i.e., RP2 and RP4, as examples. Fig. 3.8 shows the image formation of the collected signal data when a target is at RP2 or RP4 of the monitoring area in living room, corridor and outdoor space. In this figure, the first row presents the image formations of raw signals, which were collected directly by sensors, and the second row shows the variation signals obtained after BE pre-processing. It is clear that the patterns of raw data of different positions are very similar, which can hardly be classified artificially. After the BE pre-processing, not only the patterns of images appeared clear, but also the differences of patterns between RP2 and RP4 became more obvious, which benefited the feature extraction.

In practical DFL applications, the signal data are generally collected involving a

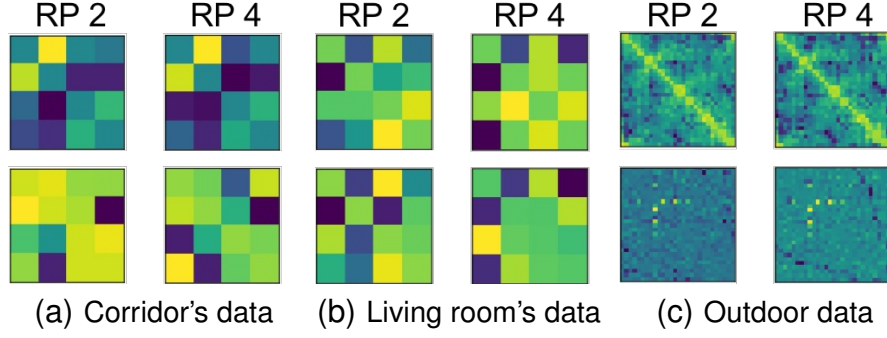


Figure 3.8: Comparisons of raw RSS matrices (Top row) and background eliminated RSS matrices (Bottom row). Here, the data of reference position 2 (RP2) and reference position 4 (RP4) are taken as the examples.

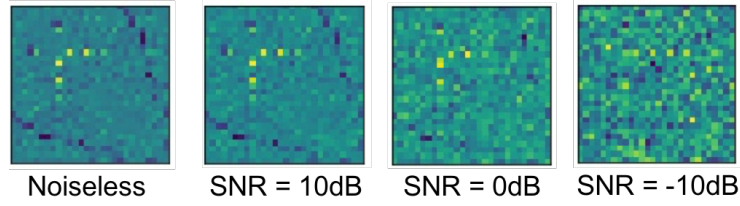


Figure 3.9: Featured images of noiseless signal and noisy signal with different SNR. Here, the outdoor data are taken as the examples, corresponding to the reference position 4 (RP4).

certain level of noise. There may be some adverse dynamic information, such as the electro-magnetic interference from the surrounding electronic devices or equipment, which may lead to various noise with different degrees in the RSS signals. Due to the fact that the RSS is easily affected by noise, the robustness ability with respect to noise is important to DFL algorithms.

To evaluate the robustness against noise of the proposed scheme, we performed experiments by adding noise of different levels. Fig. 3.9 shows the image formations of pre-processed raw noiseless signal data and the associated noisy signal with different degrees of the signal-to-noise ratio (SNR). The SNR is defined as:  $\text{SNR}(\text{dB}) = 10 \log_{10}(P_{\text{signal}}/P_{\text{noise}})$ , where  $P_{\text{noise}}$  and  $P_{\text{signal}}$  represent the power of the noise and the power of the signal, respectively. Note that the noiseless data denotes the raw data without adding Gaussian noise in this chapter. In Fig. 3.9, except for the noiseless signal, the other three RSS images are noisy signals with SNR of 10 dB, 0 dB, and  $-10$  dB. It shows a clear tendency that the noisy condition of the images becomes more and more serious as the SNR decreased, which makes their features more difficult to be extracted. Note that different levels of noise were added into the entire dataset including the training samples and testing samples. All these noise cases were exploited to evaluate the performance of our proposal with respect to the localization accuracy and robustness.

Table 3.1: Optimal parameters of the BE-CNN for indoor DFL system.

Key parameters	Optimization
Convolutional layer number	4
Convolutional filter size	$3 \times 3$
Filter number of each layer	16-16-32-32
Max pooling layer	Without pooling
Maximum epoch number	600
Learning rate	$10^{-4}$
Batch size	200
Dropout rate	0.8

### 3.5.3 Performance of the BE-CNN for indoor DFL

In this subsection, we exploit the localization performance of the BE-CNN approach in the indoor environment, including the living room and the corridor. According to our preliminary experimental results, there may exist the “over-fitting” problem. It is considered that the main reasons are the small number of the employed wireless sensors in these DFL systems and the limited number of collected data samples. Therefore, we adopt “early stopping” to avoid “over-fitting”. In order to apply the “early stopping”, we employ the validation data to evaluate the localization accuracy every epoch. The stop-training strategy is shown as below: we compare the average accuracy and the loss of the validation data every 10 epochs. If the calculated accuracy is declining as well as the loss is increasing, the training procedure will be stopped.

After performing several experiments by trial and error for the localization performance, the hyperparameters of the BE-CNN for indoor DFL are as summarized in Table 3.1. There are four convolutional layers with the corresponding filter number of 16, 16, 32 and 32 for each layer. Instead of taking use of max pooling operation, we employ the  $3 \times 3$  convolutional filter with stride 2 to perform down-sampling. Although the maximum epoch number is set as 600, the training procedure will stop automatically according to the stop-training strategy. The batch size is 200 and the neurons of the fully connected layer will be randomly dropped at the rate of 0.8 for the training of each batch.

Table 3.2: Impact of the number of training samples on localization accuracy.

Number of samples for training	Number of samples for validation		
	350	700	1050
1750	<b>91.0%</b> (90.2%)	86.8%	85.3%
1050	86.2%	83.5%	82.8%
350	79.6%	78.5%	78.7%

Note: The result in the parentheses denotes the average result by performing K-Fold cross validation for 50 times, where K equals 5.



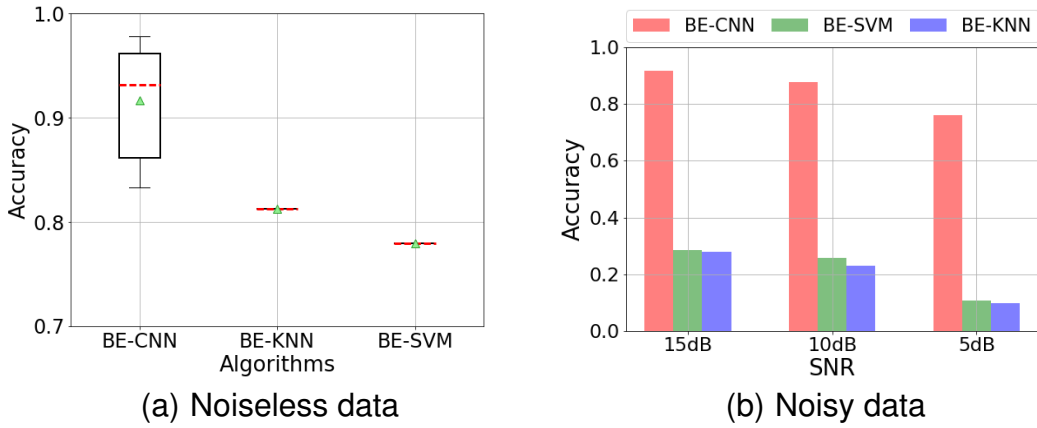


Figure 3.10: Indoor localization performance of BE-CNN, BE-KNN and BE-SVM on the noiseless and noisy data.

To explore the effect of changing the number of training samples, we perform experiments of 9 conditions. The experiments for each condition are performed 50 times. Table 3.2 shows the average localization accuracy by employing different number of samples for training and validation. The experiments for each condition are performed for 50 times by employing BE processed noiseless data. It is clear that either decreasing the number of training samples or validation samples will drop the localization accuracy. When we employ 1750 samples for training and 350 for validation, the highest average accuracy of 91.0% can be obtained. The accuracy of 90.2% in the parentheses in this table is the corresponding average result by adopting K-Fold cross validation strategy.

After deciding the number of samples for training, validation and testing, we compare the BE-CNN with multi-class SVM and the KNN to demonstrate its value. The average localization results of 50 times experiments for each condition are shown in Fig. 3.10. Specifically, Fig. 3.10(a) shows the box-plot of localization accuracy on noiseless testing data. Because of the “early stopping” training strategy, the localization accuracies obtained by BE-CNN are distributed from the minimum value 85.4% to the maximum value 97.8% among the 50 times experiments. Although the distributed range seems wide, the probability of the accuracy which is higher than 86.24% is 0.75. In addition, the median is 92.1% which means 50% of the obtained localization accuracies are higher than 92.1% and among them, 25% of the accuracies are higher than 96.1% which are exact enough for most applications. Note that the average localization accuracy obtained by BE-CNN is 91.0%, which is 9.1% and 13.0% higher than using BE-KNN and BE-SVM, respectively. Even the minimum value 85.4% obtained by BE-CNN is higher than 81.3% of BE-KNN and 78.0% of BE-SVM.

Fig. 3.10(b) reveals the average localization accuracy of the three approaches on noisy data of different SNR, including 15 dB, 10 dB and 5 dB. Note that by employing

BE-KNN or BE-SVM, the obtained localization accuracies are all lower than 30% when SNR is less than 15 dB, which indicates that the target is failed to be located. With regard to the BE-CNN, there is a tendency that the localization accuracies are dropping with the reduction of SNR. When the SNR equals 15 dB, the average accuracy achieved by BE-CNN is 89.9%. When the SNR declines to 10 dB, the average accuracy drops to 87.7%. However, even the SNR of data is reduced to 5dB, the BE-CNN can still obtain the average localization accuracy of 76.1%, which is much higher than using BE-KNN and BE-SVM.

Based on the experimental results of the indoor DFL, the highest localization accuracy on noiseless data achieved by the BE-CNN can be 97.8% and the corresponding average accuracy is 91.0%, which are significantly superior to BE-KNN and BE-SVM. Furthermore, the BE-CNN is robust to noisy data with SNR equals to 15 dB.

### 3.5.4 Performance of the BE-CNN for outdoor DFL

After performing several experiments by trial and error for the localization performance, the hyperparameters of the BE-CNN for outdoor DFL are summarized in Table 3.3. As shown in this table, we employ two convolutional layers with 32 filters for each layer. In addition, the size of the filters is  $9 \times 9$  for the first layer and  $3 \times 3$  for the second layer. The feature maps learned from the two layers are concatenated before flattening. To avoid the “overfitting” problem, we set a dropout rate of 0.4 for the training of each batch.

Based on the designed BE-CNN, we exploited the localization performance of the proposed scheme together with other compared approaches. In order to validate the merit of BE pre-processing, we performed experiments on both raw data and the BE processed data by employing CNN, KNN, and SVM, as shown in Fig. 3.11.

Note that Fig. 3.11(a) shows the impact on localization accuracy by employing BE processing. Experiments were performed on the noisy dataset with SNR equal to 15 dB. In addition, the accuracy for each condition shown in this figure was the average

Table 3.3: Optimal parameters of the BE-CNN for outdoor DFL system.

Key parameters	Optimization
Convolutional layer number	2
Concatenated convolutional filter size	$9 \times 9, 3 \times 3$
Filter number for each layer	32
Subsampling operation	Without pooling
Epoch number	100
Learning rate	$10^{-4}$
Batch size	300
Dropout rate	0.4

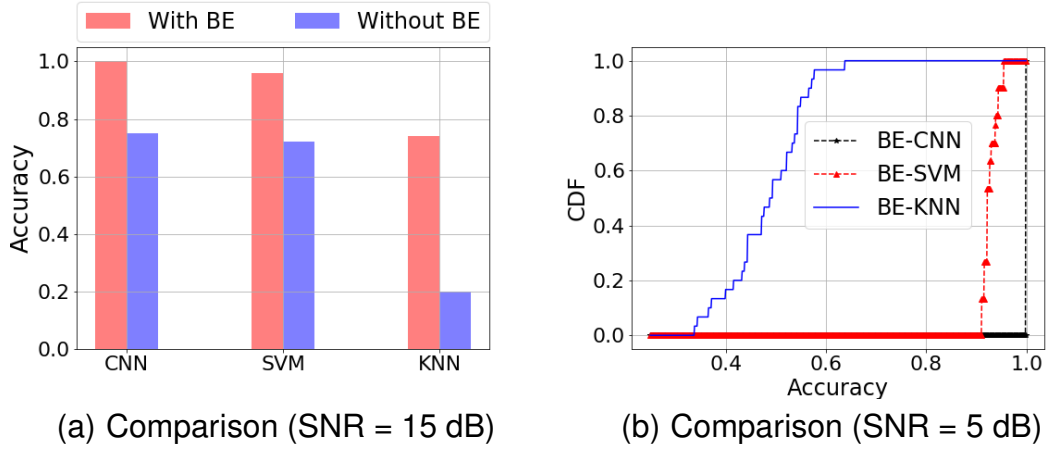


Figure 3.11: Localization performance comparisons by using CNN, KNN, and SVM. Here, all data are with noise. BE is short for background elimination.

result of 30 times experiments. When employing BE-CNN, the localization accuracy was 100%, while the accuracy obtained without BE was 74.6%. The accuracy obtained by BE-SVM and BE-KNN were 17.7% and 54.2% higher than the results obtained by corresponding ones without BE. It can be concluded that no matter which method is employed, the BE-based ones can get higher accuracy. Fig. 3.11(b) shows the cumulative distribution function (CDF) result of the localization accuracy by employing BE-CNN, BE-KNN, and BE-SVM. The input data was noisy signals with the SNR equal to 5 dB. Experiments were performed 30 times for each condition. Among the 30 experiments, the BE-CNN could always accurately locate the target with the accuracy of 100%, which was obviously higher than 88.9% obtained by the BE-SVM. The accuracies obtained by BE-CNN were all lower than 60%, which indicated that the BE-KNN could hardly locate the target when the level of noise was less than 5 dB. This figure demonstrates that the BE-CNN outperforms the other two methods in the outdoor DFL, even when the noise is severe.

Except for the abovementioned two baseline methods, to demonstrate the prior-

Table 3.4: Localization accuracy compared with other methods.

	BE-CNN	BE-SVM	BE-KNN	BE-AE	SC-ISTA	SRC-CVX
Accuracy	<b>100%</b>	88.9%	50.6%	97.2%	2.8%	2.8%

Note: Experiments were performed on the noisy dataset with the SNR = 5 dB.

ity of the BE-CNN, we compared its performance with a kind of deep neural network, called autoencoder. In this chapter, we utilized the suggested architecture from [101], which achieved good classification performance. In the corresponding experiments, there were three hidden layers that had respectively 200, 100 and 50 neurons for the encoder part. Furthermore, two more methods using the same open dataset were also evaluated. One is SRC-CVX and the other one is the sparse coding method based on

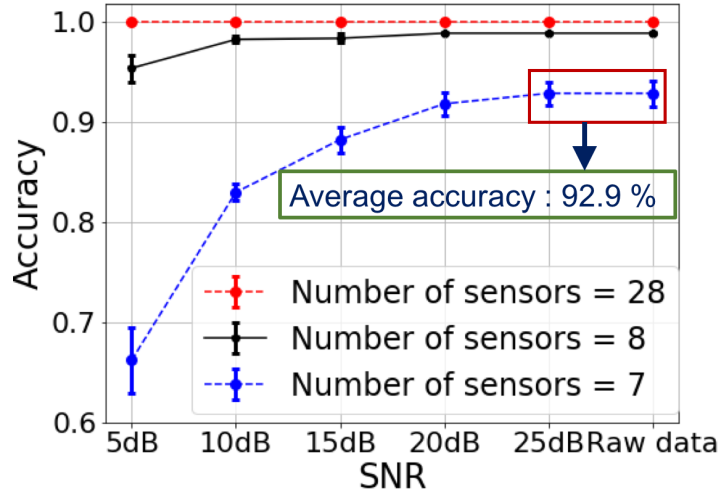


Figure 3.12: Comparisons of localization accuracy by employing different numbers of sensors. Here, the distance between the adjacent sensors is 3 feet, 9 feet, and 12 feet for sensor numbers of 28, 8, and 7, respectively.

the iterative shrinkage-thresholding algorithm (SC-ISTA) [76]. The dataset used in the two works was the raw RSS signals without conducting the BE process.

The comparison results are shown in Table 3.4, which were based on the noisy dataset with SNR equal to 5 dB. The localization accuracies in this table are all average results based on 30 times experiments for each condition. It is obvious that the BE-CNN can achieve stably the highest localization accuracy of 100%, which outperformed all the other compared schemes under the noisy condition.

Additionally, to explore the impact of different scenarios of the DFL system, we performed experiments by reducing the number of sensors employed in the system. In other words, the distance between two adjacent sensors became farther. As shown in Fig. 3.12, we compared the localization accuracy by employing 7, 8 and all 28 sensor nodes. The distance between every two adjacent sensors by employing 7 and 8 sensors was about 12 feet and 9 feet, respectively. Note that both raw data without noise and noisy data with different levels of SNR from 5–25 dB, were employed in these experiments. Note that the experiments for each condition were performed for 30 times. In this figure, there was a tendency that with the increasing of the SNR, the accuracy went up gradually and then became stable. When 7 sensor nodes were employed in the DFL system, the localization accuracy obtained by BE-CNN was 92.9%. If we increased the number of sensor to 8, the localization accuracy went up sharply to 98.9% when the SNR equaled 10 dB. It can be concluded that the least number of sensors in the DFL system for accurate localization was 8, and each sensor was about 9 feet away from each other.

To summarize, the proposed BE-CNN scheme can maintain the highest localization

accuracy of 100% when the SNR of data is greater than 5 dB, which means the proposed BE-CNN has great anti-noise ability. Combining the experimental results from subsection 3.5.3 and subsection 3.5.4, the BE-CNN has the most optimal performance compared with BE-KNN and BE-SVM on DFL, especially for noisy datasets.

## 3.6 Summary

Aiming at solving the problems of low accuracy and low robustness in DFL approaches, we first treated the RSS signal as an RSS-image matrix and conducted a process of eliminating the background to dig out the variation components with distinguished features. Then, we made use of these feature-rich images by formulating DFL as an image classification problem. Furthermore, a deep CNN was designed to extract features automatically for classification.

The localization performance of the BE-CNN is validated in our real testbeds of indoor DFL system and a real-world dataset of outdoor DFL. In addition, we also validate the robust performance of the proposal by conducting numerical experiments with different levels of noise. Experimental results show that:

**(a) Localization performance in the indoor experiments.**

Results on DFL system of the indoor environment include the living room and the corridor. Among 50 trials experiments on the noiseless dataset, the highest localization accuracy obtained by the BE-CNN is 97.8%. As an average result of performing on noisy dataset with SNR equals to 15 dB, the BE-CNN can still achieve localization accuracy of 91.7%, while the BE-KNN and BE-SVM are failed to locate the target.

**(b) Localization performance in the outdoor experiments.**

When conducting experiments on the noiseless dataset, the BE-CNN could maintain the highest localization accuracy of 100%. For the noisy dataset with SNR equal 15 dB, the localization accuracies of the BE-based methods were all higher than the corresponding raw data-based methods. It demonstrates the value of BE pre-processing. In addition, the BE-CNN could maintain a high accuracy of 100% on the noisy dataset with an SNR equal 5 dB, which is better than all the compared methods.

In summary, the experimental results clearly demonstrated that the BE-CNN could achieve high accuracy localization results in both indoor and outdoor DFL. In addition, the anti-noise ability of the proposed approach were better than the comparison methods especially under the conditions with heavy noise. All these results demonstrated the good performance of the BE-CNN in solving the DFL problem.

## Chapter 4

# An Accurate and Robust Approach of Device-Free Localization with Convolutional Autoencoder

### 4.1 Introduction

The wireless localization market has spawned extensive Internet of Things (IoT) applications in Smart Cities, such as healthcare at home or in the hospitals (e.g., for detecting locations and activities of elderly people and patients), location-based services in smart spaces (e.g., airports, shopping centers, touristic sites, etc.), emergency rescue (e.g., location detection of firefighters and survivors in the fire), and for detecting and tracking an intruder's location in security safeguards. To support the abovementioned scenarios, various wireless localization techniques [68, 69, 102] have been developed. For some localization techniques, for example, GPS [71] and ultrasound [103], the targets must carry or be equipped with wireless electronic devices, such as a smartphone, to perform localization. However, in some conditions, such as emergency rescue, it is impossible to preinstall devices on targets. To address this problem, device-free sensing technology [104, 105] is proposed based on WSNs. Particularly, the DFL [72, 106], as an emerging wireless localization technology, is exploited to locate targets who do not carry any attached devices. As shown in Fig. 4.1, WSNs are employed to sense signals on the target's location in the DFL system and then send the collected data to the server to perform the data processing. After the data processing, the location information can be obtained for user access. As described in Fig. 4.1, the user refers to a doctor or a guardian for the healthcare of the elderly, security personnel or a property administrator for security safeguards, a manager for the location services in smart spaces, etc. Thus, the DFL model can be applied to the IoT-based applications [74] mentioned at the beginning of this section, such as elderly healthcare, security safeguards and so on.

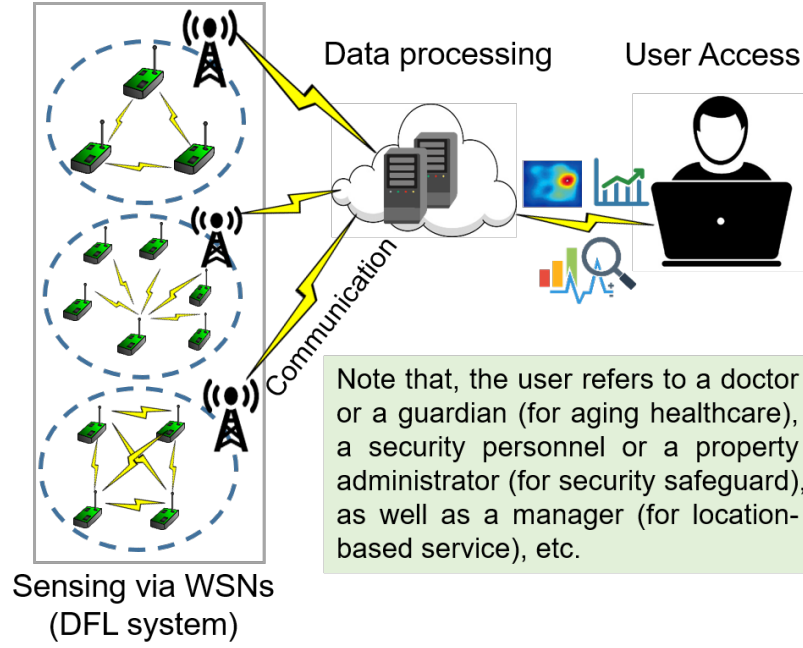


Figure 4.1: Fundamental blocks of IoT with respect to the device-free localization (DFL) model. Here, WSN is short for wireless sensor network.

Studies have shown that the so-called RSSI can be used in the DFL problem; moreover, RSSI is easily acquired and it is affected differently by different human positions. In a DFL system, wireless sensor nodes are used to sense targets by collaboratively transmitting and receiving wireless signals with each other. Every communication between two sensors is called a ‘wireless link’. If the target changes its location from one place to another, it will surely affect some of the wireless links, which means the target in different locations will derive correspondingly specific wireless signals. Therefore, the DFL system can estimate the target’s location by the wireless signal variations produced by the targets. However, in practice, the location-RSSI correspondence relationship is not directly accessible. Furthermore, the wireless signal variations caused by a target are extremely weak, and the complex noise in the environment degrades the signal quality. To address these problems, many pioneering works have been conducted including fingerprinting approaches [40], a radio tomographic imaging approach [85], a compressive sensing approach [42], an artificial neural network (ANN) [107, 108], etc. Among these existing methods, an ANN is especially attractive because of its ability to process a large amount of data, extract discriminative features automatically and obtain high accuracy in various applications. The most commonly used algorithms in DFL are the SVM [92], the CNN [5, 109], the AE [43, 110], etc.

In the above ANN-based methods, most of them work by converting the collected RSS signal matrix in each location into a corresponding vector to create the dataset. Because of the specific features underlying vectors in different locations, they formulated the DFL problem as a classification problem and employed ANNs to classify input

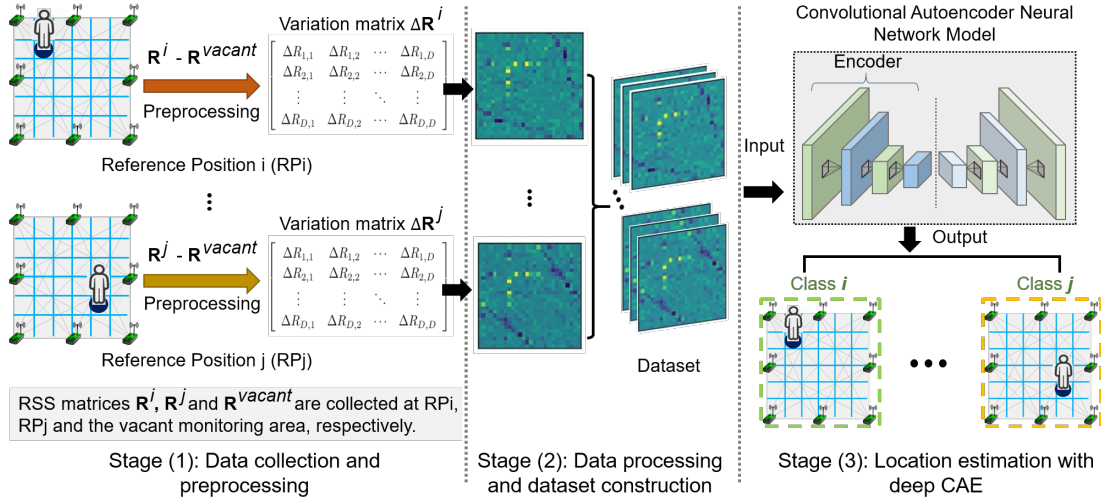


Figure 4.2: Illustrations of data patterns and the flowchart of the system.

vectors.

By discriminating with the abovementioned methods, instead of processing the collected signals into vectors, we propose to treat each RSS matrix as an image and formulate the DFL problem as an image classification problem. As is well known, natural images of the same class, such as images of cats, have many common features or similar underlying distributions in pixel matrices. In contrast, images of different classes have their respective features, such as in images of cats and dogs. Analogously, in a DFL system, different targets' arrangements will produce different features in the RSS matrix if we view the RSS matrix as an image matrix. For instance, if there is only one target, we can obtain a different image when the target is shifted from one location to another. That is, different RSS images (RSS matrices) reflect different patterns that are associated with the different target's locations, as shown in Stage (1) and Stage (2) in Fig. 4.2. The situation is similar if there are multiple targets.

In addition, there are some studies [111, 112] that convert the raw data into equivalent images and obtain good performance. From this point of view, the DFL problem can be transformed into an image classification problem even when the RSS matrix or the image matrix is corrupted by a certain level of noise. Furthermore, in recent years, image classification problems based on a deep neural network (DNN), especially a CNN, have achieved outstanding results [113, 114]. Based on the abovementioned considerations, we formulate the DFL problem as an image classification problem by implementing DNN and expect to achieve a better localization performance.

To solve this image classification problem, we designed a convolutional autoencoder (CAE) neural network which essentially combines the architectures and advantages of CNN and AE. In addition, we expected to achieve a more robust DFL with better localization performance. In detail, a CNN can capture local features of input images because of its spatially located convolutional filters. Moreover, an AE can learn



features from unlabeled data directly, which is an unsupervised pretraining method to learn the nuances of data to improve the classification performance.

The CAE architecture is shown in the Stage (3) of Fig. 4.2. It first uses unsupervised pretraining to initialize the parameters at a good local minimum. In addition, it then fine-tunes the forward encoder part and follows with a classifier to learn spatially specific features of targets in images in a supervised way. After the entire training procedure, the deep learning model can be used to estimate the target's location, i.e., in a testing mode. The major contributions of our study can be summarized as follows:

- We convert RSS matrix data into an image formation and formulate the DFL problem as an image classification problem.
- We design a three-layer CAE architecture and an effective algorithm to learn the parameters of CAE, and then, in the testing mode, the trained neural network can output the locations of DFL targets based on RSS features automatically.
- The localization performance of the CAE is verified as superior to other compared state-of-the-art approaches on a real-world dataset.

The remainder of this chapter is organized as follows. In Section 4.2, the pioneering related works are introduced briefly. Section 4.3 describes the DFL as an image classification problem. The proposed algorithm is presented in Section 4.4. Section 4.5 demonstrates the results of the performance evaluation. Finally, Section 4.6 concludes this work.

## 4.2 Related Works

In this section, we will briefly discuss the studies focusing on DFL problems and the works related to approaches solving these DFL problems.

### 4.2.1 Device-free localization

Youssef et al. [83] first introduced the concept of device-free passive localization, which did not require the targets to carry any device during positioning. In their work, they modeled the problem as a machine learning problem and realized DFL with a fingerprint matching method. Moussa et al. [84] formulated the localization problem as a fingerprint-matching problem and used nominal WiFi equipment for intrusion detection without using any extra hardware. Wilson et al. [85] presented a linear model for using RSS measurements to obtain images of moving objects and first proposed the radio tomographic imaging (RTI) technology. Zhang et al. [86] increased the localization area and introduced more sensor nodes to eliminate the noise effects. Based on these

pioneering works, many subsequent studies [88, 89] based on analyzing the shadowing effect of the target on wireless links have been proposed. For example, Wang et al. [115] proposed an approach based on RSS information from the shadowed and non-shadowed links to achieve robust location estimation performance with low computational cost. Ciunzio et al. [116–118] approached the distributed detection of a noncooperative target with a WSN by observing the amplitude attenuation of the signal depending on the position of the target. Sabek et al. [87] proposed a probabilistic energy minimization framework to locate multiple targets for WiFi-based DFL. [119, 120] took use of radio maps constructed of channel state information (CSI) to improve localization accuracy for single or multiple targets. Wang et al. [44] proposed a deep learning approach for indoor localization based on the CSI matrix.

The above studies provided a foundation for further DFL studies. In addition, to achieve high localization accuracy and improve detection efficiency, many DFL methods have been proposed.

### 4.2.2 DFL algorithms

The existing DFL methods mainly include the RTI approach [121], geometric approach [122], sparse representation approach [123], compressive sensing approach [124], fingerprint-based approach [125], deep learning [126], etc.

The RTI-based localization method images the RSS attenuation, which is caused by targets, with inexpensive and standard hardware. Bocca et al. [127] proposed methods that use RTI in an indoor environment to track multitargets in real time. Liang et al. [91] proposed a half-quadratic optimization algorithm to achieve robustness against outliers in distributed multiple-input multiple-output radar. Zhang et al. [122] proposed a typical geometric approach, the best-cover algorithm, to track multiple targets with calibration on the wireless links. Wang et al. [128] proposed a compressive sensing theory-based dynamic statistical model and applied a Bayesian greedy matching pursuit algorithm to locate the target. The fingerprint method usually involves experiments for constructing a training database in advance and then locates a targets position from new testing data by a classification algorithm. Mager et al. [129] focused on the degradation when the environment changes and make use of the machine learning method, random forests to locate the target. Hong et al. [92] considered signal features on spatial and temporal averaging as training data and then employed SVMs to evaluate the localization performance in two different environments. Zhang et al. [110] collected wireless signals in different periods and proposed a 4-layer stacked denoising autoencoder neural network which obtained unrivaled localization performance. Wang et al. [43] extended wireless RSS matrices into vectors and designed a sparse autoencoder neural network to realize a target's localization. Sun et al. [78] also implemented the RSS matrix as training data

and designed a 3-layer perceptron network with a nonlinear function to achieve a comparable localization performance in the DFL problem. In addition, the studies based on converting signals into images were also proposed. Cai et al. [130] built DFL-related images using the collected raw CSI signals and employed a CNN for classification in an indoor localization task. Wang et al. [61] and Huang et al. [76] transformed the DFL process to a sparse representation-based image classification problem, and employed SC algorithms for locating the targets.

The proposed approach is available for both CSI and RSS. Moreover, the RSS matrices are employed in this chapter. To avoid processing the RSS matrices into vectors, as summarized at the end of the Introduction section, we treat each collected RSS matrix as an image matrix and pre-process the collected raw data by eliminating the effect of the background, which results in more clear patterns in the data. In addition, we make use of the different patterns of image matrices associated with different target locations. In terms of the algorithm, we design a CAE neural network, which combines the advantages of CNN and AE, to perform image classification as well as locating the target.

## 4.3 Problem Statement

The DFL problem in this chapter is same with the one described in Chapter 3. As described in Section 3.3, to collect the data of the DFL system, there are mainly four steps as follows:

- 1) Divide the monitoring area into grids. The grids are divided according to the monitoring area and the cross-section area of the target, as shown in Fig. 4.3.
- 2) RSS matrices collection. In the DFL system, each sensor node performs as a transmitter by turns, and the others perform as receivers following a schedule. Once a schedule is finished, the data of one trial is obtained.
- 3) Data pre-processing. Process the collected RSS matrices by subtracting the corresponding measurements collected without any target in the monitoring area.
- 4) Build image data according to RSS matrices. Each RSS image is built by using the values of measurements as the pixels of the image.

### 4.3.1 Description of the DFL problem

Fig. 4.3 is an illustration of a DFL system model. The monitoring area is surrounded by several wireless transmitter-receivers or called APs. All the APs in a DFL system can communicate with each other and the obstructions will absorb, scatter or diffract the communication transmitted from the APs. Then, each receiver will receive the RSS signals, derived by the target, consisting of absorption, scatter and diffraction.

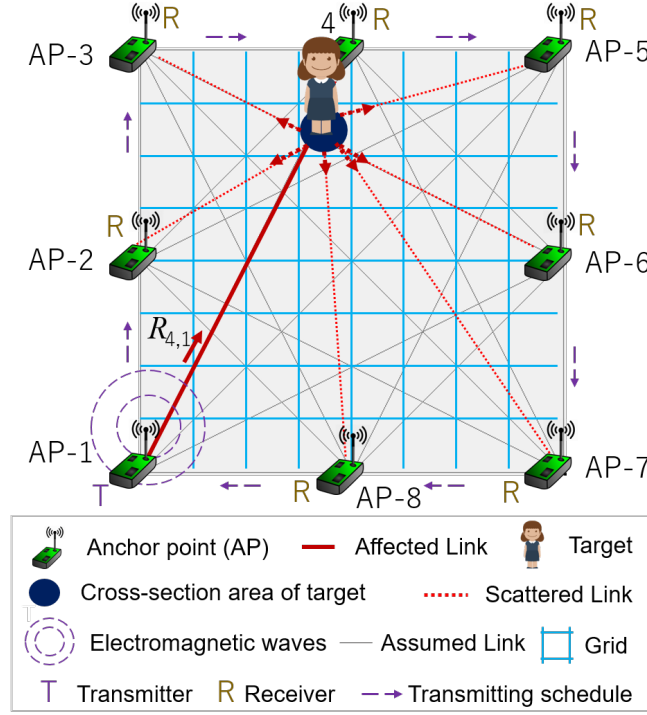


Figure 4.3: Illustration of a device-free localization (DFL) system model.

The details are illustrated in subsection 3.3.1.

As described in subsection 2.2.1,  $R_{m,n}^{\text{vacant}}$  and  $R_{m,n}^{\text{target}}$  denote the measurements of wireless links that are transmitted from the  $n$ -th AP to the  $m$ -th AP without any target and with a target existing in the monitoring area, respectively. In addition,  $\Delta R_{m,n}$  denotes the variation between the two measurements, as shown in Eq. (2.1). In addition, considering the variations of all  $D \times D$  wireless links between all pairs of APs, the variation matrix,  $\Delta \mathbf{R}$ , is established as Eq. (2.2). Similarly, let  $\mathbf{R}^{\text{vacant}}$  and  $\mathbf{R}^{\text{target}}$  represents the RSS matrix consisting of the measurements collected from all the APs. Here,  $D$  denotes the number of APs in the DFL system. Additionally, since the RSS measurements are easily acquired,  $\Delta \mathbf{R}$  can be employed to estimate the position of the target.

### 4.3.2 Formulation of the DFL as an image-classification problem

As described in Section 3.3.3, each RSS matrix can be converted into an image matrix by regarding RSS measurements as pixels of the RSS image. Moreover, the image matrices contain their specific patterns corresponding to the target's positions. In details, the image matrices obtained from close target positions will produce similar features and image matrices obtained from different positions containing respective features. From this point of view, the DFL problem can be formulated as an image classification problem, which can be well solved by a deep learning methods [113].

Assume that the DFL monitoring area is discretized into  $L$  grids as shown in Fig.

4.3 and each bin on the grid acts as one RP, i.e., one class. Thus, all the potential RPs compose  $L$  classes in this DFL problem. For each RPl,  $l = 1, 2, \dots, L$ , we perform experiments with  $p = 1, \dots, P$  trials. In each trial, we can obtain a RSS matrix data,  $V_{lp} \in \mathbf{R}^{D \times D}$ , for each position. Here, let  $S$  denote the total number of data samples, where  $S = L \times P$ . Then, we can obtain the training data with the position information for all grids and trials as  $V = [V_1 \ V_2 \ \dots \ V_s \ \dots \ V_S]$ .

## 4.4 Proposed Methods

In this section, we intend to provide the theoretical background of the proposed deep learning approach for solving the DFL problem.

### 4.4.1 Background

The DNN, one kind of ANN, has attracted tremendous attention because of its frontier performance and advantages. As a strong branch of the DNN family, the AE [19] implements unsupervised pretraining to initialize the parameters for feed-forward neural network (FNN), which in fact improves the FNN in classification performance. The CNN [98, 131] is another popular architecture of the DNN, which can locally extract features from raw spatial data (such as images or video) directly without complex pre-processing. Its local connection and spatial feature detection can not only reduce computational cost but also improve feature extraction ability. The followings part is the brief introduction of the AE and the CNN.

#### Autoencoder (AE)

The AE is an unsupervised learning method that reconstructs the input data as output under certain restrictions. In this architecture, each layer is fully connected with next layer by linear multiplication followed by an activation function. Fig. 4.4(a) is the illustration of an AE architecture that includes the encoder part and decoder part. The encoder part is used to learn the low-dimensional representation features of input data, called the code layer, and the decoder part is employed to reconstruct the code layer back to the original dimension. Note that the raw data, regardless of the type, should be processed into vectors as input data.

The AE is capable of learning features from input data without any labels. It learns to maintain as much information underlying the input data as possible by minimizing the reconstruction error function via backpropagation. Under this pretraining strategy, AEs can extract features from the data themselves, which in fact not only improves the classification performance but also regularizes the network to prevent over-fitting. After pretraining, the decoder part is usually dropped and a classifier, for example, the

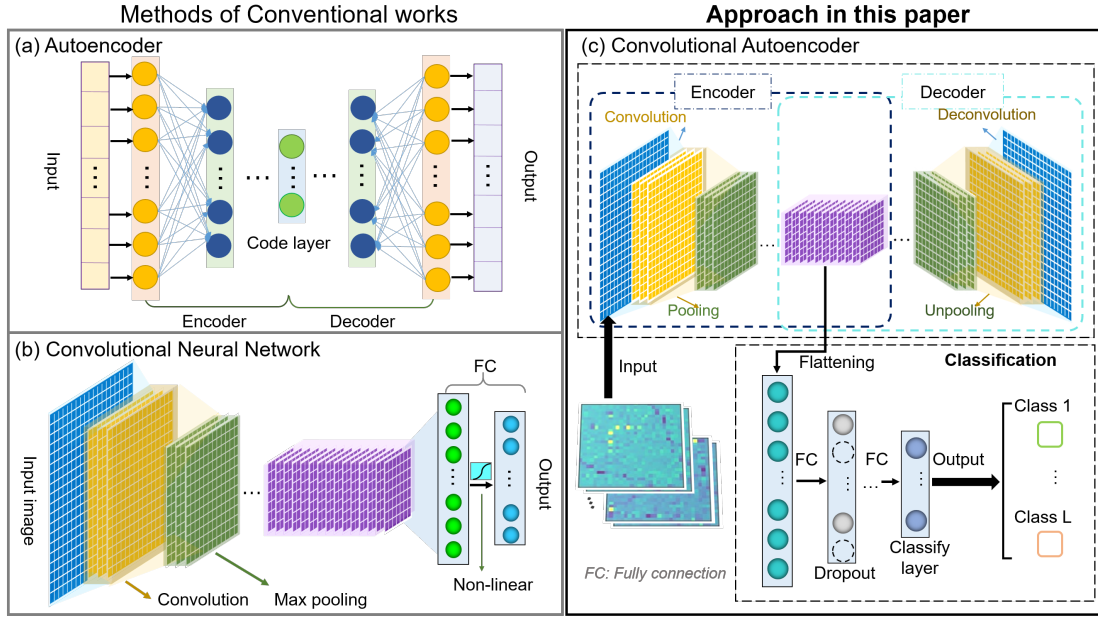


Figure 4.4: Schematic of a convolutional autoencoder (CAE) architecture. All dashed circles in the fully connected layer indicate randomly dropped neurons.

Softmax function [101], is added at the end of the encoder part to train the encoder neural network for classification by using the labeled data.

### Convolutional Neural Network (CNN)

CNNs have attracted considerable attention because of their outstanding ability to deal with two- or three-dimensional data, such as images or videos. Generally, a CNN architecture includes convolutional layers, pooling layers and fully connected layers. Fig. 4.4(b) is an illustration of a CNN architecture including the abovementioned three kinds of layers together with input image data and output classes' possibilities.

Unlike conventional fully connected FNNs, CNNs only focus on specific regions of input feature maps. This local connection benefits the training performance, especially on image classification. In a convolutional layer, a set of 2-dimensional kernels are convolved with the receptive field of the input images or the previous layer's feature maps in a sliding window style to learn data-specific feature maps. The pooling layer, always following a convolutional layer, can subsample to reduce the amount of parameters so that it is not prone to over-fitting. After several convolutional layers and pooling layers, there are generally one or two fully connected layers at the end of CNNs. Similar to the layers in FNNs, the fully connected layers in CNNs are used to learn the combinations of extracted features from previous layers in a nonlinear manner. Finally, an output layer follows to predict the class possibilities.

### 4.4.2 Convolutional autoencoder (CAE) - the proposed architecture

The CAE neural network [132] integrates the merits of convolutional spatial learning of CNNs and the unsupervised pretraining of AEs. Fig. 4.4(c) illustrates the flowchart of the CAE neural network. First, images are taken as inputs to perform unsupervised pretraining by minimizing the differences between the input data and reconstructions. During the pretraining stage, the features underlying the images are extracted. Then, we add several fully connected layers and a classifier to fine-tune the neural network by employing labeled data for classification. Different from the fully connected layers in an AE, a CAE consists of a convolutional encoder and the corresponding deconvolutional decoder. The encoder part is implemented to extract feature maps from input images, while the decoder part is used to reconstruct information from the learned feature maps. As described in Fig. 4.4, the output of the previous layer is employed as the input of the next layer. Here, let  $U$  denote the output feature maps of the previous layer. Then, the encoder computes a nonlinear mapping of the feature maps as follows:

$$A_s = f(c + W * U) \quad (4.1)$$

where  $*$  denote the convolutional operation,  $W$  and  $c$  denote the connection weights and bias terms in the convolutional layer, respectively,  $A_s$  denotes the encoder result of the  $s$ -th sample of the input data and  $f(\cdot)$  denotes the nonlinear activation function. In this chapter, we adopt the ReLU as the activation function defined as  $f(z) = \max(0, z)$ , where  $z$  indicates the input. The decoder reconstruction can be obtained by

$$Z_s = f(c' + W' * A_s) \quad (4.2)$$

where  $Z_s$  is the reconstruction result of the  $s$ -th input and  $W'$  and  $c'$  denote the connection weights and bias terms in the deconvolutional layer, respectively.

In this chapter, max pooling is performed after each convolutional layer as mentioned in subsection 4.4.1. After max pooling, only the maximum value of the receptive field remains, as shown in Eq. (4.3).

$$F_i^q = \max(U_i^q) \quad (4.3)$$

where  $U_i^q$ , ( $q = 1, \dots, Q$ ) denotes the  $q$ -th receptive field of the  $i$ -th feature map and  $F_i^q$  represents the  $q$ -th neuron of the  $i$ -th output feature map. In addition, there is a max unpooling following each deconvolutional layer to restore information. We adopt the nearest interpolation method to scale-up feature maps so that we can avoid too many zeros in the unpooled feature maps. Furthermore, the optimization of the unsupervised

pretraining can be performed by minimizing the error between the input image  $V_s$  and the reconstructed  $Z_s$ , as shown in Eq. (4.4).

$$E(\mathbf{W}, \mathbf{c}) = \frac{1}{S} \sum_{s=1}^S (\mathbf{Z}_s - \mathbf{V}_s)^2 \quad (4.4)$$

After pretraining the CAE, the decoder part is dropped and the fully connected layers together with a Softmax layer are added. Based on the pre-trained parameters, this encoder architecture can learn more abstract features and perform classification with labeled data. Additionally, there are ‘dropout’ operations in the fully connected layers to randomly drop some neurons in each iteration. These operations enable the neural network to learn different combinations of features from the preceding layers as well as to avoid over-fitting. In the supervised training procedure, a set of labeled training data  $(\mathbf{V}, \mathbf{Y})$  is used. We employ the cross-entropy error function as the cost function, as shown in Eq. (2.15), which computes the error between predictions  $\mathbf{Y}'$  and real labels  $\mathbf{Y}$  to update the parameters. The optimization algorithm for both pretraining and fine-tuning is Adam. Additionally, the dropout rate of 0.4 is performed in the fully connected layer.

In addition, the computational time of an algorithm is closely related to the computational complexity. For a CNN based neural network, the total time complexity of all convolutional layers is given as follows [133]:

$$O\left(\sum_h N_h^2 K_h^2 C_{h-1} C_h\right) \quad (4.5)$$

where  $h$  is the index of a convolutional layer.  $C_{h-1}$  is the number of input channels of the  $h$ -th layer, and  $C_h$  is the number of filters of the  $h$ -th layer.  $K_h$  is the filter size, and  $N_h$  is the length of the output feature map. In addition, according to [134], the total time complexity of all fully connected layers is given as follows:

$$O\left(\sum_g B_{g-1} B_g\right) \quad (4.6)$$

where  $g$  is the index of a fully connected layer,  $B_{g-1}$  denotes the number of input neurons of the  $g$ -th layer, and  $B_g$  is the number of output neurons of the  $g$ -th layer.

## 4.5 Performance Evaluation

In this section, we evaluate the performance of our proposal by using a real-world experimental dataset, the 2010 Outdoor RTI Data Set [85], which is same this the outdoor DFL dataset employed in Chapter 2 and Chapter 3. All the experiments are im-



plemented in TensorFlow 1.2.0 open source software on a system with a GeForce GTX 1080 GPU and 32 GB of memory.

### 4.5.1 Experiment configurations

The layout of the wireless sensor network is same as illustrated in Fig. 3.7. As described in Section 3.5.1, the network is deployed on a wide grassy area approximately 15 feet away from the building and employs 28 Crossbow TelosB wireless sensor nodes. Each node is placed 3 feet apart along the  $21 \times 21$  feet perimeter square area and 3 feet away from the ground. In addition, the monitoring area is discretized into 36 grids.

During the signal collection procedure, the DFL system uses a simple token passing protocol to avoid transmission collisions. Each AP is implemented as a transmitter by following a schedule. When an AP transmits, the other APs perform as receivers. When a schedule (or trial) is finished, a base station node collects these RSS measurements and then feeds them to a computer using a USB port for the processing of building RSS images. Each time the computer receives the data, it processes the RSS signals into images and then executes the DFL algorithm for the location data.

In the experiments, 30 trials of the RSSs were measured in a short time interval at every bin on the grid with one target in the region of every grid. Note that in different trials, the target may be at different places, although in the same grid. We select all the 36 RPs as test points for the performance evaluation. The total RSS sample matrices of each bin on the grid are split into two sets, where 25 trials are used as training data, and the remaining 5 are used as testing data. Therefore, in the following experiments, there are 900 training samples and 180 testing samples.

**Metric:** (1) Accuracy. We take the percentage of the count of correctly predicted samples with respect to the count of all samples as the localization accuracy to perform the evaluation of the CAE. (2) Stability. To evaluate the stability of the algorithms, a number of repeated trials are conducted. We employ the probability of reaching the highest localization accuracy, 100%, to evaluate the stability of the algorithm.

### 4.5.2 Data pre-processing

Fig. 4.5 shows an example of the imaging of the signal data together with the pre-processing of the collected raw data. In this figure, the signal data are collected when a target is at RP4 and RP16. Moreover, there are three samples for each RP. In addition, as illustrated in subsection 3.5.2, different colors in the images reveal the values of RSS matrices as shown in the color bars. In Fig. 4.5, there are three kinds of signals, including the original signals  $\mathbf{R}^{\text{target}}$ , signals  $\mathbf{R}^{\text{vacant}}$  and the variation signals  $\Delta \mathbf{R}$ , which is constructed by subtracting  $\mathbf{R}^{\text{vacant}}$  from  $\mathbf{R}^{\text{target}}$ . This figure clearly shows that when the samples are collected at the same RP, they have similar image patterns. However, if the

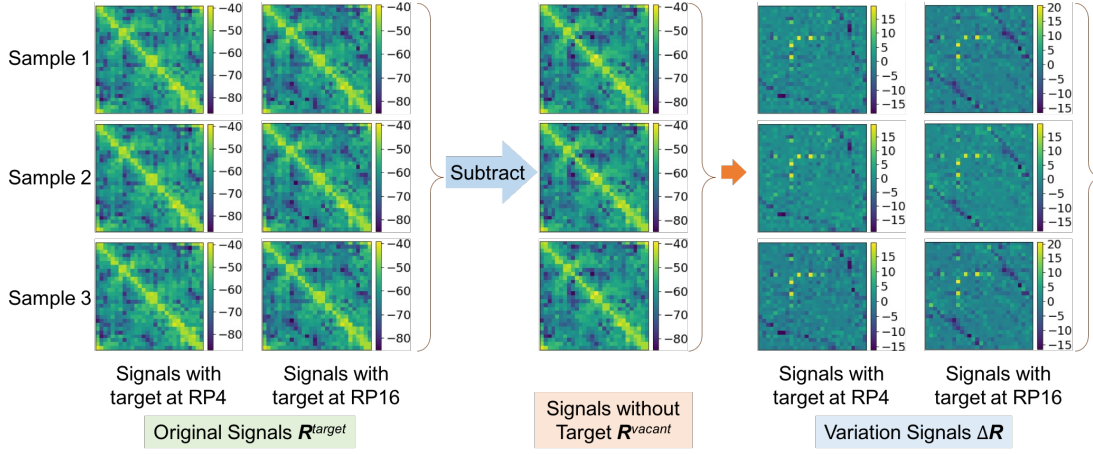


Figure 4.5: Imaging process of RSS matrices from original signals to variation signals at two different reference positions (RPs). Note that different colors in the RSS images reveal the values of the corresponding RSS matrices. The unit for each value of the color bar is dBm.

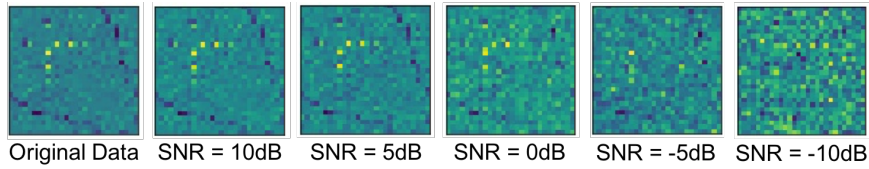


Figure 4.6: Imaging of noisy data collected at the reference position 4 (RP4) with different SNRs.

target's RP is shifted to another, it produces different features in the images. Nevertheless, the differences between samples of different RPs in the original signals  $R^{target}$  are difficult to be recognized artificially. Compared with signals  $R^{target}$ , after subtracting the signals collected without target  $R^{vacant}$ , the features become much clearer in the variation images  $\Delta R$ , which can improve the feature extraction performance. Hence, we adopt the variation signal data as the input data in our experiments.

Fig. 4.6 shows the imaging of noisy variation signal data with different SNRs based on the same RP4. In this figure, except for the beginning original data on the left, the noise with SNR equal to 10 dB, 5 dB, 0 dB, -5 dB and -10 dB are added separately. There is a tendency that with the decreasing of SNR, the data becomes noisier, which means that it becomes more difficult to extract features from signals with the rising of noise degree. The noise seems serious when SNR equals 0 dB, and it is difficult to recognize information from this image with human eyes when the SNR equals -10 dB. All these noisy data are employed in our experiments to evaluate the localization performance of the proposed approach.

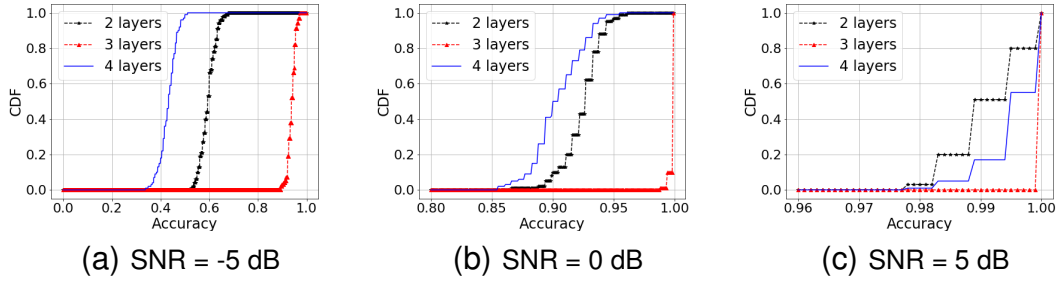


Figure 4.7: Cumulative distribution function of localization accuracy on noisy testing data by using CAE with different convolutional layers.

### 4.5.3 Performance of the CAE in DFL

In this subsection, we exploit the optimal parameters of the CAE for the DFL task. Considering the properties of the CNN and the AE, we mainly discuss 5 factors that will affect the performance of CAE, including convolutional layer numbers, max pooling layers, the size of convolutional layer filters, learning rate and training epoch number. In addition, the abovementioned parameters will be evaluated by trial and error based on the performance of each designed architecture. The optimizing procedure with respect to these factors is shown as follows:

#### Convolutional Layer Number

The cumulative distribution function (CDF) of a real-valued random variable  $X$  is the function given by  $F(X) = P(X \leq x)$ , where the right-hand side represents the probability that the random variable  $X$  takes on a value less than or equal to  $x$ . Fig. 4.7 shows the CDF of the localization accuracy performed by CAE with 2, 3 and 4 convolutional layers on noisy data. Here, we show the cases in which the number of convolutional filters are 32-64, 32-64-128 and 32-64-128-256 for CAEs with 2, 3 and 4 convolutional layers, respectively. The size of all of the convolutional filters is set to  $3 \times 3$ , and the pooling operation is not adopted. The learning rate for unsupervised pretraining is  $10^{-3}$ , and for supervised fine-tuning is  $10^{-5}$ . Moreover, the epoch number for the two stages are 300 and 100. Experiments are performed 100 times on the noisy testing data. The three subfigures 4.7(a)-4.7(c) show results on noisy data with SNR equal to -5 dB, 0 dB and 5 dB, respectively. When the SNR equals -5 dB, as shown in Fig. 4.7(a), all the localization accuracies obtained by the 3-layer CAE are over 89.4%, while the accuracies obtained by the 2-layer and 4-layer CAE are less than 66.7%. When the SNR equals 0 dB, the probability of the 100% localization accuracy obtained by the 3-layer CAE is 0.9. Although the localization accuracy of the other two architectures improves obviously, they can only achieve the localization accuracy of 96.1% at most. When the SNR increases to 5 dB, the probabilities of reaching 100%

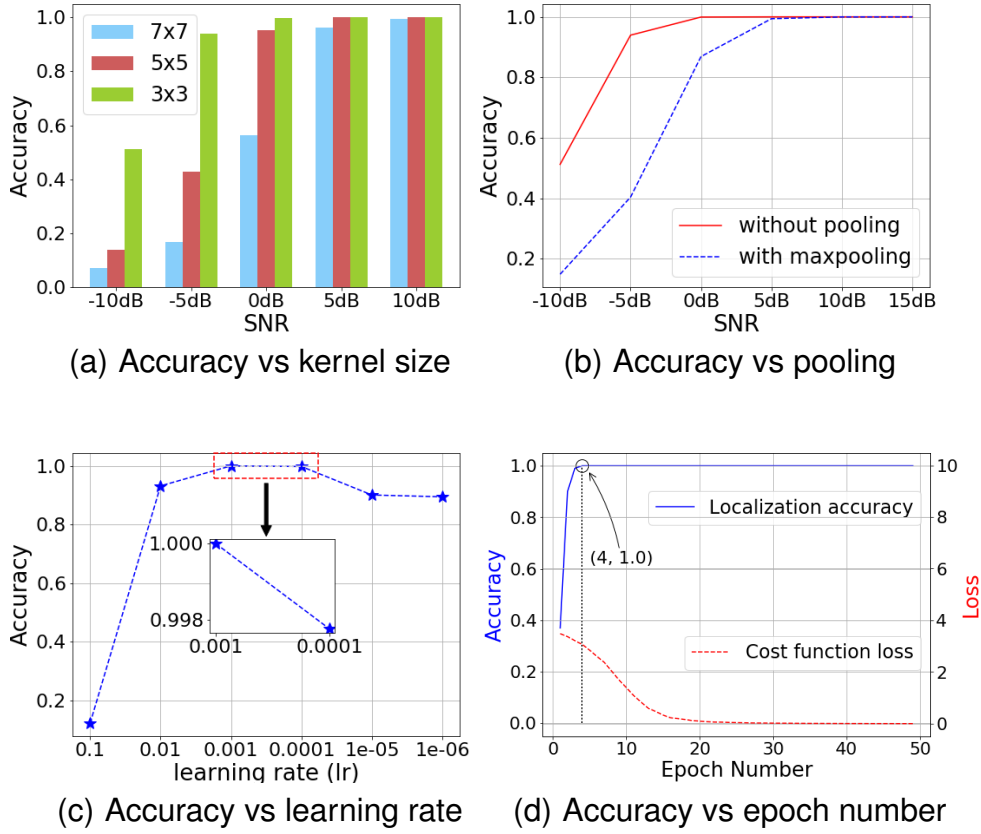


Figure 4.8: Localization performance of the convolutional autoencoder (CAE) with different optimization parameters. For (a) and (b), testing signals are noisy.

localization accuracy are 0.2, 1.0 and 0.45 for the 2-, 3- and 4-layer CAE, respectively. Although there is a tendency that the localization performance of the 3 architectures obviously increases with the increasing of SNR, the 3-layer CAE is superior to the other two on noisy data with all of the different SNR. Hence, in our CAE architecture, 3 convolutional layers are employed.

### Convolutional Filter Size

After deciding on the number of convolutional layers, the size of the convolutional filters needs to be considered. The bar chart in Fig. 4.8(a) shows the localization accuracy of CAE with convolutional filter sizes of  $3 \times 3$ ,  $5 \times 5$  and  $7 \times 7$  on noisy data with the SNR equal to -10 dB, -5 dB, 0 dB, 5 dB and 10 dB. Experiments are performed 100 times on the testing data. As seen from the Fig. 4.8(a), regardless of the filter size used, the accuracy rises sufficiently high for practice when the SNR is larger than 5 dB. When the SNR equals -10 dB, which means the noise is very severe, none of the three CAE architectures can locate the target. If the SNR increases to -5 dB, the CAE with the  $3 \times 3$  filter size can achieve an average localization accuracy of 94.0%, which is 51.1%

higher than the  $5 \times 5$  one. Additionally, when the SNR is equals to 0 dB, with the  $3 \times 3$  convolutional filter, the CAE achieves 100% localization performance. Under the same SNR, the accuracies of the CAE with the  $5 \times 5$  and  $7 \times 7$  kernel sizes are approximately 95.4% and 56.2%, respectively. When the SNR rises to 10 dB, all three CAE architectures can achieve the highest 100% localization accuracy. According to this bar chart, when the SNR is less than 5 dB, it is obvious that the CAE with the  $3 \times 3$  filter always has the highest localization performance. To summarize,  $3 \times 3$  is the most appropriate size of convolutional layer filters and has a more robust ability to deal with noise.

### **Number of Convolutional Filters in Each Layer**

The number of convolutional filters that are used to generate activation maps are also very important to the performance of the CAE. Generally, the more filters there are, the more information we can obtain on the input volume. According to the previous studies of classical CNN, such as LeNet [135], AlexNet [31] and ResNet [114], the convolutional filter number in the first layer differs. Referring to the abovementioned neural networks and considering the lower memory cost, we empirically choose 32, 64 and 128 as the filter numbers of the CAE architecture in this study.

### **Max Pooling Layer**

The most common CNN architecture is composed of several alternating convolutional and max pooling layers, followed by one or two fully connected layers. However, Springenberg and Dosovitskiy et al. [100] proposed that replacing max pooling by a convolutional layer with an increase in stride can also achieve high accuracy on some image recognition tasks with a shallower architecture. After choosing the convolutional layer number, convolutional filter size and channel number, we perform experiments on two architectures: one is the CAE without pooling layers and the other one includes max pooling after each convolutional layer. Fig. 4.8(b) shows the localization accuracy of the CAE with and without the pooling layer performed on noisy data with various SNRs. Experiments are performed 100 times on the testing data. According to this graph, the localization accuracies of both CAE architectures increase sharply when the SNR rises from -10 dB to 0 dB. When the SNR equals 0 dB, the CAE without the pooling layer can achieve the localization accuracy of 100%, which is better than the 86.9% achieved by the CAE with max pooling. When SNR is greater than 10 dB, both CAE architectures can achieve the highest 100% localization accuracy. In the SNR range of -10 dB to 10 dB, which means severely noisy signals, it is clear that the CAE architecture without pooling performs better. In summary, the graph in Fig. 4.8(b) proves the dominance of the CAE architecture without pooling on the localization task with noisy data.

## Learning Rate

The learning rate mainly decides the updating rate of the parameters, and it is a critical factor in the performance of DNN. The learning rate is usually difficult to decide directly because with a large learning rate, a DNN may typically miss the global minimum value; however, with a small learning rate, it always takes a much longer time to train. Therefore, only when the learning rate is properly coordinated with the other hyperparameters does the neural network have a great probability to work well. In our experiments, the learning rate in the fine-tuning stage is fixed at a small value,  $10^{-5}$ , to avoid distorting pre-trained parameters too quickly and too much. The following discussion regarding the learning rate refers to pretraining, which is vital for the performance of CAE. Fig. 4.8(c) reveals the average testing localization accuracy of the CAE trained with several learning rates,  $10^{-1}$ ,  $10^{-2}$ ,  $10^{-3}$ ,  $10^{-4}$ ,  $10^{-5}$  and  $10^{-6}$ . Experiments are performed 30 times in the testing stage. In this graph, a very noticeable trend is the initial increase and later decrease of the average testing localization accuracy with the decline of the learning rate. According to Fig. 4.8(c), although the accuracy achieves close to 100% when the learning rate is  $10^{-3}$  and  $10^{-4}$ , the average accuracy of the CAE with the former one is slightly higher than the latter at 99.8%. Considering the higher localization accuracy and lower time cost in the training procedure, the learning rate  $10^{-3}$  has obvious advantages over the others.

## Epoch Number

An epoch is a measurement of the time used by all of the training vectors are to update the weights once. In some ways, the epoch number can affect the performance of the algorithm. If only a small epoch number is used in training, poor or under-fitted results would be obtained. On the other hand, if the neural network is trained too long, it may also result in over-fitting. This result is because the neural network would ‘memorize’ the desired results for the training inputs in supervised learning. Fig. 4.8(d) shows the average convergence epoch number of the CAE neural network in experiments of 10 trials on the training data. It is clear that the training accuracy increases sharply from 40% to 100% within only 5 epochs and remains constant at the highest 100%. According to Fig. 4.8(d), a very noticeable trend is the steady decrease in the loss with the increasing epoch numbers and the decline to approximately 0 at 20 epochs. In light of the classification loss function in Eq. (2.15), when the loss is lower than approximately 3.54, the target can be located correctly. The reason for the decreasing of loss when the epoch number is greater than 5 is that the outputs are moving closer to the desired labels with the training processing. However, although it seems that 5 epochs are sufficient for this neural network in the training stage, we prefer to apply 20 or more epochs (the number of epochs for loss convergence) to ensure the performance on the testing

Table 4.1: Optimal parameters of the CAE neural network

Key parameters	Optimization
Layer number	3
Convolutional filter size	$3 \times 3$
Filter number of each layer	32-64-128
Max pooling layer	Without pooling
Epoch number in pretraining stage	300
Epoch number in fine-tuning stage	$\geq 20$
Learning rate in pretraining stage	$10^{-3}$
Learning rate in fine-tuning stage	$10^{-5}$

localization performance.

In summary, as shown in Table 4.1, there are 3 convolutional layers with an increasing number of filters, 32, 64 and 128 in the CAE. Instead of using max pooling layers, we employ the convolution strategy of a  $3 \times 3$  convolutional filter with a stride of 2 to perform downsampling. After encoding, the bottleneck compresses the data from  $28 \times 28$  into  $4 \times 4$  with 128 channels. After unsupervised pre-training, only 20 epochs are needed to fine tune this CAE neural network, and the learning rate is  $10^{-3}$ .

#### 4.5.4 Localization performance and comparison of CAE, CNN and AE

After finding the CAE architecture parameters, this subsection will discuss the localization performance of the CAE together with the CNN and AE. To conduct a comparison, we employ the same architecture in the CAE and CNN. Regarding the AE, we adopt the suggested architecture of Seyfioglu et al. [101] who used an AE to implement classification task and achieved good performance. There are three hidden layers with 200, 100 and 50 neurons in the encoder part and the classifier is a Softmax function. Note that the working process of these approaches can be divided into two phases:

(1) Training phase. After designing the architectures of these neural networks, they are trained by employing the training data with labels of RPs in the offline form.

(2) Locating phase (or testing phase). In the online locating stage, the signal for a short period is collected when a target enters the monitoring area. The signal will be processed into images and then fed into the well trained model. Finally, the model estimates the location of the target by outputting the RP with the highest probability.

Note that in the normal DFL experiments, the sensing data are usually collected from wireless sensors in a static or relatively clean environment. However in practice, there is a variety of harmful dynamic information, such as the vibration of transceivers caused by electromagnetic interference of surrounding wireless devices, which will result in collecting various levels of noise in the RSS signals. Because RSS is easily

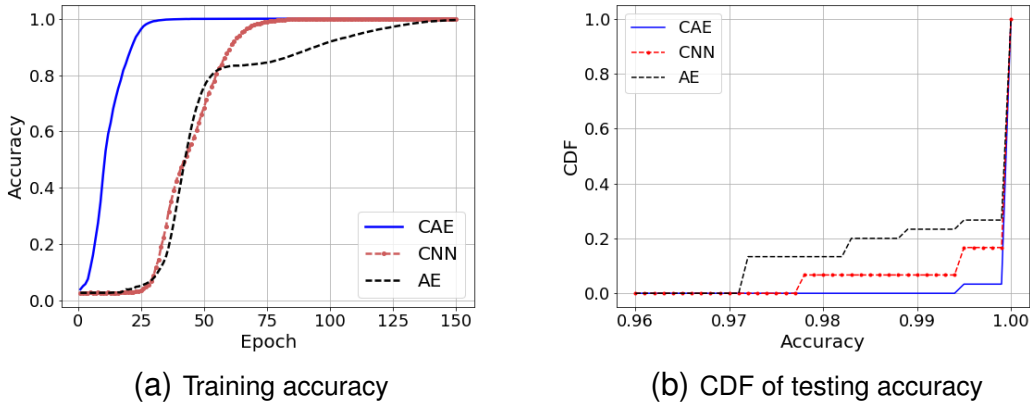


Figure 4.9: Localization accuracy of convolutional autoencoder (CAE), convolutional neural network (CNN) and autoencoder (AE) on noisy training data with SNR= 0 dB.

affected by noise in real environments, to show the robust ability against noise of these algorithms, we perform experiments in three conditions including adding noise to the training data, adding noise to the testing data and adding noise to the entire dataset.

#### Add Noise into the Training Data

In the following experiments, we input noisy data with an SNR equal to 0 dB into the CAE, the CNN and the AE as training data and test their localization performance by using raw testing data. The results in Fig. 4.9(a) indicate the performances of the three abovementioned algorithms on noisy training data with an SNR equal to 0 dB. This chart shows the average performance on 30 trials of the training procedure. It indicates that the training accuracy rises with the increasing of epoch number. Moreover, all three architectures can finally achieve the accuracy of 100% within 150 epochs. To achieve the rate of the highest training accuracy, the CNN needs approximately 80 epochs, the AE requires approximately 150 epochs, and the CAE takes only 30 epochs. This result reveals that the accuracy can increase with the least epochs by employing the CAE architecture.

Fig. 4.9(b) shows the localization performances on raw testing data, which are performed 30 times. The figure shows the CDF of the localization accuracy performed by the CAE, CNN and AE. This chart indicates that within the 30 experiments, the localization accuracy obtained by the AE, CNN and CAE distributes from 97.2% to 100%, 97.8% to 100% and 99.4% to 100%, respectively. It can be seen that the probabilities of the localization accuracies achieved by the AE, CNN and CAE architectures for the highest 100% are approximately 0.73, 0.83 and 0.97, respectively. It can be concluded that the distribution range of localization accuracies obtained by the CAE is the smallest among the three architectures. In addition, the CAE can reach and maintain the highest



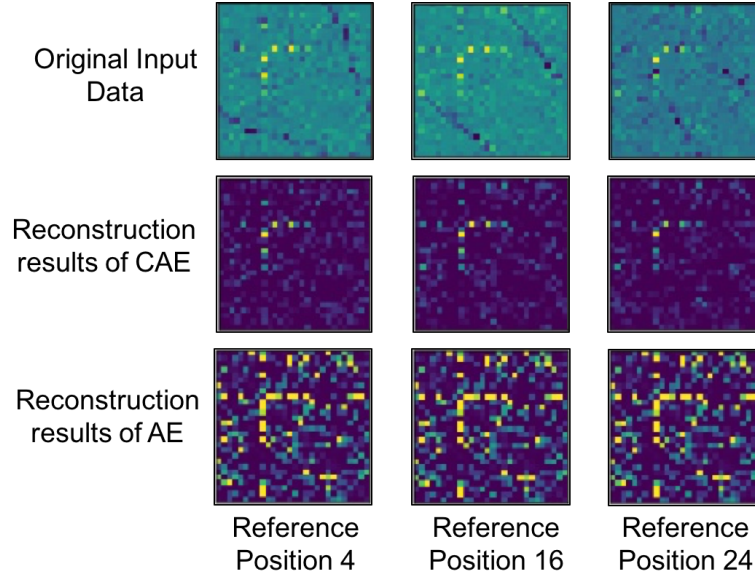


Figure 4.10: Reconstruction performance of convolutional autoencoder (CAE) compared with autoencoder (AE). Here, the signals of reference position 4, 16 and 24 are taken as examples.

100% localization accuracy. It is obvious that CAE performs better than the other two methods in achieving a higher and more stable localization accuracy.

#### Add Noise into the Testing Data

The reconstruction procedure of the CAE is the process of learning the features from the data themselves, which can improve classification performance. Fig. 4.10 shows the examples of the reconstructed signal at three different RPs, which are obtained by using the CAE and AE. The first column shows the original input images, the middle three images are the signals reconstructed by the CAE, and the images in the final column are the reconstructions by the AE. From this figure, it is obvious that there are many visible outliers in the images obtained by the AE. In comparison, although the reconstructed results of the CAE only maintain part of the obvious features of the original data, they have much less noise than the results of the AE. From this figure, we can understand that both the AE and CAE have the ability to learn features directly from unlabeled data in an unsupervised fashion; however, it is obvious that the CAE architecture achieves better performance in our experiment.

To indicate the robust ability of the CAE, AE and CNN, we add several different level of noise to the testing data for the performance evaluation. Fig. 4.11 shows the average localization accuracy of the CAE, CNN and AE on noisy data with SNR equal to -10 dB, -5 dB, 0 dB, 5 dB, 10 dB and 15 dB. Experiments are performed 100 times in the testing stage. Note that during the range of -10 dB to 0 dB, the CAE maintains the highest localization accuracy among the three approaches and achieves the localization

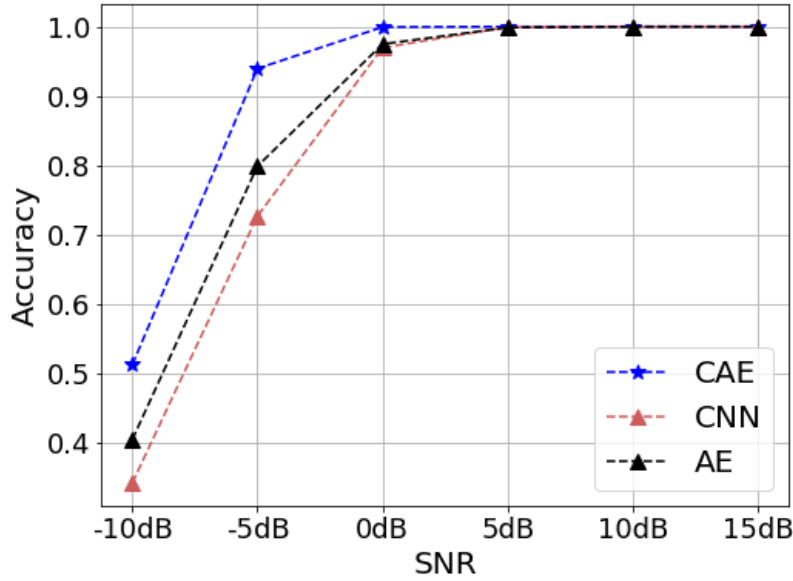


Figure 4.11: Comparison of the average localization accuracy among the convolutional autoencoder (CAE), convolutional neural network (CNN) and autoencoder (AE) on noisy testing data.

accuracy of 100% at 0 dB. Furthermore, even with severe noise such that SNR equals -5 dB, the CAE can still achieve the localization accuracy of 94.0%, which is 21.3% higher than that of the CNN. According to Fig. 4.11, the localization accuracy of the three architectures increases concomitantly with the increasing SNR, and all architectures achieve the highest accuracy of 100% when the SNR is greater than 5 dB. Although all three architectures have a robust ability to noise, the graph proves the dominance of the CAE's robustness to severe noisy data compared to the CNN and AE.

### Add Noise into Both Training and Testing Data

In this subsection, we perform an experiment on a noisy dataset where both training and testing data are noisy. Fig. 4.12 shows the average localization accuracy of the CAE, CNN and AE on the noisy dataset with the SNR equal to -5 dB, 0 dB, 5 dB, 10 dB and 15 dB. Experiments are performed 30 times in the testing stage. There is an obvious tendency that with the increasing SNR, the accuracies obtained by all three approaches rise. Note that in the range from -5 dB to 15 dB, the average localization accuracies achieved by the CAE maintain the highest performance. When the SNR equals or is higher than 5 dB, all the average localization accuracies obtained by the CAE is 100%, which is higher than the 96.3%, 97.5% and 97.6% achieved by the AE. In the same range from 5 dB to 15 dB, the average accuracies obtained by the CNN, 97.9% , 98.9% and 99.6%, are also lower than that of the CAE. Even when the noise is severe such that

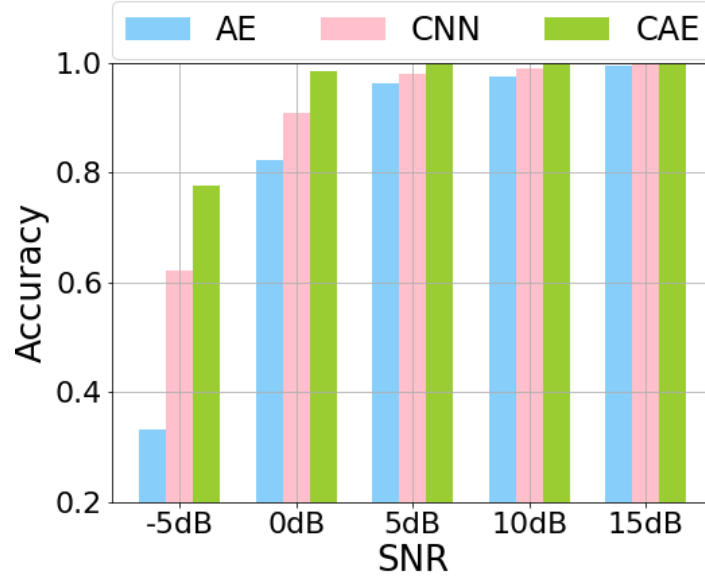


Figure 4.12: Comparison of the average localization accuracy among the convolutional autoencoder (CAE), convolutional neural network (CNN) and autoencoder (AE) on the noisy dataset

the SNR decreases to 0 dB, the CAE can still achieve an average localization accuracy of 98.5%, which is 7.4% and 16.2% higher than the average accuracies obtained by the CNN and AE, respectively. The localization performance in this subsection reveals the superiority of anti-noise ability of the CAE.

Here, we also show the example of misclassified position. Fig. 4.13 shows an instance of the confusion matrix of different RPs, which is a testing performance of the CAE based on a noisy dataset with SNR equal to 0 dB. From this figure, we can draw two conclusions. (a) Although the SNR of the dataset is serious, the CAE could classify most of the testing data with a high accuracy of 98.9%. (b) Closer locations are more easily misclassified. After testing the total set of 180 samples, only 2 samples that belong to RP2 and RP23, shown in this figure, are mistakenly located at RP8 and RP30, respectively. However, according to the distribution of the corresponding RPs in Fig. 3.7, RP8 is a neighbor position of RP2, and RP30 is very close to RP23. Therefore, considering that the noise added to the dataset is random and serious, misclassification is more likely in these instances.

Combining the experimental results from subsection 4.5.4, subsection 4.5.4 and subsection 4.5.4, the CAE architecture has the most stable performance on localization for noisy data regardless of the training stage, testing stage or both.

### Conclusion of Localization Performance

To provide further comparison and demonstrate the distinguished performance of the CAE, several existing methods are compared except for the previously discussed

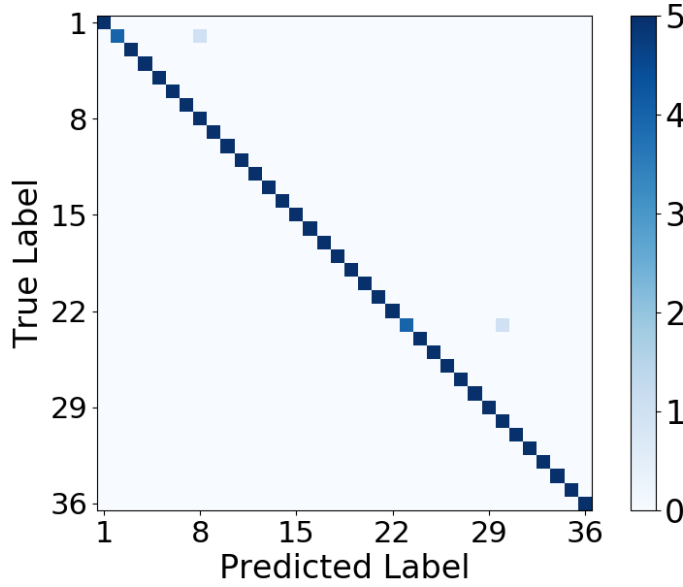


Figure 4.13: Confusion matrix of different reference positions (RPs) for the convolutional autoencoder (CAE). The sample shows the experiment result of the localization performance on the noisy dataset in the testing stage with the SNR equal to 0 dB. The overall accuracy is 98.9%.

CNN and AE. Specially, two methods were evaluated by using the same open dataset described in this chapter. One is the SRC-CVX [61], and the other one is the sparse coding with iterative shrinkage-thresholding algorithm (SC-ISTA) [76]. According to the work of [61] and [76], the data used in their experiments is the original RSS signal without eliminating the background information. Furthermore, we also perform experiments by using the SVM [136] and CNN-1D [137], which are commonly used for classification. In detail, for SVM, a radial basis function (RBF) kernel and the one-vs-one strategy are utilized for the multi-class classification task of this chapter.

Table 4.2 shows the average localization accuracies obtained by the previously mentioned approaches in the testing stage. In this table, there are three experimental conditions, including adding noise to training data, adding noise to testing data and adding noise to the entire dataset. From this table, it can be seen that when performing experiments by using noisy training data with SNR equal to -5 dB and a noisy dataset with SNR equal to 0 dB, the AE, CNN, and CNN-1D can obtain high localization accuracies of over 81.0%. However, the performances of the three methods on another condition are not quite well by getting localization accuracies of under 80.0%. Although the SVM can achieve the accuracy of 91.4% when testing on noisy data with SNR equals -5 dB, its performance on the other two conditions are terrible by obtaining accuracies under 50%. With regard to SC-ISTA and SRC-CVX, the obtained localization accuracies are lower than 80%. Overall, it is clear that the CAE has the most stable performance on noisy data especially when the noise is severe. Also, the CAE can obtain the high localization accuracies, which are over 94.0%, under all the three conditions.

Table 4.2: Comparison of Localization performance

	Noisy training data (-5 dB)	Noisy testing data (-5 dB)	Noisy data set (0 dB)
CAE	100%	94.0%	98.5%
BE-CNN	94.1%	72.7%	91.1%
BE-AE	81.2%	79.9%	82.3%
CNN-1D [137]	97.0%	65.8%	86.2%
BE-SVM [136]	46.5%	91.4%	2.8%
SC-ISTA [76]	78.9%	11.1%	2.8%
SRC-CVX [61]	19.4%	33.3%	2.8%

Note: For the noisy training data (-5 dB), training is performed with noisy data with the SNR equal to -5 dB, and testing is performed with raw data. For the noisy testing data (-5 dB), training is performed with raw data, and testing is performed with noisy data with the SNR equal to -5 dB. For the noisy dataset (0 dB), noise is added to both training and testing data with the SNR equal to 0 dB.

#### 4.5.5 Analysis of computational complexity

According to Eq. (4.5) and Eq. (4.6), the computational complexity is associated with the dimension of the input data and the parameters of the neural network. Here, the dimension of each sample of the input data is  $D \times D$  according to (2.2), and the number of classes is  $L$ . In this chapter, for the CNN and CNN-1D, the computational complexity of training is  $O(4424D^2 + 1024L)$ . For the AE, the computational complexity in the pretraining stage and training stage is  $O(400D^2 + 50000)$  and  $O(200D^2 + 50L + 25000)$ , respectively. Additionally, the computational complexity of the SVM for a single prediction is  $O(n_{SV} \times D^2)$  [138], where  $n_{SV}$  denotes the number of support vectors. By performing our designed CAE, the computational complexity of the training stage is the same as that of the CNN and CNN-1D and is  $O(11880D^2)$  for the pretraining stage.

A comparison of the computational time and the memory of the neural network parameters is given in Table 4.3. Considering that float32 occupies 4 bytes, the memory of the parameters listed in the table is calculated as 4 times the number of the parameters in each neural network. The time costs are based on running the methods by utilizing TensorFlow 1.2.0 on a system with a GeForce GTX 1080 GPU, as mentioned in Section 4.5, except SC-ISTA and SRC-CVX, for which a CPU is feasible. The number of epochs for pretraining is 300, and for training, the number is 250. The training times for all the methods are tested by running all the raw training data (a total of 900 samples). The testing times for all the methods are tested on all the raw testing data (a total of 180 samples) under the same environment. In addition, the testing times in this table are the average results of 10 times results. Although the memory of the parameters and the training time of the CAE are relatively higher than the other methods, the pretraining of the CAE indeed improves the localization performance, which is much more important for DFL. In addition, the training procedure of the CAE is finished in the offline stage,

Table 4.3: Computational Complexity for All Methods

	Params. for Pretraining	Params. for Training	Training Time	Testing Time
CAE	0.2 M	2.2 M	124 s	4 ms
BE-CNN	-	2.2 M	72 s	3 ms
BE-AE	0.36 M	0.2 M	86 s	1.4 ms
CNN-1D [137]	-	1.8 M	69 s	3 ms
BE-SVM [136]	-	-	1.8 s	20 ms
SC-ISTA [76]	-	-	-	14.5 ms
SRC-CVX [61]	-	-	-	0.3 s

Note: All the methods, excepting SC-ISTA and SRC-CVX in the table, are performed by utilizing TensorFlow 1.2.0 on a system with a GeForce GTX 1080 GPU. The number of epochs for pretraining is 300, and for training, the number is 250. The training times for all the methods are tested by running all the raw training data (in total, 900 samples). The training time for the CAE and the AE includes the time cost in both pretraining stage and fine-tuning stage. The testing times for all the methods are tested on all the raw testing data (in total, 180 samples) under the same environment.

which has little influence on the time cost of online locating. It should be noted that the average time cost of the CAE proposal for testing is 4 ms, which is sufficiently short for online locating.

#### 4.5.6 Disadvantages and improvements of the CAE

Regarding the discussion in Subsection 4.5.5, because of the unsupervised pretraining, the memory of parameters and the total training time of the CAE are slightly increased. However, for the localization task, the training process is usually completed in an offline stage. Therefore, it will not significantly increase the time cost of the online testing stage. Moreover, in practical applications of DFL, localization accuracy is much more important. According to the localization performance mentioned in the previous sections, such as Section 4.5.4, the CAE, which benefits from pretraining, has superior performance. Therefore, a slight increase in the training time of the CAE is acceptable.

With regard to the training computational complexity, in our designed CAE, CNN and CNN-1D, the number of neurons in the fully connected layer is large, which results in more memory cost of the parameters. To overcome this problem, the fully convolutional layers could be considered to replace the fully connected layers in future work.

Moreover, the memory overflows as the size of the problem increases significantly. A promising and effective way to address this problem is reducing the data size by some dimension reduction methods. These include principal component analysis and linear discriminant analysis, which have been reported in the literature for improving the efficiency of localization process [76, 139].

Table 4.4: Impact of the number of sensors for the CAE

Number of sensors	Raw data set	Noisy data set (0 dB)
4	63.9%	23.9%
8	96.7%	57.2%
12	99.4%	88.3%
16	99.4%	92.8%
20	100%	95.6%
24	100%	96.1%
28	100%	98.5%

Note: For the noisy dataset (0 dB), noise is added to both training and testing data with an SNR equal to 0 dB.

### 4.5.7 Effect of different system parameters for the CAE

As discussed in 4.5.1, there are 28 wireless sensors and 36 RPs in the DFL system. In addition, the RSSs are measured in 30 trials for each one of the 36 RPs, where 25 trials are used as training data, and the remaining 5 trials are used for testing. Therefore, in all the abovementioned experiments, there are 900 training samples and 180 testing samples. To explore the impact of the dataset for the CAE, we discuss the impacts of the number of sensors and the number of packets (or samples) for training as follows:

#### Impact of the Number of Wireless Sensor Nodes in DFL System

To evaluate the influence of the number of wireless sensor nodes employed in the DFL system, we perform experiments by employing different numbers of sensor nodes under two conditions: adding noise to both training and testing data with the SNR equal to 0 dB and raw data. To ensure the reliability of the experiments, we use the same number of sensor nodes in each side of the monitoring area. As shown in Table 4.4, the localization performances of employing different numbers of sensor nodes based on the two previously mentioned conditions are revealed. It is clear that with the increasing number of sensor nodes, the localization accuracy rises obviously, especially on noisy data. With regard to the condition with noise, the localization accuracies rise sharply with the increasing number of sensor nodes. It can be seen from the table that the localization accuracy grows noticeably from 23.9% to 98.5%. This finding indicates that the greater the number of sensor nodes that are employed in the DFL system, the better the localization performance that will be achieved, especially the antinoise performance. In this noisy condition with the SNR equal to 0 dB, using 16 sensor nodes can achieve high localization accuracy of 92.8%. When performing experiments with raw data, which means data without adding white Gaussian noise artificially, 8 sensor nodes are sufficient to locate the target with the CAE by achieving the localization accuracy of 96.7%.

Table 4.5: Impact of the number of packets for the CAE

Number of train packets	Noisy training data (0 dB)	Noisy testing data (-5 dB)	Noisy data set (0 dB)
180	99.1%	69.9%	91.6%
360	100%	77.8%	94.7%
540	100%	89.7%	96.6%
720	100%	93.2%	97.7%
900	100%	94.0%	98.5%

Note: For the noisy training data (-5 dB), training is performed with noisy data with an SNR equal to -5 dB, and testing is performed with raw data. For the noisy testing data (-5 dB), training is performed with raw data, and testing is performed with noisy data with an SNR equal to -5 dB. For the noisy dataset (0 dB), noise is added to both training and testing data with an SNR equal to 0 dB.

### Impact of the Number of Training Samples on Localization

Here, we set different numbers of samples for training data and evaluate the performance of the CAE by employing the rest of the samples as testing data. Table 4.5 shows the evaluation results of different numbers of training samples for the CAE based on three conditions: adding noise to training data with an SNR equal to 0 dB, adding noise to testing data with an SNR equal to -5 dB and adding noise to the entire dataset with an SNR equal to 0 dB. Under the first condition, the localization accuracy rises from 99.1% to 100% by increasing the number of training samples from 180 to 360. When testing on noisy data with the SNR equal to -5 dB, if the number of training samples increases from 180 to 900, the localization accuracy grows sharply from 69.9% to 94.0%. During the same period, the localization accuracy rises gradually from 91.6% to 98.5% with regard to the third condition. From this table, it can be concluded that there is a tendency that with the increasing number of training samples, the localization accuracy rises. In addition, when the number of training sample is sufficient for this task, i.e., 720 in this table, the growth of performance will become slight.

## 4.6 Summary

In this chapter, we first formulate the DFL as an image classification problem by building the DFL-related RSS images. To make full use of RSS information, we conduct a pre-process procedure of the background elimination on each RSS image to enable discrimination of the feature pattern. Then, we present a CAE neural network for performing the classification process. In particular, the CAE combines the merits of both the AE and CNN in terms of the unsupervised pretraining and convolutional spatial feature extraction strategy.

The experimental results on the real-world dataset show that the proposed approach demonstrates a superior localization performance to that of other compared DFL ap-



proaches, especially when the noise in the environment is serious. In detail, the CAE can achieve a localization accuracy of 100% after training on noisy data with an SNR equal to -5 dB, obtain a localization accuracy of 94.0% when testing on noisy data with an SNR equal to -5 dB and achieve a localization accuracy of 98.5% if noise is added with an SNR equal to 0 dB to both training and testing data. In addition, the average time cost of testing is 4 ms, which is sufficiently fast for online localization. Furthermore, we have also discussed the effects of different parameters of the DFL system on the proposed approach and demonstrated its good performance under such conditions. In detail, the results show that 8 sensor nodes are sufficient to achieve high localization accuracy of 96.7% when performing on raw data. In addition, employing 16 sensor nodes can achieve a localization accuracy of 92.8% when performing on noisy data with an SNR equal to 0 dB.

In summary, although the pretraining of the CAE will increase the computational complexity of training and memory, it indeed improves the performance of DFL. All the results clearly reveal the advantages of robustness of the CAE, even when the noise in the environment is severe.

# Chapter 5

## Conclusion

To achieve the DFL problem accurately and efficiently, we proposed three schemes in this thesis in an improving manner.

First, we formulate the DFL problem as a classification problem, present an improved autoencoder based on RBMs method, named GBRBM-AE, and conduct softmax regression function for accurate target localization of outdoor DFL. Experiment results show that when testing on noiseless data, the location accuracy can get to 97% with 20-dims data, which evidences the GBRBM-AE has good performance on dimension reduction data. When testing on noisy data, GBRBM-AE with 20-dims data is able to maintain high accuracy of 97.1% with  $\text{SNR} = 10$  dB. It indicates that the GBRBM-AE can perform well on DFL problems with the real-world dataset. To illustrate the good performance of our proposal, we also perform experiments via autoencoder without GBRBM and CNN-1D for comparison. Experimental results indicate that the GBRBM-AE outperforms the other two compared methods in both localization accuracy and robustness. However, the localization obtained by GBRBM-AE will be decreased if the noisy level of data is with SNR smaller than 10 dB.

To enhance the robustness of the DFL approach, we formulate the DFL problem as an image classification problem. First, we regard the RSS matrix as an image matrix and conduct a pre-process of BE to dig out the variation component with distinguished features. Then, we take use of these feature-rich images by formulating DFL as an image-classification-problem. Furthermore, for obtaining a robust algorithm, a deep CNN is exploited to address the DFL-related image-classification-problem. The localization performance of the proposed BE-CNN approach is validated in our real testbeds of indoor DFL system and in a real-world dataset of outdoor DFL. According to the experimental results, for indoor DFL, the proposed BE-CNN can achieve localization results with high accuracy of 91.7% with SNR equal to 15 dB; for outdoor DFL, the BE-CNN could obtain the accuracy of nearly 100% with SNR equal to 5 dB. Additionally, the localization performance of the proposed method has superiority than the comparison methods of SVM, KNN and AE, especially when the surrounding noise in

---

the environment is severe. All these results demonstrate the effectiveness of the proposed BE-CNN on solving the DFL problem.

Moreover, considering the level of noise in the practical applications is unpredictable, to improve the anti-noise ability of the DFL approach, we presented a CAE neural network for performing the classification process. In particular, the CAE combines the merits of both the AE and CNN in terms of the unsupervised pretraining and convolutional spatial feature extraction strategy. The experimental results on the real-world dataset show that the proposed approach demonstrates a superior localization performance to that of some other compared DFL approaches, especially when the noise in the environment is serious. In detail, the CAE can achieve a localization accuracy of 100% after training on noisy data with an SNR equal to -5 dB. In addition, the average testing time cost is 4 ms, which is sufficiently fast for online localization. Furthermore, we also discussed the effects of different parameters of the DFL system on the proposed approach and demonstrated its good performance under such conditions. In detail, the results show that 8 sensor nodes are sufficient to achieve good localization accuracy of 96.7% when performing on raw data. In addition, employing 16 sensor nodes can achieve a localization accuracy of 92.8% when performing on noisy data with an SNR equal to 0 dB.

According to the results of this work, we can conclude that our proposed BE-processing based GBRBM, CNN and CAE schemes have the effectiveness on solving the DFL problem. In addition, the CAE performs as the best scheme in terms of the anti-interference ability of noise and online testing time.

# Acknowledgment

Throughout the time as a doctoral student in the University of Aizu, I have received a great deal of support and encouragements from many people. Without their inspiration and assistance, I have no enough confidence to finish this dissertation. I would like to express my thanks to the people who have been very helpful to me during this time.

First and foremost I want to thank my supervisor Professor Shuxue Ding, whose expertise was invaluable in the formulation of the research topic and methodology in particular. During my doctoral study, Prof. Ding continuously supported my research with rigorous scientific attitude and enthusiasm at every stage. Under his guidance, I learned how to start a new research topic, solve the encountered problems step by step and publish the results. Thank you to my another supervisor Professor Chunhua Su, with whose support, I successfully completed part of the thesis. Prof. Su has not only expertise of research but also rich experience of life. Many things in his eyes are interesting. His positive attitude truly inspired my enthusiasm for life.

Besides my supervisors, I would like to thank the rest of my dissertation committee members (Professor Incheon Paik, and Professor Yoichi Tomioka, Professor Xiang Li) for their great support and invaluable advice. Thank you to Prof. Paik whose knowledge and curiosity in deep learning impelled me think more deeply about my own work. Thank to Prof. Tomioka for his patience when I went to his office for discussion and for his warm words of encouragements. I am also grateful to Prof. Li, who gave me many assistance and advises during my doctoral study. Specially, he always shares with me his tremendous experience in dealing with the encountered challenges and encourages me to believe myself.

In addition, I would like to thank my parents Linxia Wang and Guangdou Zhao, who are the root and staunch backing of all my achievements. They always give me valuable advises when I have any difficulties and support my decisions with trust. Thanks to my sister who has been taking care of my parents and supporting me in many ways since I left China. She is always the one to help me without any hesitation whenever I ask.

Thank to all the member in the Cognitive Science Laboratory for the discussions of problems, for the three years day-and-night working together, and for all the fun we have had outside the research. In particular, thank to Mr. Huakun Huang who always believes in me, keeps inspiring me and does everything he can to support me. Also,

---

thanks to all my friends in the University of Aizu, who have provided so many happy memories so that I can do my research as well as enjoy the life here.

# References

- [1] H. Charvátová, A. Procházka, S. Vaseghi, O. Vyšata, and M. Vališ, “Gps-based analysis of physical activities using positioning and heart rate cycling data,” *Signal, Image and Video Processing*, vol. 11, no. 2, pp. 251–258, 2017.
- [2] L. Shangguan, Z. Yang, A. X. Liu, Z. Zhou, and Y. Liu, “Stpp: Spatial-temporal phase profiling-based method for relative rfid tag localization,” *IEEE/ACM Transactions on Networking*, vol. 25, no. 1, pp. 596–609, 2017.
- [3] C.-W. Ou, C.-J. Chao, F.-S. Chang, S.-M. Wang, G.-X. Liu, M.-R. Wu, K.-Y. Cho, L.-T. Hwang, and Y.-Y. Huan, “A zigbee position technique for indoor localization based on proximity learning,” in *2017 IEEE International Conference on Mechatronics and Automation (ICMA)*. IEEE, 2017, pp. 875–880.
- [4] F. Lazzari, A. Buffi, P. Nepa, and S. Lazzari, “Numerical investigation of an uwb localization technique for unmanned aerial vehicles in outdoor scenarios,” *IEEE Sensors Journal*, vol. 17, no. 9, pp. 2896–2903, 2017.
- [5] H. Chen, Y. Zhang, W. Li, X. Tao, and P. Zhang, “Confi: Convolutional neural networks based indoor wi-fi localization using channel state information,” *IEEE Access*, vol. 5, pp. 18 066–18 074, 2017.
- [6] J. M. Pak, C. K. Ahn, P. Shi, Y. S. Shmaliy, and M. T. Lim, “Distributed hybrid particle/fir filtering for mitigating nlos effects in toa-based localization using wireless sensor networks,” *IEEE Transactions on Industrial Electronics*, vol. 64, no. 6, pp. 5182–5191, 2017.
- [7] Y. Wang and K. Ho, “Unified near-field and far-field localization for aoa and hybrid aoa-tdoa positionings,” *IEEE Transactions on Wireless Communications*, vol. 17, no. 2, pp. 1242–1254, 2018.
- [8] X. Zhu and Y. Feng, “Rssi-based algorithm for indoor localization,” *Communications and Network*, vol. 5, no. 02, p. 37, 2013.

- [9] K. Magowe, A. Giorgetti, S. Kandeepan, and X. Yu, "Accurate analysis of weighted centroid localization," *IEEE Transactions on Cognitive Communications and Networking*, vol. 5, no. 1, pp. 153–164, 2019.
- [10] O. Cheikhrouhou, G. M Bhatti, and R. Alroobaea, "A hybrid dv-hop algorithm using rssi for localization in large-scale wireless sensor networks," *Sensors*, vol. 18, no. 5, p. 1469, 2018.
- [11] S. P. Singh and S. Sharma, "Range free localization techniques in wireless sensor networks: A review," *Procedia Computer Science*, vol. 57, pp. 7–16, 2015.
- [12] B. Wang, Q. Chen, L. T. Yang, and H.-C. Chao, "Indoor smartphone localization via fingerprint crowdsourcing: Challenges and approaches," *IEEE Wireless Communications*, vol. 23, no. 3, pp. 82–89, 2016.
- [13] X. Shen, Z. Wang, P. Jiang, R. Lin, and Y. Sun, "Connectivity and rssi based localization scheme for wireless sensor networks," in *International Conference on Intelligent Computing*. Springer, 2005, pp. 578–587.
- [14] P. Bahl, V. N. Padmanabhan *et al.*, "Radar: An in-building rf-based user location and tracking system," in *IEEE infocom*, vol. 2, no. 2000. INSTITUTE OF ELECTRICAL ENGINEERS INC (IEEE), 2000, pp. 775–784.
- [15] G. Ding, J. Zhang, Z. Tan *et al.*, "Overview of received signal strength based fingerprinting localization in indoor wireless lan environments," in *2013 5th IEEE International Symposium on Microwave, Antenna, Propagation and EMC Technologies for Wireless Communications*. IEEE, 2013, pp. 160–164.
- [16] N. Patwari and J. Wilson, "Rf sensor networks for device-free localization: Measurements, models, and algorithms," *Proc. IEEE*, vol. 98, no. 11, pp. 1961–1973, 2010.
- [17] J. McCarthy, M. L. Minsky, N. Rochester, and C. E. Shannon, "A proposal for the dartmouth summer research project on artificial intelligence, august 31, 1955," *AI magazine*, vol. 27, no. 4, p. 12, 2006.
- [18] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural networks*, vol. 61, pp. 85–117, 2015.
- [19] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.
- [20] M. A. Carreira-Perpinan and G. E. Hinton, "On contrastive divergence learning," in *Aistats*, vol. 10. Citeseer, 2005, pp. 33–40.

- 
- [21] R. Salakhutdinov, J. B. Tenenbaum, and A. Torralba, “Learning with hierarchical-deep models,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1958–1971, 2013.
  - [22] Y. Tang, R. Salakhutdinov, and G. Hinton, “Robust boltzmann machines for recognition and denoising,” in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 2264–2271.
  - [23] R. Salakhutdinov, A. Mnih, and G. Hinton, “Restricted boltzmann machines for collaborative filtering,” in *Proceedings of the 24th international conference on Machine learning*. ACM, 2007, pp. 791–798.
  - [24] B. Abdollahi and O. Nasraoui, “Explainable restricted boltzmann machines for collaborative filtering,” *arXiv preprint arXiv:1606.07129*, 2016.
  - [25] Y. Bengio, A. Courville, and P. Vincent, “Representation learning: A review and new perspectives,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, 2013.
  - [26] S. Zhang, L. Yao, A. Sun, and Y. Tay, “Deep learning based recommender system: A survey and new perspectives,” *ACM Computing Surveys (CSUR)*, vol. 52, no. 1, p. 5, 2019.
  - [27] G. Carleo and M. Troyer, “Solving the quantum many-body problem with artificial neural networks,” *Science*, vol. 355, no. 6325, pp. 602–606, 2017.
  - [28] W. Xu, H. Sun, C. Deng, and Y. Tan, “Variational autoencoder for semi-supervised text classification,” in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.
  - [29] Y. Yang, Q. J. Wu, and Y. Wang, “Autoencoder with invertible functions for dimension reduction and image reconstruction,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 48, no. 7, pp. 1065–1079, 2018.
  - [30] W.-N. Hsu, Y. Zhang, and J. Glass, “Learning latent representations for speech generation and transformation,” *arXiv preprint arXiv:1704.04222*, 2017.
  - [31] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, 2012, pp. 1097–1105.
  - [32] M. Matsugu, K. Mori, Y. Mitari, and Y. Kaneda, “Subject independent facial expression recognition with robust face detection using a convolutional neural network,” *Neural Networks*, vol. 16, no. 5-6, pp. 555–559, 2003.



- [33] Y. LeCun *et al.*, “Lenet-5, convolutional neural networks,” URL: <http://yann.lecun.com/exdb/lenet>, vol. 20, 2015.
- [34] J. Yang, P. Ren, D. Zhang, D. Chen, F. Wen, H. Li, and G. Hua, “Neural aggregation network for video face recognition,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4362–4371.
- [35] H. Lee and H. Kwon, “Going deeper with contextual cnn for hyperspectral image classification,” *IEEE Transactions on Image Processing*, vol. 26, no. 10, pp. 4843–4855, 2017.
- [36] H.-C. Shin, H. R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura, and R. M. Summers, “Deep convolutional neural networks for computer-aided detection: Cnn architectures, dataset characteristics and transfer learning,” *IEEE transactions on medical imaging*, vol. 35, no. 5, pp. 1285–1298, 2016.
- [37] Y. Kim, “Convolutional neural networks for sentence classification,” *arXiv preprint arXiv:1408.5882*, 2014.
- [38] Q. Guan, X. Yin, X. Guo, and G. Wang, “A novel infrared motion sensing system for compressive classification of physical activity,” *Ieee Sens. J.*, vol. 16, pp. 2251–2259, 2016.
- [39] M. Scherhauf, M. Pichler, and A. Stelzer, “Uhf rfid localization based on phase evaluation of passive tag arrays,” *IEEE Trans. Instrum. Meas.*, vol. 64, pp. 913–922, 2015.
- [40] M. Bal, H. Xue, W. Shen, and H. Ghenniwa, “A 3-d indoor location tracking and visualization system based on wireless sensor networks,” in *Systems Man and Cybernetics (SMC), 2010 IEEE International Conference on*. IEEE, 2010, pp. 1584–1590.
- [41] J. Wilson and N. Patwari, “Radio tomographic imaging with wireless networks,” *IEEE Trans. Mob. Comput.*, vol. 9, pp. 621–632, 2010.
- [42] L. Chang, X. Chen, Y. Wang, D. Fang, J. Wang, T. Xing, and Z. Tang, “Fit-loc: Fine-grained and low-cost device-free localization for multiple targets over various areas,” *IEEE INFOCOM 2016 - 35th Annu. IEEE Int. Conf. Comput. Commun.*, pp. 1–9, 2016.
- [43] J. Wang, X. Zhang, Q. Gao, H. Yue, and H. Wang, “Device-free wireless localization and activity recognition: A deep learning approach,” *IEEE Trans. Veh. Technol.*, vol. 66, pp. 6258–6267, 2017.

- 
- [44] X. Wang, L. Gao, S. Mao, and S. Pandey, "Csi-based fingerprinting for indoor localization: A deep learning approach," *IEEE Trans. Veh. Technol.*, vol. 66, pp. 763–776, 2017.
  - [45] X. Zhang, J. Wang, Q. Gao, X. Ma, and H. Wang, "Device-free wireless localization and activity recognition with deep learning," *2016 IEEE Int. Conf. Pervasive Comput. Commun. Workshops (Percom Workshops)*, pp. 1–5, 2016.
  - [46] S. Kamada and T. Ichimura, "Knowledge extracted from recurrent deep belief network for real time deterministic control," in *Systems, Man, and Cybernetics (SMC), 2017 IEEE International Conference on.* IEEE, 2017, pp. 825–830.
  - [47] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313 5786, pp. 504–7, 2006.
  - [48] T. K. Reddy and L. Behera, "Online eye state recognition from eeg data using deep architectures," in *Systems, Man, and Cybernetics (SMC), 2016 IEEE International Conference on.* IEEE, 2016, pp. 000 712–000 717.
  - [49] G. E. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. K. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *Ieee Signal Process. Mag.*, vol. 29, pp. 82–97, 2012.
  - [50] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *NIPS*, 2006.
  - [51] Y. Bengio, "Learning deep architectures for ai," *Found. Trends Mach. Learn.*, vol. 2, pp. 1–127, 2009.
  - [52] Y. Bengio, A. C. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, pp. 1798–1828, 2013.
  - [53] N. Lu, T. Li, X. Ren, and H. Miao, "A deep learning scheme for motor imagery classification based on restricted boltzmann machines," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 25, pp. 566–576, 2017.
  - [54] N. Wang, J. Melchior, and L. Wiskott, "Gaussian-binary restricted boltzmann machines for modeling natural image statistics," in *PloS one*, 2017.
  - [55] D. Koller and N. Friedman, "Probabilistic graphical models - principles and techniques," 2009.

- [56] G. E. Hinton, “Training products of experts by minimizing contrastive divergence,” *Neural computation*, vol. 14 8, pp. 1771–800, 2002.
- [57] Y. Bengio and O. Delalleau, “Justifying and generalizing contrastive divergence,” *Neural computation*, vol. 21 6, pp. 1601–21, 2009.
- [58] G. E. Hinton, “Learning multiple layers of representation,” *Trends Cogn. Sci.*, vol. 11 10, pp. 428–34, 2007.
- [59] M. Á. Carreira-Perpiñán and G. E. Hinton, “On contrastive divergence learning,” in *AISTATS*, 2005.
- [60] D. Heckerman and C. Meek, “Models and selection criteria for regression and classification,” in *UAI*, 1997.
- [61] D. Wang, X. Guo, and Y. Zou, “Accurate and robust device-free localization approach via sparse representation in presence of noise and outliers,” in *Digital Signal Processing (DSP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 199–203.
- [62] W. Meng, “Intrusion detection in the era of iot: Building trust via traffic filtering and sampling,” *Computer*, vol. 51, no. 7, pp. 36–43, 2018.
- [63] C. Kolias, G. Kambourakis, A. Stavrou, and S. Gritzalis, “Intrusion detection in 802.11 networks: Empirical evaluation of threats and a public dataset,” *Ieee Commun. Surv. & Tutor.*, vol. 18, no. 1, pp. 184–208, 2016.
- [64] W. Meng, E. W. Tischhauser, Q. Wang, Y. Wang, and J. Han, “When intrusion detection meets blockchain technology: A review,” *Ieee Access*, vol. 6, pp. 10 179–10 188, 2018.
- [65] C. Kolias, G. Kambourakis, and M. Maragoudakis, “Swarm intelligence in intrusion detection: A survey,” *Comput. & Secur.*, vol. 30, no. 8, pp. 625–642, 2011.
- [66] Z. Li, Z. Yang, and S. Xie, “Computing resource trading for edge-cloud-assisted internet of things,” *IEEE Trans. Ind. Inform.*, 2019.
- [67] L. Zhao, H. Huang, X. Li, S. Ding, H. Zhao, and Z. Han, “An accurate and robust approach of device-free localization with convolutional autoencoder,” *IEEE Internet Things J.*, 2019.
- [68] S. Kuutti, S. Fallah, K. Katsaros, M. Dianati, F. Mccullough, and A. Mouzakitis, “A survey of the state-of-the-art localization techniques and their potentials for

- autonomous vehicle applications,” *IEEE Internet Things J.*, vol. 5, no. 2, pp. 829–846, 2018.
- [69] B. Liu, W. Zhou, T. Zhu, L. Gao, T. H. Luan, and H. Zhou, “Silence is golden: Enhancing privacy of location-based services by content broadcasting and active caching in wireless vehicular networks,” *IEEE Trans. Veh. Technol.*, vol. 65, no. 12, pp. 9942–9953, 2016.
- [70] M. Scherhäufl, M. Pichler, and A. Stelzer, “Uhf rfid localization based on phase evaluation of passive tag arrays,” *IEEE Trans. Instrum. Meas.*, vol. 64, no. 4, pp. 913–922, 2015.
- [71] H. Aly, A. Basalamah, and M. Youssef, “Accurate and energy-efficient gps-less outdoor localization,” *ACM Trans. Spat. Algorithms Syst. (TSAS)*, vol. 3, no. 2, p. 4, 2017.
- [72] S. Kianoush, S. Savazzi, F. Vicentini, V. Rampa, and M. Giussani, “Device-free rf human body fall detection and localization in industrial workplaces,” *IEEE Internet Things J.*, vol. 4, no. 2, pp. 351–362, 2017.
- [73] H. Huang, S. Guo, P. Li, and T. Miyazaki, “Stochastic analysis on the deactivation-controlled epidemic routing in dtns with multiple sinks,” *Ad Hoc & Sens. Wirel. Netw.*, vol. 38, no. 1-4, pp. 143–167, 2017.
- [74] F. Zafari, A. Gkelias, and K. Leung, “A survey of indoor localization systems and technologies,” *Arxiv Prepr. Arxiv:1709.01015*, 2017.
- [75] K. Wu, J. Xiao, Y. Yi, M. Gao, and L. M. Ni, “Fila: Fine-grained indoor localization,” in *INFOCOM, 2012 Proceedings IEEE*. IEEE, 2012, pp. 2210–2218.
- [76] H. Huang, H. Zhao, X. Li, S. Ding, L. Zhao, and Z. Li, “An accurate and efficient device-free localization approach based on sparse coding in subspace,” *IEEE Access*, vol. 6, no. 1, pp. 61 782–61 799, 2018.
- [77] W. Ruan, Q. Z. Sheng, L. Yao, X. Li, N. J. Falkner, and L. Yang, “Device-free human localization and tracking with uhf passive rfid tags: A data-driven approach,” *J. Netw. Comput. Appl.*, vol. 104, pp. 78–96, 2018.
- [78] Y. Sun, X. Zhang, X. Wang, and X. Zhang, “Device-free wireless localization using artificial neural networks in wireless sensor networks,” *Wirel. Commun. Mob. Comput.*, vol. 2018, 2018.

- [79] L. Zhao, H. Huang, S. Ding, and X. Li, "An accurate and efficient device-free localization approach based on gaussian bernoulli restricted boltzmann machine," in *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2018, pp. 2323–2328.
- [80] M. Woźniak and D. Połap, "Object detection and recognition via clustered features," *Neurocomputing*, vol. 320, pp. 76–84, 2018.
- [81] D. Połap, A. Winnicka, K. Serwata, K. Kesik, and M. Woźniak, "An intelligent system for monitoring skin diseases," *Sensors*, vol. 18, no. 8, p. 2552, 2018.
- [82] D. Połap, M. Woźniak, W. Wei, and R. Damaševičius, "Multi-threaded learning control mechanism for neural networks," *Future Gener. Comput. Syst.*, vol. 87, pp. 16–34, 2018.
- [83] M. Youssef, M. Mah, and A. Agrawala, "Challenges: Device-free passive localization for wireless environments," in *Proceedings of the 13th annual ACM international conference on Mobile computing and networking*. ACM, 2007, pp. 222–229.
- [84] M. Moussa and M. Youssef, "Smart cevides for smart environments: Device-free passive detection in real environments," in *Pervasive Computing and Communications, 2009. PerCom 2009. IEEE International Conference on*. IEEE, 2009, pp. 1–6.
- [85] J. Wilson and N. Patwari, "Radio tomographic imaging with wireless networks," *IEEE Trans. Mob. Comput.*, vol. 9, no. 5, pp. 621–632, 2010.
- [86] D. Zhang, Y. Liu, X. Guo, and L. M. Ni, "Rass: A real-time, accurate, and scalable system for tracking transceiver-free objects," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 5, pp. 996–1008, 2013.
- [87] I. Sabek, M. Youssef, and A. V. Vasilakos, "Ace: An accurate and efficient multi-entity device-free wlan localization system," *IEEE Trans. Mob. Comput.*, vol. 14, no. 2, pp. 261–273, 2015.
- [88] Y. Guo, K. Huang, N. Jiang, X. Guo, Y. Li, and G. Wang, "An exponential-rayleigh model for rss-based device-free localization and tracking," *IEEE Trans. Mob. Comput.*, vol. 14, no. 3, pp. 484–494, 2015.
- [89] M. Seinfeldin, A. Saeed, A. E. Kosba, A. El-Keyi, and M. Youssef, "Nuzzer: A large-scale device-free passive localization system for wireless environments," *IEEE Trans. Mob. Comput.*, vol. 12, no. 7, pp. 1321–1334, 2013.

- 
- [90] Q. Gao, J. Wang, X. Ma, X. Feng, and H. Wang, "Csi-based device-free wireless localization and activity recognition using radio image features," *IEEE Trans. Veh. Technol.*, vol. 66, no. 11, pp. 10 346–10 356, 2017.
  - [91] J. Liang, D. Wang, L. Su, B. Chen, H. Chen, and H.-C. So, "Robust mimo radar target localization via nonconvex optimization," *Signal Processing*, vol. 122, pp. 33–38, 2016.
  - [92] J. Hong and T. Ohtsuki, "Signal eigenvector-based device-free passive localization using array sensor," *IEEE Trans. Veh. Technol.*, vol. 64, no. 4, pp. 1354–1363, 2015.
  - [93] R. Zhou, X. Lu, P. Zhao, and J. Chen, "Device-free presence detection and localization with svm and csi fingerprinting," *Ieee Sens. J.*, vol. 17, no. 23, pp. 7990–7999, 2017.
  - [94] D. A. Tran, S. Gong, and Q. Vo, "Geometric-based knn localization using sensor dissimilarity information," in *Personal, Indoor, and Mobile Radio Communications (PIMRC), 2017 IEEE 28th Annual International Symposium on*. IEEE, 2017, pp. 1–6.
  - [95] K. Zheng, H. Wang, H. Li, W. Xiang, L. Lei, J. Qiao, and X. S. Shen, "Energy-efficient localization and tracking of mobile devices in wireless sensor networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 3, pp. 2714–2726, 2017.
  - [96] R. Zhou, M. Hao, X. Lu, M. Tang, and Y. Fu, "Device-free localization based on csi fingerprints and deep neural networks," in *2018 15th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. IEEE, 2018, pp. 1–9.
  - [97] H. Huang and S. Lin, "Widet: Wi-fi based device-free passive person detection with deep convolutional neural networks," in *Proceedings of the 21st ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*. ACM, 2018, pp. 53–60.
  - [98] Y. Chen, H. Jiang, C. Li, X. Jia, and P. Ghamisi, "Deep feature extraction and classification of hyperspectral images based on convolutional neural networks," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 10, pp. 6232–6251, 2016.
  - [99] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.
-

- [100] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, “Striving for simplicity: The all convolutional net,” *Arxiv Prepr. Arxiv:1412.6806*, 2014.
- [101] M. S. Seyfioğlu, A. M. Özbayğlı, and S. Z. Gurbuz, “Deep convolutional autoencoder for radar-based classification of similar aided and unaided human activities,” *IEEE Trans. Aerosp. Electron. Syst.*, vol. 54, no. 4, pp. 1709–1723, 2018.
- [102] L. Catarinucci, D. De Donno, L. Mainetti, L. Palano, L. Patrono, M. L. Stefanizzi, and L. Tarricone, “An iot-aware architecture for smart healthcare systems,” *IEEE Internet Things J.*, vol. 2, no. 6, pp. 515–526, 2015.
- [103] R. Carotenuto, M. Merenda, D. Iero, and F. G. Della Corte, “Ranging rfid tags with ultrasound,” *Ieee Sens. J.*, vol. 18, no. 7, pp. 2967–2975, 2018.
- [104] M. Gochoo, T. H. Tan, S. H. Liu, F. R. Jean, F. Alnajjar, and S.-C. Huang, “Unobtrusive activity recognition of elderly people living alone using anonymous binary sensors and dcnn,” *IEEE Journal of Biomedical and Health Informatics*. [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/8355255/>
- [105] M. Gochoo, T. H. Tan, V. Velusamy, S. H. Liu, D. Bayanduuren, and S. C. Huang, “Device-free non-privacy invasive classification of elderly travel patterns in a smart house using pir sensors and dcnn,” *IEEE Sensors Journal*, vol. 18, no. 1, pp. 390–400, 2018.
- [106] O. Kaltiokallio, H. Yigitler, and R. Jäntti, “A three-state received signal strength model for device-free localization,” *IEEE Trans. Veh. Technol.*, vol. 66, no. 10, pp. 9226–9240, 2017.
- [107] X. Wang, L. Gao, and S. Mao, “Csi phase fingerprinting for indoor localization with a deep learning approach,” *IEEE Internet Things J.*, vol. 3, no. 6, pp. 1113–1123, 2016.
- [108] M. Mohammadi, A. Al-Fuqaha, M. Guizani, and J.-S. Oh, “Semisupervised deep reinforcement learning in support of iot and smart city services,” *IEEE Internet Things J.*, vol. 5, no. 2, pp. 624–635, 2018.
- [109] X. Wang, X. Wang, and S. Mao, “Deep convolutional neural networks for indoor localization with csi images,” *IEEE Trans. Netw. Sci. Eng.*, 2018.
- [110] 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.

- 
- [111] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé, “Mobile encrypted traffic classification using deep learning,” in *2018 Network Traffic Measurement and Analysis Conference (TMA)*. IEEE, 2018, pp. 1–8.
- [112] W. Wang, M. Zhu, X. Zeng, X. Ye, and Y. Sheng, “Malware traffic classification using convolutional neural network for representation learning,” in *Information Networking (ICOIN), 2017 International Conference on*. IEEE, 2017, pp. 712–717.
- [113] E. Maggiori, Y. Tarabalka, G. Charpiat, and P. Alliez, “Convolutional neural networks for large-scale remote-sensing image classification,” *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 2, pp. 645–657, 2017.
- [114] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [115] J. Wang, Q. Gao, P. Cheng, Y. Yu, K. Xin, and H. Wang, “Lightweight robust device-free localization in wireless networks,” *IEEE Trans. Ind. Electron.*, vol. 61, no. 10, pp. 5681–5689, 2014.
- [116] D. Ciuonzo and P. S. Rossi, “Distributed detection of a non-cooperative target via generalized locally-optimum approaches,” *Information Fusion*, vol. 36, pp. 261–274, 2017.
- [117] D. Ciuonzo, P. S. Rossi, and P. Willett, “Generalized rao test for decentralized detection of an uncooperative target,” *IEEE Signal Processing Letters*, vol. 24, no. 5, pp. 678–682, 2017.
- [118] D. Ciuonzo and P. S. Rossi, “Quantizer design for generalized locally optimum detectors in wireless sensor networks,” *IEEE Wireless Communications Letters*, vol. 7, no. 2, pp. 162–165, 2018.
- [119] J. Xiao, K. Wu, Y. Yi, L. Wang, and L. M. Ni, “Pilot: Passive device-free indoor localization using channel state information,” in *Distributed computing systems (ICDCS), 2013 IEEE 33rd international conference on*. IEEE, 2013, pp. 236–245.
- [120] K. Wu, J. Xiao, Y. Yi, D. Chen, X. Luo, and L. M. Ni, “Csi-based indoor localization,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 7, pp. 1300–1309, 2013.



- [121] C. Alippi, M. Bocca, G. Boracchi, N. Patwari, and M. Roveri, "Rti goes wild: Radio tomographic imaging for outdoor people detection and localization," *IEEE Trans. Mob. Comput.*, vol. 15, no. 10, pp. 2585–2598, 2016.
- [122] D. Zhang, J. Ma, Q. Chen, and L. M. Ni, "An rf-based system for tracking transceiver-free objects," in *Pervasive Computing and Communications, 2007. PerCom'07. Fifth Annual IEEE International Conference on*. IEEE, 2007, pp. 135–144.
- [123] X. Li, S. Ding, Z. Li, and B. Tan, "Device-free localization via dictionary learning with difference of convex programming," *Ieee Sens. J.*, vol. 17, no. 17, pp. 5599–5608, 2017.
- [124] R. Zhang, W.-D. Zhong, K. Qian, S. Zhang, and P. Du, "A reversed visible light multitarget localization system via sparse matrix reconstruction," *IEEE Internet Things J.*, vol. 5, no. 5, pp. 4223–4230, Oct 2018.
- [125] S. Yiu, M. Dashti, H. Claussen, and F. Perez-Cruz, "Wireless rssi fingerprinting localization," *Signal Processing*, vol. 131, pp. 235–244, 2017.
- [126] L. Zhao, H. Huang, S. Ding, and X. Li, "An accurate and efficient device-free localization approach based on gaussian bernoulli restricted boltzmann machine," in *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Oct 2018, pp. 2319–2324.
- [127] M. Bocca, O. Kaltiokallio, N. Patwari, and S. Venkatasubramanian, "Multiple target tracking with rf sensor networks," *IEEE Trans. Mob. Comput.*, vol. 13, no. 8, pp. 1787–1800, 2014.
- [128] J. Wang, Q. Gao, H. Wang, P. Cheng, and K. Xin, "Device-free localization with multidimensional wireless link information," *IEEE Trans. Veh. Technol.*, vol. 64, no. 1, pp. 356–366, 2015.
- [129] B. Mager, P. Lundrigan, and N. Patwari, "Fingerprint-based device-free localization performance in changing environments," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 11, pp. 2429–2438, 2015.
- [130] C. Cai, L. Deng, M. Zheng, and S. Li, "Pilc: Passive indoor localization based on convolutional neural networks," in *2018 Ubiquitous Positioning, Indoor Navigation and Location-Based Services (UPINLBS)*. IEEE, 2018, pp. 1–6.
- [131] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *Arxiv Prepr. Arxiv:1409.1556*, 2014.

- 
- [132] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber, “Stacked convolutional auto-encoders for hierarchical feature extraction,” in *International Conference on Artificial Neural Networks*. Springer, 2011, pp. 52–59.
  - [133] K. He and J. Sun, “Convolutional neural networks at constrained time cost,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 5353–5360.
  - [134] J. Wu, C. Leng, Y. Wang, Q. Hu, and J. Cheng, “Quantized convolutional neural networks for mobile devices,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4820–4828.
  - [135] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
  - [136] Z. Wen, J. Shi, Q. Li, B. He, and J. Chen, “ThunderSVM: A fast SVM library on GPUs and CPUs,” *Journal of Machine Learning Research*, vol. 19, pp. 1–5, 2018.
  - [137] S. Kiranyaz, T. Ince, and M. Gabbouj, “Real-time patient-specific ecg classification by 1-d convolutional neural networks,” *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 3, pp. 664–675, 2016.
  - [138] M. Claesen, F. De Smet, J. A. Suykens, and B. De Moor, “Fast prediction with svm models containing rbf kernels,” *arXiv preprint arXiv:1403.0736*, 2014.
  - [139] J. Liu, N. An, M. T. Hassan, M. Peng, Z. Cui, and S. Zhao, “Redundancy reduction for indoor device-free localization,” *Pers. Ubiquitous Comput.*, vol. 21, no. 1, pp. 5–15, 2017.