DSLN: Securing Internet of Things Through RF Fingerprint Recognition in Low-SNR Settings

Weiwei Wu[1], Su Hu[1*], Di Lin[1], Zilong Liu[2]
[1] University of Electronic Science and Technology of China
[2] School of Computer Science and Electronics Engineering, University of Essex, United Kingdom
Correspondence author: Su Hu, Email: husu@uestc.edu.cn

***Abstract*—The explosive growth of Internet of things (IoT) has mandated the security of data access. Although authentication methods can enhance network security, their vulnerability to malicious attacks may be a barrier for the wide deployments in IoT scenarios. To address the security issue, we advocate the use of physical layer security through radio-frequency (RF) fingerprint recognition. Observing that most RF fingerprint recognition methods show a degradation of performance under low signal-to-noise ratio (SNR) environments, we present a dynamic shrinkage learning network (DSLN) to enhance security for IoT applications, particularly in the setting of low SNR. We design a novel dynamic shrinkage threshold for improving the accuracy of recognition under low-SNR environments. Additionally, we design an identity shortcut for reducing the running time of RF fingerprint recognition. In comparison with convolutional neural network (CNN), recurrent neural network (RNN) and a hybrid CNN+RNN network (CRNN), our proposed DSLN yields accuracy improvements of up to 20%. Moreover, DSLN can reduce running time by up to 60%, indicating its great potential to a real-time IoT system, e.g., an intelligent automotive system.**

***Index Terms*—Internet of things, Network security, RF fingerprint recognition, Deep learning, Low SNR.**

I. INTRODUCTION

As a promising technology, Internet of things (IoT) is restructuring many sectors, including transportation, healthcare, business, etc [1]. A typical IoT system comprises multiple interconnected devices for the exchange of massive data via wireless communication infrastructures. It has been predicted that more than 50 billion IoT devices will be connected through Internet by 2025 [2]. The explosive growth of wireless devices has imposed a strong need for IoT networks to collect and process an unprecedented amount of data for various purposes such as monitoring and decision making, thereby facilitating innovations in the intelligent industry [3].

However, the widespread IoT applications also raise severe security and trustworthiness concerns [4]. A cluster of servers at the cloud is employed for data storage and user authentication in a typical IoT system. Due to the private nature of data at the servers, it is important to identify the authentication of mobile devices for the control of user access. In the setting of IoT, most authentication methods by using IP or MAC addresses (e.g., digital signature) may be inapplicable since these methods are vulnerable to malicious attacks, e.g., duplicating and changing IP addresses [5]. Such a security problem has become a bottleneck that restricts the further applications of IoT.

Against this background, deploying RF fingerprint recognition for IoT user authentication has received increasing research attention [6]–[13]. RF fingerprints are designed to reflect the unique features of an individual device. These features attribute to manufacturing tolerances, component aging, and changes in working environments. The typical examples of the RF fingerprints include frequency deviation of oscillators, phase noise, non-linear distortion of power amplifiers, filter distortion, I/Q imbalance, phase imbalance, frequency error, etc. There have been methods of extracting fingerprint features and identifying the devices by comparing the similarities of features [6]–[9]. These methods need a manual selection of fingerprint features, which is highly dependent on domain knowledge. To remove the process of manual feature selection, an alternative direction is by intelligently learning fingerprint features with deep-learning algorithms [10]–[13]. The idea is to apply a few typical deep learning algorithms, e.g., convolutional neural network (CNN) and recurrent neural network (RNN), to recognize mobile devices to prevent impersonation. As shown in Fig.1, the RF signals emitted from different devices may have similar waveforms in the time domain, and therefore, it is challenging to recognize devices through their signals both in time and in frequency domains. However, by extracting the RF fingerprint features from these signals, e.g., the high-order moments, box dimension, fractal dimension, one can identify the devices by analyzing their features [7].

Existing learning algorithms for RF fingerprint recognition in an IoT system may not work well in low-SNR settings [14]. For example, an intelligent automotive IoT system uses a millimeter-wave radar for environment perception, information sharing, and decision making [15], where the millimeter-wave signals may be difficult to pass through buildings and tend to have a high noise level. In this case, a low SNR needs to be dealt with in order to cover the subtle differences of RF fingerprints between mobile devices. At an SNR of 0 dB, convolutional neural network (CNN) and recurrent neural network (RNN) may suffer from accuracy degradation of 30% [16].

Despite a rich body of literature on RF fingerprint

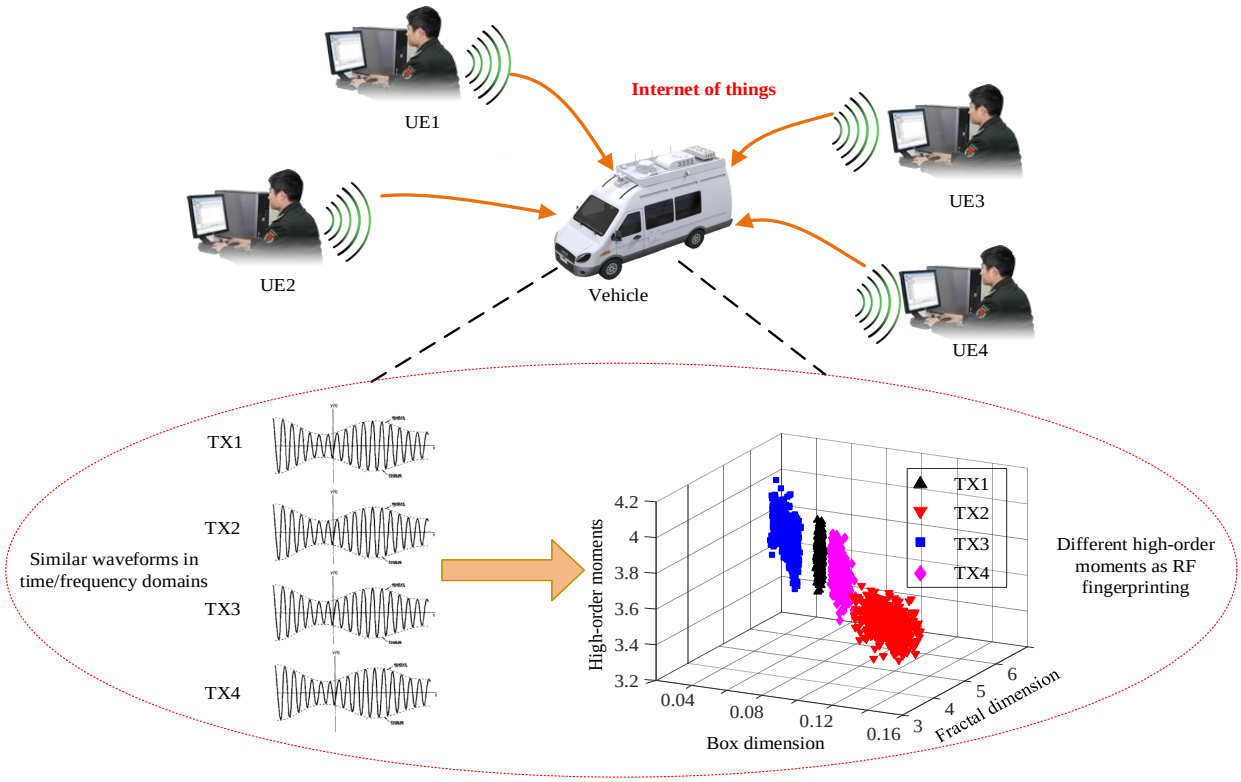


Fig. 1. RF fingerprint recognition for device identification in IoT systems.

recognition, most existing works are effective for modest to high SNRs only [6]–[13]. Compared with low SNR settings, it is easier to recognize devices' RF fingerprints in high SNR settings since a low level of noise has little impact on the recognition results. However, when the SNR is low, without an effective denoising method, a high level of noise is most likely to mislead a model and generate an incorrect recognition result. The effective methods for high SNR can theoretically be used in low SNR, but their performance would dramatically degrade in practice. Thus, the key of achieving a high recognition accuracy at low SNRs is to design a denoising method in the model when recognizing a device's RF fingerprint. Towards this end, we develop a dynamic shrinkage learning network (DSLN) to train RF fingerprint features of mobile devices. The main contributions of this paper are summarized as follows:

(1) To the best of our knowledge, it is the first work which studies on RF fingerprint recognition under a low-SNR environment for IoT systems. The research on this topic has the great potential of enhancing security for widespread IoT applications.

(2) To improve the accuracy of RF fingerprint recognition at low SNRs, we design a dynamic shrinkage threshold in the DSLN algorithm. The dynamic shrinkage threshold can preserve more signal features than the existing learning algorithms, thereby achieving higher accuracy of recognizing RF fingerprints.

(3) To reduce the running time of RF fingerprint recognition, we present an identity shortcut in the DSLN algorithm. An identity shortcut is used to skip one or more convolutional layers, thus directly connecting a particular layer to a latter layer with an identity mapping. Critical features can be preserved even though particular layers are skipped [17]. Due to identity mappings, the proposed network can facilitate training process, thus yielding a reduced number of iterations.

(4) We also develop a hardware prototype of DSLN for RF fingerprint recognition. Extensive experiments illustrate that our proposed DSLN can achieve an improvement of up to 20% in recognition accuracy than CNN and RNN. Moreover, the performance of running time in DSLN also gets significantly reduced by $60\% - 80\%$ in comparison to that of CNN and RNN.

The rest of this paper is organized as follows. In Section II, we review the progress of RF fingerprint recognition and its related learning methods. In Section III, we demonstrate the internal hardware features which lead to the external differences of signals. Then, the proposed DSLN algorithm is described in Section IV, including its architecture and its training pipeline. Section V illustrates the experiment results. The conclusion and future work is discussed in Section VI.

II. RELATED WORK

The development of RF fingerprint recognition is primarily composed of two research streams: One is knowledge-based RF fingerprint recognition, whereas the other is data-based RF fingerprint recognition. In the first stream, one needs

to carefully select appropriate fingerprint features and then employ the feature difference to identify mobile devices. Specifically, machine-learning algorithms are used to recognize devices based on a model with the features as system inputs. The second stream is based on deep learning algorithms which can automatically train the raw data of signals to recognize mobile devices.

A. Knowledge-based RF fingerprint recognition

RF fingerprint recognition can be carried out by the measurement of similarities which compares the observed RF fingerprint with reference RF fingerprint in a database. An RF fingerprint method was used to collect and analyze the probe request frames of devices with Bayesian algorithms in order to recognize IEEE 802.11 mobile devices [6]. However, the recognition of RF fingerprint might be fragile to the variation of wireless channels [7]. Thus, a wavelet analysis algorithm was proposed to identify the mobile devices with the inter-arrival time of TCP/UDP packets.

Alternatively, one can classify different mobile devices by analyzing I/Q imbalance, phase imbalance, frequency deviation of oscillators, phase noise, non-linear distortion of power amplifiers, filter distortion, etc [18]. [8] proposed a method of extracting spectral features and analyzing preamble within a packet with k-nearest neighbor (kNN) algorithms. To improve the accuracy of k-nearest neighbor (kNN) algorithms, a combination of support vector machine (SVM) and kNN algorithms was presented to classify 802.11 devices by analyzing the RF fingerprint for an accuracy of 95%.

Additionally, a method of artificial neural networks with an accuracy of 99% was proposed to identify mobile devices by analyzing their hardware compositions [9]. Although the above-mentioned works can achieve high accuracy at high SNRs, these methods have their limitations. Most work only provided insights into the performance of learning models in high SNR settings (e.g., 15-30 dB), and did not consider low SNR scenarios.

B. Data-based RF fingerprint recognition

Recently, deep learning has been applied to tackle RF fingerprint recognition problems. [11] used an ANN model to identify IEEE 802.11, IEEE 802.15.4, and IEEE 802.15.1 devices, and achieved a higher recognition accuracy than machine-learning models. But they did not consider the problem of multi-path interferences. To improve the accuracy of device identification in multi-path channels, [10] preprocessed original data by using the Multiple Signal Classification algorithm and then used a deep-learning-based model for RFID-based activity identification in a multi-path indoor environment. It was shown that their model demonstrated high classification accuracy in a rich multipath environment. In a cognitive radio network, the impersonation of devices could be recognized by detecting their physical-layer features [12] for increased the utilization efficiency of the network. A few more methods were proposed to reduce the complexity of identifying mobile devices. An acceleration method was proposed to analyze the features between similar devices with a CNN classifier [12]. The key idea of their paper was to directly train the raw I/Q samples to distinguish mobile devices. A revised network with long short term memory (LSTM) in [13] was proposed, and this model could reduce running time compared to a CNN model. The revised network needs fewer parameters than the CNN model, and the forward and back propagation iterations of CNN models are fairly time-consuming in nature. Moreover, the revised model could improve classification accuracy in comparison with the CNN model. Experimental results showed that the convolutions fail to capture the highly discriminative features from original data. Additionally, extensive experimental data from multiple sources provided a comparison between CNN, RNN, and hybrid CNN & RNN. Their results showed that hybrid CNN & RNN model outperforms other deep learning methods to detect false-reading attacks [19].

C. Comparison of knowledge-based and data-based RF fingerprint recognition

Heavily relying on human experiences of experts, traditional machine learning methods mostly require a time-consuming process of feature selection. By solving the above-mentioned issue, deep learning methods can avoid the time-consuming process of feature selection. Instead, one only needs to input raw data (i.e. I/Q samples) to the learning models. The critical task of a deep learning algorithm is to construct a multi-layer network and optimize the relevant parameters. Particularly, deep learning algorithms could be a comparatively better option than machine learning ones in the case of insufficient prior experiences. For example, it is fairly difficult to identify the primary factors or features that impose significant impacts on the final results. In this case, it is appropriate to build learning models with deep learning algorithms.

In this paper, we use deep learning algorithms to detect mobile devices. The signals from transmitters are represented by raw I/Q data. On the other hand, if we choose to use traditional machine learning algorithms, there is a long process of feature selection to construct a series of mathematical and statistical indicators, including frequency offset, spectral characteristics, time domain envelope, wavelet coefficients, etc. This process is fairly time consuming. Moreover, it might lose a large amount of information.

III. Analysis on RF fingerprint features

The RF fingerprint features are inherent to the hardware of each mobile device. The identification process is carried out at the receiving end by analyzing a number of hardware mismatches such as frequency offset of the oscillator, nonlinear distortion of the power amplifier, and the distortion of filters. In this section, we introduce and study the mechanism of RF fingerprint generated by a modulator. With different modulation schemes, we can collect the subtle differences in modulator hardware. For quadrature modulators, non-linear distortion is often introduced by I/Q channel gain imbalance, phase imbalance, delay imbalance and carrier leakage.

A. Signal difference caused by I/Q imbalance

With quadrature modulation, we can denote the ideal signal $X(t)$ at the transmitter as

$$X(t) = X_I(t)\cos(w_0 t) + X_Q(t)\sin(w_0 t). \tag{1}$$

Due to the hardware differences, the actual signal $\hat{X}(t)$ may have subtle differences in amplitude (Δ) and phase (ϵ) in comparison with the ideal signal. Hence, the former can be denoted as

$$\hat{X}(t) = X_I(t)(1-\Delta)\cos(w_0 t - \epsilon) + X_Q(t)(1+\Delta)\sin(w_0 t + \epsilon). \tag{2}$$

The difference between the actual transmit signal and the ideal transmit signal $\bar{X}(t)$ can be expressed as

$$\bar{X}(t) = \bar{X}_\epsilon(t) + \bar{X}_{\epsilon,\Delta}(t). \tag{3}$$

In (3), $\bar{X}_\epsilon(t)$ represents the signal difference only caused by phase difference (ϵ) as

$$\begin{aligned}\bar{X}_\epsilon(t) = 2\sin\tfrac{\epsilon}{2}[&\cos\tfrac{\epsilon}{2}(X_Q(t)\cos(w_0 t) \\ &+ X_I(t)\sin(w_0 t)) \\ &- \sin\tfrac{\epsilon}{2}(X_Q(t)\sin(w_0 t) \\ &+ X_I(t)\cos(w_0 t))].\end{aligned} \tag{4}$$

Moreover, $\bar{X}_{\epsilon,\Delta}(t)$ represents the signal difference caused by both amplitude difference and phase difference (Δ and ϵ) as

$$\begin{aligned}\bar{X}_{\epsilon,\Delta}(t) = \Delta[&\cos\epsilon(X_Q(t)\sin(w_0 t) \\ &- X_I(t)\cos(w_0 t)) \\ &+ \sin\epsilon(X_Q(t)\cos(w_0 t) \\ &+ X_I(t)\sin(w_0 t))].\end{aligned} \tag{5}$$

When ϵ is close to 0, given $\phi = \arctan\frac{X_I}{X_Q}$, $\bar{X}(t)$ can be simplified as

$$\begin{aligned}\bar{X}(t)|_{\epsilon=0} &= \Delta(X_Q(t)\sin(w_0 t) - X_I(t)\cos(w_0 t)) \\ &= \Delta\sqrt{X_Q^2(t) + X_I^2(t)}\sin(w_0 t - \phi).\end{aligned} \tag{6}$$

When Δ is close to 0, given $\phi = \arctan\frac{X_I}{X_Q}$, $\bar{X}(t)$ can be simplified as

$$\begin{aligned}\bar{X}(t)|_{\Delta=0} = \sqrt{X_Q^2(t) + X_I^2(t)}[&\sin\epsilon\cos(w_0 t - \phi) \\ &+ (\cos\epsilon - 1)\sin(w_0 t + \phi)].\end{aligned} \tag{7}$$

In the following, we will use the subtle difference between the ideal signal and the actual signal (shown in Fig. 2) as the internal features of mobile devices to identify them. We will use the amplitude difference and phase difference in (3), (6), (7) for simulation, which will be shown in Section V.

B. Inherent nature of RF fingerprint features

RF fingerprint represents a collection of inherent features extracted from wireless devices, and usually has the following characteristics:

(1) Versatility: Even in the same batch, each device has its own imperfections deviating slightly from nominal specifications, and thus each device has its own features in hardware due to the slight imperfection [18], [20]. RF fingerprints are designed to reflect the unique features of an individual device. The features attribute to manufacturing tolerances, component aging, and changes in working environments. The typical examples of features include frequency deviation of oscillators, phase noise, non-linear distortion of power amplifiers, filter distortion, I/Q imbalance, phase imbalance, frequency error, etc. We can identify each of the wireless devices by properly selecting these fingerprint features.

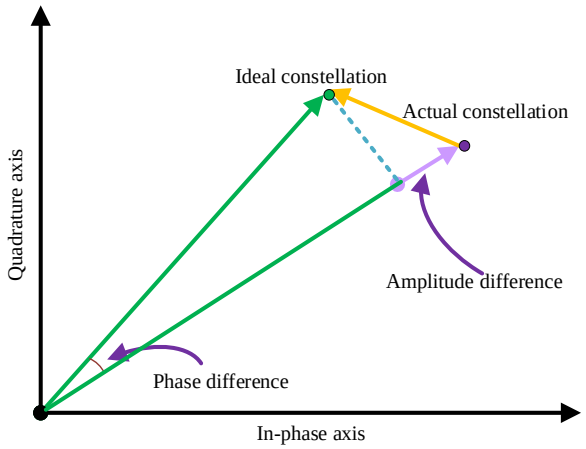


Fig. 2. Illustration of the subtle difference between devices reflected in the constellation.

(2) Short-term invariance: Unlike human beings who can remain their fingerprint features invariant throughout their lives, the components of wireless devices will inevitably face aging, and long-term use will lead the actual fingerprint features to be different from those registered in the fingerprint feature database. However, the aging of components takes a long time, so its impact on RF fingerprint features during a short period is relatively low [21]. The study in [21] shows that the RF fingerprint features extracted by a wireless network can remain unchanged within 5 months, and the device recognition rate is quite stable.

(3) Uniqueness: As human have different fingerprint features, the RF fingerprints of wireless devices are also unique due to the uniqueness of their hardware. With the fast development of integrated circuits, the degree of integration between electronic components is dramatically rising. At the same time, the high level of integration leads to a more subtle difference between devices. Thus, how to identify the wireless devices with their unique fingerprint features is a primary issue.

(4) Robustness: The robustness of an RF fingerprint feature refers to the level of stability in the RF fingerprint of a wireless device when its communication environment changes. The authors in [23] present the influence of antenna polarization direction, multi-path effect, transceiver distance, voltage, power, and noise interference on the extraction of RF fingerprint features and the identification of devices. Therefore, it is necessary to consider the robustness issue in the selection of RF fingerprint features.

C. Identification of wireless devices with RF fingerprint features

Identifying wireless devices can be considered as a classification problem in which each device represents a specific class. We use the features of all the wireless devices, selecting a device as a positive class and the other devices as a single negative class. In the identification issue with K wireless devices, we can use a length-K vector $[p(w_i; y_1), p(w_i; y_2), \ldots, p(w_i; yx_N))]$ to represent the estimated probabilities of matching the received signal with each of K devices, where y_i $(i = 1, 2, \ldots N)$ denotes the ith segment of data at the receive end and w_i the ith wireless device.

We can use the signal data y to identify a wireless device with the highest probability of matching. Mathematically, we have

$$\hat{w}_i = \text{argmax}_{w_i}[p(w_i; y_1), p(w_i; y_2), \ldots, p(w_i; yx_N))]. \quad (8)$$

IV. Algorithm design

In this section, we first present the architecture of our proposed DSLN algorithm, and demonstrate the layered structure of DSLN. Then, we address the DSLN training pipeline, and design each of the components in the DSLN. Finally, we implement the proposed DSLN algorithm for device recognition in IoT scenarios.

A. DSLN Architecture

As introduced above, we train the I/Q samples of RF signals within a DSLN framework to recognize IoT devices. A DSLN is a type of improved variant of ResNets which is composed of typical components such as a convolutional layer, a residual building unit (RSBU), a batch normalization/rectifier linear unit/convolutional layer unit (BRC), a global average pooling (GAP), and a fully connected (FC) layer. The structure of a DSLN is shown in Fig.3.

A convolutional layer is used to compute the convolution between input I/Q samples and a convolutional kernel. The convolution process can dramatically reduce running time compared with matrix multiplication that is widely used in the fully-connection layer. The RSBU is the core of our DSLN model, eliminating the noise features by using a dynamic threshold. Typically, an RSBU comprises two layers of BRC, one layer of GAP, one layer of FC in sequence. Additionally, an RSBU has four operations, including sigmoid function, element-wise multiplication, dynamic thresholding, element-wise summation. A BRC takes the role of nonlinear transformation to reduce the variation of features in the I/Q samples of RF signals. The BRC includes three components, i.e. a batch normalization (BN), a rectifier linear unit (ReLU), and a convolutional layer unit. A GAP layer computes the mean of features, and it can further eliminate the variation of features in the I/Q samples. An FC layer is designed to use transformation matrices for multi-class recognition tasks. In the following, these components are described in detail.

B. DSLN training pipeline

(1) A convolutional layer uses a convolutional kernel to replace matrix multiplications in a fully connected neural network, which can dramatically reduce the number of parameters to be trained. With the convolutional layer, we can achieve a high accuracy since the process of convolution may avoid the overfitting in the model. Specifically, we can compute the jth output $\mathbf{I}_j$ by convolving the ith input $\mathbf{P}_i$ and the convolutional kernel $\mathbf{C}$, adding a bias $\mathbf{B}$ as:

$$\mathbf{I}_j = \sum_i \mathbf{P}_i * \mathbf{C}_{ij} + \mathbf{B}_j. \quad (9)$$

Typically, the feature map of a convolutional layer is a 3-D tensor. In this paper, we take the 2-D I/Q samples as input data, and thus the height of a feature map is set to be 1. With the convolutional kernel sliding on the feature map, we can achieve a channel of feature map at the output end. When a convolutional layer contains multiple convolutional kernels, we can attain more than one channel as the output of feature map.

(2) A RSBU is the core of a DSLN network. The primary operation of RSBU is a dynamic threshold, and it can be expressed as

$$\mathbf{W}_i = \begin{cases} \mathbf{U}_i - \tau, & \mathbf{U}_i > \tau \\ 0, & -\tau \le \mathbf{U}_i \le \tau \\ \mathbf{U}_i + \tau, & \mathbf{U}_i < -\tau \end{cases}, \quad (10)$$

where $\mathbf{U}$ represents the features of input I/Q signals, $\mathbf{W}$ represents the output features, and τ represents a dynamic threshold which is usually a positive parameter. This dynamic threshold only sets the near-zero features to zeros, instead of setting all the negative features to zero as in the ReLU function, so that the negative features of a signal can be preserved.

The threshold used in the RSBU is expressed as follows:

$$\tau = \alpha \times \text{mean}[|\mathbf{U}|], \quad (11)$$

where mean$[|\mathbf{U}|]$ represents the mean of modulus of $\mathbf{U}$, which is the output of GAP. α represents a scaling parameter, which is expressed as

$$\alpha = \frac{1}{1 + e^{-|t|}}, \quad (12)$$

where t represents the output of fully connection networks.

Fig.3 illustrates the detailed structure of a RSBU. t is the output of a fully connection layer, and according to equation (12) α is constrained within 0.5 to 1. Because the threshold τ is a linear function of α, the threshold τ is dependent on t.

According to Fig.3, RSBU is composed of two operations: one is identity mapping, and the other is wavelet denoising. The output of identity mapping is denoted as $\mathbf{I}$, and the output of wavelet denoising is $\mathbf{W}$. By element-wise summation, the output of RSBU can be denoted as $\mathbf{O} = \mathbf{I} + \mathbf{W}$.

A specific extreme example to clarify the operation in equation (12) is presented as follows: When the output of a fully connection layer t=0.0001, according to equation (12), α is approximate to be 0.5, and $\tau = 0.5\text{mean}[|\mathbf{U}|]$; when the

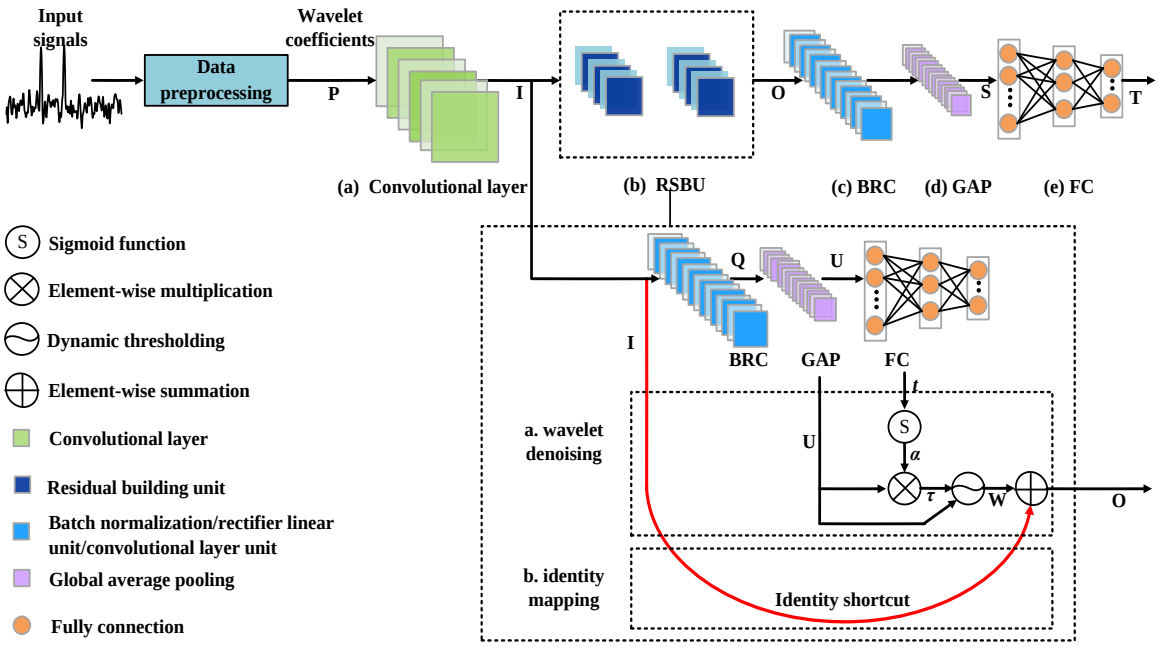


Fig. 3. Proposed DSLN based RF recognition algorithm.

output of a fully connection layer t=10000000, according to equation (12), α is approximate to be 1, and $\tau = \text{mean}[|\mathbf{U}|]$.

When $\tau = 0.5\text{mean}[|\mathbf{U}|]$, only the inputs $\mathbf{U}$ with a comparatively large value is preserved, and the preserved inputs have a high possibility to be useful features based on the wavelet de-noising theory. Therefore, we need to execute both the wavelet de-noising step and the identity mapping step by using an element-wise summation.

When $\tau = \text{mean}[|\mathbf{U}|]$, compared to the case of $\tau = 0.5\text{mean}[|\mathbf{U}|]$, more inputs $\mathbf{U}$ are set to be zeros, indicating that the inputs have a comparatively lower level of noise and the wavelet de-noising is not necessary. Therefore, we could skip the wavelet de-noising step, and only execute the identity mapping step.

To reduce the running time of RF fingerprint recognition, we design an identity shortcut in the stage of RSBU. An identity shortcut is used to skip one or more convolutional layers, and directly to connect a particular layer to a latter layer with an identical mapping. By guaranteeing the residual of the building block with an identity shortcut connection is close to zero, critical features of data would not be lost [17]. Due to identical mappings, this design is verified to accelerate the training process of RSBU, thereby reducing running time for trainning, shown later in Section V.D.

(3) In the BRC layer, a batch normalization can reduce the variation of features in each round of training process by adjusting the parameters of a convolution layer. The step of batch normalization can normalize the features to ensure that they are in a distribution with the average of zero and the standard deviation of one. Specifically, the detailed process of batch normalization can be characterized as

$$\begin{aligned} \mu &= \frac{1}{N_b}\sum_{i=1}^{N_b}\mathbf{I}_i, \\ \sigma^2 &= \frac{1}{N_b}\sum_{i=1}^{N_b}(\mathbf{I}_i-\mu)^2, \\ \hat{\mathbf{I}}_i &= \frac{\mathbf{I}_i-\mu}{\sqrt{\sigma^2+\delta}}, \\ \mathbf{Q}_i &= \alpha\hat{\mathbf{I}}_i+\gamma \end{aligned} \tag{13}$$

where $\mathbf{I}_i$ and $\mathbf{Q}_i$ denote the ith input and ith output features in a batch, respectively. μ and σ^2 represent the mean and the deviation, respectively. N_b represents the number of features in a batch. α and γ are two parameters for scaling and shifting the distributions, respectively. δ represents a constant, which is close to zero.

(4) A global average pooling layer computes the mean of feature inputs. Generally, it can reduce the number of training weights in a network and thus achieve a lower probability of overfitting. GAP can also reduce the impact of variation of input features on the output.

(5) A fully connection layer is designed to minimize the cross-entropy error of multiclass recognition. In comparison with the minimization of squared mean error, the minimization of cross-entropy error can enhance the training efficiency since the gradient of cross-entropy error is less likely to be zero. To compute the cross-entropy error, we can use a softmax function to ensure that the feature values can be kept in the range of $(0, 1)$. Specifically, we can represent a softmax function as

$$\mathbf{T}_i = \frac{e^{\mathbf{S}_i}}{\sum_k^{N_c} e^{\mathbf{S}_k}}, \tag{14}$$

where $\mathbf{S}_i$ and $\mathbf{T}_i$ denote the ith input and ith output of a cross-entropy error layer, respectively. N_c denotes the number of classes to recognize.

Assume that $P_r(\mathbf{T}_j)$ denotes the probability of an output to be classified into the jth class, we can represent the cross-entropy error as

$$E = -\sum_{i}^{N_c} P_r(\hat{\mathbf{T}}_j) \log(P_r(\mathbf{T}_j)), \tag{15}$$

where E denotes the cross-entropy error, $P_r(\hat{\mathbf{T}}_j)$ denotes the actual probability of an output to be classified into the jth class.

C. Implementation of DSLN for device recognition

Based on the above-mentioned procedures, we design a gradient descent algorithm to minimize the cross-entropy error, and train a few iterations of a DSLN model. The detailed DSLN algorithm is presented in Algorithm 1.

In Algorithm 1, given the initial parameters input signal $\mathbf{P}_i$, convolutional kernel $\mathbf{C}_{ij}$, bias $\mathbf{B}_j$, RSBU parameters δ, γ, t, we can start the process of DSLN based algorithm as follows. We first compute the output of a convolutional layer, a RSBU layer, a BRC layer, a GAP layer and a FC layer.

Algorithm 1: DSLN based algorithm for device identification with RF fingerprint

Require:
1: Input signal $\mathbf{P}_i$.
2: Convolutional kernel $\mathbf{C}_{ij}$.
3: Bias $\mathbf{B}_j$.
4: RSBU parameters δ, γ, t.
Ensure:
5: **for** each iteration in training **do**
6: **Compute** the output of a convolutional layer $\mathbf{I}_j = \sum_i \mathbf{P}_i * \mathbf{C}_{ij} + \mathbf{B}_j$ in equation (9).
7: **Compute** the output of RSBU by repeating the following updates:
$\alpha = \frac{1}{1+e^{-|t|}}$,
$\tau = \alpha \times \text{mean}[|\mathbf{U}|]$.
8: **Compute** the output of BRC by repeating the following updates:
$\mu = \frac{1}{N_b}\sum_{i=1}^{N_b} \mathbf{I}_i$,
$\sigma^2 = \frac{1}{N_b}\sum_{i=1}^{N_b}(\mathbf{I}_i - \mu)^2$,
$\hat{\mathbf{I}}_i = \frac{\mathbf{I}_i - \mu}{\sqrt{\sigma^2+\delta}}$,
$\mathbf{Q}_i = \alpha\hat{\mathbf{I}}_i + \gamma$.
9: **Update** the output of full connection with $\mathbf{T}_i = \frac{e^{\mathbf{S}_i}}{\sum_k^{N_c} e^{\mathbf{S}_k}}$ to achieve the cross-entropy error.
10: **Estimate** E with $\mathbf{T}_j = \text{argmax}_{\mathbf{T}_j}$ $E = -\sum_i^{N_c} P_r(\hat{\mathbf{T}}_j)\log(P_r(\mathbf{T}_j))$ with the output of $\mathbf{T}$.
11: **end for**
12: **Return** the label of device $\mathbf{T}$.

V. Performance evaluation and discussion

We carried the experiment over a hardware platform consisting of a NI-PXIe 1085 device and three USRP-RIO-2943 (Universal software radio peripheral-radio reconfigurable Input/Output). As a low-cost software-defined radio support device, USRP-RIO enables a wide range of applications, including broadcasting, mobile, GPS, WiFi, ISM FM, TV signals, and so on. So we can use USRP-RIO to simulate diverse IoT devices.

The hardware set is shown in Fig. 4, and the user interface of signal analysis software is shown in Fig. 5. NI-PXIe 1085 is a computer-based platform for data transmission and graphic display. The two USRP RIO-2943 (RIO2 and RIO3) contain 4 transmitters, simulating four different transmitters that need to be identified; one USRP RIO-2943 (RIO1) as a receiver, responsible for receiving signals from the four transmitters. As for the transmitters, their hardware differences can lead to amplitude difference and phase difference in (3), (6), (7). We set the parameters as follows: the pattern of the signal from transmitter 1 only indicates the changes in amplitude (using (6)), the pattern of the signal from transmitter 2 only indicates the changes in phase (using (7)), and the pattern of transmitter 3 and 4 indicates the changes in both amplitude and phase (using (3)).

We consider CNN, RNN, and a hybrid model CRNN (CNN+RNN) [22] as benchmarks to compare with our proposed model. The hybrid CRNN is composed of three layers: a CNN based transcription layer, a RNN based recurrent layer, and a connectionist temporal classification (CTC) layer. The transcription layer takes the role of feature extraction from input data. The recurrent layer is used to predict the feature sequence, learn each feature vector in the sequence, and output the predicted labels. The CTC layer employs CTC loss to convert a series of labels from the recurrent layer into the final label sequence.

A. Setting of experiments

In the experiments, we used a NI-PXIe 1085 computer to collect the data from multiple RIO-2943 USRPs, and extracted the RF fingerprint features for device identification. Afterward, we used MATLAB 2018b to simulate wireless channels, including AWGN and Rayleigh channels. In MATLAB, we set the sampling rate as 20 Mbps, the number of subcarriers as 64, the guard interval length of 0.8μs for the transmitted signals, the root-mean-squared delay spread (RDS) as 50 ns and 100 ns with 16 paths in Rayleigh channels, simulating the environment of home and office (shown in Table I) respectively [23].

Specifically, four transmitters cyclically transmit the data with quadrature phase shift keying (QPSK) modulation to the receiver. We collected 20000 independent groups of signals emitted from four transmitters, and each consisting of 190-200 I/Q samples. Hence we have 3834920 I/Q signal samples in total. 80% of these data (around 16000 independent groups of 3067936 samples) are used for training, and 20% of the data (around 4000 independent groups of 766984 samples) are used for testing.

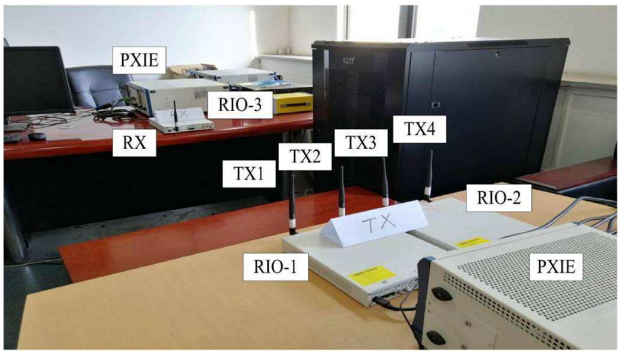


Fig. 4. Experimental settings for data collection.

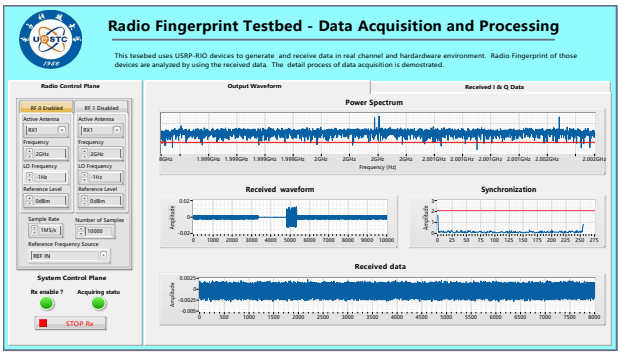


Fig. 5. User interface of signal analysis software.

Besides the data collected in our experiment, we also use the Oracle dataset generated by Genesys lab, Northeastern University [24] to compare the performance of various deep-learning networks. The Oracle dataset is collected by using more than 100 WiFi devices in an open recreation area with much fewer reflections, while the data in our experiment are collected in a closed lab with more reflections. Thus, the dataset can be viewed as a complement to the data collected in our lab. For all the data either collected from our lab or recorded in the Oracle dataset, we randomly select 80% of data for model training while using the other data for model testing.

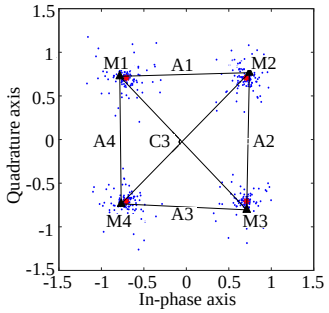
B. Constellation of I/Q signal samples

The original I/Q signal sample points appear around 4 constellation points (0.5, 0.5), (0.5, -0.5), (-0.5, -0.5) and (-0.5, 0.5) with a deviation, shown in Fig. 6 (a), which depends on the subtle difference among wireless devices. After passing through AWGN or Rayleigh channels, the difference around 4 constellation points is fairly scattered.

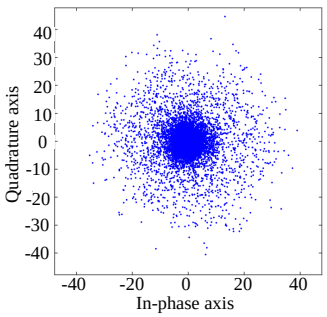
Fig. 6 (b) and Fig. 6 (c) show that the constellation points range from -10 to 10 in both horizontal axis (In-phase) and vertical axis (Quadrature) through AWGN channels, while the constellation points range from -5 to 5 through a Rayleigh channel at an RDS of 100 ns when SNR equals to 0 dB. This result indicates that device recognition in Rayleigh channels is more difficult than that in AWGN channels, since the constellation points are widely dispersed, which shows the difference of constellation points among devices.

TABLE I
PARAMETERS OF RAYLEIGH CHANNELS

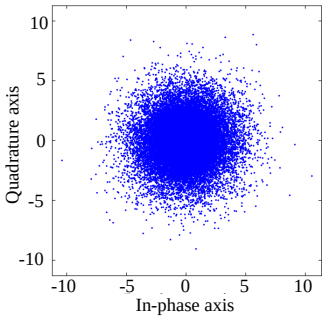
Environment	RDS (ns)	Number of paths
Home	50	16
Office	100	16
Environment	**Sampling rate (Mbps)**	**Guard interval length (μs)**
Home	20	0.8
Office	20	0.8



(a) At the transmit end.



(b) At the receive end (AWGN).



(c) At the receive end (Rayleigh).

Fig. 6. Constellation of RF fingerprint (SNR=0 dB) at both transmit and receive ends after passing through AWGN or Rayleigh channels.

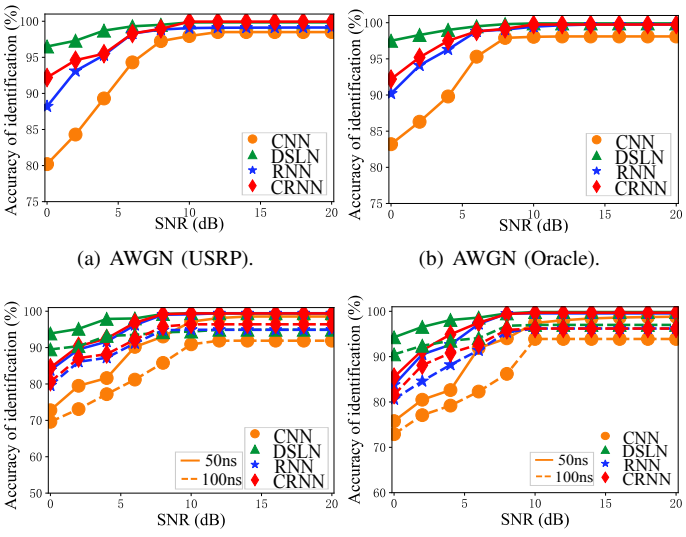
C. Accuracy of RF fingerprint recognition

The identification accuracy in AWGN channels with our collected dataset and that with the Oracle dataset are respectively shown in Fig. 7 (a) and Fig. 7 (b). The results show that our proposed DSLN algorithm outperforms CNN, RNN, CRNN (CNN+RNN) algorithms at different levels of SNRs. In the low-SNR region, the proposed algorithm can achieve a dramatically higher level of accuracy than that of the other algorithms. Moreover, the gap in the level of accuracy between these algorithms increases with the decrease of SNR. Specifically, at a very low SNR (e.g., 0 dB), DSLN outperforms CRNN by 5%, outperforms RNN by 8%, and outperforms CNN by 16%. The confusion matrix for the four device identification is shown in Table II. Each entry refers to the number of groups, illustrated in Subsection V.A. In addition, within the range of 0 dB to 20 dB, the accuracy of DSLN varies from 95.5% to 99.5%, which illustrates that DSLN is more robust than the other algorithms. In brief,

TABLE II
CONFUSION MATRIX FOR 4 DEVICE IDENTIFICATION PROBLEM AT VERY LOW SNR (NUMBER OF GROUPS)

CNN	AWGN				Rayleigh (RDS=50 ns)				Rayleigh (RDS=100 ns)			
	Device 1	Device 2	Device 3	Device 4	Device 1	Device 2	Device 3	Device 4	Device 1	Device 2	Device 3	Device 4
Device 1	733	88	88	91	725	95	91	89	700	95	105	100
Device 2	75	780	73	72	83	750	83	84	113	681	101	106
Device 3	74	74	772	80	90	90	730	90	97	97	710	96
Device 4	75	73	70	782	86	87	87	740	100	99	99	702
DSLN	**AWGN**				**Rayleigh (RDS=50 ns)**				**Rayleigh (RDS=100 ns)**			
	Device 1	Device 2	Device 3	Device 4	Device 1	Device 2	Device 3	Device 4	Device 1	Device 2	Device 3	Device 4
Device 1	959	15	11	15	952	16	11	21	901	33	33	33
Device 2	19	945	18	18	21	940	19	20	32	901	32	35
Device 3	14	13	961	12	14	15	947	14	30	36	892	42
Device 4	15	14	15	956	20	21	20	959	32	32	31	905
RNN	**AWGN**				**Rayleigh (RDS=50 ns)**				**Rayleigh (RDS=100 ns)**			
	Device 1	Device 2	Device 3	Device 4	Device 1	Device 2	Device 3	Device 4	Device 1	Device 2	Device 3	Device 4
Device 1	850	50	45	55	840	60	50	50	807	64	65	64
Device 2	42	873	40	45	55	835	56	54	73	783	72	72
Device 3	56	60	834	50	41	51	836	72	71	58	813	58
Device 4	45	45	44	862	51	51	82	845	51	71	72	806
CNN+RNN	**AWGN**				**Rayleigh (RDS=50 ns)**				**Rayleigh (RDS=100 ns)**			
	Device 1	Device 2	Device 3	Device 4	Device 1	Device 2	Device 3	Device 4	Device 1	Device 2	Device 3	Device 4
Device 1	880	40	40	40	845	56	49	50	817	60	60	63
Device 2	40	880	35	45	51	839	56	54	70	793	70	67
Device 3	54	57	840	50	41	45	845	69	61	59	821	59
Device 4	40	42	44	870	51	51	72	855	51	66	67	816

compared to CNN, RNN, CRNN, our proposed DSLN is less sensitive to noise and is more capable of learning the RF fingerprint features from wireless signals even when they are contaminated by a high level of noise.



(a) AWGN (USRP). (b) AWGN (Oracle).

(c) Rayleigh (USRP). (d) Rayleigh (Oracle).

Fig. 7. Accuracy comparison of CNN, RNN, CRNN (CNN+RNN), and DSLN. ('50 ns' represents $RDS = 50ns$, '100 ns' represents $RDS = 100ns$.)

We also evaluate the device identification with RF fingerprint features in Rayleigh channels. These results from our collected data and from Oracle dataset are shown in Fig. 7 (c) and Fig.7 (d), respectively. As shown in Fig. 7 (c) and Fig. 7 (d), the accuracy of device identification with CNN, RNN, CRNN (CNN+RNN) and DSLN increases with the rise of root-mean-squared delay spread (RDS), i.e., the accuracy in the home environment is higher than that in the office environment. Also, in comparison with AWGN channels, the accuracy in the Rayleigh channels decreases. This is due to the multi-path effect of the Rayleigh channels that leads to the change of a signal segment, thus making it difficult to identify a device in the Rayleigh channels.

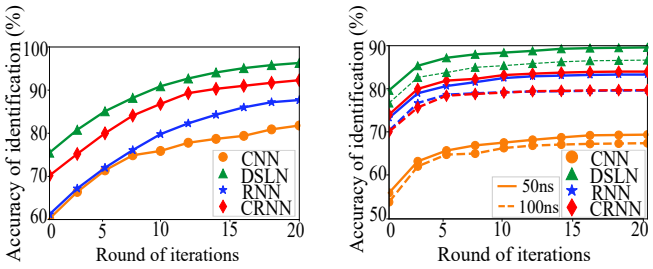
As shown in Fig. 7 (c), in comparison with other algorithms, our proposed DSLN can achieve a higher level of accuracy. Specifically, when SNR is 0 dB and RDS is 50 ns, DSLN outperforms CRNN by 9%, outperforms CNN by 20%, and outperforms RNN by 11%. The confusion matrix for 4-device identification with various learning algorithms is illustrated in Table II with the RDS of 50 ns and 100 ns, respectively.

The key of our network to achieve a better performance in high-level noise environments is the use of dynamic thresholding for activation functions, shown in equation (10). In the data pre-processing procedure, original data is transformed into wavelet coefficients for de-noising. In the following, we use a dynamic thresholding based activation function, which sets near-zero inputs (between $-\tau$ and τ) to zeros, since these inputs have a high probability of being noise without any useful information according to the theory of signal de-noising. Specifically, our idea is borrowed from the Wavelet threshold de-noising method [25]. Because wavelets localize data features to different scales, important signal or data features can be preserved after removing noise. That is, a wavelet transform leads to a sparse representation for real-world signals, and it merely concentrates signals in a few large-magnitude wavelet coefficients. It should be noted that the wavelet coefficients with small values are regularly

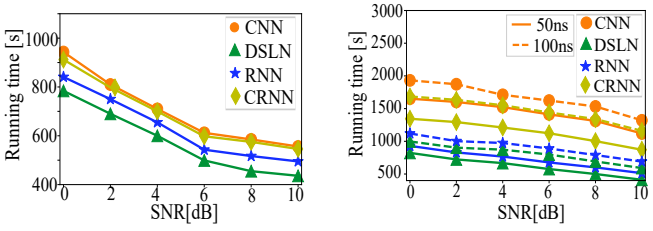
considered as noise.

D. Convergence of RF fingerprint recognition algorithms

This section investigates the convergence of RF fingerprint recognition by comparing CNN, RNN, CRNN (CNN+RNN), and our proposed DSLN algorithms with our collected dataset and the Oracle dataset. As shown in Fig. 8 (a) and (b), it takes around 16-18 iterations in RNN, CRNN and DSLN to achieve the convergence in AWGN channels when SNR is 0 dB, while it takes 18-20 iterations in CNN for convergence. In addition, it takes around 15 iterations in RNN, CRNN and DSLN to achieve convergence in Rayleigh channels when SNR is 0 dB, while it takes 18 iterations in CNN for convergence. The reason is that compared to an AWGN channel, a Rayleigh channel is significantly influenced by the effect of multiple paths, and thus it needs more iterations to learn signal features.



(a) Convergence (AWGN). (b) Convergence (Rayleigh).



(c) Running time (AWGN). (d) Running time (Rayleigh).

Fig. 8. Convergence and running time comparison of CNN, RNN, CRNN (CNN+RNN), and DSLN (SNR = 0 dB) ('50 ns' represents $RDS = 50ns$, '100 ns' represents $RDS = 100ns$.)

We then evaluate the running time of CNN, RNN, CRNN (CNN+RNN) and DSLN, including both data training and testing time. As shown in Fig.8 (c), it takes around 600-950 seconds to achieve convergence with CNN, around 500-850 second to achieve convergence with RNN, around 600-940 second to achieve convergence with CRNN, and around 430-780 seconds to achieve convergence with DSLN at different levels of SNR in AWGN channels. In Fig. 8, the numbers of iterations in RNN, CRNN and DSLN are almost the same, but DSLN needs less time for training and testing than other algorithms. This is due to the fact that DSLN employs an identity shortcut that could facilitate parameter update process [26].

In Rayleigh channels, DSLN needs 400-800 seconds or 600-1000 seconds to achieve convergence when the RDS is 50 ns or 100 ns, respectively. In comparison with CNN, our proposed DSLN algorithm can reduce the running time by $50-60\%$ in total. In comparison with RNN, our proposed DSLN algorithm can reduce the running time by $10-20\%$. In comparison with CRNN, our proposed DSLN algorithm can reduce the running time by $25-40\%$.

E. Computation complexity of RF fingerprint recognition algorithms

In this section, we discuss the computation complexity in the training phase of various learning algorithms. Specifically, we present the computation complexity of DSLN, CNN, RNN and CRNN. The authors in [27] proved that the learning process of a neural network with a depth of d takes a time of $O\left(s^{2^d}\right)$, where s refers to the input dimension and $O(.)$ represents the level of complexity of an algorithm. The operation of convolution has additional time complexity due to the forward and backpropagation units, which can be shown as $O\left(\sum_{m=1}^{M} c_m s_m^2 f_m \rho_m^2\right)$ where M represents the number of convolutional layers, m represents the index of a convolutional layer, c_m represents the size of input feature map at the mth layer, s_m represents the size of filters at the mth layer, f_m represents the number of filters at the mth layer, ρ_m represents the size of output feature map at the mth layer [28].

In the proposed DSLN model with depth d, we have one convolutional layer, and $d-1$ non-convolutional layers. The time complexity of DSLN model can be shown as

$$TC_D = O\left(c_1 s_1^2 f_1 \rho_1^2\right) + O\left(s^{2^d}\right). \tag{16}$$

In the CNN model, we have d convolutional layers, and thus the time complexity of CNN model can be shown as

$$TC_C = O\left(\sum_{m=1}^{d} c_m s_m^2 f_m \rho_m^2\right) + O\left(s^{2^d}\right). \tag{17}$$

In the RNN model, we have no convolutional layer, but merely d non-convolutional layers. The time complexity of CNN model can be shown as

$$TC_R = O\left(s^{2^d}\right). \tag{18}$$

In the CRNN model, we have $\hat{d}$ convolutional layers ($\hat{d} < d$) and $d - \hat{d}$ non-convolutional layers. The time complexity of CRNN model can be shown as

$$TC_{CR} = O\left(\sum_{m=1}^{\hat{d}} c_m s_m^2 f_m \rho_m^2\right) + O\left(s^{2^{\hat{d}}}\right) + O\left(s^{2^{d-\hat{d}}}\right). \tag{19}$$

By comparing equations (16), (17), (18), (19), we find that CNN and CRNN requires more time than other algorithms, attributing to convolutional layers that consume a large amount of computation time. The proposed DSLN model takes slightly more time than the RNN model since the former has one convolutional layer. But the gap of time complexity between DSLN and RNN tends to diminish with the increase of depth d. Moreover, DSLN needs fewer epochs for convergence than other models, resulting in less running time than RNN, CNN, and CRNN, as shown in Fig. 8.

F. Summary of performance evaluation

We have proposed a DSLN algorithm to recognize devices by analyzing RF fingerprints in a low-SNR IoT system. In comparison with CNN, RNN and CRNN that are most widely used deep learning algorithms, our proposed DSLN algorithm not only dramatically improves the accuracy of device recognition, but also reduces the running time for training and testing. These improvements stem from two aspects.

On one hand, in terms of the device recognition accuracy, the proposed algorithm outperforms CNN, RNN, CRNN by yielding improvements of 10% to 20%, under a typical low-SNR IoT environment. Also, the proposed DSLN algorithm is more robust than other algorithms. DSLN can keep a level of accuracy between $90 - 99\%$ in various AWGN and Rayleigh channels, while other algorithms can only achieve an accuracy below 80% at low SNRs. As a result, the design of a dynamic shrinkage threshold in our proposed DSLN algorithm can effectively improve the recognition accuracy at low SNRs and keep robust at various SNRs.

On the other hand, the proposed DSLN algorithm could reduce 20% to 60% of running time in comparison with CNN, RNN, CRNN. For DSLN, the design of an identity shortcut can skip convolutional layers and directly connect a particular layer to a latter layer with an identity mapping, thus greatly reducing running time. It should be noted that running time is a critical indicator to evaluate the performance of a real-time IoT system.

Conclusion

In this paper, we have proposed a novel algorithm, called DSLN, to identify IoT devices by analyzing their RF fingerprints. DSLN could mitigate the problems in the low SNR settings of IoT applications, e.g., in a radar system for Internet of vehicles. Specifically, we develop a dynamic shrinkage threshold to improve device recognition accuracy. Moreover, an identity shortcut has been presented for accelerating the training process. The detailed analysis and extensive experiments has shown the robustness of DSLN in various SNR scenarios. Finally, the deployment of a 4-device network has demonstrated that DSLN comprehensively outperforms CNN, RNN, CRNN, both in terms of the recognition accuracy and running time, thereby enabling security in real-time IoT scenarios.

To further improve the robustness of our RF recognition methods in complicated scenarios, we plan to extend our study to the scenario of both high-level noise and adversarial attacks. Different from [13], which assumes that RF signal data can be collected by a server for model training, we might not be able to send the original signal data to a server for model training in a high SNR setting. Instead, we can only complete the learning process in a distributed way, with a small amount of data exchange between the server and the devices, e.g., the key parameters of learning models. In a distributed computing architecture, an invalid user is more likely to mislead the learning process of device recognition by intentionally spoofing or imposing perturbations on RF signals. As a future work, it is interesting to consider distributed adversarial learning under low SNR environments.

Acknowledgement

This work is partially funded by Science and Technology Program of Sichuan Province (2021YFG0330), partially funded by Grant SCITLAB-0001 of Intelligent Terminal Key Laboratory of SiChuan Province, and partially funded by Fundamental Research Funds for the Central Universities (ZYGX2019J076).

References

[1] C. Kai, H. Li, L. Xu, et al., "Energy-efficient device-to-device communications for green smart cities," *IEEE Trans. Ind. Informat.*, vol. 14, no. 4, pp. 1542–1551, Apr. 2018.

[2] H. N. Dai, Z. Zheng, and Y. Zhang, "Blockchain for Internet of Things: A survey," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8076–8094, Oct. 2019.

[3] J. Huang, L. Kong, G. Chen, et al., "Towards secure industrial IoT: Blockchain system with credit-based consensus mechanism," *IEEE Trans. Ind. Informat.*, vol. 15, no. 6, pp. 3680–3689, Jun. 2019.

[4] Y. Sun, L. Zhang, G. Feng, et al., "Blockchain-enabled wireless Internet of Things: Performance analysis and optimal communication node deployment," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 5791–5802, Jun. 2019.

[5] M. A. Ferrag, M. Derdour, M. Mukherjee, et al., "Blockchain technologies for the Internet of Things: Research issues and challenges," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2188–2204, Apr. 2019.

[6] X. Wang, X. Wang and S. Mao, "RF Sensing in the Internet of Things: A General Deep Learning Framework," *IEEE Commun. Mag.*, vol. 56, no. 9, pp. 62-67, Sept. 2018.

[7] X. Zhou, A. Hu, G. Li, et al., "A Robust Radio Frequency Fingerprint Extraction Scheme for Practical Device Recognition," *IEEE Internet Things J.*, doi: 10.1109/JIOT.2021.3051402, Jan. 2021.

[8] T. Jian et al., "Deep Learning for RF Fingerprinting: A Massive Experimental Study," *IEEE Internet Things J.*, vol. 3, no. 1, pp. 50-57, March 2020.

[9] S. V. Radhakrishnan, A. S. Uluagac, and R. Beyah, "Gtid: A Technique for Physical Device and Device Type Fingerprinting," *IEEE Trans. Dependable Secure Comput.*, vol. 12, no. 5, Sept. 2015, pp. 519–532.

[10] X. Fan, F. Wang, F. Wang, et al., "When RFID Meets Deep Learning: Exploring Cognitive Intelligence for Activity Identification," *IEEE Trans. Wireless Commun.*, vol. 26, no. 3, pp. 19-25, June 2019.

[11] B. Chatterjee, D. Das, S. Maity, et al., "RF-PUF: Enhancing IoT Security Through Authentication of Wireless Nodes Using In-Situ Machine Learning," *IEEE Internet Things J.*, vol. 6, no. 1, pp. 388-398, Feb. 2019.

[12] S. Riyaz, K. Sankhe, S. Ioannidis, et al., "Deep Learning Convolutional Neural Networks for Radio Identification," *IEEE Commun. Mag.*, vol. 56, no. 9, pp. 146-152, Sept. 2018.

[13] D. Roy, T. Mukherjee, M. Chatterjee, et al., "RFAL: Adversarial Learning for RF Transmitter Identification and Classification," *IEEE Trans. Cogn. Commun. Netw.*, vol. 6, no. 2, pp. 783-801, June 2020.

[14] G. A. Akpakwu, B. J. Silva, G. P. Hancke, et al., "A Survey on 5G Networks for the Internet of Things: Communication Technologies and Challenges," *IEEE Access*, vol. 6, pp. 3619-3647, 2018.

[15] P. C. Ng, J. She and R. Rong, "Compressive RF Fingerprint Acquisition and Broadcasting for Dense BLE Networks," *IEEE Trans. on Mobile Comput.*, doi: 10.1109/TMC.2020.3024842, Sep 2020.

[16] X. Tian, X. Wu, H. Li and X. Wang, "RF Fingerprints Prediction for Cellular Network Positioning: A Subspace Identification Approach," *IEEE Trans. on Mobile Comput.*, vol. 19, no. 2, pp. 450-465, Feb 2020.

[17] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 770-778, 2016.

[18] L. Peng, J. Zhang, M. Liu, et al., "Deep Learning Based RF Fingerprint Identification Using Differential Constellation Trace Figure," *IEEE Trans. Veh. Technol.*, vol. 69, no. 1, pp. 1091-1095, Jan. 2020.

[19] M. M. Badr, M. I. Ibrahem, M. Mahmoud, et al., "Detection of False-Reading Attacks in Smart Grid Net-Metering System," *IEEE Internet Things J.*, 2021(Early access).

[20] G. Shen, J. Zhang, A. Marshall, et al, "Radio Frequency Fingerprint Identification for LoRa Using Spectrogram and CNN," *IEEE Int. Conf. Comput. Commun.*, 2021 (to be published).
[21] Q. Wu, C. Feres, D. Kuzmenko, et al, "Deep Learning Based RF Fingerprinting for Device Identification and Wireless Security," *Electron. Lett.*, vol.54, no. 24, pp. 1405-1407, 2018.
[22] S. Baoguang, B. Xiang, Y. Cong, "An End-to-End Trainable Neural Network for Image-Based Sequence Recognition and Its Application to Scene Text Recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*. PP.101-109, 2016.
[23] S. Singh, N. Singh, "Internet of Things (IoT): Security Challenges, Business Opportunities and Reference Architecture for E-commerce," 2015 *Int. Conf. on Green Comput. and Internet of Things*, Noida, 2015, 1577-1581.
[24] K. Sankhe, M. Belgiovine, F. Zhou, et al., "No Radio Left Behind: Radio Fingerprinting Through Deep Learning of Physical-Layer Hardware Impairments," *IEEE Trans. on Cogn. Commun. Netw., Special Issue on Evolution of Cognitive Radio to AI-enabled Radio and Networks*, 2019.
[25] J. Zhong, X. Bi, Q. Shu, et al., "Partial Discharge Signal Denoising Based on Singular Value Decomposition and Empirical Wavelet Transform," *IEEE Trans.on Instrum. Meas.*, vol. 69, no. 11, pp. 8866-8873, Nov. 2020.
[26] M. Zhao, S. Zhong, X. Fu, et al., "Deep Residual Shrinkage Networks for Fault Diagnosis," *IEEE Trans. Ind. Informat.*, vol. 16, no. 7, pp. 4681-4690, July 2020.
[27] K. He and J. Sun, "Convolutional neural networks at constrained time cost," *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 5353–5360.
[28] K. Merchant, S. Revay, G. Stantchev, et al., "Deep Learning for RF Device Fingerprinting in Cognitive Communication Networks," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 1, pp. 160-167, Feb. 2018.