Deep Learning Enhanced Radio Frequency Fingerprint Identification for LoRa

Thesis submitted in accordance with the requirements of the University of Liverpool for the degree of Doctor in Philosophy by

Guanxiong Shen

June 2023

Abstract

Wireless security is becoming a critical concern in the era of the Internet of Things (IoT). Radio frequency fingerprint identification (RFFI) is an authentication technique that identifies wireless devices by analyzing their physical attributes. The wireless signals are generated by the transmitter chain, which is composed of numerous analog components such as the oscillator, mixer, power amplifier, antenna, etc. The specifications of the hardware components will deviate slightly from their nominal values due to the manufacturing variation. Therefore, each wireless transmitter has unique hardware characteristics, termed the radio frequency fingerprint (RFF). The emitted wireless signals are distorted by the transmitter impairments, which can be captured and analyzed by the receiver for device identification. RFF is difficult to modify because it is the physical attribute of a wireless transmitter. RFFI can be deployed with a customized receiver without changing the transmitter hardware, making it suited for both legacy and future wireless networks. It becomes a potential authentication technique for low-cost IoT applications.

In recent years, RFFI has been significantly enhanced by deep learning. The specially designed neural networks (NNs) can predict the device identity by analyzing the received signal. Although deep learning leads to a huge performance improvement, there are still remaining challenges. Firstly, the distortion caused by

the transmitter impairments is minute and difficult to detect in the time domain. The received signals need to be converted into suitable signal representations as inputs to the NN to facilitate feature extraction. Secondly, the RFFI performance is severely affected by the wireless channel variations. However, many IoT devices are mobile and channel changes are unavoidable. It is necessary to explore how to mitigate the impact of the wireless channel on RFFI performance. Thirdly, the analog receiver chain can distort the received signal as well. Using a new receiver for signal reception can affect the characteristics of the received signal and thus degrade the RFFI performance. However, it is common for a wireless device to be served by different access points/gateways, and training an NN for each receiver is impractical and time-consuming.

This thesis addresses the challenges discussed above and takes LoRa communication technology as a case study. Firstly, it is proposed to convert the received LoRa IQ samples to a spectrogram before feeding them into the NN because the characteristics of LoRa chirp signals are more evident in the time-frequency domain. Convolutional neural networks (CNNs) are then found to be more efficient in processing time-frequency domain spectrograms than multilayer perception (MLP) and long short-term memory (LSTM) models. Secondly, a channel-independent spectrogram was created as the signal representation in order to minimize the impact of wireless channels and make the system location-independent. Moreover, it is also proposed to use data augmentation during training to combat channel effects. More specifically, the training data is fed into a channel simulator to emulate signals under various channel conditions. Finally, a receiver-agnostic and collaborative RFFI protocol is designed, in which the adversarial training is leveraged to train a receiver-agnostic classification NN that is deployed on multiple receivers to make a collaborative inference. As the trained NN may not perform satisfactorily on some

receivers, a fine-tuning technique is further proposed to improve its performance. The proposed fine-tuning scheme only requires a few labeled packets to reach a satisfactory performance. All the proposed LoRa-RFFI protocols are experimentally evaluated using commercial-off-the-shelf (COTS) LoRa devices and software-defined radio (SDR) boards in real communication scenarios.

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my supervisors Dr Junqing Zhang and Prof Alan Marshall. This thesis would not have been finished without their insightful advice and tremendous support. As Dr Zhang's first Ph.D. student, I am grateful for his hands-on guidance in both academic and daily life. It is my great honor to have had this opportunity to work with him in the past three years. I will always cherish the ways of doing research that I learned from them.

I would also like to thank all my friends and colleagues at Liverpool, particularly Dr Tianyuan Jia, Dr Yixuan Zhang, Dr Chen Chen, Mr Guolin Yin, Mr Jiaxu Liu, Mr Ningze Yuan, Mr Senhao Gao, Miss Jie Ma, Miss Yijia Guo. Thank them for the productive discussions and enjoyable times.

Finally and most importantly, I would like to thank my parents and family. It has been more than three years since the last time I was home. I deeply appreciate their unwavering love and support throughout my life.

List of Publications

Journal Publications

- **G. Shen**, J. Zhang, A. Marshall, L. Peng, and X. Wang, "Radio frequency fingerprint identification for LoRa using deep learning," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 8, pp. 2604–2616, 2021.
- **G. Shen**, J. Zhang, A. Marshall, and J. R. Cavallaro, "Towards scalable and channel-robust radio frequency fingerprint identification for LoRa," *IEEE Trans. Inf. Forensics Security*, vol. 17, pp. 774–787, 2022.
- **G. Shen**, J. Zhang, A. Marshall, M. Valkama, and J. Cavallaro, "Towards length-versatile and noise-robust radio frequency fingerprint identification," *IEEE Trans. Inf. Forensics Security*.
- **G. Shen**, J. Zhang, and A. Marshall, "Deep Learning-Powered Radio Frequency Fingerprint Identification: Methodology and Case Study," *IEEE Commun. Mag.*.
- **G. Shen**, J. Zhang, A. Marshall, R. Woods, J. Cavallaro, and L. Chen, "Towards receiver-agnostic and collaborative radio frequency fingerprint identification," *IEEE Trans. Mobile Comput.*, Under Review.

Conference Publications

- **G. Shen**, J. Zhang, A. Marshall, L. Peng, and X. Wang, "Radio frequency fingerprint identification for LoRa using spectrogram and CNN," in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, Virtual Conference, May 2021, pp. 1–10.

- **G. Shen**, J. Zhang, A. Marshall, M. Valkama, and J. Cavallaro, "Radio frequency fingerprint identification for security in low-cost IoT devices," in *Proc. Asilomar Conf. Signals, Systems, and Computers (Asilomar)*, 2021, pp. 309–313.

Publications (Collaboration)

- G. Yin, J. Zhang, **G. Shen**, and Y. Chen, "FewSense, Towards a Scalable and Cross-Domain Wi-Fi Sensing System Using Few-Shot Learning," *IEEE Trans. Mobile Comput.*, Early Access, 2022

- J. Ma, J. Zhang, **G. Shen**, A, Marshall, and C. Chang. "White-Box Adversarial Attacks on Deep Learning-Based Radio Frequency Fingerprint Identification," accepted by *Proc. IEEE Int. Conf. Commun. (ICC)*, 2023.

- H. Ruotsalainen, **G. Shen**, J. Zhang, and R. Fujdiak, "LoRaWAN physical layer-based attacks and countermeasures, a review," Sensors, vol. 22, no. 9, p. 3127, 2022.

- P. Yin, L. Peng, **G. Shen**, J. Zhang, M. Liu, H. Fu, A. Hu, and X. Wang. "Multi-Channel CNN-Based Open-Set RF Fingerprint Identification for LTE Devices," *IEEE Trans. on Cogn. Commun. Netw.*, Under Review.

- J. Zhang, **G. Shen**, W. Saad, and K. Chowdhury. "Radio Frequency Fingerprint Identification, Device Authentication for Internet of Things," *IEEE Commun. Mag.*, Under Review.

Dataset

- **G. Shen**, J. Zhang, A. Marshall, Feb. 3, 2022, "LoRa_RFFI_dataset,", IEEE Dataport, [Online], Available: https://dx.doi.org/10.21227/qqt4-kz19

Contents

Acronyms

IoT Internet of Things

RFFI radio frequency fingerprint identification

SEI specific emitter identification

RF radio frequency

RFF radio frequency fingerprint

DL deep learning

IQ in-phase and quadrature

CFO carrier frequency offset

NN neural network

CNN convolutional neural network

LSTM long short-term memory

MLP multiple layer perceptron

GRU gated recurrent unit

GAN generative adversarial network

LTE long-term evolution

RFID radio frequency identification

NFC near-field communication

DUT device under test

AWGN additive white Gaussian noise

i.i.d. independent and identically distributed

COTS commercial off-the-shelf

SDR software defined radio

UAV unmanned aerial vehicle

DCTF differential constellation trace figure

FFT fast Fourier transform

SVM support vector machine

CSS chirp spread spectrum

ADC analog-to-digital converter

STFT short-time Fourier transform

SNR signal-to-noise ratio

RMS root mean square

USRP universal software radio peripheral

BN batch normalization

ReLU rectified linear unit

Adam adaptive moment estimation

GPU graphics processing unit

t-SNE t-distributed stochastic neighbor embedding

LOS line-of-sight

kNN k-nearest neighbor

NLOS non-line-of-sight

PDP power delay profile

RMSprop root mean squared propagation

SGD stochastic gradient descent

Symbols

K Number of DUTs in the RFFI-secured wireless network

$y(t)$ Received physical layer baseband signal

$x(t)$ Ideal transmitted signal without distortion

$\mathcal{G}(\cdot)$ Distortion caused by the receiver impairments

$h(\tau, t)$ Channel impulse response

$\mathcal{F}^k(\cdot)$ Distortion caused by the hardware impairments of transmitter k

$n(t)$ Additive white Gaussian noise

$c(t)$ A baseband linear chirp, i.e., an unmodulated LoRa symbol

Λ Amplitude

BW Bandwidth

T_{sym} LoRa symbol time

SF LoRa spreading factor

T_s The sampling interval of receiver ADC

$x[n]$ Transmitted IQ samples

$y[n]$ Received IQ samples

$\mathbf{x}$ Transmitted IQ samples in vector form

$\mathbf{y}$ Received IQ samples in vector form

L_{pre} Length of the preamble part of a LoRa packet

L_{sym} Length of a LoRa symbol

$S_{a,b}$ The element in the a^{th} row and b^{th} column of the complex matrix $\mathbf{S}$

$\mathbf{S}$ The complex matrix obtained by STFT, i.e., spectrogram

B The number of columns of $\mathbf{S}$

A The number of rows of $\mathbf{S}$

$w[n]$ The window function used in STFT

L_{hop} The hop size used in STFT

$\widetilde{\mathbf{S}}$ The spectrogram in dB scale

$M[n]$ The metric during LoRa packet detection

$y_{rcv}[n]$ The received signal before LoRa packet detection

λ_{det} The threshold used in LoRa packet detection

$x_{lo}[n]$ The locally generated ideal baseband LoRa preamble part

$f_{lo}[n]$ The instantaneous frequency of $x_{lo}[n]$

$f_{rcv}[n]$ The instantaneous frequency of $y_{rcv}[n]$

Ψ The index of the starting point located by the fine synchronization algorithm

$y'_{rcv}[n]$ The signal after synchronization and before CFO compensation

$f'_{rcv}[n]$ The instantaneous frequency of $y'_{rcv}[n]$

Δf The CFO in the received signal

$f_{sym}[n]$ The instantaneous frequency of a received LoRa preamble

$\widehat{f}_{coarse}$ The CFO estimated by the coarse estimation algorithm

$y''_{rcv}[n]$ The signal after coarse CFO compensation

$\widehat{f}_{fine}$ The CFO estimated by the fine estimation algorithm

$\Delta\widehat{f}$ The overall estimated CFO after coarse and fine estimation

$y'''_{rcv}[n]$ The signal after CFO compensation and before normalization

$\mathcal{D}^{train}$ The training dataset

$\mathbf{p}_m$ One-hot encoded label of the m^{th} training sample

$O(\cdot)$ One-hot encoding operation

ℓ_m The DUT label of the m^{th} training sample

M_{train} The number of training samples

$f(\cdot;\Theta)$ The neural network with parameters Θ

Θ The parameters of the neural network

$\mathcal{L}_{ce}(\cdot)$ The cross-entropy loss function

$\mathbf{y}'$ The received siganl in the inference stage

$\hat{\mathbf{p}}$ The prediction made by the neural network, i.e., a probability vector

$\hat{\ell}$ The predicted DUT label

z_k The output of the k^{th} neuron before the softmax activation

$\widetilde{\mathcal{F}}^k(\cdot)$ The distortion caused by the transmitter impairments in the frequency domain

$\mathbf{X}_b$ The ideal frequency spectrum of the b^{th} signal segment in the STFT operation

$\mathbf{H}_b$ The channel frequency response experienced by the b^{th} signal segment

$\mathbf{C}$ The channel independent spectrogram generated from the STFT matrix $\mathbf{S}$

$\widetilde{\mathbf{C}}$ The channel independent spectrogram in dB scale

$P[d]$ The power of the d^{th} path in the exponential PDP model

τ_d The RMS delay spread in the exponential PDP model

d_{max} The index of the last path in the exponential PDP model

$\Omega(f)$ The spectrum of the Jakes model

f_d The maximum Doppler shift

I Number of receivers in the adversarial training

J Number of receivers in the collaborative inference

$\mathbf{q}_m$ The one-hot encoded receiver label of the m^{th} training sample

$\mathcal{X}^{train}$ The dataset used for adversarial training

$g(\cdot)$ The feature extractor in the adversarial training

θ The learnable parameters in the feature extractor

$\mathcal{L}_{tx}$ The loss function used for transmitter classifier

$\mathcal{L}_{rx}$ The loss function used for receiver classifier

μ The learning rate during training

γ^j The SNR at the j^{th} receiver

$\hat{\mathbf{p}}^j$ The prediction made by the j^{th} receiver

$\hat{\mathbf{p}}^{\mathbf{fused}}$ The fusion result of collaborative inference

$\hat{\ell}^{fused}$ The predicted label of collaborative inference

$\mathcal{X}^{tune}$ The dataset used for fine-tuning

M_{tune} The number of fine-tuning packets

List of Figures

List of Tables

Chapter 1

Introduction

In the era of Internet of Things (IoT), wireless devices are becoming ubiquitous in our daily lives [1]. For example, more and more household appliances are equipped with wireless connectivity, which enables the homeowner to control and access them remotely. In addition, wireless sensors installed in factories can exchange collected data or transfer it to a central controller for analysis, thereby enhancing manufacturing efficiency. It is estimated that there will be more than 30 billion IoT devices collecting and exchanging data around the world by 2025 [2]. Although the fast-developing wireless technologies greatly facilitate our life and manufacturing process, security is becoming a pressing concern. The attacker can compromise a user's privacy if an IoT device is not properly secured because some IoT devices contain private information. Moreover, IoT devices are widely used in medical applications. A compromised medical IoT device can be used to interfere with a patient's healthcare and even cause physical harm [3]. Another well-known example is the Mirai botnet in 2016, which launched a distributed denial-of-service attack to interrupt Internet services using insecure IoT devices like cameras [4]. Therefore, security is crucial for IoT applications in order to help prevent cybercrime.

Device authentication is the first defense for any wireless network, which prevents unauthorized devices from gaining access to the network and confidential resources. Current authentication solutions rely on symmetric or public key-based cryptographic algorithms and challenge-response protocols. Nevertheless, they are

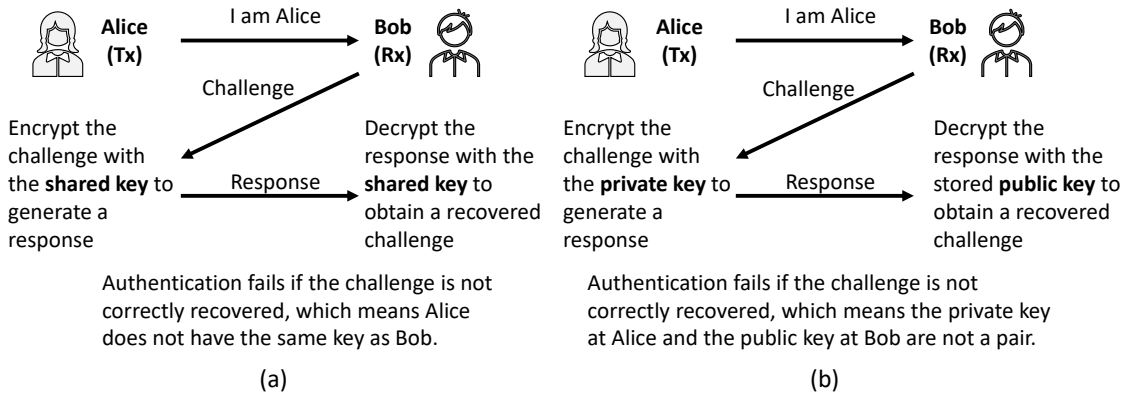


Figure 1.1: Typical authentication schemes based on cryptographic algorithms and challenge-response protocols. (a) Symmetric key-based approach. (b) Public key-based approach.

still vulnerable to a variety of risks. Examples of current authentication schemes are shown in Fig. 1.1. In the symmetric key-based approach, transmitter (Alice) and receiver (Bob)[1] have a shared key that can be used for encryption and decryption. Bob sends a challenge to Alice after receiving the authentication request. Then Alice encrypts the challenge with the shared key and responds to Bob. If Bob can use his key to decrypt the response correctly, this means Alice has the same key as Bob and the authentication succeeds. The symmetric key-based authentication is fast and easy to implement. However, it requires a shared key agreed upon between Alice and Bob, while the key distribution process can be insecure.

The public key-based authentication method addresses certain limitations that are present in the symmetric key-based approach. In the public key-based authentication process as illustrated in Fig. 1.1(b), Alice creates a pair of keys, one of which is made public and shared on a server while the other remains private. Bob sends a challenge to Alice, which Alice then encrypts using her private key. Bob then uses the public key, which was previously published by Alice, to decrypt the message. If the challenge is not properly decrypted, it indicates that the private key held by Alice does not match the public key held by Bob and the authentication process fails.

[1]In cryptographic systems and protocols, the two parties that exchange messages are named Alice and Bob, respectively.

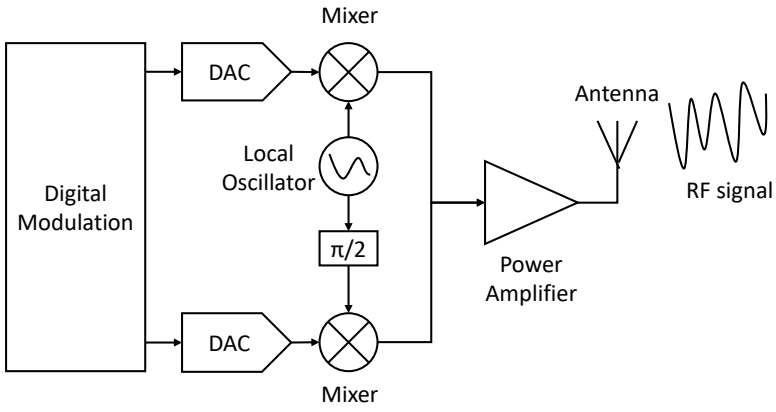


Figure 1.2: The analog front-end of a typical wireless transmitter.

In contrast to the symmetric key-based approach, Alice does not need to exchange the private key and therefore the key leakage is prevented. However, the public key-based approach requires complicated mathematical operations that can heavily increase the power consumption of IoT devices. Moreover, some IoT devices even cannot operate public key-based algorithms because of limited computing resources. In summary, it is difficult for a symmetric key-based authentication system to ensure the security of the key distribution and management process, while the public key-based method is not suitable for IoT applications due to its complexity. The strength of cryptographic authentication approaches is highly related to the security of keys, but it is difficult to distribute keys securely in a large IoT network due to the huge population of devices. Therefore, a lightweight, low-cost and reliable authentication mechanism is needed to enhance wireless security.

Radio frequency fingerprint identification (RFFI), also known as specific emitter identification (SEI) or radiometric identification, is a potential physical layer authentication method suitable for low-cost wireless devices. In contrast to the cryptographic authentication solutions that verify what the transmitters have, i.e., keys, RFFI systems perform identification by validating the physical attributes of the transmitters. As shown in Fig. 1.2, the radio frequency (RF) signals are generated by the analog front-end of wireless transmitters, which consists of mixers, oscillators, power amplifiers, antennas, etc. These components inevitably deviate from nomi-

nal specifications due to variations in the manufacturing process. Therefore, each wireless transmitter contains device-specific RF hardware impairments, termed radio frequency fingerprint (RFF). Similar to a biometric fingerprint, RFF is unique and hard to tamper with since it is the physical attribute of the transmitter hardware. The RF impairments cause unique distortion to the emitted RF signals, which allows us to extract RFF by analyzing the physical layer signal captured at the receiver. Then the RFFI system can uniquely identify from which device the signal is sent. All the RFFI operations are implemented at the receiver side and no modification is required at the transmitters. This unique feature makes RFFI extremely suitable for legacy and future wireless networks, as RFFI can be deployed with a specialized receiver without upgrading the transmitter hardware. In addition, RFFI does not require a dedicated packet and can piggyback on existing packets, which will not drain transmitter energy and can achieve a per-packet authentication.

The existing RFFI work can be roughly categorized into traditional handcrafted RFF-based and deep learning (DL)-based approaches. The former relies on a manually designed RFF extraction algorithm to obtain hardware features such as in-phase and quadrature (IQ) imbalance [5, 6], carrier frequency offset (CFO) [6–11], power amplifier non-linearity [12,13], beam pattern [14,15], etc. However, such schemes are highly dependent on the quality of the designed feature extraction algorithms and require a deep understanding of the adopted communication protocol. Apart from this, the hardware characteristics are interrelated with each other, making it challenging to extract each feature individually. In recent years, RFFI has benefited greatly from the fast development of deep learning techniques. DL-based RFFI systems leverage a neural network (NN) to process physical layer signals and directly infer device identity without the need for feature engineering, which has attracted wide attention. A variety of advanced NNs are employed to enhance RFFI performance, including convolutional neural network (CNN) [16–37], long short-term memory (LSTM) [19,23,30,38,39], and multiple layer perceptron (MLP) [16,19,23], gated recurrent unit (GRU) models [23,40], generative adversarial network (GAN) [23,41–43], transformer [40, 44, 45], etc. DL is progressively becoming the dominant technology in RFFI because of its excellent performance. However, there are still a number of

challenges to overcome in DL-RFFI, which will be discussed in detail in the following section.

Recent studies have demonstrated that RFFI has the capability to secure a variety of daily-life communication systems such as WiFi [5, 18, 24, 46–49], Zigbee [11, 35, 50–56], LoRa [19, 21, 22, 38, 40, 57], long-term evolution (LTE) [58], radio frequency identification (RFID) [31, 59], near-field communication (NFC) [60], Bluetooth [61], etc. Among them, LoRa is particularly suitable for investigating RFFI design methodologies and is therefore used as a case study in this thesis. The decision to use LoRa as a primary example is based on two factors. Firstly, LoRa devices are manufactured with low-cost components and therefore have abundant hardware impairments, which are suitable for RFFI. Moreover, the commercial LoRaWAN specification defines cryptography-based device authentication schemes. The root keys are assigned to the end nodes during fabrication and secure storage of them is a huge challenge. The proposed RFFI can provide an alternative method to authenticate LoRa devices.

1.1 Problem Statement and Existing Challenges

A closed-set[2] RFFI system aims to classify K IoT end nodes, i.e. device under test (DUT), in a wireless network. As shown in Fig. 1.3, the input to the RFFI system is the received physical layer baseband signal $y(t)$, which is mathematically given as

$$y(t) = \mathcal{G}\Big(h(\tau, t) * \mathcal{F}^k(x(t))\Big) + n(t) \quad \text{for} \quad k = 1, 2, \cdots, K, \tag{1.1}$$

where $x(t)$ is the ideal modulated signal without distortion. $\mathcal{G}(\cdot)$ denotes the hardware effects of the receiver, $h(\tau, t)$ is the time-varying wireless channel impulse response, i.e., the delay spread function with respect to delay τ and time t. $\mathcal{F}^k(\cdot)$ represents the transmitter chain effect of DUT k, $n(t)$ is the additive white Gaussian noise (AWGN) and $*$ denotes the convolution operation. The goal of an RFFI

[2]The closed-set RFFI is first introduced because recent RFFI studies are often in a closed-set setting. The discussion about closed-set and openset RFFI is detailed in Chapter 4.

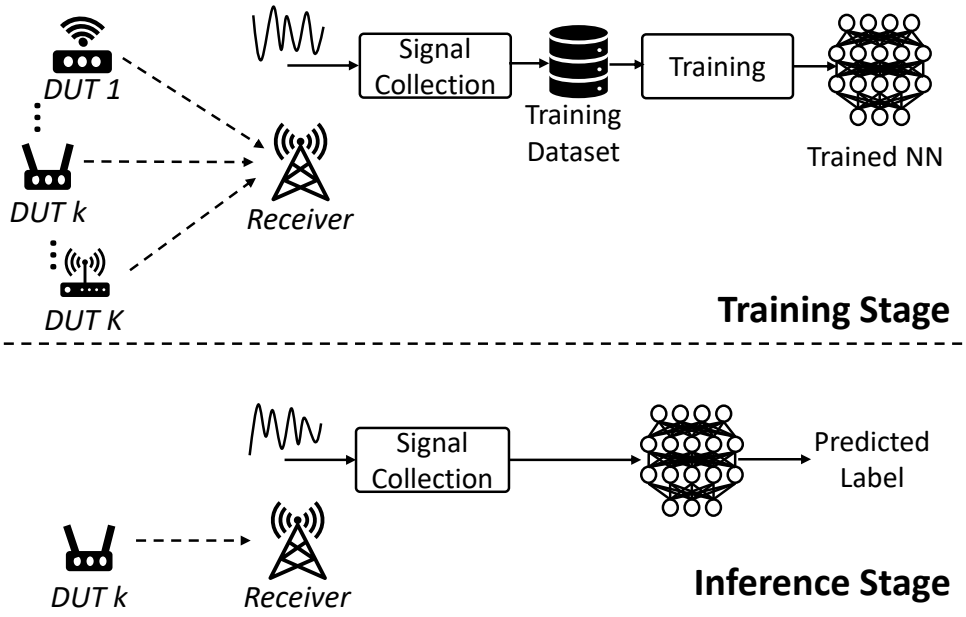


Figure 1.3: A typical DL-based closed-set RFFI system.

system is to predict the transmitter label k by analyzing the collected signal, $y(t)$. The DL-based RFFI system first trains the NN to approximate a mapping function between the received signal $y(t)$ and device label k, namely the training stage. Then the trained NN can act as a classifier to classify the newly received signals, thus achieving identification functionality.

There are currently obstacles to the development of DL-based RFFI. First, the distortion caused by the transmitter chain $\mathcal{F}^k(\cdot)$ is extremely minor; otherwise, the quality of the communication will be compromised. It is hard for the NN to extract discriminative RFFs directly from the time-domain signal $y(t)$. Second, as demonstrated in (1.1), $y(t)$ is affected by the channel impulse response $h(\tau, t)$. The NN will likely make inaccurate predictions when $h(\tau, t)$ deviates from that in the training data since the fundamental independent and identically distributed (i.i.d.) assumption of deep learning is violated. In fact, $h(\tau, t)$ can frequently change since many wireless devices are designed to be mobile. The RFFI system can not be used to identify mobile wireless devices without applying appropriate channel mitigation solutions. Finally, using a new receiver for signal reception can change $\mathcal{G}(\cdot)$ and

further affect the characteristics of $y(t)$, which breaks the i.i.d. assumption as well. However, wireless devices are likely to be served by different gateways/access points because of their mobility, thus assuming the same receiver is used during training and inference is unreasonable. Therefore, designing an RFFI protocol that can accurately identify wireless devices and is robust to location and receiver changes is still challenging.

Additionally, the RFFI system is also required to be capable of identifying devices that are not present during training, namely openset identification. This is crucial since the rogue devices that the RFFI system aims to defend are never accessible in the training stage. In the closed-set setting depicted in Fig. 1.3, the signal from a rogue device will always be labeled as legitimate since the NN can only give predictions from DUT 1-K, i.e., legitimate DUTs used in the training stage, and cannot detect the presence of a rogue device. It is, therefore, necessary to upgrade the RFFI protocol from closed-set to openset.

1.2 Contributions and Thesis Outline

This thesis takes LoRa as a case study to address the above-discussed research challenges. The main reason for choosing LoRa is that LoRa end nodes are low-cost and cannot run complicated cryptographic-based authentication algorithms. In contrast, RFFI is completely deployed at the gateway and does not require any modifications to the end node, which is suitable for LoRa end nodes with limited computing power. The LoRa-RFFI testbed is constructed using commercial off-the-shelf (COTS) LoRa development boards as DUTs to be identified and software defined radio (SDR) boards as LoRa receivers to capture the physical layer signal. Note that the conclusions and designed RFFI protocols in this thesis are applicable to other communication techniques and are not specific to LoRa. Chapter 2 presents a literature review and some background knowledge, including the LoRa and LoRaWAN, the LoRa signal collection program running at the receiver, and the procedure of a conventional DL-based RFFI system.

Chapters 3, 4, 5 are the research contributions of this thesis. The three chap-

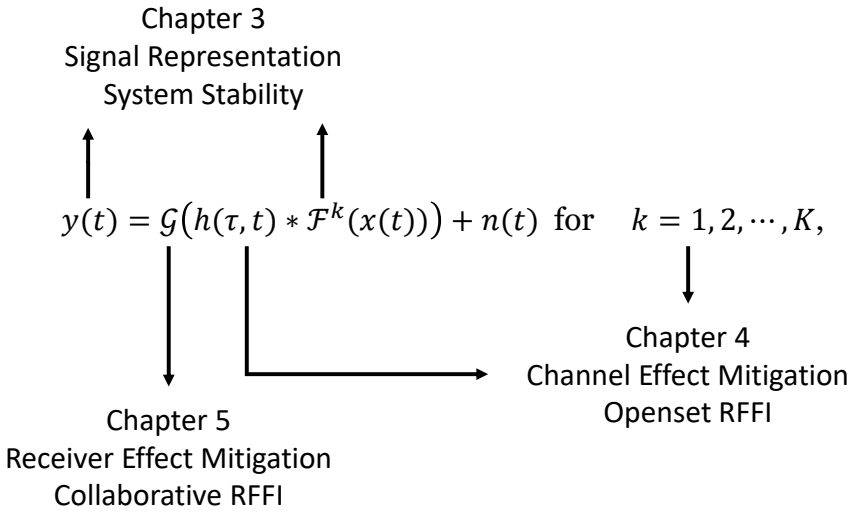


Figure 1.4: Summary of technical chapters.

ters are on the basis of equation (1.1). A summary of technical chapters is shown in Fig. 1.4. Chapter 3 investigates the stability of the hardware distortion $\mathcal{F}^k(\cdot)$ and the characteristics of the modulated signal $x(t)$ in (1.1). The content in this chapter is based on the work published in IEEE International Conference on Computer Communications (INFOCOM) [21] and IEEE Journal on Selected Areas in Communications (JSAC) [19]. The contributions are listed as follows.

- The distortion $\mathcal{F}^k(\cdot)$ is often not evident in the captured time-domain signal. Therefore, we study how to transform $y(t)$ into appropriate signal representations to make the distortion $\mathcal{F}^k(\cdot)$ easier to extract. The design of signal representation should consider the characteristics of $x(t)$. For instance, the captured LoRa signal $y(t)$ is converted to the time-frequency domain representation, i.e., spectrogram, since its frequency changes linearly over time.

- Different varieties of NNs are suited for processing different signal representations due to the diversity of their architectures and fundamental components. We present examples demonstrating how to select and construct NNs in consideration of the characteristics of signal representation, e.g., CNN is efficient to process image-like spectrograms.

- The experimental results demonstrate that $\mathcal{F}^k(\cdot)$ is not stable over time. That is, the hardware characteristics may deviate from those during training and thus degrades inference accuracy. It is experimentally found that the oscillator frequency of low-cost IoT devices is not consistent with time, which is the main reason for the drift of $\mathcal{F}^k(\cdot)$. We show that integrating CFO compensation into the RFFI system can effectively mitigate performance degradation and thus improve system robustness.

- Although the CFO should be compensated for RFFI stability, it is also a hardware attribute that can be used to distinguish across wireless devices. Integrating this into the RFFI system can potentially increase identification accuracy. For this reason, we propose a hybrid identification method that calibrates the output of NN using a pre-built CFO database. The hybrid method can significantly enhance identification performance while maintaining system stability.

Chapter 4 mainly focuses on mitigating the impact of channel variations in RFFI, i.e., the channel impulse response $h(\tau, t)$ in (1.1), as well as the design of openset RFFI systems. The work has been published in IEEE Transactions on Information Forensics and Security (TIFS) [22]. The contributions are detailed as follows:

- An openset RFFI protocol is proposed in this chapter. We leverage deep metric learning to train an NN-based feature extractor, also known as an RFF extractor, instead of a classification NN. Then an RFF database containing RFFs of legitimate devices is created. In this approach, the RFFI system can identify signals sent by devices that do not exist during training, thus achieving rogue device detection functionality and system scalability.

- We design a signal representation for LoRa signal named channel independent spectrogram. It is less susceptible to the variations of wireless channel $h(\tau, t)$ and the transmitter distortion $\mathcal{F}^k(\cdot)$ is still detectable. This is achieved by signal processing in the time-frequency domain.

- Data augmentation is shown to be effective in improving the channel robustness of the RFFI system. The well-designed wireless channel simulator can emulate

numerous channel impulse responses $h(\tau, t)$. Feeding the received signal $y(t)$ into the channel simulator can assist generate more training data influenced by various channel conditions $h(\tau, t)$. The NN trained with the augmented data generalizes well to the unseen environments because similar channel conditions are likely already present in the training dataset.

Chapter 5 focuses on reducing the impact of receiver distortion $\mathcal{G}(\cdot)$ on RFFI performance. On this basis, we also propose a collaborative RFFI protocol that employs multiple receivers to enhance identification performance. This chapter is based on a journal paper that is still under review [62]. The detailed contributions are as follows.

- It is experimentally confirmed that the variation of the receiver effect $\mathcal{G}(\cdot)$ can degrade RFFI performance. First, using a new receiver that is different from the training one can greatly change $\mathcal{G}(\cdot)$. In addition, the drift in hardware characteristics of some low-cost receivers can also cause changes in $\mathcal{G}(\cdot)$, which reduces RFFI performance. Therefore, mitigating the receiver effect is necessary.

- We propose a training strategy to obtain a receiver-agnostic NN. In the training procedure, the NN is guided to learn receiver-independent features, i.e., features not related to $\mathcal{G}(\cdot)$. The trained receiver-agnostic NN can be deployed on any receiver without significantly reducing identification performance.

- The broadcast nature of wireless signals enables us to design a collaborative RFFI protocol. The receivers in the communication network are equipped with the receiver-agnostic NN to perform independent inference on the captured signal. The predictions are then fused in a central server to achieve higher identification accuracy.

Chapter 6 finally concludes the thesis and discusses future work.

Note that the RFFI solutions/protocols proposed in the thesis can be directly applied to other wireless technologies except for the signal representation of spectrogram/channel-independent spectrogram. For example, the openset RFFI protocol discussed in

Chapter 4 is applicable to any wireless technology. Although the spectrogram/channel-independent spectrogram is designed specifically for LoRa modulation, we believe the design concept can be transferred to other wireless protocols.

Chapter 2

Literature Review and Background

2.1 Literature Review

RFFI is historically used for military purposes to identify radar signals [63–68], and it has recently demonstrated the ability to secure a variety of daily-life communication technologies, including WiFi [5,18,24,46–49], Zigbee [11,35,50–56], LoRa [19,21,22, 38,40,57,69], LTE [58], RFID [31,59], NFC [60], Bluetooth [61], etc. Moreover, some studies also utilize RFFI to identify aircraft [70–73], satellites [74, 75], unmanned aerial vehicle (UAV) [28] and ships [29]. Although DL-RFFI shows great potential for enhancing wireless security, it is still in a rapid development stage and there are many obstacles that need to be addressed.

2.1.1 Stability of RFFI Systems

As a device identification system, stability is fundamentally required for RFFI. Robyns *et al.* indicate that the accuracy of the designed DL-RFFI system drops over time and inferred that this is probably due to the frequency drift of the oscillator [16]. However, they do not provide in-depth analysis or a mitigation method. Andrews *et al.* experimentally examined the effect of temperature variation on different analog components, e.g., oscillator, power amplifier, phase-locked loop, mixer, etc., and concluded that the oscillator is particularly sensitive to temperature fluc-

tuations [76]. Even within 15 minutes, it is observed that low-cost ZigBee devices experienced considerable CFO variations [11, 25]. Nonetheless, there are currently no comprehensive studies on the effect of CFO drift on the performance of DL-RFFI systems.

2.1.2 Signal Representation in RFFI Systems

The signal distortion caused by the transmitter impairments is minor and difficult to observe in the time domain. Therefore, the received IQ samples are often converted to more discriminative signal representations to enhance RFFI performance. There are numerous signal representations designed in previous studies. Merchant *et al.* calculated the error signal by subtracting the ideal signal from the received one [32]. He *et al.* leveraged signal processing techniques to decompose the received signals [39]. Gong *et al.* extracted the grey histogram of bispectrum to enhance individual discriminability [77]. Other signal representations include power spectral density [78–80], Hilbert-Huang spectrum [81, 82], differential constellation trace figure (DCTF), etc. Specific to LoRa-RFFI, Robyns *et al.* performed fast Fourier transform (FFT) on the received signal to make the CFO more evident [16]. Das *et al.* directly used IQ samples as the system input [38]. Jiang *et al.* converted the received LoRa signals to specially-designed differential constellation trace figures [83]. However, none of them have considered the unique characteristics of LoRa modulation, which may limit the identification performance. Therefore, a signal presentation specially designed for LoRa signals is required.

2.1.3 Openset RFFI Systems

Many previous DL-RFFI schemes are designed as close-set, making them lack system scalability and rogue device detection capability [19, 20, 25, 84]. More specifically, a close-set RFFI system neither supports efficient device joining and leaving nor can it distinguish rogue devices from legitimate ones. This is because previous methods usually rely on the softmax layer for classification, whereby once the training is completed, the number of neurons in this output layer cannot be changed [45]. When

a new device joins or an old device leaves, the NN should be updated by retraining, which is not practical and time-consuming. Chen *et al.* used transfer learning to speed up the retraining process, but it still requires the RFFI system to have a GPU [85]. Furthermore, the softmax layer-based systems can only output the label presented in the training set, but rogue devices are never available during training. In this case, the rogue devices will always be classified as legitimate devices which is unacceptable. To overcome these limitations, researchers have started to consider RFFI as an open-set problem rather than a close-set one. Xie *et al.* proposed a similarity-based RFFI system, which is the most relevant work to ours [35]. They trained an RFF extractor using the softmax-based loss. Then the cosine similarity and a threshold are leveraged for the back-end authentication task. The similarity-based open-set solution supports efficient device joining without high complexity. Hanna *et al.* evaluated a number of open-set classifiers, including autoencoder, OpenMax, One Vs All (OvA), etc [86]. They concluded that the OvA classifier can reach the best performance. Gritsenko *et al.* proposed a novel device detection scheme that leveraged the probabilistic characteristics of classification NNs [87]. The device is labeled as a new one when the confidence level during classification is low. Soltani *et al.* further extended this scheme to the case of multiple classifiers [28]. The GAN is also used for open-set RFFI problem [23]. Before feeding the signal to a typical classification NN, a GAN is additionally introduced to determine whether the device is a rogue or not.

2.1.4 Channel Effects in RFFI Systems

Previous work demonstrates that DL-based RFFI is not robust to wireless channels if no specific solution is applied [18]. Sankhe *et al.* proposed the ORACLE system to combat channel effects [84, 88]. However, it needs to intentionally introduce impairments to the transmitter, which is costly and not suitable for IoT applications. Morin *et al.* indicated that the training dataset should contain as many channel conditions as possible to make the NN automatically learn how to mitigate it [89, 90]. However, this may dramatically increase the cost of collecting the training set. Data

augmentation is an effective alternative to this approach, in which we can collect the training set in a static scenario and pass it into a well-designed channel simulator to generate signals under various channel conditions [24, 27, 30, 31, 52]. However, designing an accurate channel simulator that matches the real application scenarios is challenging.

2.1.5 Receiver Effects in RFFI Systems

A major challenge for RFFI is that the received signal not only contains the characteristics of the transmit chain but is also affected by the receiver chain. The changes in receiver hardware characteristics can seriously affect RFFI performance, but their impacts have been usually overlooked in previous studies. Most existing RFFI works assume that the same receiver is used during training and inference and that the receiver characteristics do not change over time [16, 18, 19, 21, 30, 38]. However, this assumption does not always hold in practical IoT applications. For instance, mobile IoT devices will be served by different access points/gateways, depending on their coverage. Furthermore, even if the same receiver is used, the hardware characteristics of low-cost receivers may vary over time. To the best knowledge of the author, there have been few studies investigating the receiver effects. Zhang *et al.* [20] demonstrated how the change of receiver characteristics affects the RFFI performance, but the work is mostly simulation-based and does not present a countermeasure. Merchant *et al.* [51] undertook experiments using high-end receivers and observed the performance degradation caused by the receiver effect. However, the low-end receivers were not investigated. Elmaghbub *et al.* [33] experimentally revealed that using different receivers during training and inference degrades system performance, but no solutions were designed. As will be experimentally demonstrated in Section 5.4, both changing a new receiver for inference and the drift of receiver features over time can seriously degrade RFFI performance. There is an urgent need for a receiver-agnostic RFFI system that can be deployed in a highly practical manner.

2.1.6 Collaborative RFFI Protocols

As wireless transmissions are broadcast and can be captured by any receivers within range, it is, therefore, possible to design a collaborative RFFI protocol that can enhance system performance. In IoT applications, multiple receivers can be present with numerous gateways in LoRaWAN and multiple access points in WiFi enterprise networks, but critically, to the best of our knowledge, there are only two papers that have explored using multiple receivers in RFFI systems [39, 91]. Andrews *et al.* [91] have investigated how to combine the observations from multiple antennas and compared three combination methods, but it is based on traditional frequency features and is not available in deep learning-based RFFI systems. He *et al.* [39] employed a support vector machine (SVM), MLP, and LSTM to fuse the extracted decomposed features and compared the fusion performance. However, it was mainly based on simulation with limited experimental results. A collaborative RFFI scheme for deep learning-based approaches needs to be designed and experimentally evaluated.

2.2 LoRa and LoRaWAN Background

LoRa is a physical layer modulation technique patented by Semtech in 2014 [92], which has been widely used for long-range IoT applications. LoRaWAN defines the upper networking layer protocol and is managed by the open association LoRa Alliance. This section presents the background knowledge of LoRa and LoRaWAN.

2.2.1 LoRaWAN

LoRaWAN defines a star-of-stars network topology, which is shown in Fig. 2.1. It allows one LoRa end node to establish wireless communication links (dashed lines) with multiple gateways at the same time. The LoRa gateways are connected to a server by using, e.g., WiFi, cellular communications or Ethernet (solid lines). They relay the captured messages to the network server for collaborative decoding. The server runs network and application layer protocols.

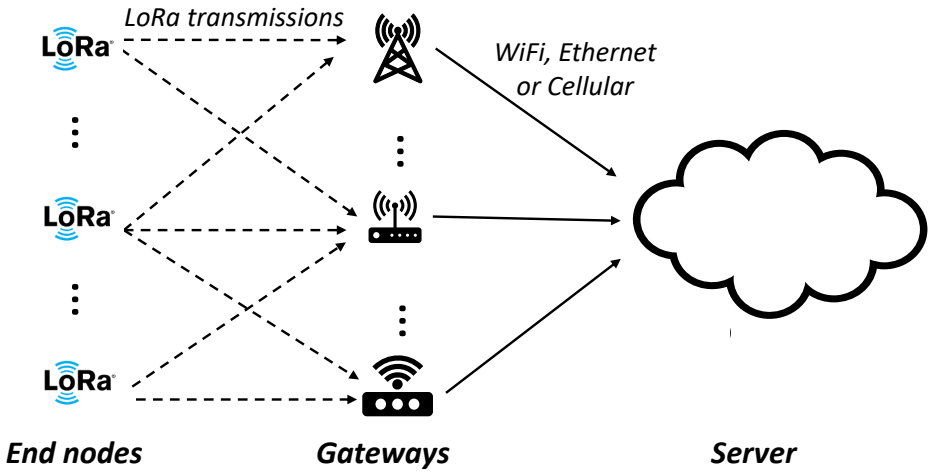


Figure 2.1: LoRaWAN network topology.

2.2.2 LoRa Signal Modelling

LoRa is developed from chirp spread spectrum (CSS) modulation, which uses frequency-varying chirp signals for communication. A baseband linear chirp, i.e. an unmodulated LoRa symbol, is denoted as $c(t)$ and mathematically given as

$$c(t) = \Lambda e^{j(-\pi BWt + \pi \frac{BW}{T_{sym}} t^2)}, \tag{2.1}$$

where Λ, BW, and T_{sym} are amplitude, bandwidth, and symbol time duration, respectively. In LoRa modulation, the symbol time T_{sym} is adjustable by modifying the spreading factor SF and bandwidth BW, given as

$$T_{sym} = \frac{2^{SF}}{BW}. \tag{2.2}$$

The LoRa standard usually specifies several repeated baseband chirps at the beginning of a packet as the preamble part [93], denoted as $x(t)$.[1] Note that $x(t)$ is the same for all the LoRa packets. The signal $x(t)$ is affected by the wireless channel before it reaches the receiver. The received baseband signal is denoted as $y(t)$. Then

[1]Note that $x(t)$ only refers to the packet preamble part since the payload information is not leveraged in this thesis.

an analog-to-digital converter (ADC) is applied for sampling and converts $y(t)$ to the discrete form, given as

$$y(t) \xrightarrow{ADC} y[nT_s], \tag{2.3}$$

where T_s is the ADC sampling interval.[2] Similarly, $x[nT_s]$ refers to the discrete form of the transmitted baseband signal $x(t)$. For convenience, $x[n]$ and $y[n]$ are used for denotation, or $\mathbf{x}$ and $\mathbf{y}$ in vector form. Note that $y[n]$ is a vector composed of complex numbers, i.e., IQ samples. Its real and imaginary parts are named the I and Q branches, respectively. The length of $y[n]$, L_{pre}, is given as

$$L_{pre} = 8 \cdot L_{sym} = 8 \cdot \frac{T_{sym}}{T_s}, \tag{2.4}$$

where L_{sym} is the symbol length. L_{pre} is eight times L_{sym} because the packet preamble part contains eight[3] unmodulated LoRa symbols. Fig. 2.2(a) shows the I-branch of $y[n]$ and Fig. 2.2(b) is the zoom of the first preamble.

2.2.3 Short-time Fourier Transform

The LoRa signal is usually analyzed in the time-frequency domain because of its non-stationary property. Short-time Fourier transform (STFT) is an efficient time-frequency analysis algorithm that can reveal the time-frequency features of the signal. The discrete STFT is mathematically written as

$$\begin{aligned} S_{a,b} &= \sum_{a=0}^{A-1} y[n] w[n - bL_{hop}] e^{-j2\pi \frac{a}{A} n} \\ &\text{for } a = 1, 2, ..., A \text{ and } b = 1, 2, ..., B, \end{aligned} \tag{2.5}$$

[2]Although the sampling offset can affect the received waveform, the signals of various offsets have been collected into the training dataset and the NN can learn how to deal with it.

[3]The number of symbols in the preamble is changeable. We use the default setting of eight in this thesis.

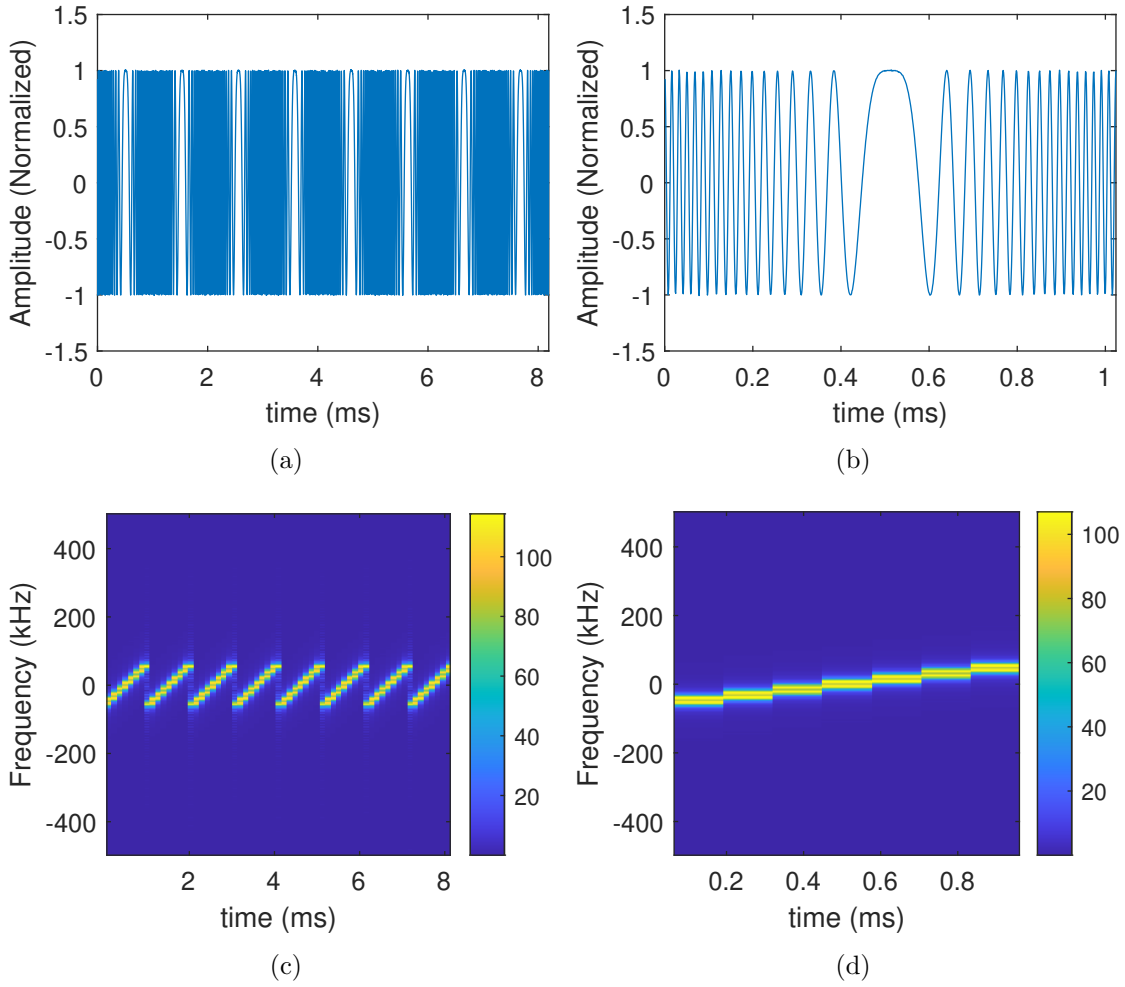


Figure 2.2: Preamble part of a LoRa packet. (a) Time domain signal of all the preambles (I branch). (b) Time domain signal of the first preamble (I branch). (c) Spectrogram of all the preambles. (d) Spectrogram of the first preamble.

where $S_{a,b}$ is the element of the STFT complex matrix $\mathbf{S}$. B is number of columns of $\mathbf{S}$, given as

$$B = \frac{8 \cdot \frac{2^{SF}}{BW} \cdot \frac{1}{T_s} - A}{L_{hop}} + 1. \tag{2.6}$$

A is the number of rows of $\mathbf{S}$, which is also the length of window function $w[n]$. L_{hop} is the hop size. Note that BW and SF are parameters in LoRa and can only be

selected from several integers, i.e., BW can be 125 kHz, 250 kHz, 500 kHz, and SF is from seven to 12. The sampling frequency, i.e., $\frac{1}{T_s}$, is set to an integer multiple of BW. L_{hop} should be carefully selected to make B an integer. The spectrogram is then represented in dB scale, $\widetilde{\mathbf{S}}$, with each element being

$$\widetilde{S}_{a,b} = 10\log_{10}(|S_{a,b}|^2), \forall a, b, \tag{2.7}$$

where $|\cdot|$ returns the amplitude of the input element. The spectrogram of the LoRa preamble part $\mathbf{y}$ is shown in Fig. 2.2(c) and Fig. 2.2(d) is the zoom of the first one.

Note that the phase of the channel frequency response is not as stable as the amplitude. It is susceptible to phase noise and carrier frequency offset, therefore it is noisy and sensitive to temperature variations. In addition, the phase can also be affected by the sampling time offset at the receiver side, making it more difficult to be leveraged. Therefore, we did not use phase information in the thesis.

2.3 LoRa Signal Collection

The LoRa signal collection program is detailed in this section, which consists of synchronization, CFO compensation and normalization.

2.3.1 Synchronization

Packet Detection

The packet detection algorithm is first performed to roughly detect the presence of LoRa packets and coarsely locate the packet start point. The well-known Schmidl-Cox algorithm is exploited, which is based on the repeating property of the preambles [94]. A detection metric $M[n]$ is calculated from the received signal $y_{rcv}[n]$, which is expressed as

$$M[n] = \frac{\left|\sum_{i=0}^{L_{sym}-1} y_{rcv}[n+i] \cdot y^*_{rcv}[n+i+L_{sym}]\right|}{\sum_{i=0}^{L_{sym}-1} y_{rcv}[n+i+L_{sym}] \cdot y^*_{rcv}[n+i+L_{sym}]}, \tag{2.8}$$

where $y^*_{rcv}[n]$ denotes the conjugate of $y_{rcv}[n]$. $M[n]$ changes from low to high when the LoRa packet arrives. We compare it with a predefined threshold λ_{det}, the packet is detected when $M[n] > \lambda_{det}$. λ_{det} is often a fixed number between 0.5 and 0.8 in a practical wireless system and is set to 0.7 in our signal collection program.

Fine Synchronization

The method described in the previous subsection can only detect the arrival of a LoRa packet but not the accurate starting point [95]. A more precise synchronization algorithm is essential to locate the exact starting point of the packet. Robyns *et al.* [95] proposed an effective fine synchronization algorithm according to the unique characteristics of LoRa signals. First, an ideal baseband preamble $x_{lo}[n]$ is locally generated and its instantaneous frequency $f_{lo}[n]$ is calculated as

$$f_{lo}[n] = \frac{1}{2\pi T_s}(\angle x_{lo}[n+1] - \angle x_{lo}[n]) \tag{2.9}$$

where $\angle(\cdot)$ returns the angle. Then we calculate the instantaneous frequency $f_{rcv}[n]$ of the received synchronized signal $y_{rcv}[n]$, given as

$$f_{rcv}[n] = \frac{1}{2\pi T_s}(\angle y_{rcv}[n+1] - \angle y_{rcv}[n]). \tag{2.10}$$

Finally, a sliding-window cross-correlation is performed between $f_{rcv}[n]$ and $f_{lo}[n]$. The index of the maximum value is chosen as the index of the accurate starting point, Ψ, which is given as

$$\Psi = \arg\max_i \Big(\sum_{n=0}^{L_{pre}-1} f_{rcv}[n] \cdot f_{lo}[n+i] \Big). \tag{2.11}$$

The segment before Ψ is discarded to obtain the synchronized signal $y'_{rcv}[n]$.

2.3.2 CFO Compensation

Coarse Compensation

We first calculate the instantaneous frequency $f'_{rcv}[n]$ of the synchronized signal $y'_{rcv}[n]$, given as

$$f'_{rcv}[n] = \frac{1}{2\pi T_s}(\angle y'_{rcv}[n+1] - \angle y'_{rcv}[n]). \tag{2.12}$$

As $y'_{rcv}[n]$ is composed of eight unmodulated linear chirps, the graph of $f'_{rcv}[n]$ is eight repeated straight lines, with each expressing as

$$f_{sym}[n] = -\frac{BW}{2} + \frac{BW}{T_{sym}} + \Delta f, \tag{2.13}$$

where Δf is the CFO to compensate. Thanks to the linearity of $f_{sym}[n]$ and repeatability of $f'_{rcv}[n]$, the CFO can be coarsely estimated by calculating the mean value of $f'_{rcv}[n]$. The estimated CFO $\widehat{f}_{coarse}$ is given as

$$\Delta\widehat{f}_{coarse} = \frac{1}{L_{pre}} \sum_{n=0}^{L_{pre}-1} f'_{rcv}[n]. \tag{2.14}$$

Then $y'_{rcv}[n]$ can be compensated using the estimated $\widehat{f}_{coarse}$, given as

$$y''_{rcv}[n] = y'_{rcv}[n] \cdot e^{-j2\pi\Delta\widehat{f}_{coarse}nT_s}, \tag{2.15}$$

where $y''_{rcv}[n]$ is the coarsely compensated signal.

Fine Compensation

We further employ a fine CFO estimation algorithm since the coarse compensation is not accurate enough. The residual CFO, $\widehat{f}_{fine}$, can be estimated based on the repeating property of preambles, given as

$$\Delta\widehat{f}_{fine} = -\frac{1}{2\pi T_s L_{sym}}\angle\Big(\sum_{n=0}^{L-1} y''_{rcv}[n] \cdot y''^{*}_{rcv}[n+L_{sym}]\Big). \tag{2.16}$$

The received signal can be further finely compensated as

$$y'''_{rcv}[n] = y''_{rcv}[n] \cdot e^{-j2\pi\Delta\widehat{f}_{fine}nT_s}.l \tag{2.17}$$

As the phase can only be resolved in $[-\pi, \pi]$, the range of CFO that can be estimated by (2.16) is

$$|\Delta\widehat{f}_{fine}| < \frac{\pi}{2\pi T_s L_{sym}} = \frac{BW}{2^{SF+1}}. \tag{2.18}$$

When the LoRa transmission is configured with SF=7 and BW=125 kHz, the estimation capability is within ±488.3 Hz. Commonly, the oscillator drift of LoRa devices is ±10 ppm [96], approximately 8.68 kHz for the 868 MHz carrier frequency, which is much higher than 488.3 Hz. Hence, the coarse CFO estimation should be employed first to limit the residual offset before the fine CFO estimation. After the coarse and fine CFO estimation, the overall estimated CFO, $\Delta\widehat{f}$, can be represented as

$$\Delta\widehat{f} = \Delta\widehat{f}_{coarse} + \Delta\widehat{f}_{fine}. \tag{2.19}$$

Simulation is carried out to evaluate the performance of CFO compensation using MATLAB. Numerous baseband LoRa preambles $x_{sim}[n]$ with CFO Δf_{sim} are generated. Then the coarse CFO estimation is performed and the residual CFO can be calculated by $\Delta f_{sim} - \Delta\widehat{f}_{coarse}$. Similarly, the fine estimated CFO, $\Delta\widehat{f}_{fine}$, can be obtained by (2.16) and the residual CFO after fine compensation can be calculated by $\Delta f_{sim} - \Delta\widehat{f}_{coarse} - \Delta\widehat{f}_{fine}$. We ran the simulation 10,000 times with Δf_{sim} uniformly distributed between -10,000 Hz and +10,000 Hz. The signal-to-noise ratio (SNR) was 20 dB.[4] As the histograms shown in Fig. 2.3, the residual CFO after coarse and fine compensation is between 5 Hz to 20 Hz and between -1 Hz to +1 Hz, respectively. This is accurate enough for RFFI applications.

[4]In the simulation, the signal amplitude is set to one and the noise level is accordingly computed to reach an SNR of 20 dB.

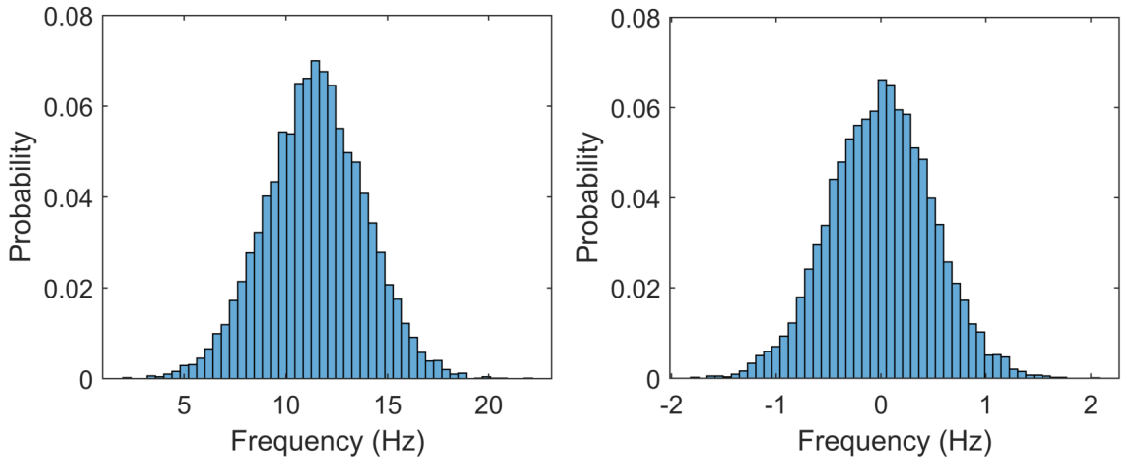


Figure 2.3: Histograms of residual CFO. (a) Residual CFO after coarse compensation. (b) Residual CFO after fine compensation.

2.3.3 Normalization

Normalization is a technique used in RFFI to prevent the system from using received signal strength to identify devices. The compensated signal $y'''_{rcv}[n]$ is normalized by dividing its root mean square (RMS) value, which is mathematically given as

$$y[n] = \frac{y'''_{rcv}[n]}{\sqrt{\frac{1}{L_{pre}} \sum_{n=0}^{L_{pre}-1} |y'''_{rcv}[n]|}}, \tag{2.20}$$

where $y[n]$ is the collected signal as the input to the RFFI system.

2.4 Conventional Deep Learning RFFI System

As shown in Fig. 1.3, a conventional DL-based RFFI system[5] comprises two stages, namely training and inference. In the training stage, the receiver first collects signals from the K end nodes operating in the IoT network. The signal collection procedure

[5]In this thesis, a conventional RFFI refers to a typical closed-set DL-RFFI system.

is elaborated in Section 2.3. The collected signals are stored as a training dataset, $\mathcal{D}^{train}$, given as

$$\mathcal{D}^{train} = \{(\mathbf{y}_m, \mathbf{p}_m)\}_{m=1}^{M_{train}}, \tag{2.21}$$

where $\mathbf{y}_m$ is the m^{th} training sample and $\mathbf{p}_m$ is the corresponding one-hot encoded DUT label, given as

$$\mathbf{p}_m = O(\ell_m), \tag{2.22}$$

where $O(\cdot)$ denotes one-hot encoding operation, ℓ_m is the ground truth DUT label of the m^{th} training sample, and M_{train} is the number of training samples. After building the training dataset, we define a neural network $f(\cdot;\Theta)$ and optimize its parameters Θ using $\mathcal{D}^{train}$ as defined below

$$\Theta = \underset{\Theta}{\operatorname{argmin}} \frac{1}{M_{train}} \sum_{(\mathbf{y},\mathbf{p})\in\mathcal{D}^{train}} \mathcal{L}_{ce}(f(\mathbf{y};\Theta), \mathbf{p}), \tag{2.23}$$

where $\mathcal{L}_{ce}(\cdot)$ is the cross-entropy loss. In the inference stage, the receiver captures a signal $\mathbf{y}'$ and feeds it into the well-trained neural network $f(\cdot;\Theta)$ for prediction. A probability vector $\hat{\mathbf{p}}$ is obtained via inference and is mathematically defined as

$$\hat{\mathbf{p}} = f(\mathbf{y}';\Theta), \tag{2.24}$$

where $\hat{\mathbf{p}} = \{\hat{p}_1, ..., \hat{p}_k, ..., \hat{p}_K\}$ is a probability vector over all the K DUTs, and $\hat{p}_k$ is the estimated probability for the k^{th} DUT. The predicted transmitter label, $\hat{\ell}$, is derived by simply selecting the index of the element with the highest probability as defined below

$$\hat{\ell} = \underset{k}{\operatorname{argmax}}(\hat{\mathbf{p}}). \tag{2.25}$$

Chapter 3

Towards Stable and Efficient RFFI

3.1 Introduction

This chapter aims to explore the stability of the DL-driven LoRa-RFFI system and enhance the identification performance by designing appropriate signal representations and deep learning models. First, stability is critical for any authentication system but the DL-RFFI is experimentally shown to be unstable over time. Previous studies demonstrate that the oscillator frequency is sensitive to temperature variations [76], but its effect on DL-RFFI systems has not been comprehensively investigated. Second, LoRa modulation employs the linear chirp signal to encode information. Considering the unique frequency-varying property of chirp signals, appropriate signal representations can be designed to enhance LoRa-RFFI performance. Additionally, the most suitable NN type for the designed signal representation should be investigated for future design reference.

This chapter is motivated to address the above research challenges by designing a DL-based RFFI system to classify LoRa devices. We carry out an in-depth investigation and extensive experiments involving 25 LoRa DUTs and a universal software radio peripheral (USRP) N210 SDR platform. The technical contributions of this chapter are summarized as follows.

- We carry out extensive experiments over seven months to measure the CFO

variation and demonstrate that CFO drifts over time and degrades the RFFI stability. A bespoke setup is established by connecting each LoRa DUT and the USRP platform with an attenuator to eliminate the multipath and Doppler effects. CFO compensation is demonstrated to be effective mitigation for RFFI stability, which can improve the classification accuracy from 83.53% to 95.35% for the spectrogram-CNN[1] model.

- We investigate three signal representations for LoRa signals, namely IQ samples, FFT results and spectrograms, corresponding to the time domain, frequency domain and time-frequency domain analysis, respectively. Then we build MLP, CNN and LSTM models to evaluate each signal representation. Experimental results show that the spectrogram-CNN can reach the highest accuracy of 95.35% with the least system complexity and the shortest training time.

- A hybrid classifier is designed, which uses a pre-built CFO database to calibrate the output of NNs. This enables the CFO to contribute to the device identification without compromising system stability. It is experimentally validated that the designed hybrid classifier can significantly improve the classification accuracy, namely from 58.26% to 82.81% in the best case for the FFT-LSTM model.

The rest of the chapter is organized as follows. Section 3.2 gives an overview of the designed RFFI system, and Section 3.3 further details it. Section 3.4 experimentally demonstrates CFO drift and its effect on the RFFI system and Section 3.5 evaluates the performance of the proposed RFFI systems in a real wireless environment. Section 3.6 concludes the chapter.

[1]Spectrogram-CNN means using the spectrogram as signal representation and CNN as the DL model. Similar descriptions are used throughout the chapter.

3.2 System Overview

The architecture of the proposed RFFI system is shown in Fig. 3.1. In the training stage, numerous packets are collected from legitimate DUTs with each packet correctly labeled. The preamble parts of these packets are then pre-processed, including synchronization, CFO compensation, normalization, and conversion to the selected signal representation, i.e., IQ samples, FFT results, and spectrogram. In the pre-processing stage, a CFO database is created, which records the CFO of legitimate DUTs. After the pre-processing, the training data is used to train the NN. Once the training is completed, the newly received packet preamble can be converted to the designed signal representation and fed into a hybrid classifier. The deep learning model first makes a prediction, and then the CFO database is used for calibration. The output of the hybrid classifier is the predicted device label.

3.3 System Design

3.3.1 Signal Collection

The LoRa signal collection program is detailed in Section 2.3, which consists of synchronization, CFO compensation, and normalization. The synchronization detects the signal arrival and locates the packet start point. Then the CFO is estimated and compensated to improve system stability. Normalization is finally applied to prevent the system from learning the received signal strength.

3.3.2 CFO Database Creation

A CFO database is created during training for use in the subsequent hybrid classification. The database records the maximum and minimum CFO for each DUT in the training dataset denoted as $\Delta\widehat{f}^k_{max}$ and $\Delta\widehat{f}^k_{min}$, respectively. In the hybrid classifier, the CFO database $\{(\Delta\widehat{f}^k_{max}, \Delta\widehat{f}^k_{min})\}_{k=1}^K$ is used to calibrate the softmax output of DL models. The calibration algorithm is detailed in Section 3.3.5.

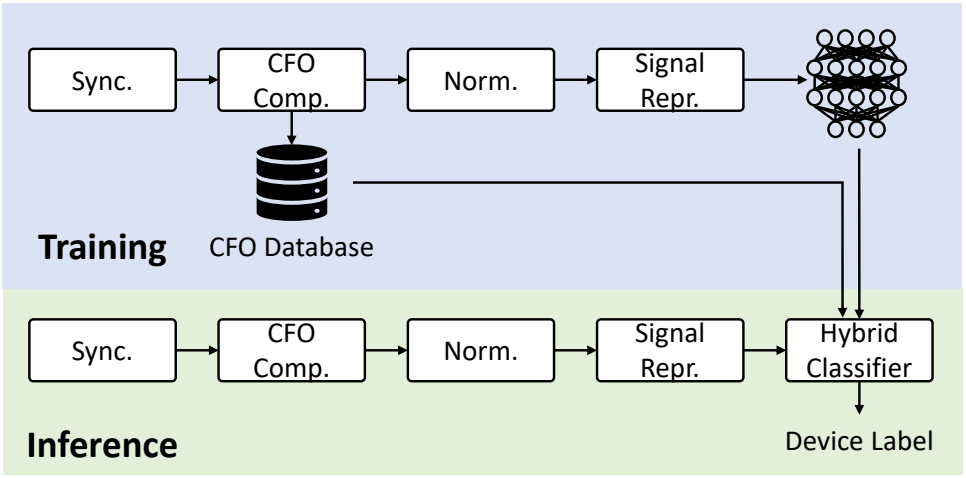


Figure 3.1: A DL-based RFFI system. CFO compensation is implemented to prevent the system performance from declining over time.

3.3.3 Signal Representation

The received IQ samples can be directly fed into the DL model. They can also be converted to other signal representations such as spectrograms, FFT results, etc. This helps reveal the underlying signal characteristics, making it easier for the NN to learn.

IQ Samples: IQ samples, consisting of complex numbers from the I and Q branches, represent the time-domain signals which are captured from the receiver chain directly. Some previous studies directly employ IQ samples as system inputs [18, 25, 32, 84, 88]. However, this possibly makes the training of DL models difficult and time-consuming because some signal characteristics are not obvious in the time domain.

FFT Results: FFT transforms the signal into the frequency domain, making some features easier to observe. For example, the CFO causes a phase difference in the time domain, which manifests as a more easily observed spectrum shift in the frequency domain [16].

Spectrogram: The spectrogram is generated by STFT, which is detailed in Section 2.2. It converts the signal into the time-frequency domain, which not only

provides information about frequency components but also demonstrates how they change over time. It is useful for analyzing non-stationary wireless signals, including the chirps employed by LoRa.

3.3.4 Deep Learning Models

DL models can automatically extract features from signal representations without the need for feature engineering. Considering the characteristics of IQ samples, FFT results, and spectrogram, three well-known DL models are investigated, namely MLP, CNN, and LSTM. The preamble part of LoRa packets used in our experiments contains 8,192 IQ samples. The FFT result also consists of 8,192 complex numbers. As the DL model cannot process complex numbers, we split the I and Q branches of IQ data, and the amplitude and phase of FFT result as two independent dimensions. Therefore, the input dimension of DL models designed for IQ/FFT data is 2×8192. Unlike IQ and FFT data, the spectrogram can be considered a 2D image, resulting in different input dimensions. We generate the spectrogram with a rectangular window of length 256 and hop size 128, which leads to a 256×63 spectrogram. We further crop it to a smaller size of 102×63 as the top and bottom parts contain nearly no useful information.

Fig. 3.2 demonstrates the proposed NNs, which are developed from well-known DL models. For instance, the CNN for spectrogram is developed from LeNet but its structure is further experimentally optimized to adapt to our applications [97].

Convolutional Neural Network: CNN was popular in recent years thanks to its excellent performance in image recognition and computer vision [98]. CNN is usually composed of convolutional layers, fully connected layers, and some pooling layers. CNN is designed for image-like inputs. Among the three signal representations introduced in Section 3.3.3, the spectrogram can be considered as a 2D image and thus can be processed by CNN.

The architecture of CNN for spectrogram is illustrated in Fig. 3.2(a). It consists of three 2D convolutional layers with 8, 16, and 32 3×3 filters, respectively. Each convolutional layer is followed by a batch normalization (BN) layer and activated

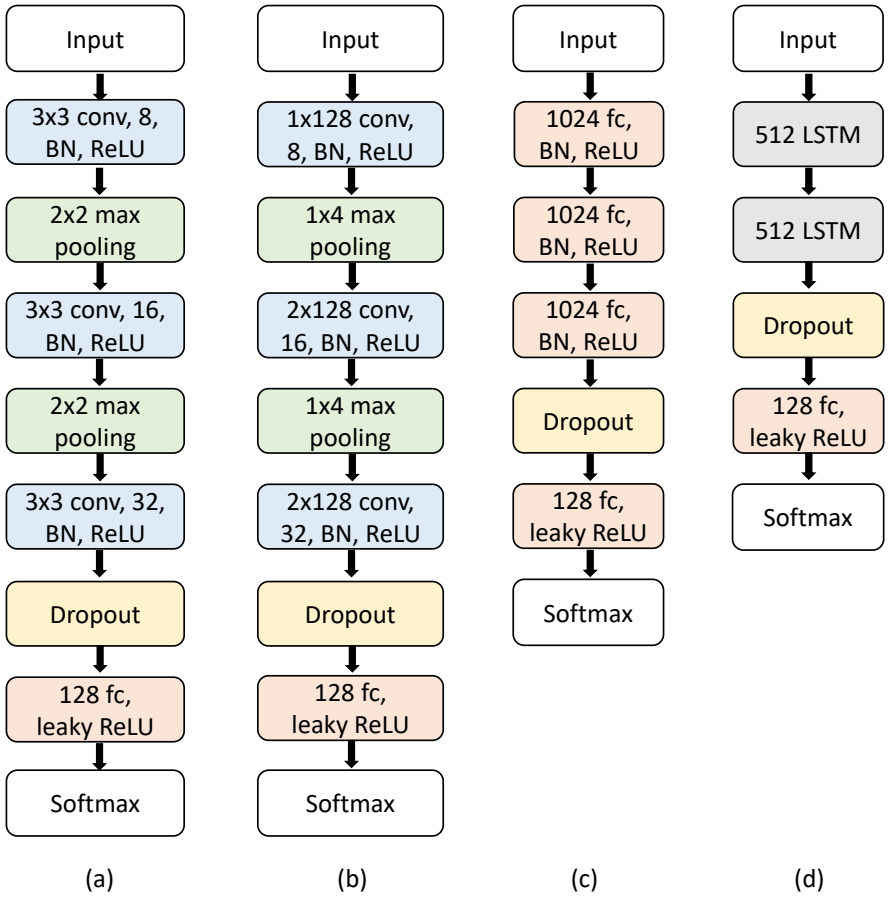


Figure 3.2: Architectures of DL models. (a) CNN for spectrograms. (b) CNN for IQ/FFT data. (c) MLP for IQ/FFT data and spectrograms. (d) LSTM for IQ/FFT data and spectrograms.

by the rectified linear unit (ReLU) function. There are two maxpooling layers of size 2×2 following the first and second convolutional layers. The last convolutional layer is connected to a fully connected layer activated by leaky ReLU, which is mathematically given as

$$f(x) = \begin{cases} x, & x \geq 0 \\ \text{scale} \cdot x, & x < 0 \end{cases}, \tag{3.1}$$

where the scale is set to 0.01 in this chapter. Dropout is adopted before fully connected layers and padding is used in each convolutional layer.

The architecture of CNN for IQ/FFT is illustrated in Fig. 3.2(b), which consists of three 2D convolutional layers and one fully connected layer. The three convolutional layers are composed of 8, 16, and 32 filters with filter sizes of 1×128, 2×128, and 2×128, respectively. There are two 1×4 max pooling layers following the first and second convolutional layers. Other settings are the same as the CNN designed for spectrogram. Note that deeper CNNs with residual connections can enhance the performance, which will be discussed in Chapter 4.

Multilayer Perceptron: MLP is the forerunner of CNN and LSTM, which entirely consists of fully connected layers. The number of learnable parameters is huge since all the neurons are fully connected in MLP, which can result in redundancy and increased complexity. MLP has no preferred input data type.

The MLPs for spectrogram and IQ/FFT have the same network architecture except for the input dimension. The architecture is shown in Fig. 3.2(c). It is implemented with four fully connected layers. The first three layers have 1024 neurons with ReLU activation and the last one has 128 neurons with leaky ReLU activation. Dropout is adopted after the third fully connected layer.

Long Short-Term Memory Network: LSTM is a kind of recurrent neural network that is designed for temporal sequences such as speech and acoustic signals [99]. It is efficient in capturing the temporal correlation in the input sequences. In the three signal representations, the IQ samples and the spectrogram are time-dependent, therefore LSTM can be used for processing.

Similar to the MLP, the LSTMs designed for spectrogram and IQ/FFT have the same network architecture except for the input dimension. We use two LSTM layers with 512 units, tanh is adopted as the activation function. Then we add one fully connected layer with leaky ReLU activation after the second LSTM layer. Dropout is used after the second LSTM layer.

Softmax Output: In classification problems, the softmax function is usually employed in the last layer of a DL model to map the outputs to a list of probabilities

$\hat{\mathbf{p}} = \{\hat{p}_1, \cdots, \hat{p}_k, \cdots, \hat{p}_K\}$ over all the classes, which is mathematically given as

$$\hat{p}_k = \frac{e^{z_k}}{\sum_{k=1}^{K} e^{z_k}}, \tag{3.2}$$

where z_k is the output of the k^{th} neuron before the softmax activation. K is the number of DUTs. The straightforward strategy to make predictions is to select the class with the highest probability as the final predicted label, which is used by most of the DL-based RFFI systems.

Training Settings: The DL models are trained with the same parameters. We select adaptive moment estimation (Adam) as the optimizer and the initial learning rate is set to 0.0003. The learning rate drops every ten epochs with a drop factor of 0.3. The mini-batch size is set to 32 and the L2 regularization factor is 0.0001. The training stops when the maximum epochs are reached, i.e., 60 epochs. The training and validation loss plateaus when they reached the maximum epochs so that all the DL models can be considered fully trained. All the networks are implemented with the MATLAB DL Toolbox[2] and trained on the same PC with a graphics processing unit (GPU) of NVIDIA GeForce GTX 1660.

3.3.5 Hybrid Classifier

Although the CFO of the received IQ samples should be compensated before feeding into the NN due to system stability, it is still a discriminative feature that can aid in device identification. Therefore, we integrate the CFO-based identification into the RFFI system, which is achieved by calibrating the NN output using CFO information. In the inference stage, we first estimate the CFO of the received signal, $\Delta\widehat{f}$. Then we can calibrate each element of the DL model output, $\hat{\mathbf{p}} = \{\hat{p}^1, \hat{p}^2, \cdots, \hat{p}^K\}$, using $\Delta\hat{f}$ and the CFO database $\{(\Delta\widehat{f}^k_{max}, \Delta\widehat{f}^k_{min})\}_{k=1}^{K}$. The calibration process is formulated as

$$\begin{cases} \hat{p}^k = 0, when\ \Delta\widehat{f} > \Delta\widehat{f}^k_{max}\ or\ \Delta\widehat{f} < \Delta\hat{f}^k_{min}, \\ \widehat{p}^k = \widehat{p}^k, when\ \Delta\hat{f}^k_{min} < \Delta\widehat{f} < \Delta\widehat{f}^k_{max}. \end{cases} \tag{3.3}$$

[2]https://mathworks.com/products/deep-learning.html

Table 3.1: LoRa DUTs.

DUT Index	Model	Chipset
1 - 5	SX1272MB2xAS mbed shield[3]	SX1272
6 - 10	SX1261MB2xAS mbed shield[4]	SX1261
11 - 15	Pycom FiPy[5]	SX1272
16 - 20	Pycom LoPy[6]	SX1276
21 - 25	Dragino SX1276 shield[7]	SX1276

The above calibration is carried out for each element of the NN output $\mathbf{p}$. After the calibration, the DUT with the highest probability in $\mathbf{p}$ is selected as the final predicted label. The CFO database $\{(\Delta\widehat{f}^k_{max}, \Delta\widehat{f}^k_{min})\}_{k=1}^K$ should be updated periodically to achieve adequate identification performance.

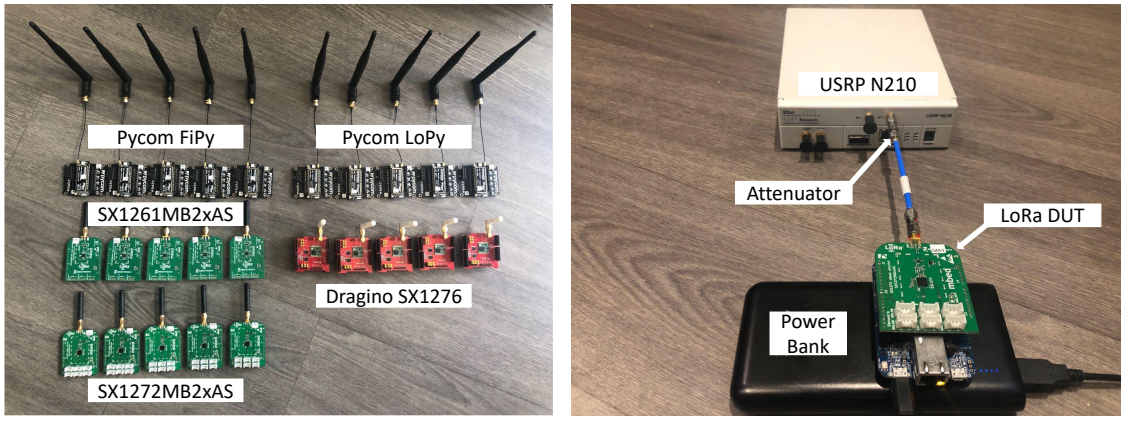
3.4 Experimental Results of CFO Drift

Stability is one of the most crucial attributes of an RFFI system. The RFFs are required to be time-invariant in the presence of environmental and time changes. In this section, we experimentally demonstrate that CFO drifts over time and should be compensated to prevent system performance from deteriorating.

3.4.1 Experimental Setup

Ten LoRa DUT of two models were employed, namely five SX1272MB2xAS and five SX126xMB2xAS mbed shields, as listed in Table 3.1 and shown in Fig. 3.3(a). The LoRa DUTs were configured with $SF = 7$, bandwidth $BW = 125$ kHz, and the carrier frequency was 868.1 MHz. The receiver was a USRP N210 SDR with a

[3]https://os.mbed.com/components/SX1272MB2xAS/
[4]https://os.mbed.com/components/SX126xMB2xAS/
[5]https://pycom.io/product/fipy/
[6]https://pycom.io/product/lopy4/
[7]https://www.dragino.com/products/lora/item/102-lora-shield



(a) (b)

Figure 3.3: Experimental devices and setup. (a) LoRa DUTs. (b) The LoRa DUT and the USRP receiver are connected by a 40 dB attenuator.

sampling rate of 1 MS/s. We used the Communications Toolbox Support Package for USRP Radio of MATLAB to configure the USRP N210 SDR and access the received data[8]. To eliminate channel effects and focus on CFO variations, we created a bespoke setup connecting the LoRa DUT to the USRP N210 receiver through a 40 dB attenuator, which is shown in Fig. 3.3(b).

3.4.2 CFO Drift Observation

To evaluate the long-term drift of CFO, we carried out extensive experiments spanning seven months, namely April and September, October, and November 2020. The five SX1272MB2xAS LoRa DUTs were tested as a case study. We collected 3,000 LoRa packets per day from each DUT, and each collection lasted about one hour.

The CFO of each packet is estimated and presented in Fig. 3.4. The CFO decreased over the first 20 minutes and then remained relatively constant. This is reasonable because the temperature gradually increases after the DUT is powered on, i.e., self-heating, and the oscillator is sensitive to temperature variations [76]. It is also observed in Fig. 3.4 that there is a non-negligible and unpredictable CFO

[8]https://mathworks.com/help/supportpkg/usrpradio/

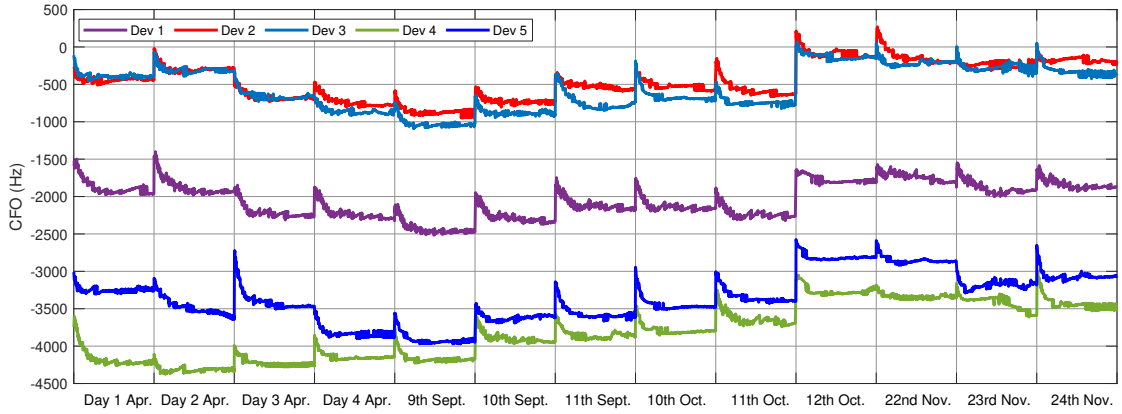


Figure 3.4: CFO drift over seven months. The experiments were carried out in April, September, October, and November 2020. The data collection in April for each device is not completed on a single day and thus cannot be labelled with a specific date.

change over seven months. The drift is probably caused by uncontrollable environmental conditions such as room temperature. However, the CFO remained relatively consistent over several days, likely due to the receiver and DUTs being in a stable environment with stable room temperature.

In DL studies, the training and test data are often required to have the same data distributions, i.e., the i.i.d. assumption. Otherwise, the trained NN performs poorly on the test data. The data distribution of the received IQ samples cannot be visualized directly because it contains thousands of dimensions. The t-distributed stochastic neighbor embedding (t-SNE) is a well-known dimensionality reduction technique to visualize high-dimensional data, which has been used for data visualization in a wide range of applications including RFFI [16]. Hence, we can use t-SNE to visualize the IQ data collected on different days and observe the performance of CFO compensation. The results are given in Fig. 3.5. We use the data of DUT 1 as an example. Each point represents one LoRa packet after dimension reduction, and the colors of the points denote the day on which the packet was collected. From Fig. 3.5(a), the IQ data collected on different days gather as several clusters when CFO was not compensated. After CFO compensation, the IQ data collected on

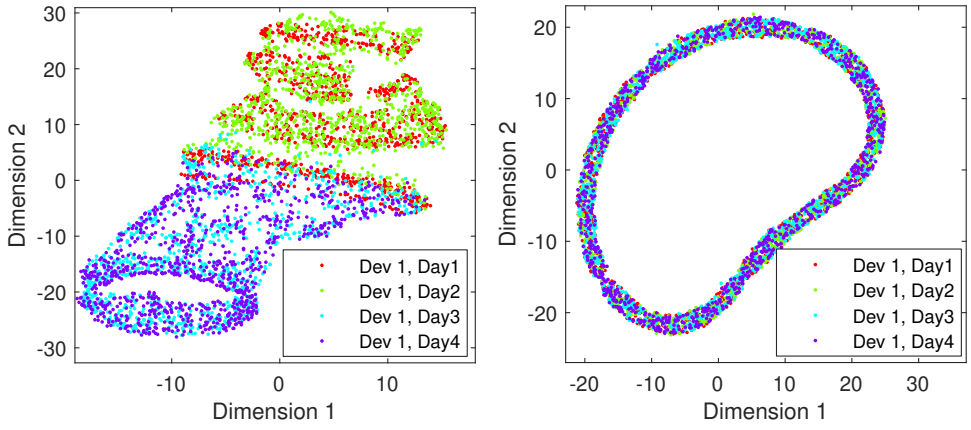


Figure 3.5: t-SNE visualization of IQ samples collected from DUT 1. (a) Without CFO compensation. (b) With CFO compensation.

different days were mixed and cannot be separated, as shown in Fig. 3.5(b). This is required for RFFI since the distribution of data collected from the same device should be time-invariant. Therefore, CFO compensation is a necessary process in RFFI systems for stability.

3.4.3 The Effect of CFO Drift on RFFI

In addition to the five SX1272MB2xAS LoRa DUTs used in the previous subsection, we also collected data from five SX1261MB2xAS LoRa DUTs. The data collected from these ten DUTs on four days were used to evaluate the effect of CFO drift on RFFI. The spectrogram was selected as the signal representation and the CNN model in Fig. 3.2(a) was employed. The CNN was trained with the first 1,000 packets of each DUT ($1,000 \times 10$ packets in total) from the Day 1 dataset, among which 90% was randomly selected for training and the rest 10% was for validation. Then we used another 1,000 packets of each DUT from the Day 1 dataset to test the trained CNN. For Day 2-4 datasets, the first 1,000 packets of each device were used as the test data. This allows us to evaluate the trained CNN with signals collected on four different days.

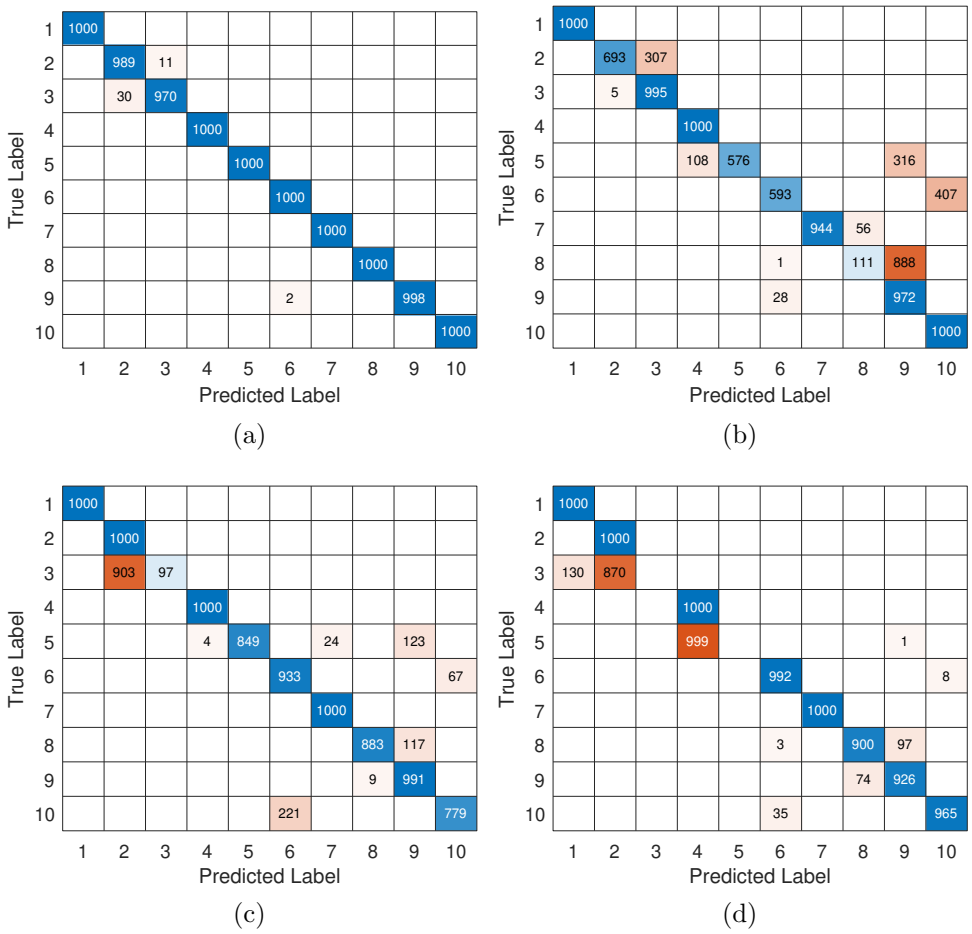
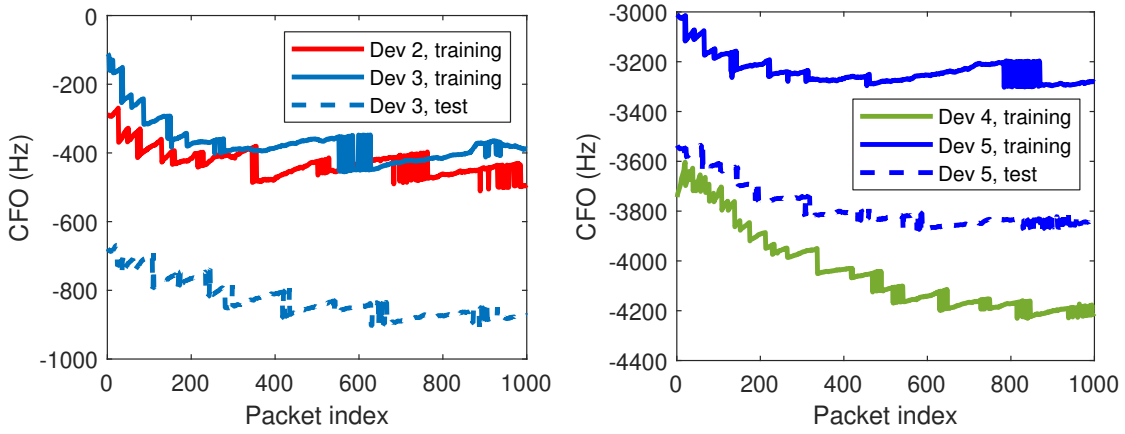


Figure 3.6: Experimental results of the spectrogram-CNN model without CFO compensation. (a) Day 1 train, Day 1 test, overall accuracy: 99.57%. (b) Day 1 train, Day 2 test, overall accuracy: 78.84%. (c) Day 1 train, Day 3 test, overall accuracy: 85.32%. (d) Day 1 train, Day 4 test, overall accuracy: 77.83%.

Fig. 3.6 shows the confusion matrices obtained by the spectrogram-CNN model when the CFO compensation was not applied. Figs. 3.6(a), (b), (c) and (d) represent the classification results when the test data was collected on Day 1, Day 2, Day 3, and Day 4, respectively. When the training and test data were collected on the same day, i.e., Fig. 3.6(a), the classification accuracy reached 99.57% which was almost no classification error. In contrast, when the training and test data were collected on



(a) (b)

Figure 3.7: The comparison of CFO between the Day 1 training data and Day 4 test data. (a) Comparison between DUT 2 and DUT 3. (b) Comparison between DUT 4 and DUT 5.

different days, i.e., Figs. 3.6(b), (c), (d), the classification results were unacceptable as several DUTs were almost completely misclassified, e.g., DUT 3 and DUT 5 in Fig. 3.6(d). As shown in Fig. 3.7(a), it can be observed that the CFO of DUT 3 drifted by hundreds of hertz from Day 1 to Day 4. The CFO of DUT 3 test data was closer to DUT 2 training data rather than DUT 3 training data. Similarly, as shown in Fig. 3.7(b), the CFO of DUT 5 test data was closer to DUT 4 training data rather than DUT 5 training data. It is inferred that CFO drift is the main reason for performance degradation and a slight drift of CFO would cause the NN to make an incorrect prediction.

Fig. 3.8 shows the classification results after CFO compensation was applied. The accuracy was always maintained above 96% on the four days. These results revealed that CNN can classify DUTs with high accuracy after CFO compensation and performance degradation is significantly mitigated.

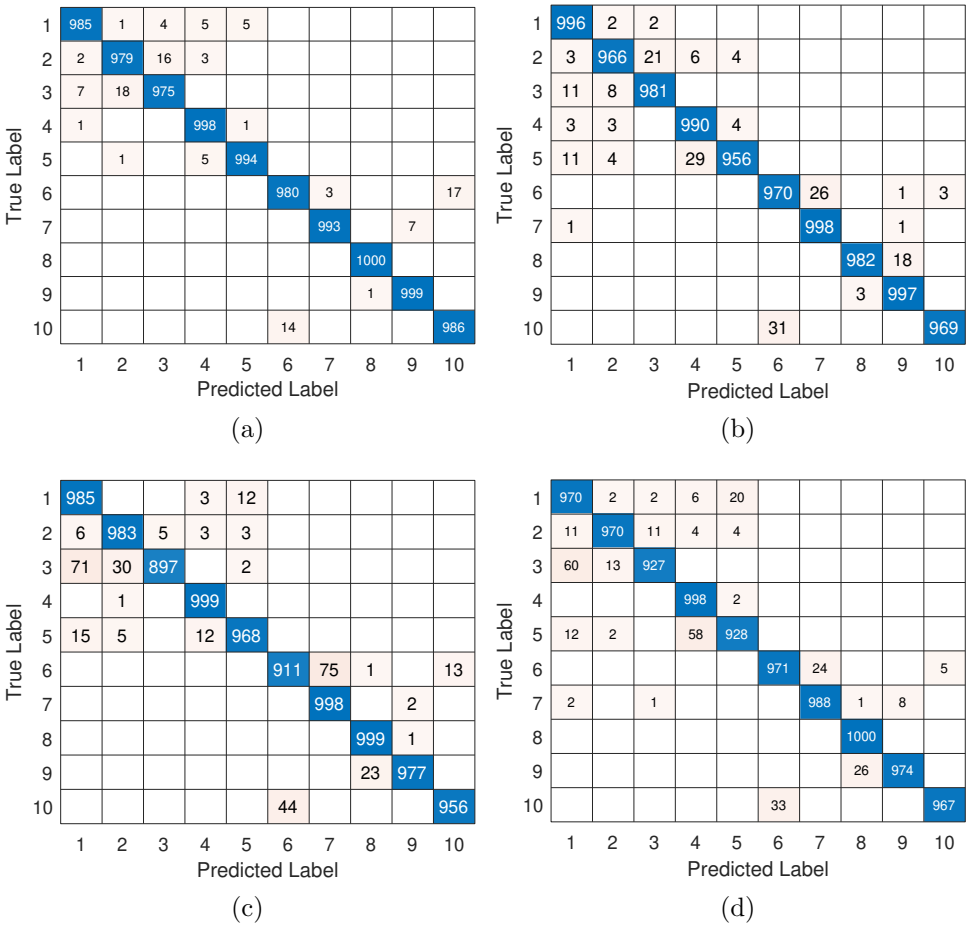


Figure 3.8: Experimental results of the spectrogram-CNN model with CFO compensation. (a) Day 1 train, Day 1 test, overall accuracy: 98.89%. (b) Day 1 train, Day 2 test, overall accuracy: 98.05%. (c) Day 1 train, Day 3 test, overall accuracy: 96.73%. (d) Day 1 train, Day 4 test, overall accuracy: 96.93%.

3.5 Experimental Evaluations in a Real Wireless Environment

To focus on the CFO effect, the LoRa DUT and USRP were connected using an attenuator in Section 3.4. However, this is not a practical application scenario. The proposed RFFI system was evaluated in a real wireless environment in this section.

Table 3.2: Classification accuracy of different models, number of parameters and required training time.

Signal Representation	DL Model	Accuracy			# of Parameters	Training Time
		w/o CFO Comp.	w/ CFO Comp.	Hybrid		
IQ samples	MLP	54.08%	55.73%	78.26%	19,018,009	25 min
	CNN	64.10%	92.26%	**98.11%**	4,361,545	75 min
	LSTM	61.16%	89.54%	95.14%	4,267,289	70 min
FFT results	MLP	55.44%	94.48%	96.17%	19,018,009	25 min
	CNN	61.14%	82.10%	85.58%	4,361,545	75 min
	LSTM	49.20%	58.26%	82.81%	4,267,289	69 min
Spectrogram	MLP	88.60%	91.82%	95.95%	8,821,017	22 min
	CNN	83.53%	**95.35%**	96.40%	**1,545,193**	20 min
	LSTM	68.16%	89.50%	98.04%	3,427,609	80 min

3.5.1 Experimental Setup

The number of LoRa DUTs is increased to 25 in this section. As shown in Table 3.1 and Fig. 3.3(a), these LoRa DUTs were from five manufacturers. The same USRP N210 SDR was used as the receiver. The LoRa DUTs and USRP were configured with the same parameters as described in Section 3.4.1.

The experiments were carried out in a typical indoor environment, with chairs and tables distributed in the room. The distance between the LoRa DUT and USRP was about three meters and there was a line-of-sight (LOS) between them. The SNR of the received signals is estimated to be over 30 dB. We collected 2,000 packets continuously from each DUT, which lasted about 15 minutes. All the DUTs were placed at the same location and the environment was kept unchanged. Therefore, the same channel condition can be assumed for all the signal transmissions.

We evaluated different signal representations and DL models. We used the first 1,000 packets of each DUT as the training data, 90% of which were randomly selected for training and the rest 10% were for validation. The second 1,000 packets of each DUT were used as the test data to evaluate the RFFI system. The experimental results are presented in Table 3.2. We analyze the results from three aspects: the impact of CFO in a wireless environment, the selection of signal representation and

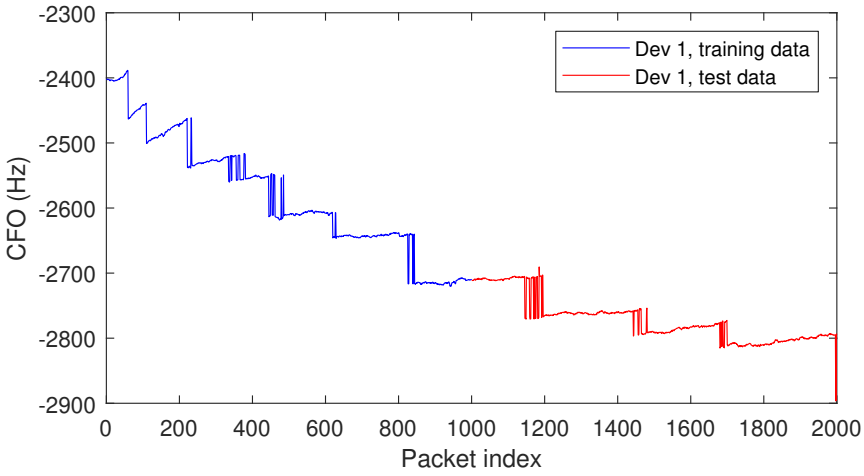


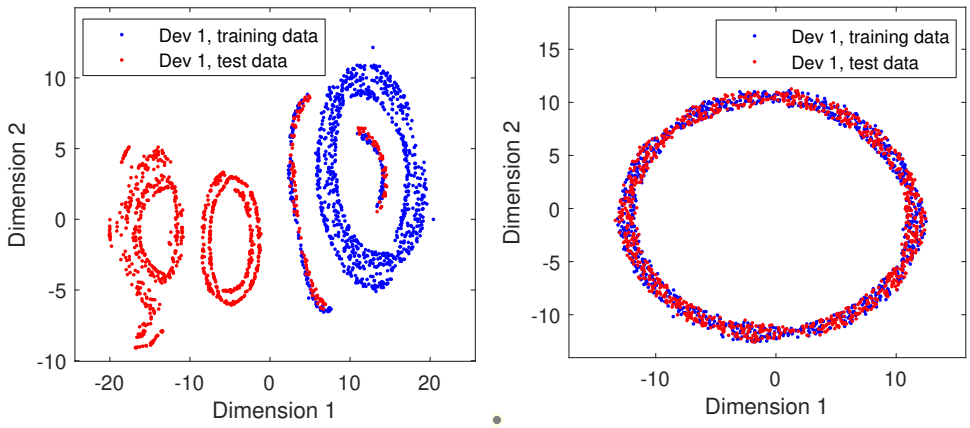
Figure 3.9: CFO of each packet in the dataset of DUT 1.

DL models, and the performance of the proposed hybrid classifier.

3.5.2 Impact of CFO Drift

We take DUT 1 as an example to illustrate the CFO variation of the collected data. Fig. 3.9 shows the CFO of each packet collected from DUT 1 and presents a similar pattern with Fig. 3.4 that the CFO decreased after the device was powered on. In the wireless experiments, we used packets 1-1,000 to train the DL model and packets 1,001-2,000 to evaluate its performance. The packets in the test set have different CFOs from those in the training set.

Similar to Section 3.4.2, we use t-SNE to visualize the training and test data and the results are shown in Fig. 3.10. There are 2,000 points and the blue points represent packets 1-1,000 used for training and the red points represent packets 1,001-2,000 used for the test. It can be observed from Fig. 3.10(a) that there are distinct clusters when there is no CFO compensation, which indicates the training and test data have different distributions. In contrast, as shown in Fig. 3.10(b), the blue and red points are mixed after CFO compensation and cannot be separated. This is expected because the RFFs of each DUT should be time-invariant after CFO



(a) (b)

Figure 3.10: t-SNE visualization of the training and test sets of DUT 1. (a) Without CFO compensation. (b) With CFO compensation.

compensation.

The classification results are shown in Table 3.2. The accuracy was not satisfying for all the signal representations and DL models when CFO compensation was not applied. Taking spectrogram-CNN as an example, the overall accuracy was only 83.53%. In contrast, the accuracy significantly increased to 95.35% after CFO compensation was applied. The confusion matrices were given in Fig. 3.11(a) and Fig. 3.11(b) which provide more detailed information.

3.5.3 Selection of Signal Representation and Deep Learning Model

A crucial step to build a DL-based RFFI system is to design an appropriate signal representation and construct a suitable DL model for it. System performance can be evaluated by three metrics: classification accuracy, system complexity, and training time.

Classification Accuracy: As shown in Table 3.2, the spectrogram-CNN model achieved the highest classification accuracy, i.e., 95.35%. The accuracy of the FFT-

MLP model reached 94.48% while the IQ-CNN model was 92.26%. In addition to this, we observed two under-performing combinations: IQ-MLP and FFT-LSTM, whose accuracy was only 55.73% and 58.26%, respectively. For IQ-MLP, the designed MLP lacks the ability to extract distinctive features from LoRa IQ samples. For FFT-LSTM, FFT data does not have time dependence and is not suitable for input to the LSTM network.

System Complexity and Training Time: System complexity and training time should also be considered. The RFFI systems are desired to have low complexity, i.e., few parameters, so that the demand on the authenticator hardware such as memory, and computing power can be reduced. Table 3.2 shows that the spectrogram-based CNN has the least amount of parameters, namely 1,545,193, and the trained CNN only takes up 5,679 kb of storage space.

The training time is another evaluation metric. As demonstrated in Table 3.2, the spectrogram-based CNN requires 20 mins for training, which is only a quarter of the spectrogram-based LSTM. When there are many DUTs running in a communication network, the training dataset can be very large, and the short training time is especially important.

Comprehensive Comparison: The spectrogram-CNN model reached the highest classification accuracy with the least complexity and training time. Although the accuracy of the FFT-MLP model reached 94.48%, only 0.87% lower than the spectrogram-CNN model, the MLP has 19,018,009 learnable parameters, almost 12 times as many as the parameters of the spectrogram-CNN model. For LoRa signals, the spectrogram-CNN model is highly recommended. The LoRa hardware imperfections can be revealed in the time-frequency domain spectrogram, and CNN is particularly efficient at extracting the features hidden in 2D spectrograms.

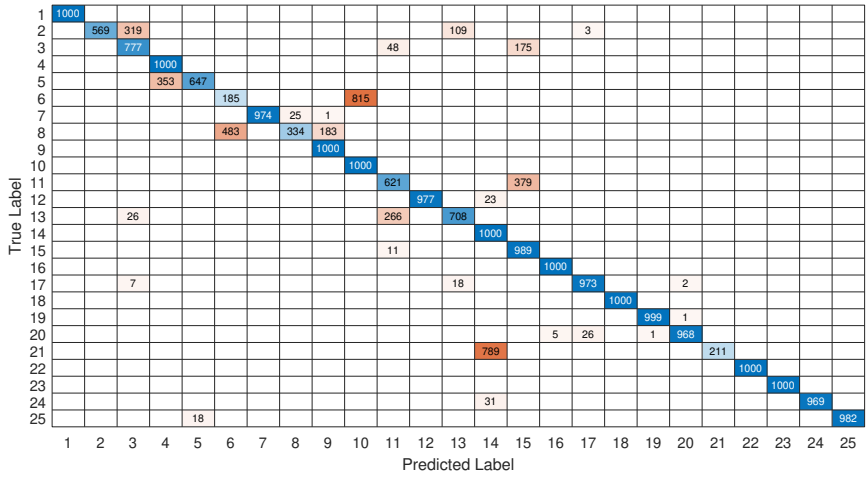
3.5.4 Performance of the Hybrid Classifier

The hybrid classifier introduced in Section 3.3.5 improves the identification accuracy by calibrating the softmax output of the DL model , and the results are given in Table 3.2. It can be observed that the hybrid classifier can increase the accuracy for

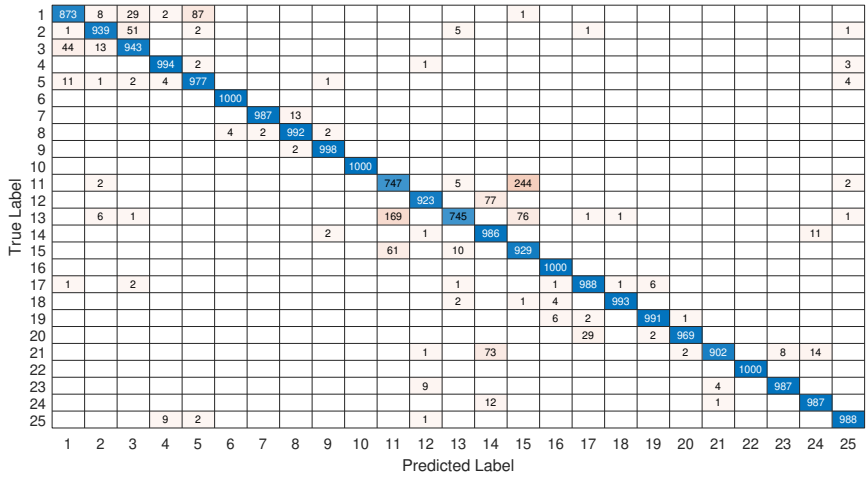
all the signal representations and DL models. The most significant improvement is the FFT-LSTM; the accuracy with the hybrid classifier is increased from 58.26% to 82.81%, which is a 24.55% improvement. The cost-effective enhancement of hybrid classification is accomplished by K more comparison steps, which consumes limited computing resources.

3.6 Conclusion

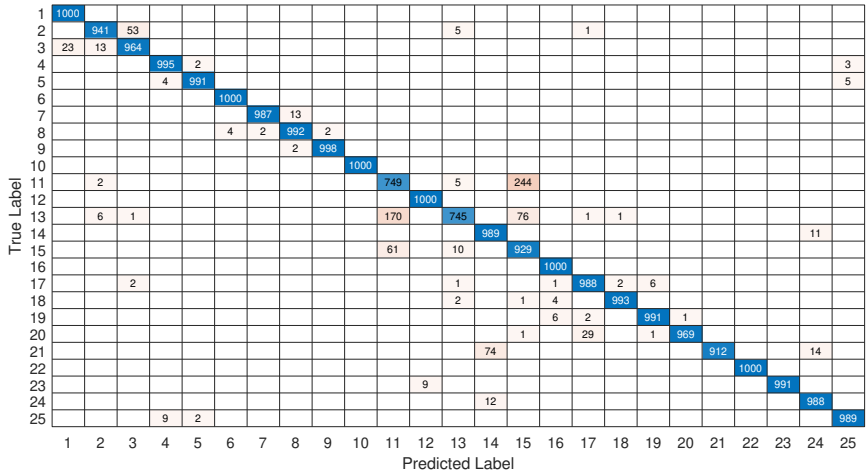
In this chapter, we proposed a DL-based RFFI scheme to classify LoRa DUTs and carry out extensive experimental evaluations involving 25 LoRa DUTs from five manufacturers and a USRP N210 SDR as the receiver. First, the CFO of LoRa DUTs was experimentally found to vary over time and compromise the system stability. The CFO compensation was demonstrated to be effective in mitigating performance degradation. Second, we investigated three signal representations, i.e., IQ samples, FFT results, and spectrograms, and built three DL models to process them, i.e., MLP, LSTM, and CNN. The spectrogram-CNN model achieved the highest classification accuracy with the least complexity and the shortest training time. Finally, a hybrid classifier was proposed to calibrate the softmax output of DL models using the estimated CFO. The range of CFO variations was found to stay relatively stable over several continuous days so it is helpful to rule out predictions when the estimated CFO deviates greatly from the reference CFO. The proposed RFFI system achieved an accuracy of 96.40% in classifying 25 LoRa DUTs in real wireless environments. In this chapter, the experiments were carried out with all the DUTs and receivers fixed at the same location so that the wireless channel can be assumed to be static. However, this setting does not well match the actual communication scenario. In the next chapter, the impact of a changing wireless channel will be experimentally evaluated and corresponding solutions will be proposed.



(a)



(b)



(c)

Figure 3.11: Classification results of the spectrogram-based CNN. (a) Without CFO compensation, overall accuracy: 83.53%. (b) With CFO compensation, overall accuracy: 95.35%. (c) With CFO compensation, hybrid classifier, overall accuracy: 96.40%.

Chapter 4

Towards Scalable and Channel-Robust RFFI

4.1 Introduction

This chapter aims to design an openset RFFI protocol that is robust to the variations of wireless channels. Previous studies often define DL-RFFI as a closed-set problem [17, 18, 100], which is however unsuitable for an authentication system. The closed-set RFFI systems typically use NNs with softmax layers for classification, which can only output labels that are present in the training stage. However, the rogue devices are never accessible during training. This implies that a rogue device is always labeled as one of the legitimate devices. In other words, the closed-set RFFI system lacks rogue device detection ability. Moreover, the classification NN needs to be retrained when a new device requests to join the communication network because the softmax layer consists of a fixed number of neurons. The retraining is time-consuming and significantly increases the deployment cost. Additionally, the wireless signal is affected by the multipath and Doppler effects. Previous work demonstrates that the RFFI system is severely affected by channel variations [18, 24]. However, most IoT devices are mobile by design, and the wireless channel changes frequently. It is therefore necessary to design an RFFI protocol that is robust to

channel variations.

The RFFI system proposed in this chapter introduces an enrollment stage and uses k-nearest neighbor (kNN) to replace the classification NN, making it scalable and able to detect rogue devices. Moreover, the channel-independent spectrogram is proposed to mitigate channel effects. Extensive experiments were carried out with 60 COTS LoRa devices of four models with various channel conditions to demonstrate the excellent performance of the proposed RFFI system. The contributions are highlighted as follows.

- A scalable RFFI framework is developed, based on a deep metric learning-powered RFF extractor, which enables device joining and leaving without the need for retraining. It maintains an RFF database by enrolling a new device using the pre-trained RFF extractor or deleting the record of a leaving device.
- A channel-robust RFFI protocol is proposed by constructing the channel-independent spectrogram and exploiting data augmentation. The channel-independent spectrogram can mitigate the channel effect in the time-frequency domain while reserving the RFF of the LoRa signal. The data augmentation is carefully designed to represent real channel conditions with both multipath and Doppler shift.
- Extensive experiments are conducted involving different LoRa devices, various channel conditions, and antenna polarization. We experimentally demonstrate that the RFF extractor can extract features from devices that are not present during training, even if they are produced by other manufacturers. The proposed channel-independent spectrogram is shown to be effective in mitigating the channel effect and data augmentation can further increase the system's robustness. Antenna polarization is found to affect classification performance.

The codes[1] and datasets[2] created in this chapter are openly available online.

[1] https://github.com/gxhen/LoRa_RFFI
[2] https://ieee-dataport.org/open-access/lorarffidataset

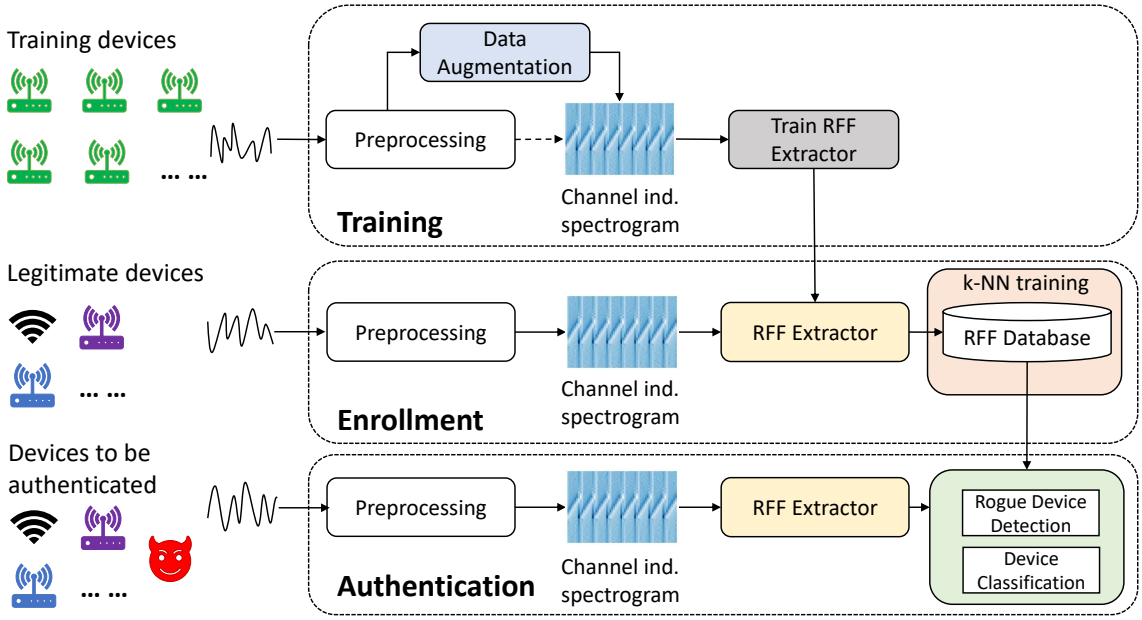


Figure 4.1: System diagram of the proposed RFFI system.

The rest of this chapter is organized as follows. Section 4.2 shows the system overview. Section 4.3 presents the designed scalable and channel-robust RFFI system. Section 4.4 provides extensive evaluation results to show the system's performance. Section 4.5 finally concludes the chapter.

4.2 System Overview

The proposed system is based on the DL-powered RFF extractor. As shown in Fig. 4.1, it involves training, enrollment, and authentication stages.

Training: The training stage produces a DL-powered RFF extractor rather than a classification NN. Specifically, a large number of labeled packets are collected from numerous training DUTs. Data augmentation is then used to increase channel diversity. After augmentation, the training signals are converted to channel-independent spectrograms to mitigate channel effects. Finally, the triplet loss in deep metric learning is used to train the RFF extractor. The input of the extractor is a 2D

channel-independent spectrogram, and the output is a vector consisting of 512 feature elements, which is the unique RFF of that device.

The training only needs to be done once as the trained RFF extractor is able to extract unique RFFs from out-of-library (unseen) DUTs. The training DUTs are not necessarily the same as the ones for enrollment and authentication. The training can be done by a third party that owns massive data collected from a huge number of devices to train an RFF extractor with excellent generalization capability.

Enrollment: The enrollment will obtain the RFFs of legitimate DUTs using the RFF extractor. These are the actual working devices in an IoT network, which are probably different from the training DUTs. These legitimate devices are required to send several packets to the RFFI system. Then RFFs can be extracted from the received packets using the trained RFF extractor and stored in a database. The RFFs of the newly joined devices will be added to the database and the RFFs of devices that leave the system will be deleted, which allows the system to be updated efficiently. The enrollment should be carried out in a controlled environment to ensure the RFFs of rogue devices are not enrolled.

Authentication: A complete authentication system should consist of two parts, namely rogue device detection[3] and device classification. Rogue device detection first determines whether the transmitter belongs to the legitimate group, i.e., previously enrolled DUTs, and device classification further infers its label. Both are implemented by the kNN algorithm.

4.3 System Design

4.3.1 Signal Preprocesssing

The LoRa signal collection and preprocessing algorithms are detailed in Section 2.3. It consists of synchronization, CFO compensation, and normalization. The collected signals are discrete IQ samples.

[3]Some work uses the term 'identification' to refer to detecting rogue devices.

4.3.2 Channel Independent Spectrogram

The received signal, i.e., IQ samples, is not only distorted by the hardware impairments but also by the wireless channel. While many IoT devices are mobile and the wireless channel frequently changes. In this subsection, the channel-independent spectrogram is proposed to mitigate the channel effect in the time-frequency domain while preserving the device-specific characteristics.

Observation of Channel Effect

LoRa supports three bandwidth options, namely 125 kHz, 250 kHz, and 500 kHz. LoRa transmissions are sometimes assumed to experience flat fading where the wireless channel causes the same magnitude and phase changes to all the frequency components.

However, it was experimentally found that the wireless channel significantly distorts the LoRa signal, and the effect can be easily observed from the received waveform. We collected LoRa packets in LOS stationary and non-line-of-sight (NLOS) stationary scenarios as well as LOS mobile and NLOS mobile scenarios. Detailed experimental settings can be found in Section 4.4. The I-branch of the packet preamble is shown in Fig. 4.2. It can be observed that the waveforms collected under different scenarios are distinct. Section 4.3.3 demonstrates that the time dispersion causes the sawtooth shapes, i.e., the multipath effect, and the amplitude variation is due to channel changes, i.e., the Doppler effect.

Constructing Channel Independent Spectrogram

In Chapter 3, the STFT-generated spectrogram is used as the input to the NN because it suits the frequency-changing property of LoRa signals. However, the spectrogram is affected by the wireless channel. To mitigate channel effects, we convert the spectrogram to a specially designed channel-independent spectrogram, which is achieved by dividing the adjacent columns.

As introduced in Section 2.2, STFT can be considered as splitting $y[n]$ into B segments of A samples, then performing FFT on each segment. According to the

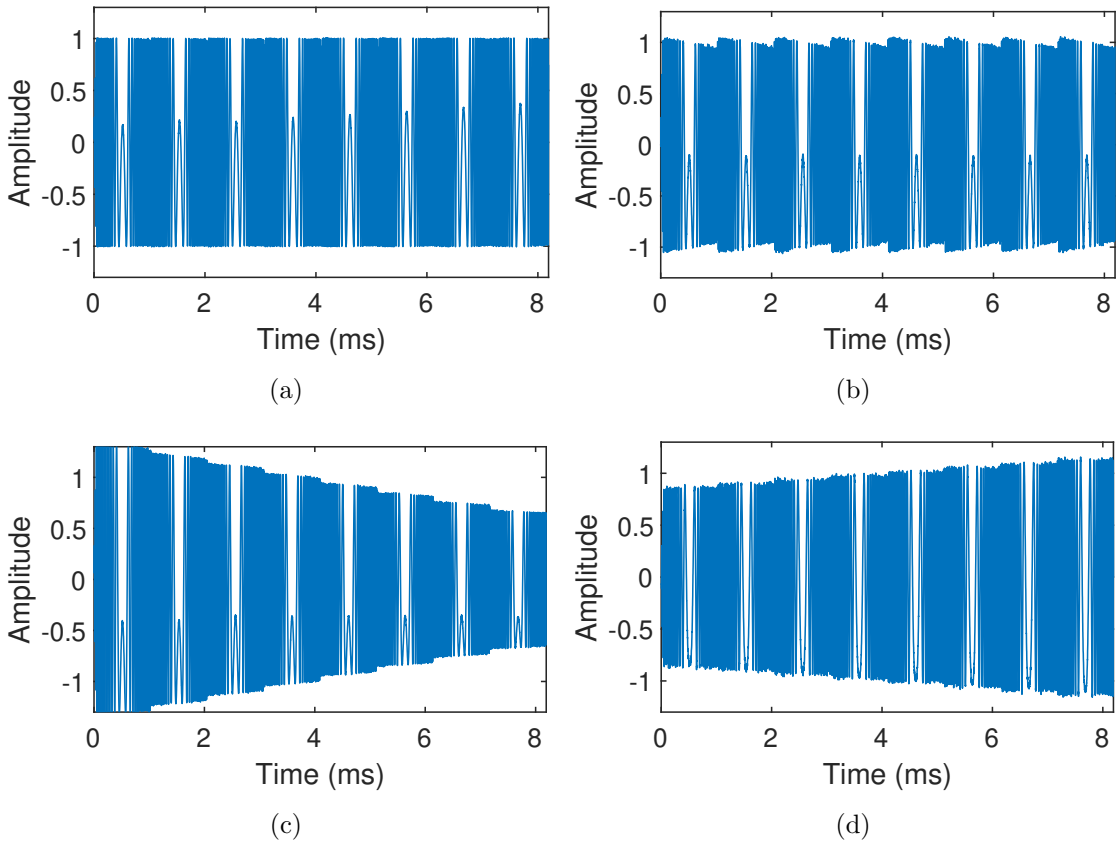
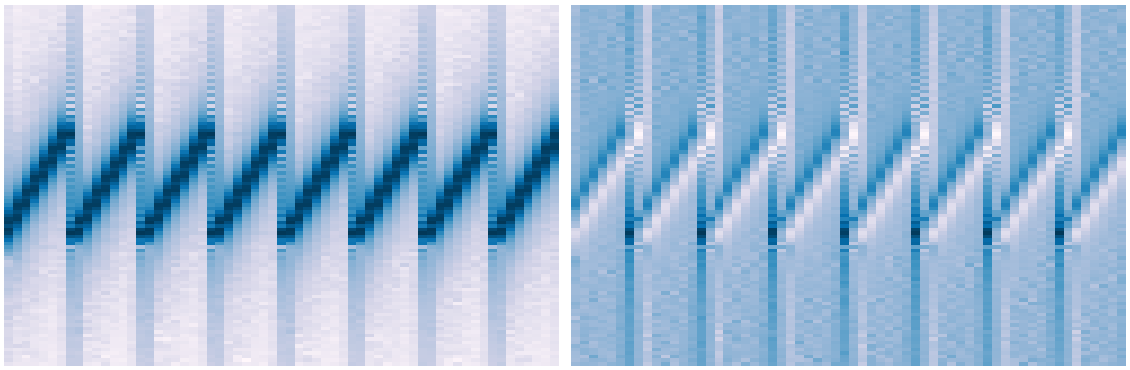


Figure 4.2: Normalized received LoRa waveform. (a) LOS stationary scenario. (b) NLOS stationary scenario. (c) LOS mobile scenario. (d) NLOS mobile scenario.

STFT implementation in this chapter, $B = 63$ and $A = 256$. The overall effect of hardware distortion on the transmitted signal in the frequency domain is denoted as $\widetilde{\mathcal{F}}^k(\cdot)$. Therefore, the STFT complex matrix $\mathbf{S}$ can be arranged as

$$\begin{aligned}\mathbf{S} &= \begin{bmatrix} H_{1,1}\widetilde{\mathcal{F}}^k(X_{1,1}) & H_{1,2}\widetilde{\mathcal{F}}^k(X_{1,2}) & \cdots & H_{1,B}\widetilde{\mathcal{F}}^k(X_{1,B}) \\ H_{2,1}\widetilde{\mathcal{F}}^k(X_{2,1}) & H_{2,2}\widetilde{\mathcal{F}}^k(X_{2,2}) & \cdots & H_{2,B}\ \widetilde{\mathcal{F}}^k(X_{2,B}) \\ \vdots & \vdots & \cdots & \vdots \\ H_{A,1}\widetilde{\mathcal{F}}^k(X_{A,1}) & H_{A,2}\widetilde{\mathcal{F}}^k(X_{A,2}) & \cdots & H_{A,B}\widetilde{\mathcal{F}}^k(X_{A,B}) \end{bmatrix} \\ &= [\mathbf{H}_1 \odot \widetilde{\mathcal{F}}^k(\mathbf{X}_1) \quad \mathbf{H}_2 \odot \widetilde{\mathcal{F}}^k(\mathbf{X}_2) \ \cdots \ \mathbf{H}_B \odot \widetilde{\mathcal{F}}^k(\mathbf{X}_B)], \end{aligned} \tag{4.1}$$



(a) (b)

Figure 4.3: (a) Spectrogram of LoRa preambles. (b) Channel independent spectrogram of LoRa preambles.

where $\mathbf{X}_b = [X_{1,b}, X_{2,b} \cdots X_{A,b}]^T$ denotes the ideal frequency spectrum of the b^{th} signal segment, the column vector $\mathbf{H}_b = [H_{1,b}, H_{2,b} \cdots H_{A,b}]^T$ represents the channel frequency response experienced by the b^{th} signal segment and $\odot$ denotes element-wise product.

The phase information of $\mathbf{S}$ is noisy and can be affected by several issues such as phase noise, CFO, and sampling time offset. Therefore, only the amplitude of each element in $\mathbf{S}$ is used, expressed as $|\mathbf{S}|$. The result is rearranged as

$$|\mathbf{S}| = [|\mathbf{H}_1| \odot \left|\widetilde{\mathcal{F}}^k(\mathbf{X}_1)\right| \quad |\mathbf{H}_2| \odot \left|\widetilde{\mathcal{F}}^k(\mathbf{X}_2)\right| \ \cdots \ |\mathbf{H}_B| \odot \left|\widetilde{\mathcal{F}}^k(\mathbf{X}_B)\right|]. \tag{4.2}$$

where $|\cdot|$ computes the amplitude of each element in the input matrix $\mathbf{S}$. In our experimental settings, the time gap between two adjacent signal segments is only 128 μs. It is reasonable to assume $|\mathbf{H}_b| \approx |\mathbf{H}_{b+1}|$, i.e., the corresponding element in $\mathbf{H}_b$ and $\mathbf{H}_{b+1}$ are almost the same since the wireless channel will not change dramatically in such a short period. Based on this assumption, we can divide the b^{th} column by the $(b+1)^{th}$ one so that the channel-related information can be eliminated.

The matrix after division is given as

$$\mathbf{C} = \Big[\frac{\left|\widetilde{\mathcal{F}}^k(\mathbf{X}_2)\right|}{\left|\widetilde{\mathcal{F}}^k(\mathbf{X}_1)\right|} \quad \frac{\left|\widetilde{\mathcal{F}}^k(\mathbf{X}_3)\right|}{\left|\widetilde{\mathcal{F}}^k(\mathbf{X}_2)\right|} \quad \cdots \quad \frac{\left|\widetilde{\mathcal{F}}^k(\mathbf{X}_B)\right|}{\left|\widetilde{\mathcal{F}}(\mathbf{X}_{B-1})\right|}\Big]. \tag{4.3}$$

Compared to (4.2), the channel information is largely eliminated but the device-specific hardware distortions $\widetilde{\mathcal{F}}^k(\cdot)$ are preserved. Similar to (2.7), the amplitude of $\mathbf{C}$ is expressed in dB scale

$$\widetilde{\mathbf{C}} = 10\log_{10}(|\mathbf{C}|^2). \tag{4.4}$$

$\widetilde{\mathbf{C}}$ is the proposed channel-independent spectrogram which is used as the input of the RFF extractor. It is also worth noting that the top and bottom of $\widetilde{\mathbf{C}}$ are cropped by 30% since these parts are far from the central frequency and therefore contain nearly no useful information but noise. The size of the cropped channel independent spectrogram is 102×62 according to our experimental settings. The channel-independent spectrogram of the LoRa preamble part is shown in Fig. 4.3b.

4.3.3 RFF Extractor Training

The RFF extractor is the core module in the proposed RFFI system. It should extract channel-independent and discriminative RFFs from the received signal and generalize well on previously unseen devices. In the proposed system, the training data can be collected in a controlled environment, and correctly labeling each LoRa packet is not difficult. This enables us to use supervised learning approaches rather than unsupervised ones such as autoencoders to better learn identity-related features.

Data Augmentation

Data augmentation is an efficient approach in deep learning to improve performance and has been recently applied in the area of RFFI to increase its robustness to the wireless channel [24, 27, 52]. Data augmentation can generate more training data to increase the performance of the RFF extractor and reduce the overhead for

Table 4.1: Parameter of the Channel Simulator

Paramter	Range
RMS delay spread τ_d (ns)	[5,300]
Maximum Doppler frequency f_d (Hz)	[0,10]
Rician K-factor	[0,10]
SNR (dB)	[20,80]

data collection. It can also mitigate the channel effect by injecting various channel distortions into the training data so that the RFF extractor automatically learns how to deal with them.

1) Channel Effects: In this chapter, the channel effect for data augmentation includes both multipath and Doppler shift. The multipath is described by the power delay profile (PDP). The exponential PDP is selected with each path $P[d]$ mathematically given as

$$P[d] = \frac{1}{\tau_d} e^{-dT_s/\tau_d}, \quad d = 0, 1, \cdots, d_{max}, \tag{4.5}$$

where τ_d is the RMS delay spread and d_{max} is the index of the last path. The PDP is normalized by dividing the sum of all values in $P[d]$.

This chapter also considers the Doppler shift, which is overlooked in previous work as the channel is assumed to be constant in a packet time [24, 27, 52]. However, this assumption might not hold for some LoRa transmissions. Each LoRa preamble typically lasts about 1 ms and the channel effect may not remain constant in mobile scenarios. Actually, the Doppler effect is observed from the received waveform such as Fig. 4.2c and Fig. 4.2d. The Doppler effect can be characterized by the Doppler spectrum. This chapter adopts the popular Jakes model whose spectrum $\Omega(f)$ is defined as

$$\Omega(f) = \frac{1}{\pi f_d \sqrt{1 - (f/f_d)^2}}, \tag{4.6}$$

where f_d is the maximum Doppler shift.

2) Procedure: The data augmentation is carried out as follows.

1. The training data should be collected in a short-distance LOS stationary scenario so that it can be assumed as experiencing frequency-flat slow fading.

2. The number of training samples is increased by replication. In our implementation, the training set is doubled. Note that more replications may improve system performance but will significantly increase the requirements on disk and memory storage.

3. The channel effect with multipath and Doppler effect is generated with the parameters randomly selected within specific ranges given in Table 4.1.

4. The packet generated in step 2 passes through the channel from step 3. The AWGN is added to the signal to emulate different SNR levels.

The augmentation is completed by repeating the above process for all the training packets.

We show the waveform of three cases, namely strong multipath[4] no Doppler effect, strong Doppler effect with weak multipath and both strong multipath and Doppler effects in Fig. 4.4b, Fig. 4.4c and Fig. 4.4d, respectively. It can be observed that the augmented waveforms match well with the real collected ones shown in Fig. 4.2. We can also conclude that the wireless channel distorts the received LoRa signal significantly. The sawtooth shapes are caused by multipath and the amplitude variation is caused by the Doppler effect.

Model Architecture

The channel-independent spectrogram can be regarded as a 2D image thus CNN is used to extract RFFs from it. The CNN in this chapter is designed with reference to the well-known ResNet [98]. It is further optimized to be more lightweight and suitable for the size of channel-independent spectrograms.

[4]The level of multipath relies on both the symbol period and the RMS delay spread. As LoRa has a long symbol period, it does not suffer from multipath much. 'Strong multipath' in this chapter

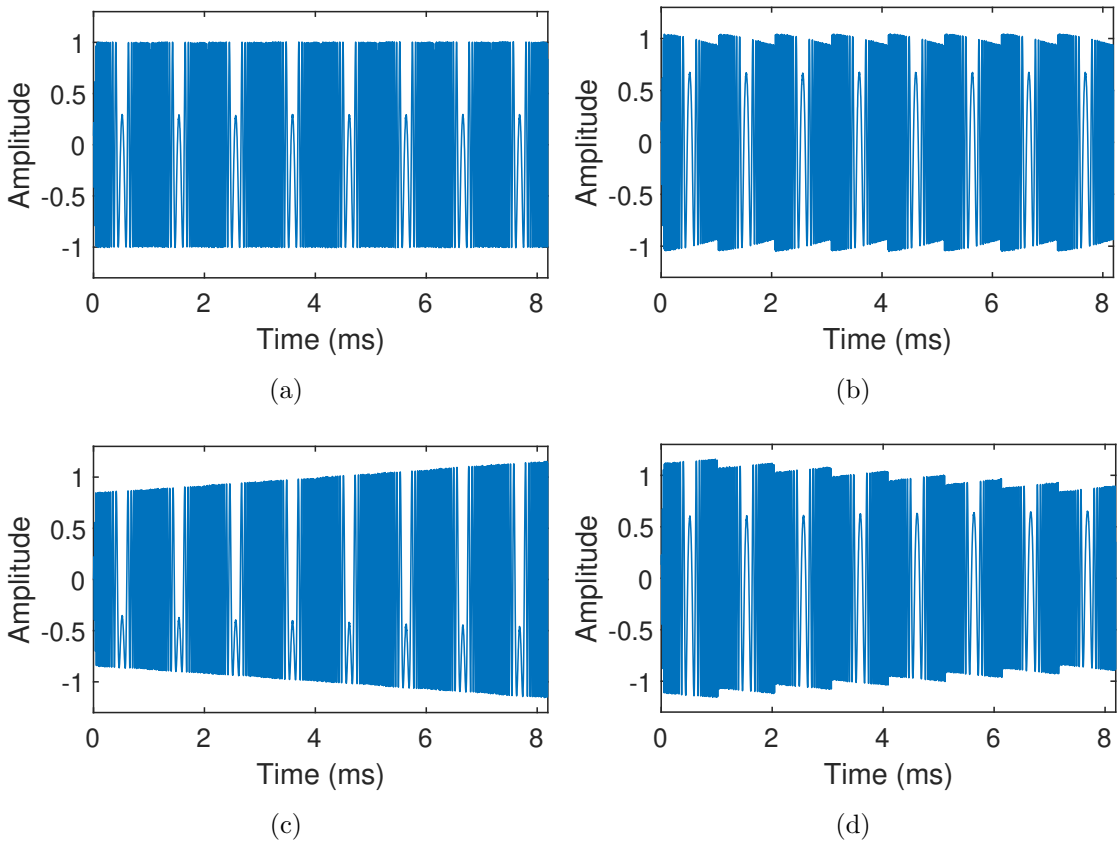


Figure 4.4: Augmented LoRa preambles. (a) Original waveform. (b) Strong multipath no Doppler effect ($\tau_d = 300$ ns, $f_d = 0$ Hz). (c) Strong Doppler effect with weak multipath ($\tau_d = 5$ ns, $f_d = 10$ Hz). (d) Both multipath and Doppler effects are strong ($\tau_d = 300$ ns, $f_d = 10$ Hz).

The architecture of the RFF extractor is shown in Fig. 4.5, where /2 denotes strides in convolution operation is two. It consists of nine convolutional layers, one average pooling layer, and one dense layer of 512 neurons, residual structure is also adopted. The first convolution layer uses 32 7×7 filters with stride 2, the second to the fifth layers use 32 3×3 filters, and the sixth to the ninth layers employ 64 3×3 filters. All the convolutional layers are activated by ReLU and padding is

is just a contrast to the 'weak multipath' scenario of 5 ns RMS delay. Similarly, the 'strong Doppler effect' is a contrast to the 'no Doppler' scenario.

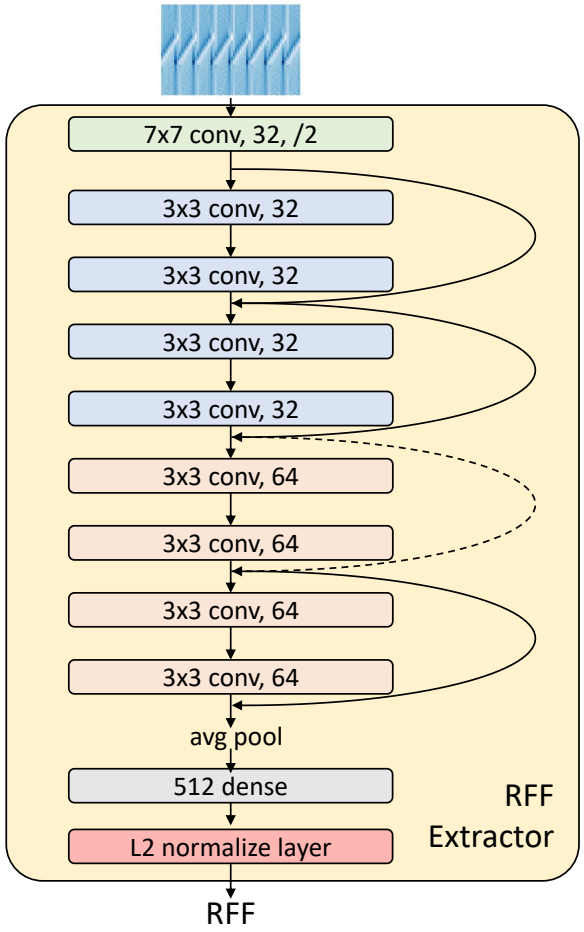


Figure 4.5: Architecture of the RFF extractor, revised from ResNet.

used. We leverage an L2 normalization layer to make the RFF extractor learns better features [101]. The output of the RFF extractor is a vector consisting of 512 elements which can be considered as the RFF extracted from the received packet. Note that the model hyper-parameters such as the dimension of feature vectors are adjusted based on LoRa-RFFI, and can be optimized on a third-party dataset. The designed feature extractor has 12,458,496 learnable parameters, which is affordable for many embedded systems.

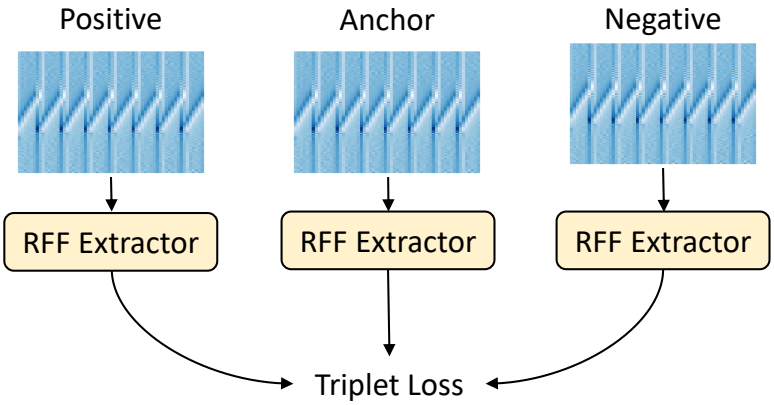


Figure 4.6: Triplet loss.

Deep Metric Learning

Deep metric learning aims to train an NN that maps the input to a 1D vector/embedding. There are many loss functions available in deep metric learning such as contrastive loss, L2-softmax loss, etc. Interested readers, please refer to [102, 103] for detailed information.

Triplet loss is a well-known ranking loss used in deep metric learning and has been successfully adopted in face recognition tasks [104]. It projects the input data into a space where similar samples are close to each other and dissimilar ones are far away. During the training process, a triplet consisting of anchor, positive and negative samples is selected from the training dataset at each step. Specific to RFFI, anchor and positive samples are packets from the same device and the negative sample is from a different one. As shown in Fig. 4.6, the RFFs of anchor, positive and negative samples are extracted, and then the triplet loss is calculated. The goal of triplet loss is to minimize the Euclidean distance between the anchor and positive samples while maximizing the distance between the anchor and negative samples, which is expressed as

$$\mathcal{L}_{triplet} = \max(E(Anc, Pos) - E(Anc, Neg) + \alpha, 0) \tag{4.7}$$

where Anc, Pos, Neg denote features extracted from anchor, positive and negative

samples, respectively. $E(\cdot, \cdot)$ denotes the Euclidean distance between two vectors. α denotes the margin between positive and negative pairs, which is a hyper-parameter and is set to 0.1 in our implementation.

4.3.4 Enrollment and Authentication

Enrollment

The legitimate devices are required to send several packets for enrollment before joining the system. These enrollment packets are first preprocessed and transformed to channel independent spectrograms. Then the RFF extractor is used to extract the RFF templates and store them in a database. The enrollment is actually the training phase of a kNN classifier, which simply memorizes all the training samples. Unlike deep learning, kNN is not a data-hungry model. This is very desirable as it can significantly decrease the overhead of data collection. For example, 100 packets from each legitimate DUT are sufficient for enrollment, which is experimentally presented in Section 4.4.

Authentication

A complete authentication system should include two parts, namely rogue device detection and device classification. In the proposed implementation, they are both based on the kNN algorithm with simple distance measures. The RFF extractor is trained with Euclidean distance-defined triplet loss. The kNN algorithm also relies on Euclidean distance measures thus their principles match well.

1) Rogue Device Detection: Rogue device detection is to determine whether the received packet is from an enrolled legitimate device. It is necessary before device classification, otherwise, the RFFI system will assign legitimate labels even to rogue devices. The rogue device detection is implemented by the distance-based kNN anomaly detection algorithm. After receiving a packet, its RFF is extracted and the average distance to its K_{knn}[5] nearest neighbors in the RFF database is calculated as

[5] K_{knn} is a hyperparameter that needs to be experimentally determined. It is advised to test various values of K_{knn} in a small dataset to find the optimal option.

the detection score, which can be mathematically expressed as

$$E_{avg} = \frac{1}{K_{knn}} \sum_{k=1}^{K_{knn}} E_k, \tag{4.8}$$

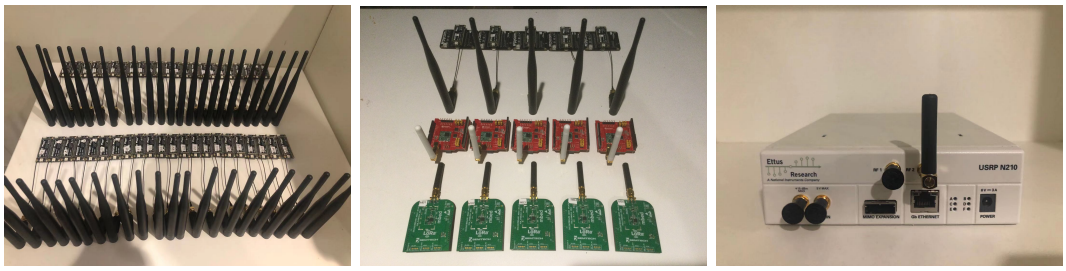
where E_k is the Euclidean distance to the k^{th} neighbor. Then a predefined threshold λ_{rd} is used to determine whether the received packet is from an enrolled device. When the detection score E_{avg} is above λ_{rd}, the packet is considered to be sent from a rogue device and the authentication fails. In contrast, the packet is considered to come from an enrolled device when E_{avg} is below the threshold λ_{rd} and will be further classified. This can be formulated as

$$Decision = \begin{cases} \text{enrolled device}, when\ E_{avg} \leq \lambda_{rd} \\ \text{rogue device}, when\ E_{avg} > \lambda_{rd} \end{cases} \tag{4.9}$$

The detection threshold λ_{rd} can be determined based on the application requirement. A higher λ_{rd} leads to higher security, while a lower λ_{rd} makes the RFFI system more user-friendly, i.e., access will not be denied frequently. When there is no special requirement, the method to find the optimal λ_{rd} will be introduced in Section 4.4.2.

2) Device Classification: Device classification is to infer the specific label of a transmitter from which the received packet is sent, which is a classification problem. It outputs a previously enrolled label according to the templates stored in the RFF database.

The proposed device classification system is implemented with the majority voting kNN algorithm. The RFF of the received packet is first extracted and its K_{knn} nearest neighbors are selected from the database according to the Euclidean distance. This packet is then assigned to the label that is most frequent among the K_{knn} neighbors.



(a) (b) (c)

Figure 4.7: Experimental devices. (a) DUTs 1-45: LoPy4. (b) DUTs 46-60: mbed SX1261 shields, FiPy and Dragino SX1276 shields. (c) Receiver: USRP N210 SDR.

Table 4.2: LoRa DUTs.

DUT index	Model	Chipset
1 - 45	Pycom LoPy4	SX1276
46 - 50	mbed SX1261 shield	SX1261
51 - 55	Pycom FiPy	SX1272
56 - 60	Dragino SX1276 shield	SX1276

4.4 Experimental Evaluation

4.4.1 Experimental Settings

DUT and Receiver

We employed 60 COTS LoRa devices as DUTs, and a USRP N210 SDR platform as the receiver, as shown in Fig. 4.7. Detailed DUTs information can be found in Table 4.2. The carrier frequency is 868.1 MHz and the transmission interval is set to 0.3 s. The receiver sampling rate is 1 MHz.

Training RFF Extractor

We first collect 500 packets from each of DUTs 1-30 in a residential room with LOS between the DUT and the receiver. The distance is about half a meter and DUT

Table 4.3: Extractor Information.

	Extractor Input	Training DUTs	Data Augmentation
Extractor 1	Channel ind. spectrogram	DUT 1-30	Yes, multipath and Doppler
Extractor 2	Channel ind. spectrogram	DUT 1-20	Yes, multipath and Doppler
Extractor 3	Channel ind. spectrogram	DUT 1-10	Yes, multipath and Doppler
Extractor 4	Channel ind. spectrogram	DUT 1-30	No
Extractor 5	Spectrogram	DUT 1-30	No
Extractor 6	Channel ind. spectrogram	DUT 1-30	Yes, multipath only

antennas are vertical to the ground.

We trained six RFF extractors with their detailed configuration given in Table 4.3. Extractor 1 is trained as a baseline model involving channel independent spectrogram and data augmentation. Extractors 2-6 are trained for comparison to show the effects of different procedures during training. Extractor 1-3 are trained with the dataset augmented by both multipath and Doppler effects, while during the training of Extractor 6 only the multipath effect is emulated.

All the RFF extractors are trained with the same settings, including validation set ratio, optimizer, learning rate schedule, batch size and stop condition. 10% of the training data are randomly separated for validation. The model is optimized using root mean squared propagation (RMSprop) with an initial learning rate of 0.001. The learning rate drops every time the validation loss does not decrease for 10 epochs and the drop factor is 0.2. The batch size is set to 32. The training stops when validation loss does not decrease for 30 epochs. The model is implemented using Keras and trained with NVIDIA GeForce GTX 1660.

Table 4.4: Summary of Experimental Evaluation.

Section	Extractor	Enrollment Set	Identification/Classification Set	Purpose
Section 4.4.4	1,2,3	Legitimate DUTs (residential room)	Legitimate and rogue DUTs (residential room)	Generalization ability of rogue device detection versus the number of training DUTs
Section 4.4.5	1,2,3	Seen, unseen, and different model DUTs (residential room)	Seen, unseen, and different model DUTs (residential room)	Generalization ability of device classification versus the number of training DUTs
Section 4.4.6	1,4,5	Data collected in stationary scenarios (residential room)	Ten datasets with different channel conditions (office building)	Evaluate data augmentation and channel independent spectrogram
Section 4.4.7	1,6	Data collected in stationary scenarios (residential room)	Emulated datasets with various moving speeds	Evaluate data augmentation (Doppler effect)
Section 4.4.8	1	Datasets with different antenna directions (office)	Datasets with different antenna directions (office)	Effect of antenna polarization

Enrollment and Authentication

100 packets are collected from each DUT for enrollment. Unless otherwise specified (Section 4.4.8), the enrollment sets are collected in a residential room with LOS and vertical antenna placement.

We collect another 100 packets from each DUT for authentication. The experimental setup varies according to the tests. We use identification and classification sets to represent datasets for rogue device detection and device classification, respectively. The number of neighbors K_{knn} is set to 15 for both rogue device detection and device classification.

The RFF extractors training only needs to be done once. Enrollment and authentication are carried out multiple times for evaluating system performance in various configurations. Table 4.4 summarizes the experimental studies and their configurations.

4.4.2 Evaluation Metrics

Rogue Device Detection

Rogue device detection can be evaluated using the receiver operating characteristic (ROC) curve. As shown in (4.9), the result of rogue device detection is related to the value of the threshold λ_{rd}. This makes it unfair to compare rogue device detection

performance since RFFI systems may set different threshold λ_{rd}. The ROC curve can be leveraged to overcome this, which reveals the trade-off between a false-positive rate (FPR) and true-positive rate (TPR) at various threshold settings. More specifically, a pair of FPR and TPR are calculated at each threshold setting, and then we plot these pairs in the same figure to obtain the ROC curve. Two important metrics can be further calculated from the ROC curve, namely the area under the curve (AUC) and the equal error rate (EER). AUC refers to the area under the ROC curve, and the larger it is, the better the system performance. EER refers to the point on the ROC curve where FPR equals (1-TPR), and the smaller the better. EER can also help find the optimal threshold. The threshold λ_{rd} that makes the system meet the EER point can be selected as the optimal value.

Classification

The metrics for device classification are the confusion matrix and overall accuracy which is defined as the correctly classified samples divided by the total number of test samples.

4.4.3 System Scalability

RFFI system should be scalable to allow device joining and leaving. Previous deep learning-based RFFI systems use a single classification NN therefore the model output can only be device labels that are present during training. For example, if we train with the packets collected from DUTs 1-10, the model can only output a label between 1-10. Therefore, the deep learning model must be retrained when a new DUT joins the system, which is time-consuming and not practical.

In this chapter, we train the deep learning model as an RFF extractor rather than for classification. The training of the RFF extractor only needs to be done once. We then introduce an enrollment stage to obtain the RFFs of any devices in the IoT network via the pre-trained RFF extractor. In the enrollment stage, we only need to train a kNN classifier (store template RFFs), which can be done very quickly when new devices join. The RFF database can also be managed efficiently by

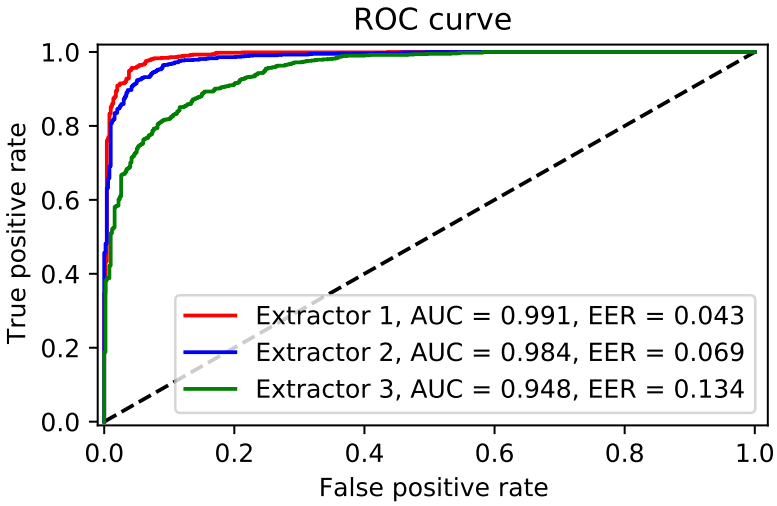


Figure 4.8: ROC curve of rogue device detection.

only recording the RFF of devices that are active and present in the IoT network. In summary, our protocol has excellent scalability in terms of maintaining an up-to-date device list.

4.4.4 Generalization Ability for Rogue Device Detection

Rogue devices are out-of-library devices that are inaccessible during training. Whether the number of training DUTs affects the performance of rogue device detection should be evaluated. The hardest case for rogue device detection is whenever the rogue device has similar hardware characteristics to the legitimate ones. Therefore, we specifically select DUT 31-40 as legitimate devices and DUT 41-45 as rogue ones, which are all LoPy4 devices. The identification dataset consists of 100 packets from each DUT 31-45, collected from the same residential room using the same setup.

The ROC curves are shown in Fig. 4.8. It can be observed that the AUC of Extractor 1 is 0.9905, indicating excellent detection performance. The AUC of Extractor 3 is the worst, which is 0.9479. This demonstrates that the more DUTs involved in the training stage, the better the rogue device detection capability is.

4.4.5 Generalization Ability for Device Classification

To achieve good scalability, the RFF extractor must be able to extract RFFs from newly added devices that are out-of-library during the training stage. In other words, the RFF extractor should have an excellent generalization ability on previously unseen devices. The rule of thumb in deep metric learning is that the more training data, the better the generalization ability. We select Extractors 1, 2, and 3 for comparison since they are trained with different numbers of DUTs.

We specifically select three groups of DUTs for evaluation, namely DUTs 1-10, DUTs 31-40, and DUTs 46-60.

- **Seen DUTs**: DUTs 1-10 are present during the training of Extractors 1, 2, and 3 therefore they are used to evaluate the extractor performance on seen devices.
- **Unseen DUTs, same manufacturer**: DUTs 31-40 are disjoint with the training devices but with the same model (LoPy4). They can be used to validate the extractor performance on unseen devices.
- **Unseen DUTs, different manufacturers**: DUTs 46-60 are produced by other manufacturers, whose hardware characteristics are likely to be different from the training LoPy4 devices. This is the most challenging case as it requires a much higher generalization ability of the RFF extractor. This is also unavoidable in practice since involving devices from all the manufacturers during training is impossible.

The classification datasets are collected from the same residential room with the same setup as the enrollment set.

The classification results on these three groups are shown in Fig. 4.9. It can be observed that Extractor 1, 2, 3 perform excellently on DUTs 1-10 and DUTs 31-40. The overall accuracies are always above 95%. The highest accuracy is reached by Extractor 1 with 98.50% on DUTs 1-10. The accuracy is 98.40% on classifying DUTs 31-40 which demonstrates the trained RFF extractor can efficiently extract RFFs from the devices that are not present during training. We can see that there is

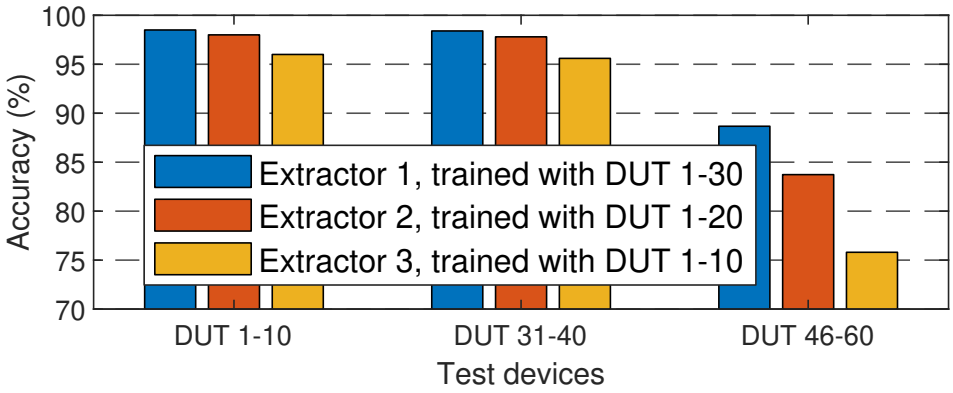


Figure 4.9: Performance of RFF extractors.

a significant performance gap between Extractor 1, 2, 3 on DUT 46-60. Extractor 3, trained with only 10 DUTs, has the worst classification result, i.e., 75.80%. Training with 30 devices (Extractor 1) can increase it to 88.67% and the confusion matrix is shown in Fig. 4.10. It is found that the DUTs of the same type have similar hardware characteristics as almost all the misclassified packets fall into the three red boxes. The classification performance on DUT 46-50 is excellent but that for DUT 56-60 is not satisfying. This may be because the hardware characteristics of DUT 56-60 (Dragino SX1276) are more different from the training DUTs (LoPy4). In summary, it is advised to include as many DUTs of various models as possible to improve the generalization ability on out-of-library devices.

4.4.6 Effect of Data Augmentation and Channel Independent Spectrogram

The RFFI system should be robust to locations and channel variations. We mitigate the channel effect in the time-frequency domain and propose the channel-independent spectrogram as the model input. Then we use data augmentation to further increase its robustness to the wireless channel.

To verify that they are effective, classification datasets are collected in an office building whose environment is completely different from the enrollment residential

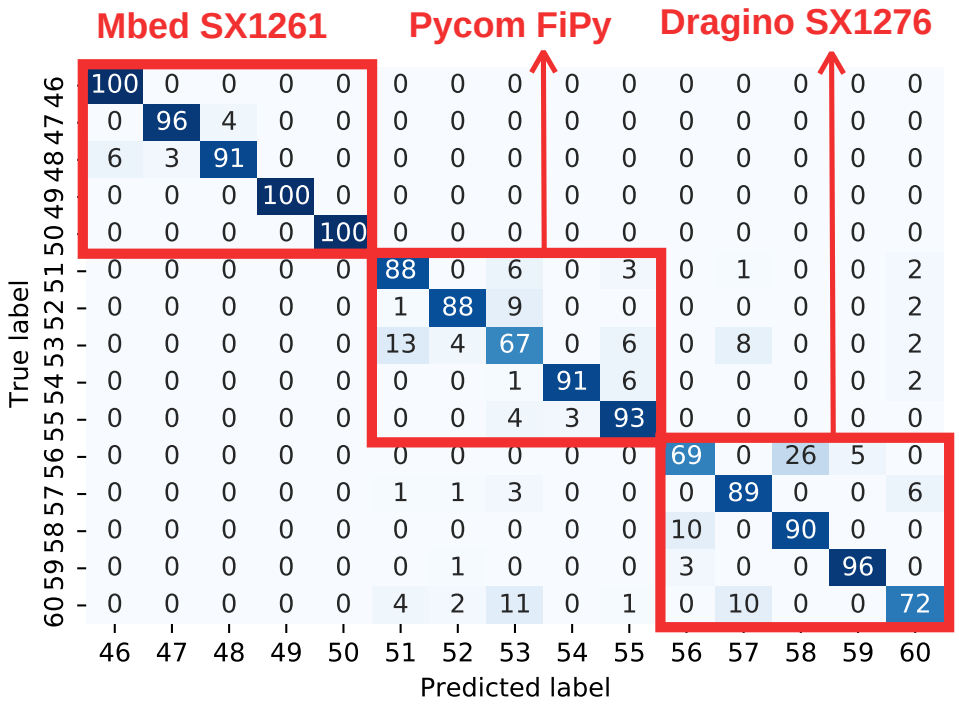


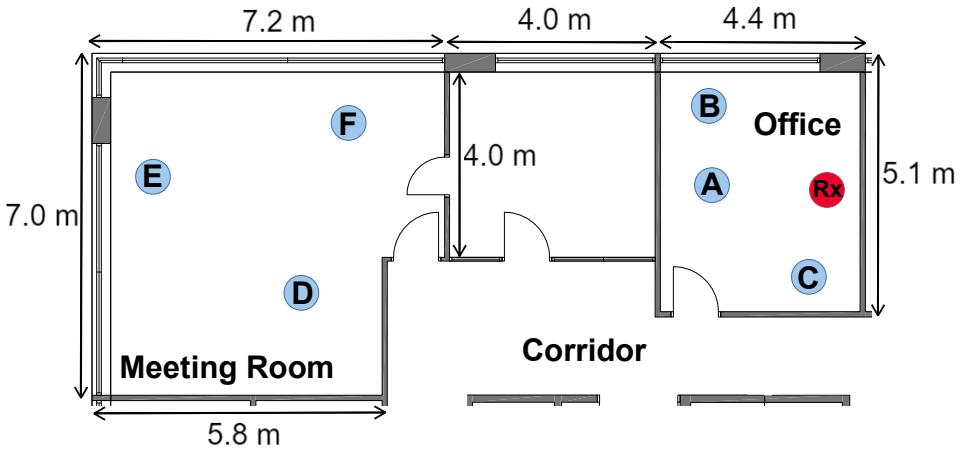
Figure 4.10: Classification result on other LoRa types, overall accuracy 88.67%.

Table 4.5: Summary of Classification Datasets Collected in an Office Room and a Meeting Room.

Evaluation Purpose	Data Augmentation and Channel Independent Spectrogram										Antenna Polarization	
Dataset	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12
Location	A	B	C	D	E	F	B	F			B	F
Channel Effect	LOS	LOS	LOS	NLOS	NLOS	NLOS	LOS	NLOS	LOS	NLOS	LOS	NLOS
Movement	Stationary						Objective moving		Mobile		Stationary	
Antenna	Vertical										Horizontal	

room. The experiments are conducted in an office and a meeting room. The photos and floor plan of them can be found in Fig. 4.11. We collect 10 classification datasets, D1-D10, and they can be categorized into three scenarios, namely stationary, the object moving, and mobile scenarios. A summary of these datasets is given in Table 4.5.

(a) (b)



(c)

Figure 4.11: Experimental environments. (a) Meeting room. (b) Office. (c) Floor plan.

Stationary Scenario

Six datasets, D1-D6, are collected at Location A-F, respectively. During the data collection, DUTs and USRP N210 are stationary and there is no object moving around.

Locations A-F lead to multipath effects of varying severity. For instance, the waveform of DUT 6 at Location F (D6) is similar to that shown in Fig. 4.2b, showing a distinct sawtooth shape. While the waveform at Location A is almost the same as

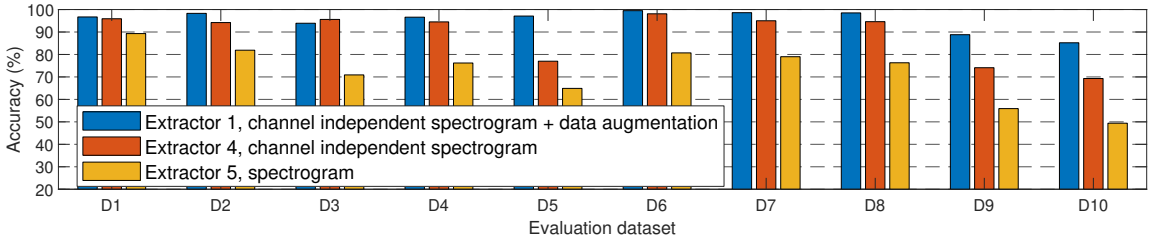


Figure 4.12: Classification results in various channel conditions.

Fig. 4.2a, which is nearly flat. As discussed in Section 4.3.3, the sawtooth is caused by the multipath effect therefore the channels at Locations A and F are different.

Object Moving Scenario

Two datasets, D7 & D8, are collected at locations B and F, respectively. The DUTs and USRP are kept stationary while a person randomly walks around the room at a speed of 2 m/s.

In this scenario, the collected LoRa packets show different waveforms since the channel changes as a result of people walking. However, the Doppler effect is not quite serious due to the low walking speed.

Mobile Scenario

Two datasets, D9 & D10, are collected in the office and meeting room, respectively. The USRP is kept stationary and a person takes the DUTs walking around at a speed of 2 m/s. There is always a clear LOS between the transmitter and receiver because the body does not obstruct it.

Both multipath and Doppler effects are serious in this scenario since the sawtooth shapes and amplitude variations can be frequently observed. Many packets show waveforms similar to those shown in Fig. 4.2c and Fig. 4.2d.

Discussion

Extractors 1, 4, and 5 are selected for comparison to demonstrate the effectiveness of channel independent spectrogram and data augmentation. Their training details can be found in Table 4.3 and the classification results are summarized in Fig. 4.12.

It can be observed that Extractor 1 performs well on all datasets, which indicates the proposed RFFI system is robust to locations and channel variations. It performs slightly worse on D9 and D10 that are collected in mobile scenarios (lower than 90%). This is possibly due to the inevitable shaking of the DUT antenna during moving, which will result in the change of antenna polarization. The effect of antenna polarization is discussed in Section 4.4.8.

Compared with Extractor 1, the training of Extractor 4 does not employ data augmentation. As can be seen, its performance on D5 is significantly worse with only 78.80% accuracy. D5 is collected at Location E which is the farthest from the receiver. This indicates the channel independent spectrogram is affected by the noise in lower SNR scenarios. Data augmentation must be used to further improve its robustness.

As for Extractor 5, it employs neither channel-independent spectrogram nor data augmentation. It can be observed that Extractor 5 only performs well on D1 which is a short-distance LOS stationary scenario. It is presumed that the channel condition at Location A is similar to the enrollment residential room. Once the channel is changed, the classification performance will degrade significantly. The worst results occur on D9 and D10 (mobile scenario) whose accuracy is 55.90% and 49.40%, respectively, which are 30% lower than Extractor 1.

Whether data augmentation can solve the channel problem without the use of the channel-independent spectrogram was also investigated. However, it was found that the training loss does not decrease during the training process. We presume this is due to the RFF extractor is not capable of extracting discriminative RFFs from the spectrogram when the training data is distorted by various wireless channels. In contrast, when we use the channel-independent spectrogram instead of the spectrogram as model input (Extractor 1), the channel effects are mitigated and device-specific

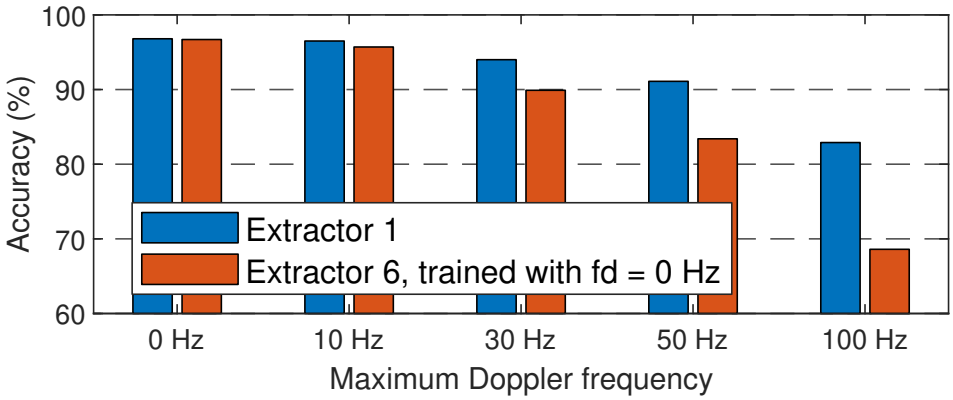


Figure 4.13: Classification results under various Doppler frequencies of the identification sets.

features can be learned so that the model is trained successfully.

4.4.7 Effect of Doppler Shift in Data Augmentation

Data augmentation must be considered to improve the classification performance in high-speed scenarios (serious Doppler effects). Due to the experimental constraints, we emulate the communication scenarios at various moving speeds by filtering the real collected D1 to a channel simulator to generate classification sets. The process is almost the same with data augmentation introduced in Section 4.3.3. The maximum Doppler frequency, f_d, is fixed to 0, 10, 30, 50, 100 Hz, respectively. These Doppler frequencies are equivalent to moving speeds of 0, 12.46, 37.33, 62.21 and 124.4 km/h for a LoRa system operating at 868 MHz. We select Extractor 1 and Extractor 6 for comparison to demonstrate the Doppler effect must be considered during data augmentation.

The results are summarized in Fig. 4.13. Compared with Extractor 1, Extractor 6 does not consider the Doppler effect during data augmentation. Its maximum Doppler frequency f_d is fixed to 0 Hz. It can be seen that both these two extractors achieved a good accuracy, i.e., around 97.50%, in stationary scenarios ($f_d = 0$ Hz).

However, when the Doppler frequency f_d increases, the performance gap between

Extractors 1 and 6 becomes larger. In the scenario of $f_d = 100$ Hz, the accuracy of Extractor 6 is only 68.60%, probably because the channel-independent spectrogram cannot perfectly eliminate channel effects in high-speed scenarios. As given in (4.2), $|\mathbf{H}_b| \approx |\mathbf{H}_{b+1}|$ holds when the Doppler effect is not severe. Involving the Doppler effect during data augmentation can mitigate this. As shown in Fig. 4.13, Extractor 1 can boost the accuracy to 80% in the $f_d = 100$ Hz scenario. However, there is still a 20% accuracy drop compared to low-mobility scenarios (e.g., $f_d = 10$ Hz), indicating that the channel effect is not fully eliminated. Therefore, algorithm enhancement is required for high-speed scenarios. One possible approach is to employ more precise channel models, which can be used to simulate high-speed scenarios and augment the training data.

4.4.8 Effect of Antenna Polarization

Antenna polarization refers to the direction of the electric field produced by an antenna. It was found to affect the transient-based RFFI system [105]. To the best knowledge of the author, there is no study on antenna polarization in deep learning-based approaches that use the non-transient signal for classification. To explore this, we additionally collected two datasets, D11 & D12, at Location B and Location F, respectively. The DUT antenna is horizontal to the ground and points toward the USRP. Omnidirectional antennas are used in the experiments. More specifically, the antenna radiation pattern is doughnut-shaped and omnidirectional in the horizontal plane. Both DUTs and USRP are kept stationary. Extractor 1 is used to extract RFFs.

Mismatched polarization refers to the antennas of the transmitter and receiver having orthogonal polarization directions. We use D2 and D11 for enrollment and classification sets, respectively, which are collected at Location B and the only difference is the antenna direction. The classification result is shown in Fig. 4.14a. It can be seen that nearly all the packets from DUT 36 are misclassified as DUT 33. A similar result is obtained when D6 is used for enrollment and D12 for classification, both of which are collected at Location F.

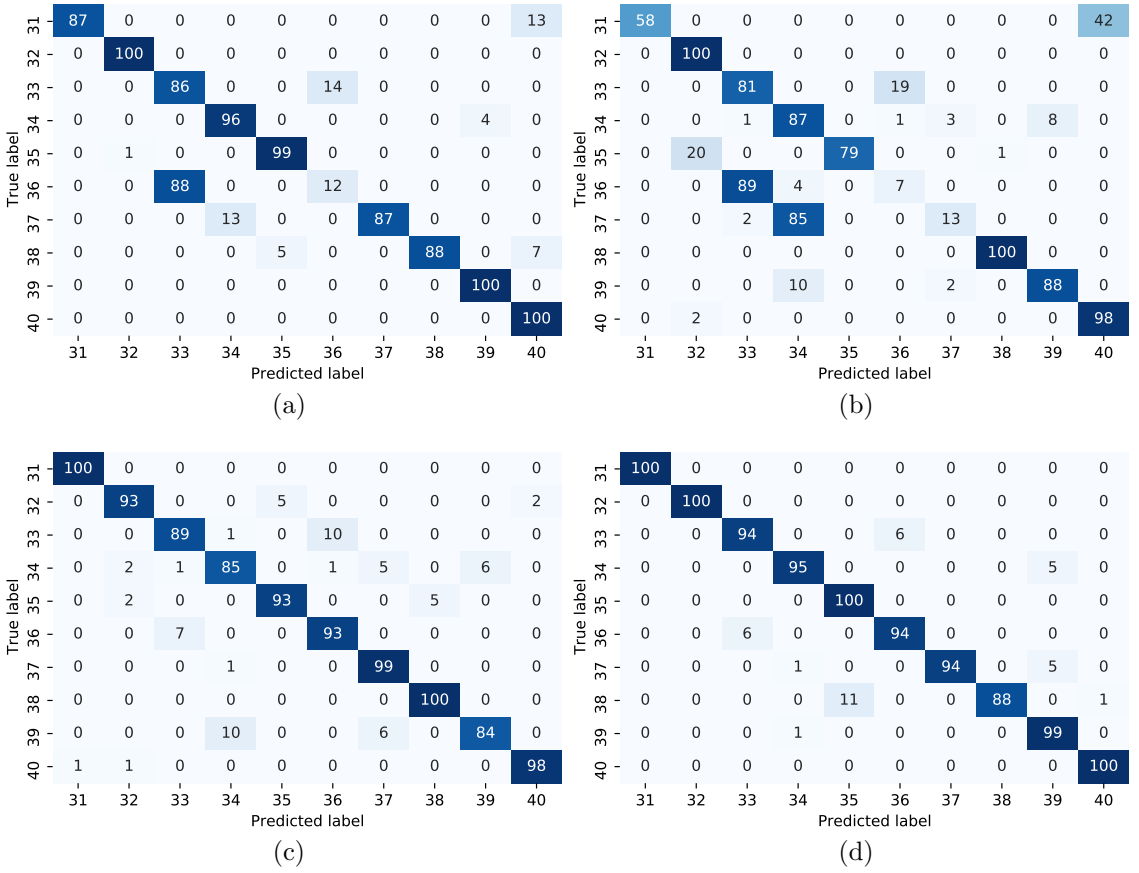


Figure 4.14: Classification results of antenna polarization. (a) Enrollment set D2, classification set D11, overall accuracy 85.50%. (b) Enrollment set D6, classification set D12, overall accuracy 71.10%. (c) Enrollment set D11, classification set D12, overall accuracy 93.40%. (d) Enrollment set D12, classification set D11, overall accuracy 96.40%.

Matched polarization refers to the antennas of the transmitter and receiver having the same polarization direction. We use D11 and D12 for enrollment and classification, respectively, which are collected at different locations but with the same antenna direction. The results are given in Fig. 4.14c and Fig. 4.14d. Their accuracies are both above 90% and there are no seriously misclassified DUTs.

The above results show the classification is severely by the transmitter antenna

direction. We infer that the reason for this is that a change in antenna orientation can significantly affect the propagation path of the RF signal, resulting in a severe change in the channel impulse response. The proposed channel-independent spectrogram may be under-effective in such cases.

4.5 Conclusion

In this chapter, a scalable and channel-robust RFFI framework that exploits the device-intrinsic hardware impairments for device authentication is proposed. Specifically, deep metric learning is leveraged to train an RFF extractor that has excellent generalization ability. When a new device joins the system, it only needs to send several packets for enrollment and the RFF extractor does not need to be retrained. The kNN algorithm is used for rogue device detection and device classification. To overcome the channel effect, we design the channel-independent spectrogram and further use data augmentation to improve the system's robustness to channel variations. We conduct extensive experiments using 60 COTS LoRa devices. We demonstrate our framework has an excellent generalization performance for both device classification and rogue device detection. The channel-independent spectrogram and data augmentation are shown to be effective under extensive tests with various channel conditions. We also find that antenna polarization affects classification performance. This chapter only uses the amplitude of the channel-independent spectrogram, and leveraging phase information can be a promising future work.

The RFFI protocols proposed in Chapter 3 and Chapter 4 assume the same receiver is used during training and inference, which is however not reasonable in real communication scenarios as the mobile devices are possibly served by different gateways. The next chapter aims to examine how variations in receiver characteristics affect RFFI performance.

Chapter 5

Towards Receiver-Agnostic and Collaborative RFFI

5.1 Introduction

This chapter aims to design a receiver-agnostic and collaborative RFFI protocol that is robust to receiver characteristic variations. First, the received signal is not only distorted by the transmitter chain but also by the receiver chain. It is experimentally found that using a new receiver for signal reception can severely degrade the NN classification accuracy. However, an IoT device is possibly served by different gateways/access points because of location changes. It is therefore necessary to train an NN that can be deployed for different receivers without sacrificing performance. Second, wireless signals can be captured by any receiver within range because of their broadcast nature. This motivates us to leverage multiple receivers to collaboratively perform RFFI to enhance identification performance.

In the RFFI protocol presented in this chapter, multiple receivers were used to collect sufficient packets from DUTs in order to train a receiver-agnostic NN. During the inference, receivers are equipped with the trained receiver-agnostic NN model. Once a packet is captured, the receivers initially make independent inferences which are then fused to permit better classification performance. For our

experimental evaluation, we used LoRa/LoRaWAN as a case study as it is a suitable technique to demonstrate the ideas. Specifically, we employed ten COTS LoRa nodes as DUTs and 20 SDR platforms as the LoRa gateways. Whilst the work focuses on LoRa/LoRaWAN as a case study, our receiver-agnostic approach is applicable to any RFFI system and the collaborative protocol is suitable for any wireless technique with multiple receivers operating simultaneously. The detailed contributions of this work include:

- The receiver effects on RFFI are experimentally investigated. The RFFI system implemented on low-end SDR receivers, e.g., RTL-SDR, shows an accuracy drop of 40% over four continuous days, which is probably due to the unstable hardware characteristics. Moreover, we show that changing the receiver in an RFFI system can result in serious performance degradation. For example, the NN trained with an RTL-SDR only achieves less than 20% accuracy when the test was using a USRP B200 SDR.

- A receiver-agnostic NN for RFFI is proposed. We guide the NN to learn receiver-independent features so that it is robust to performance degradation caused by receiver drift and change. We propose two training strategies for receiver-agnostic RFFI, namely homogeneous and heterogeneous training, depending on the diversity of the training receivers. The results show that the NN trained with heterogeneous training achieves better performance than the homogeneous one. Its classification accuracy is over 75% for all of the 20 SDRs and even exceeds 95% on receivers other than RTL-SDRs. Compared to the conventional approach, receiver-agnostic training effectively prevents drastic performance degradation on previously unseen receivers.

- A collaborative RFFI system with soft or adaptive soft fusion schemes is proposed and experimentally evaluated in both residential and office building environments. All the receivers are equipped with the same receiver-agnostic NN. They make independent inferences at the edge and upload them to a network server. The inferences are then fused by soft fusion or adaptive soft fusion

schemes to achieve higher accuracy. The experimental results show that collaborative RFFI can improve the classification accuracy by up to 20% compared to the single-receiver RFFI system.

- A further fine-tuning technique is proposed to mitigate receiver effects as even after receiver-agnostic training, the NN still may not reach satisfactory accuracy on some unseen receivers. To address this, we propose a fine-tuning approach that can collect a few packets with the new receiver to slightly adjust the NN parameters. The NN achieves higher performance after fine-tuning because it better adapts to the new receiver by re-learning the characteristics of the received signal. Experimental results show a further accuracy improvement of up to 40% on receiver-agnostic NNs.

The rest of the paper is organized as follows. Section 5.2 presents the system overview and Section 5.3 elaborates on each of the system modules. A controlled experimental evaluation of the receiver-agnostic training and collaborative RFFI system carried out in a residential room, is given in Section 5.4. Section 5.5 provides the experimental results in an office building in which the collaborative RFFI is further evaluated. Section 5.6 concludes this chapter.

5.2 System Overview

This chapter presents a receiver-agnostic and collaborative RFFI system. It involves two essential stages, as shown in Figs. 5.1(a) and (b), namely training a receiver-agnostic NN and collaborative inference of multiple receivers. There is also an optional fine-tuning stage shown in Fig. 5.1(c). These stages are summarized below.

Train a Receiver-Agnostic Neural Network

As discussed in Section 5.1, NNs trained with a conventional approach are compromised by the changes in receiver characteristics. Therefore, it is proposed to train a receiver-agnostic NN.

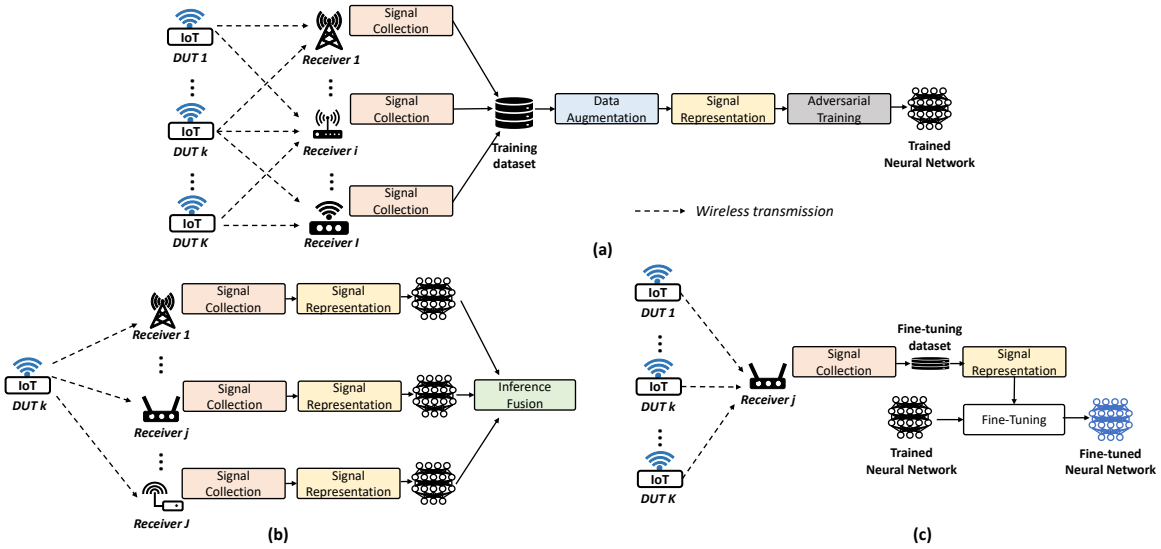


Figure 5.1: Overview of the proposed receiver-agnostic and collaborative RFFI system. (a) Training of a receiver-agnostic NN. (b) Collaborative inference using multiple receivers. The receivers send their predictions to a central server for decision fusion. (c) Fine-tuning on the underperforming receiver. Only a few labeled packets are required and the fine-tuning time is short.

During the training stage, there are K DUTs and I training receivers. Each receiver carries out the signal collection (Section 5.3.1) separately to capture wireless transmissions from the DUTs within range. The captured signals, i.e. IQ samples, along with the transmitter and receiver labels, form the training dataset. The dataset is then augmented with a wireless channel simulator to improve the channel robustness of the NN (Section 5.3.2). The augmented time-domain IQ samples can be converted to appropriate signal representations to be used as the input to the NN (Section 5.3.3). We can then obtain a receiver-agnostic NN by using a gradient reversal layer (Section 5.3.4).

Collaborative Inference of Multiple Receivers

During the inference, K end nodes and J receivers are involved. Note that the J inference receivers can be different from the I training ones. Take the k^{th} end

node as an example, where its transmission will be captured by all receivers in the range thanks to the broadcast nature of wireless transmissions. Each receiver will be equipped with a receiver-agnostic NN. They will first carry out inferences independently, and then the results will be fused to obtain a more reliable prediction of the transmitter label. This part will be explained in Section 5.3.5.

Fine-Tuning

Fine-tuning can be performed at the receiver to further improve classification accuracy when the receiver-agnostic NN does not perform well. A few packets can be collected by the under-performing receiver to slightly update the parameters of the trained NN, which will be elaborated in Section 5.3.6. Note that fine-tuning is optional, which can be adopted when further improvement is required.

5.3 System Design

5.3.1 Signal Collection

The LoRa signal collection algorithms are described in detail in Section 2.3. The processed signals along with the transmitter and receiver labels are stored in the training dataset, given as

$$\mathcal{X}^{train} = \{(\mathbf{y}_m, \mathbf{p}_m, \mathbf{q}_m)\}_{m=1}^{M_{train}}, \tag{5.1}$$

where $\mathbf{y}_m$ is the m^{th} training signal, i.e. IQ samples, and M_{train} is the total number of captured transmissions. $\mathbf{p}_m = \{p_1, ..., p_k, ..., p_K\}$ and $\mathbf{q}_m = \{q_1, ..., q_i, ..., q_I\}$ are the corresponding one-hot encoded transmitter and receiver labels, respectively. Note that $\mathcal{X}^{train}$ for receiver-agnostic training additionally includes receiver labels, $\mathbf{q}$, compared to the training dataset $\mathcal{D}^{train}$ for conventional RFFI systems given in (2.21).

5.3.2 Data Augmentation

Data augmentation is leveraged to improve the system's robustness to channel variation. The implementation details used in this chapter are exactly the same as that introduced in Section 4.3.3. The exponential PDP is employed to characterize the multipath effect and the Jakes model is used to describe the Doppler spectrum.

The training signals are replicated and passed through a channel simulator to emulate multipath and Doppler effects. With data augmentation, the training process can cover as many channel distributions as possible that may be present during inference, thus improving the system's robustness to channel variations. The dataset $\mathcal{X}^{train}$ after data augmentation is given as

$$\mathcal{X}^{train} = \{(\mathbf{y}_m, \mathbf{p}_m, \mathbf{q}_m)\}_{m=1}^{R \cdot M_{train}}, \tag{5.2}$$

where R is the replication factor that indicates how many times the training set is enlarged during data augmentation.

5.3.3 Signal Representation

After data augmentation, the time domain signals can be converted to suitable signal representations as NN inputs. There have been many types of signal representations in previous studies, such as error signal [32], channel independent spectrogram [22], frequency spectrum [16], and differential constellation trace figure [17], to name but a few. Note that the unprocessed IQ samples can also serve as NN inputs after separating the I and Q components as two independent dimensions.

The channel-independent spectrogram proposed in Section 4.3.2 is used as the signal representation. The reason is twofold. Firstly, it is an effective approach to mitigate channel effects, which has been experimentally verified in Chapter 4. Secondly, it is a time-frequency representation that is suitable for CSS modulation. Time-frequency representation has been widely used as NN inputs in LoRa-related deep learning systems [22,106]. In this chapter, it is calculated with a window length of 128 and a hop size of 64. The training dataset $\mathcal{X}^{train}$ after signal representation

is given as

$$\mathcal{X}^{train} = \left\{ (\widetilde{\mathbf{C}}_m, \mathbf{p}_m, \mathbf{q}_m) \right\}_{m=1}^{R \cdot M_{train}}, \tag{5.3}$$

where $\widetilde{\mathbf{C}}_m$ is the channel independent spectrogram converted from $\mathbf{y}_m$.

5.3.4 Receiver-Agnostic Training

Domain adaptation is an effective method to solve the problem of data distribution shift in deep learning [107]. Specific to RFFI, we leverage it to guide the NN to learn receiver-independent features. As shown in Fig. 5.2, there are three components in receiver-agnostic training, namely the feature extractor, transmitter classifier, and receiver classifier.

Feature Extractor

The feature extractor, $g(\cdot)$, converts the input signal representation, $\widetilde{\mathbf{C}}$, to a feature vector, given as

$$feature = g(\widetilde{\mathbf{C}}; \theta), \tag{5.4}$$

where θ denotes the learnable parameters[1] in the feature extractor which will be continuously updated during training. As shown in Fig. 5.2, the designed feature extractor has nine convolutional layers with skip connections and each is activated by a ReLU function. We perform L2 normalization on the extracted feature vector, which can increase system performance [35].

Transmitter Classifier

The transmitter classifier accepts the extracted feature and makes predictions on the transmitter label. The output of the transmitter classifier, $\hat{\mathbf{p}}$, is a list of probabilities.

[1] θ is different from the $\mathbf{\Theta}$ in (2.23). θ only denotes the parameters of the feature extractor while $\mathbf{\Theta}$ represents that of the entire NN.

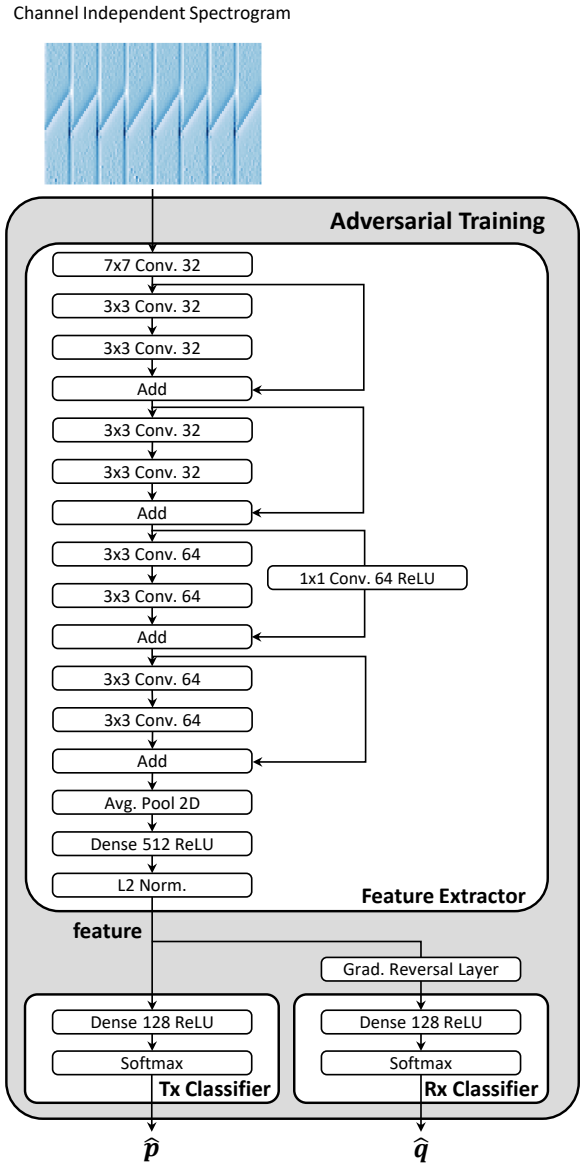


Figure 5.2: Model architecture during receiver-agnostic training.

The k^{th} element, $\hat{p}_k$, is mathematically given as

$$\hat{p}_k = \frac{e^{z_k}}{\sum_{k=1}^{K} e^{z_k}}, \tag{5.5}$$

where z_k is the output of the k^{th} neuron before the softmax activation. We use cross-entropy for the transmitter classifier loss $\mathcal{L}_{tx}$, which is widely adopted in classification problems and is mathematically given as

$$\mathcal{L}_{tx} = -\sum_{k=1}^{K} p_k \log(\hat{p}_k), \tag{5.6}$$

where p_k is the corresponding ground truth in $\mathbf{p}$. Our goal during training is to find feature extractor parameters θ that minimize $\mathcal{L}_{tx}$ to guarantee the performance of the transmitter classifier. The designed transmitter classifier consists of a 128-neuron dense layer activated by the ReLU function and a softmax layer for classification.

Receiver Classifier

The receiver classifier predicts the receiver label from the extracted feature vector. Its loss $\mathcal{L}_{rx}$ is defined as cross-entropy as well, given as

$$\mathcal{L}_{rx} = -\sum_{i=1}^{I} q_i \log(\hat{q}_i), \tag{5.7}$$

where $\hat{q}_i$ is the estimated probability of the packet being captured by the i^{th} receiver and q_i is the ground truth. Our goal during training is to find feature extractor parameters, θ, that restrict the performance of the receiver classifier. Similar to the transmitter classifier, the designed receiver classifier consists of a 128-neuron ReLU-activated dense layer and a softmax layer as well.

Gradient Reversal Layer

The gradient reversal layer was first proposed in [107] to address the distribution shift problem. It does not affect forward propagation and only takes effect during backpropagation. We leverage it to guide the feature extractor to learn receiver-independent features.

As discussed in Section 2.4, the objective of an RFFI system is to predict from

which DUT the signal is sent, which is identical to the goal of the transmitter classifier. Therefore, the performance of the transmitter classifier should be maximized during training, which can be achieved by minimizing the transmitter classifier loss, $\mathcal{L}_{tx}$. In contrast, the goal of the receiver classifier is to predict the receiver label from the extracted feature vector. However, this conflicts with the objective of the proposed receiver-agnostic RFFI protocol since we expect the feature vector to not contain any receiver-related information. This is equivalent to ensuring that the receiver classifier cannot predict the correct receiver label from the feature extractor. Therefore, the performance of the receiver classifier should be restricted during training, which can be achieved by limiting the minimization process of $\mathcal{L}_{rx}$.

The gradient reversal layer can be used to limit the performance of the receiver classifier. In a standard gradient descent process, the θ in the feature extractor is updated by

$$\theta \leftarrow \theta - \mu(\frac{\partial \mathcal{L}_{tx}}{\partial \theta} + \frac{\partial \mathcal{L}_{rx}}{\partial \theta}), \tag{5.8}$$

where μ is the learning rate. The role of the gradient reversal layer is to multiply $\frac{\partial \mathcal{L}_{rx}}{\partial \theta}$ by a negative factor $-\alpha$ during backpropagation, modifying the updating process to

$$\theta \leftarrow \theta - \mu(\frac{\partial \mathcal{L}_{tx}}{\partial \theta} - \alpha\frac{\partial \mathcal{L}_{rx}}{\partial \theta}). \tag{5.9}$$

In other words, the parameters θ of the feature extractor are updated in the opposite direction as instructed by the receiver classifier. Therefore, the receiver classifier cannot be improved as its feedback is not correctly followed. With this approach, we can train a feature extractor to extract transmitter-specific but receiver-independent information.

Once the training is completed, the receiver classifier is removed since we do not need to predict the receiver label in an RFFI system. In the inference stage, the transmitter classifier is directly connected to the feature extractor to predict the transmitter identity.

5.3.5 Collaborative Inference

The LoRaWAN adopts a star-of-stars network topology, which is briefly introduced in Section 2.2.1. As shown in Fig. 2.1, this special topology is suitable for implementing and exploring our proposed receiver-agnostic and collaborative RFFI protocol. Firstly, there are multiple gateways in a LoRaWAN network, and we desire a receiver-agnostic NN that can be directly equipped on all LoRa gateways. Secondly, LoRaWAN already allows one LoRa transmission to be captured by multiple gateways. Applying collaborative RFFI to LoRa/LoRaWAN does not require any changes to the existing communication protocol. Moreover, the LoRa network server has vast computing resources thus the computing-expensive, receiver-agnostic training can be done efficiently.

As illustrated in Fig. 5.1(b), each receiver uses the signal collection module explained in Section 5.3.1 to capture wireless transmissions. The captured signal is then converted to signal representation and fed into the receiver-agnostic NN. The output of the NN at receiver j is a list of probabilities $\hat{\mathbf{p}}^j$. After this, the predicted probability vector $\hat{\mathbf{p}}^j$ and the estimated SNR of the received packet, γ^j, are gathered for collaborative identification. This can be performed on a cloud server or a central node.

The predictions from all the J receivers are then fused. Two fusion schemes are proposed, namely soft fusion and adaptive soft fusion. In the soft fusion scheme, the predictions $\hat{\mathbf{p}}^j$ are directly summed, which is given as

$$\hat{\mathbf{p}}^{fused} = \frac{1}{J}\sum_{j=1}^{J}\hat{\mathbf{p}}^j, \tag{5.10}$$

where $\hat{\mathbf{p}}^{\mathbf{fused}} = \{\hat{p}_1^{fused}, ..., \hat{p}_k^{fused}, ..., \hat{p}_K^{fused}\}$ is the fusion result. While in the adaptive soft fusion scheme, inferences from gateways with higher SNRs are assigned higher weights in the fusion process, which is mathematically expressed as

$$\hat{\mathbf{p}}^{fused} = \frac{1}{J}\sum_{j=1}^{J}\frac{\gamma^j}{\sum_{j=1}^{J}\gamma^j}\hat{\mathbf{p}}^j. \tag{5.11}$$

After fusion, the label corresponding to the highest value in $\hat{\mathbf{p}}^{\mathbf{fused}}$ is returned as the final prediction, formulated as

$$\hat{\ell}^{fused} = \underset{k}{\text{argmax}}(\hat{\mathbf{p}}^{fused}), \tag{5.12}$$

where $\hat{\ell}^{fused}$ is the final predicted label.

5.3.6 Fine-Tuning

The trained receiver-agnostic NN performs well on most receivers. However, sometimes its performance is still not satisfactory. This issue can be alleviated by fine-tuning, which refers to slightly adjusting the parameters of the NN to make it better suited to a new task. We need to first identify the underperforming receiver j as the fine-tuning target, which can be accomplished by comparing the decoded transmitter software address, e.g., MAC address, with the predicted device identity. If there are a significant number of packets where the software addresses do not match the predicted labels, then receiver j is deemed underperforming and requires fine-tuning.

As shown in Fig. 5.1(c), we need to first collect very few labelled signals $\mathbf{y}^j$ using the under-performing receiver j and store them in a fine-tuning dataset $\mathcal{X}^{tune}$, given as

$$\mathcal{X}^{tune} = \left\{(\mathbf{y}_m^j, \mathbf{p}_m)\right\}_{m=1}^{M_{tune}}, \tag{5.13}$$

where M_{tune} is the number of fine-tuning packets. Note that compared to the training dataset $\mathcal{X}^{train}$ in (5.1), $\mathcal{X}^{tune}$ does not contain receiver labels because no receiver-agnostic training is used during fine-tuning. These packets are too few to train a NN from scratch, but sufficient for fine-tuning. Then we convert $\mathbf{y}^j$ to the selected signal representation $\widetilde{\mathbf{C}}^j$, and the data served for fine-tuning becomes $\left\{(\widetilde{\mathbf{C}}_m^j, \mathbf{p}_m)\right\}_{m=1}^{M_{tune}}$. With these data, the parameters of the receiver-agnostic NN can be adjusted with a low learning rate. The cross-entropy loss given in (5.6) is used during fine-tuning.

Once fine-tuning is completed, the fine-tuned NN is updated at the under-performing receiver to obtain higher classification performance. Fine-tuning is feasible on edge LoRa gateways because the fine-tuning data set is small and it can stop within a few

training epochs. Nevertheless, we must recall that this is an optional process.

5.4 Experimental Evaluation in Controlled Environments

In this section, we experimentally evaluate the performance of the proposed receiver-agnostic and collaborative RFFI system in controlled environments, using the LoRa-based case study implementation.

5.4.1 Experimental Setup

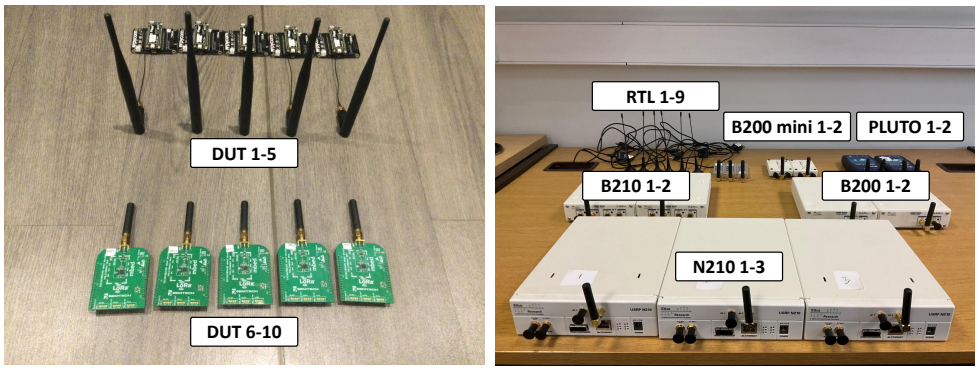
Device Information

Ten LoRa devices are used as the DUTs to be identified and 20 SDR platforms to emulate the LoRa receivers/gateways.

- LoRa DUT: As shown in Fig. 5.3(a), ten LoRa devices of two models, i.e., five of LoPy4[2] and five of mbed SX1261[3] are used. All LoRa DUTs are configured with a spreading factor of seven and a bandwidth of 125 kHz. The carrier frequency was set to 868.1 MHz.

- SDR Receiver: As shown in Fig. 5.3(b), 20 SDR platforms of six models were used to investigate the receiver effect. The detailed SDR information is given in Table 5.1. These SDR platforms are made of different RF chipsets and ADCs of different resolutions so that their hardware characteristics can be considered distinct. MATLAB Hardware Support for SDR[4] was used to access the SDR receivers. Though the MATLAB drivers for SDR receivers are different, all the SDRs run the same code to perform the signal collection procedure introduced in Section 5.3.1. Their receiver sampling rates were all set to 1 MHz.

[2]https://pycom.io/product/lopy4/
[3]https://os.mbed.com/components/SX126xMB2xAS/
[4]https://www.mathworks.com/discovery/sdr.html



(a) (b)

Figure 5.3: Experimental devices. (a) Ten LoRa DUTs. (b) 20 SDR receivers.

Table 5.1: Software-Defined Radios Receivers.

SDR name	Model	ADC	RF Chipset
RTL-1 ∼ RTL-9	RTL-SDR	8 bit	RTL2832U
PLUTO-1, PLUTO-2	ADALM-PLUTO	12 bit	AD9363
B200-1, B200-2	USRP B200	12 bit	AD9364
B200 mini-1, B200 mini-2	USRP B200 mini	12 bit	AD9364
B210-1, B210-2	USRP B210	12 bit	AD9361
N210-1 ∼ N210-3	USRP N210	14 bit	UBX RF Daughterboard

Note that we collect packets from ten DUTs with each of the 20 SDR receivers. Therefore, there are 200 DUT-SDR pairs in total in the dataset.

Dataset Description

Datasets from various SDRs were collected to evaluate the receiver effect. In this section, all the datasets were collected in a typical residential room with LOS be-

tween the DUT and SDR receiver. The distance between the DUT (transmitter) and SDR (receiver) was one meter, hence the received signal had a quite high SNR. In such controlled environments, the wireless channel remained constant. This allowed investigation of the algorithm performance with minimal influence from channel effects. 800 packets were captured from each LoRa DUT-SDR pair, which were then pre-processed and augmented to construct the training dataset. Every test dataset contained 100 packets from each DUT-SDR pair. The detailed descriptions are given in the following subsections.

CNN Training Configuration

We trained various CNN models with different configurations, which differ in two aspects, namely the number of training receivers I, and the types of training receivers. Since each subsection serves a specific evaluation purpose, we trained different CNN models with specially collected training datasets and evaluated them with corresponding test datasets. It is worth noting that all the trained CNN models have the same structure, as introduced in Section 5.3.4.

The number of training receivers, I, is increased from one to five. When $I = 1$, the training can be simplified to the conventional approach introduced in Section 2.4. In this case, the deep learning model is constructed by directly connecting the feature extractor and transmitter classifier shown in Fig. 5.2.

When there are multiple receivers in the training dataset, we further divide the receiver-agnostic training into two categories, namely homogeneous and heterogeneous schemes, based on the diversity of the I training receivers.

- Homogeneous training: a low diversity of the I training receivers, i.e., the hardware characteristics of the I receivers are similar to each other.
- Heterogeneous training: high diversity of the I training receivers, the hardware characteristics of the I receivers are significantly different from each other.

For example, when $I = 5$, homogeneous training receivers are all RTL-SDRs while heterogeneous training receivers are deliberately selected from different SDR models, namely RTL-1, PLUTO-1, B200-1, B200 mini-1, and B210-1.

Training Settings

All the CNN models in this chapter were trained with the same settings. 10% of the training data was split out for validation. The CNN parameters were optimized by the stochastic gradient descent (SGD) optimizer (momentum 0.9) with an initial learning rate of 0.001 and a batch size of 64. The validation loss was monitored during training, and the learning rate was reduced by a factor of 0.2 when the validation loss did not drop within 10 epochs. Training stopped when the validation loss did not change within 20 epochs. The deep learning model was implemented using the TensorFlow library.

When fine-tuning was adopted, a lower learning rate of 0.00001 and a smaller batch size of 32 were used. The fine-tuning process stopped after 20 epochs and no learning rate scheduler was employed.

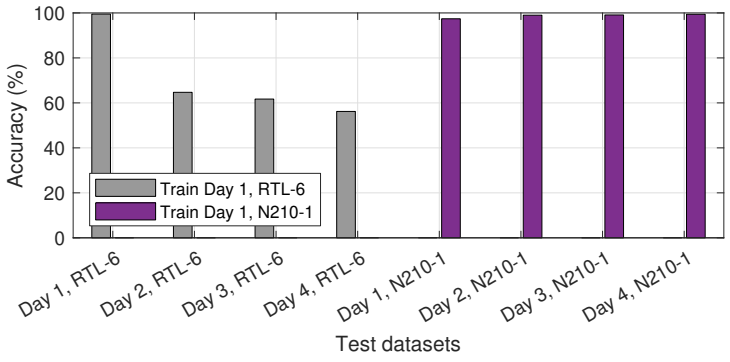
5.4.2 Evaluation of Conventional Training

In this section, the impact of the receiver on RFFI systems is experimentally examined. It is found that the hardware characteristics of low-cost SDRs drift over time, making the conventional RFFI system unstable. Moreover, changing another receiver for signal acquisition also results in serious performance degradation. As only one receiver is involved during training and inference, all the CNN models used in this subsection are trained with the conventional scheme.

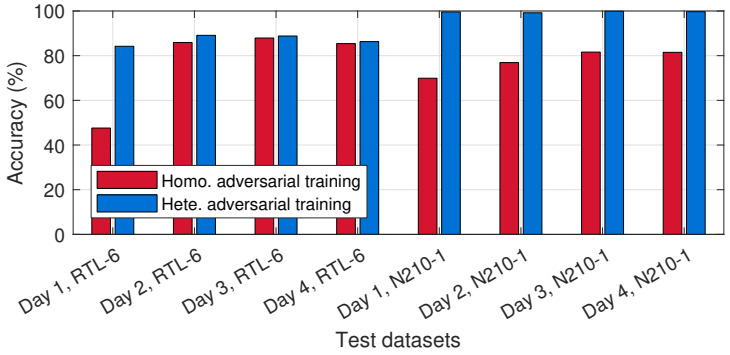
Receiver Drift

We select RTL-6 and N210-1 to represent low-end and high-end SDR platforms, respectively, to study the receiver drift problem. We specifically train two CNNs and test them with the datasets collected on four continuous days:

- Training dataset from RTL-6 Day 1. Test datasets from RTL-6 Day 1, Day 2, Day 3, and Day 4.
- Training dataset from N210-1 Day 1. Test datasets from N210-1 Day 1, Day 2, Day 3, and Day 4.



(a)



(b)

Figure 5.4: Evaluation of the receiver drift problem. (a) Conventional training. (b) Receiver-agnostic training.

The classification results are given in Fig. 5.4(a). It can be observed that the CNN trained with N210-1 is relatively stable over time. The accuracy was nearly unchanged during the four days. However, the performance of RTL-6 degrades seriously by more than 40% on Day 2, 3, and 4. The experimental settings are exactly the same except for the receiver, therefore we reckon the performance difference is due to the unstable hardware characteristics of RTL-SDR. The features of RTL-6 on Day 2, 3, and 4 may be different from Day 1.

Receiver Change

As discussed in Section 2.4, using different receivers for training and inference can lead to a sharp performance decline. To study the receiver change problem, we made the following evaluation:

- Training dataset from RTL-1. Test datasets from the 20 different SDR receivers.
- Training dataset from N210-1. Test datasets from the 20 different SDR receivers.

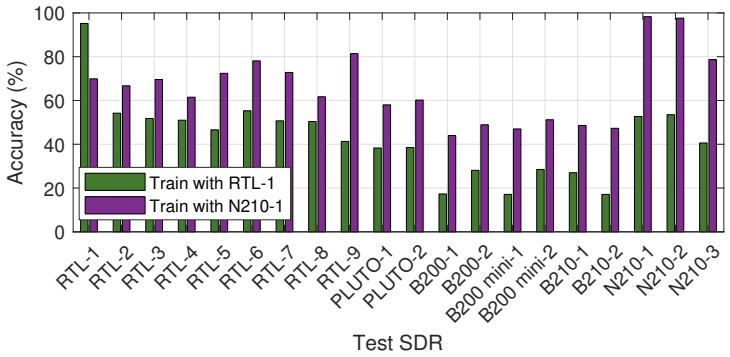
As illustrated in Fig. 5.5(a), the accuracy is high only when the same receiver is used for training and testing. The CNN trained with RTL-1 performs poorly on other receivers, especially on USRP B-series, i.e., B200, B200 mini, and B210. The B200-1 leads to the worst result, with only 20% accuracy. It drops 70% compared to the result with RTL-1. In comparison, the CNN trained with N210-1 performs slightly better. Its performance does not degrade on N210-2, which is likely because N210-2 has similar hardware characteristics to the training receiver N210-1. Beyond that, we can still observe a significant accuracy decline in other SDR receivers.

5.4.3 Evaluation of Receiver-Agnostic Training

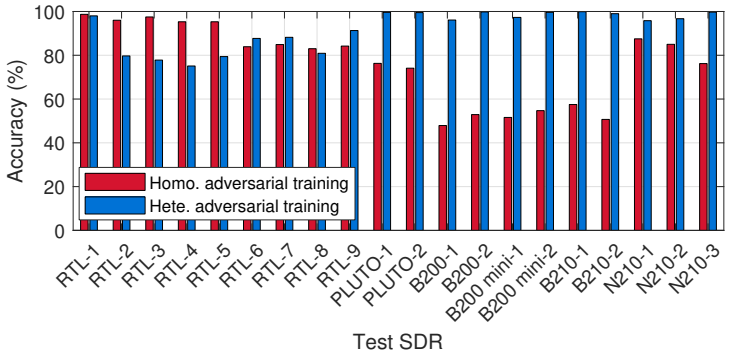
As discussed in Section 5.4.2, the drift of hardware characteristics of low-cost receivers can compromise RFFI stability. In addition, changing another receiver for RFFI also reduces system performance. The receiver-agnostic training can mitigate the performance reduction effectively. We train two CNN models using homogeneous and heterogeneous training strategies, respectively. Five training receivers are involved unless otherwise stated. The training and test configuration is emphasized in each individual subsection.

Effect on Receiver Drift

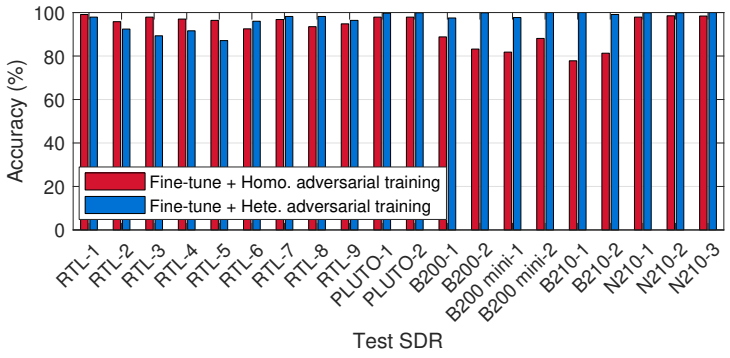
First, we evaluate the performance of receiver-agnostic training on the receiver drift problem. The following evaluation schemes were designed:



(a)



(b)



(c)

Figure 5.5: Evaluation of the receiver change problem, the effect of receiver-agnostic training, and fine-tuning. Test on 20 different receivers. (a) Conventional training. (b) Receiver-agnostic training. (c) Receiver-agnostic training with fine-tuning.

- Train with the homogeneous scheme (RTL-1 to RTL-5). Test on RTL-6 and N210-1 for four consecutive days.
- Train with the heterogeneous scheme (RTL-1, PLUTO-1, B200-1, B200 mini-1, B210-1). Test on RTL-6 and N210-1 for four consecutive days.

The results are given in Fig. 5.4(b). It can be observed that the CNN trained with the heterogeneous scheme (blue bars) performs well on all the test datasets. Moreover, we do not observe any significant performance variation on the RTL-6 datasets collected over four continuous days, indicating that the system is relatively stable after employing the receiver-agnostic training. In contrast, the CNN trained with a homogeneous scheme is still unsatisfactory. In particular, it only achieves around 50% accuracy on the RTL-6 Day 1 dataset. This suggests that the CNN trained with the homogeneous scheme has a poor generalization ability, possibly due to the low diversity of training receivers.

Effect on Receiver Change

The proposed receiver-agnostic training can mitigate the performance degradation caused by receiver changes. In other words, the CNN trained with receiver-agnostic training can be directly applied to new receivers that are not included in the training process. The following evaluations were conducted:

- Train with the homogeneous scheme (RTL-1 to RTL-5). Test on the 20 different SDR receivers.
- Train with the heterogeneous scheme (RTL-1, PLUTO-1, B200-1, B200 mini-1, and B210-1). Test on the 20 different SDR receivers.

The results of receiver-agnostic training are presented in Fig. 5.5(b). The accuracy of all test datasets is greatly improved compared to the results of conventionally trained CNNs in Fig. 5.5(a). The accuracy is always above 75% for the CNN trained with the heterogeneous scheme. We can also find that the heterogeneous training scheme performs better than the homogeneous one, except on RTL-2 to RTL-5. The

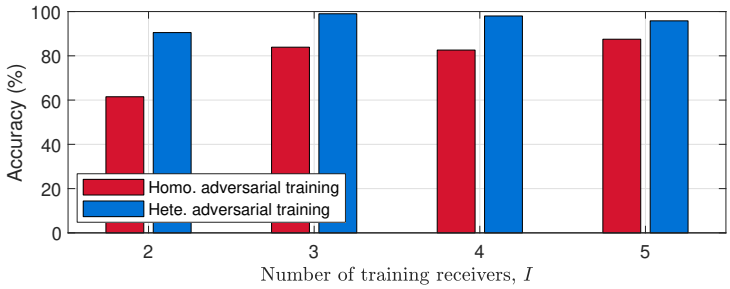


Figure 5.6: The effect of the number of training receivers I on RFFI performance. Test on N210-1, not included in the I training receivers. Both homogeneous and heterogeneous strategies are evaluated.

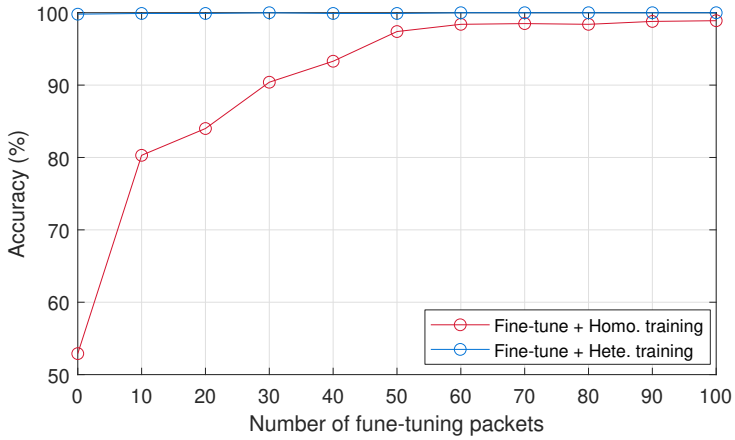
reason for the exception is that RTL-2 to RTL-5 are included during homogeneous training but not in heterogeneous training. The CNN trained with a homogeneous scheme does not generalize well to USRP B series SDRs, i.e., B200, B200 mini, and B210, which is likely because the hardware difference is huge among the training RTL-SDRs and the testing USRP B-series SDRs.

Impact of the Number of Training Receivers

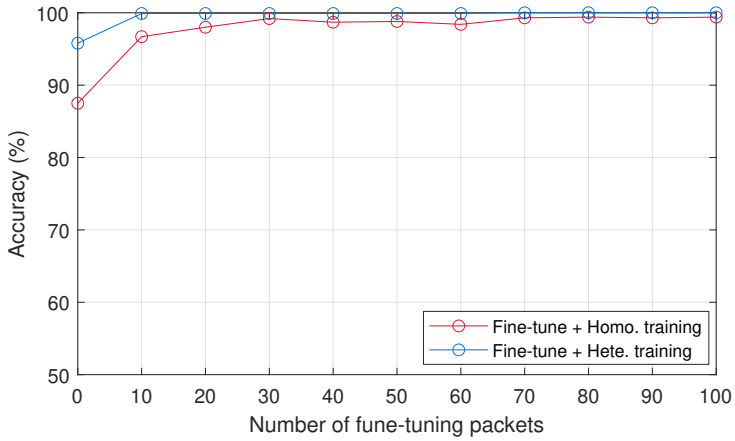
We train CNNs with different numbers of receivers both for homogeneous and heterogeneous schemes. These CNNs are then tested on N210-1 that is not included in the I training receivers. As the result given in Fig. 5.6, in both homogeneous and heterogeneous training, the accuracy gradually increases with the number of receivers. However, the improvement is marginal after I reaches three. The heterogeneous scheme finally achieves higher accuracy than the homogeneous one.

Effect of Fine-Tuning

As revealed in Fig. 5.5(b), although the CNNs trained with a receiver-agnostic scheme can be directly deployed on a new receiver, they still cannot achieve satisfactory performance in some cases. For instance, the CNN trained with the homogeneous scheme performs extremely poorly on USRP B-series SDRs, i.e., B200, B200 mini, and B210.



(a)



(b)

Figure 5.7: The effect of the number of fine-tuning packets. Both homogeneous and heterogeneous schemes are evaluated. Fine-tuning is not employed when the number of packets is zero. (a) Test on B200-2. (b) Test on N210-1.

Fine-tuning the trained CNN model can further improve the performance of new receivers. We use 20 packets from each DUT-SDR pair for fine-tuning and the results are shown in Fig. 5.5(c). It is clear that fine-tuning leads to significant improvements of up to 40% compared to the results in Fig. 5.5(b). It was also found that the homogeneous scheme still underperforms the heterogeneous counterpart even after fine-tuning.

We further investigate the effect of the number of packets used for fine-tuning. We collect different amounts of packets with B200-2 and N210-1 to fine-tune the CNNs and then evaluate the performance after fine-tuning. Note that both B200-2 and N210-1 are not included during any training process, thus we are evaluating the RFFI performance on new receivers. As shown in Fig. 5.7, the classification accuracy increases with the number of fine-tuning packets for both receivers. The CNN trained with a homogeneous scheme improves more significantly because the heterogeneous training has already achieved high accuracy. It can also be observed that after the number of fine-tuning packets reaches 50, the improvement is less noticeable.

The results show that RFFI performance can be significantly improved with less than 50 packets from each DUT-SDR pair, which is affordable for an IoT network. However, this requires the gateway to be able to retrain the NN, i.e. to be capable of forward/backward propagation, parameter updating, etc. Therefore, fine-tuning is not an appropriate solution for low-cost and energy-constrained gateways.

Summary

In conclusion, receiver-agnostic training can effectively mitigate the performance degradation caused by receiver drift and change. It is recommended to use a heterogeneous scheme since better generalization ability can be achieved. Fine-tuning can further improve the system performance even with only 20 packets from each DUT.

5.4.4 Evaluation of Collaborative RFFI

In this subsection, the proposed collaborative RFFI scheme is evaluated. Artificial AWGN is added to the test data to emulate signals collected at various SNRs.

Balanced SNR Scenario

We first consider a simple case where the signals collected by different receivers have the same SNR. In this case, the adaptive fusion in (5.11) can be simplified as the simple fusion in (5.10) since all receivers are assigned the same weights. Seven SDRs that are not included during the receiver-agnostic training are selected for evaluation,

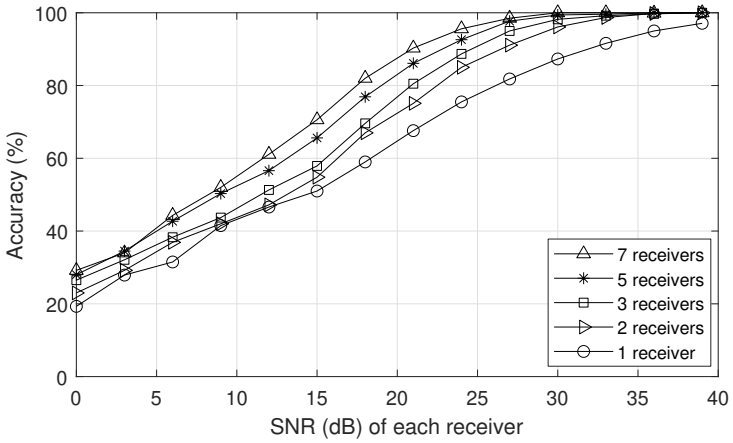


Figure 5.8: Collaborative RFFI in a balanced SNR scenario. In this case, soft fusion is equivalent to adaptive soft fusion. The CNN is trained with a heterogeneous strategy without fine-tuning. Test on seven SDR receivers that are not included during training.

namely PLUTO-2, B200-2, B200 mini-2, B210-2, N210-1, N210-2, and N210-3. Then the CNN trained with the heterogeneous scheme is directly applied without fine-tuning. The classification results are shown in Fig. 5.8. It can be observed that the improvement becomes more significant as more receivers get involved in the collaborative inference. When SNR is between 15-20 dB, the collaborative inference using seven SDR receivers performs 20% better than using an individual receiver.

Imbalanced SNR Scenario

A more common scenario in practice is that the SNRs of packets collected by different LoRa receivers are different. In this case, soft fusion and adaptive soft fusion may lead to different results.

We employ N210-1 to N210-3 for evaluation. The SNR of N210-3 was adjusted from 0 dB to 40 dB, while SNRs of N210-1 and N210-2 are fixed to 10 dB and 20 dB, respectively. The results are shown in Fig. 5.9. The accuracy of N210-3 gradually increases with SNR. It reaches close to 100% when the SNR of N210-3 is over 35 dB. We also show the average accuracy of N210-1 and N210-2 in the figure, which is

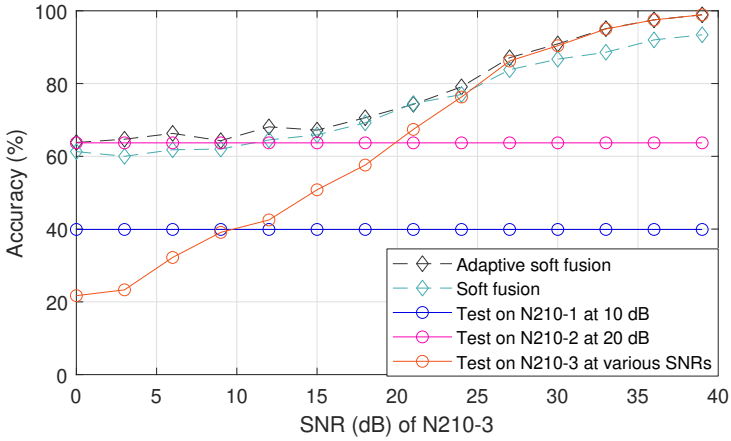


Figure 5.9: Collaborative RFFI in an imbalanced SNR scenario. Three SDR receivers not included during training are used. The SNR of N210-3 is adjusted from 0 dB to 40 dB, while N210-1 and N210-2 are fixed at 10 dB and 20 dB, respectively.

around 40% and 60%, respectively.

In this imbalanced SNR scenario, both soft fusion and adaptive soft fusion schemes are effective when the SNR of N210-3 is below 25 dB. Their accuracy is always higher than any individual receiver. However, we can also see that adaptive soft fusion is less effective when the SNR of N210-3 is over 25 dB. The reason is the SNR of N210-3 is high and the inferences from N210-1 and N210-2 are assigned very low weights. Compared to the adaptive soft fusion, when the SNR of N210-3 is over 25 dB, the soft fusion scheme without weighting leads to even lower accuracy than N210-3 itself. This indicates assigning weights according to SNR is necessary for the collaborative RFFI, thus the adaptive soft fusion scheme is recommended.

5.5 Experimental Evaluation in an Office Environment

In this section, the proposed receiver-agnostic and collaborative RFFI scheme is further investigated and experiments that are closer to practical applications are conducted. Although a preliminary evaluation has been made in Section 5.4, it

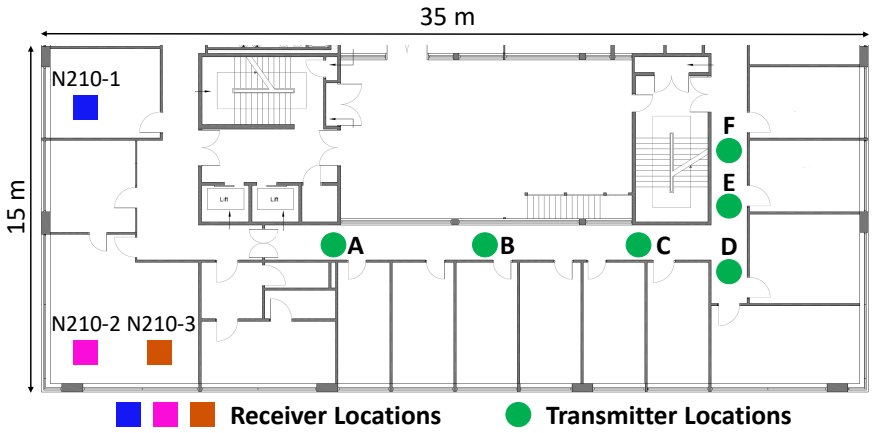


Figure 5.10: Floor plan.

is achieved by adding artificial noise to emulate different SNRs, which still cannot perfectly match the real applications. To further evaluate the collaborative RFFI scheme, we emulate a LoRaWAN network by deploying three SDRs in an office building.

5.5.1 Experimental Setup

The experimental settings are basically the same as in Section 5.4. We collected the test datasets in a typical office building using N210-1, N210-2, and N210-3. The floor plan is given in Fig. 5.10. N210-1 is placed in an office while N210-2 and N210-3 are placed in another meeting room. The LoRa DUTs are in turn located at six locations A-F. Specifically, we run the three SDR receivers simultaneously to collect 300 packets from each DUT-SDR pair in one place and then repeat the collection after moving the DUTs to the following location. The average estimated SNRs of the collected packets at locations A-F are shown in Fig. 5.11. The SNR is estimated by calculating the ratio between the received signal power and the noise floor.

We directly use the CNN trained with the heterogeneous scheme to classify the signals collected at location A-F to evaluate the collaborative RFFI scheme. It is also worth noting that the training data is collected in a residential room in a LOS scenario, which is different from the test environments.

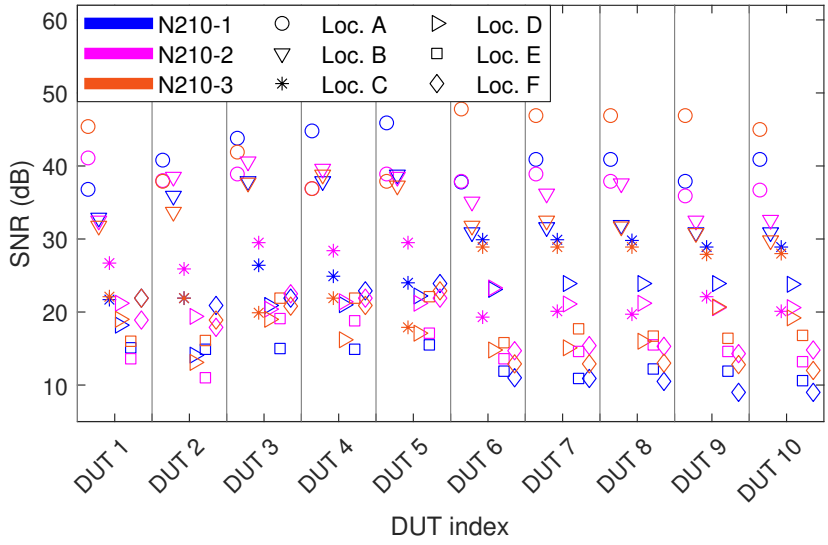


Figure 5.11: Estimated SNRs of the received LoRa packets at locations A-F. Marker colors and symbols represent the SDR receiver and location, respectively.

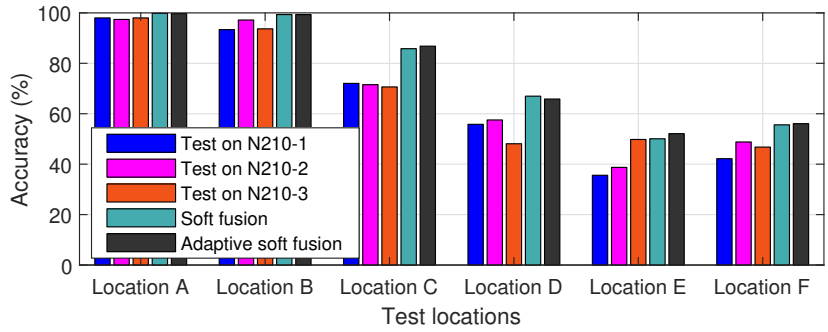


Figure 5.12: Collaborative RFFI in an office building. The DUTs are in turn placed at six locations. N210-1, N210-2, and N210-3 act as LoRa gateways. The CNN trained with a heterogeneous scheme is used without fine-tuning.

5.5.2 Experimental Results

The classification results are illustrated in Fig. 5.12. At locations A-B, our system performs well on each individual receiver with an accuracy of over 85%, thanks to the high SNR, which demonstrates the receiver-agnostic NN is effective. As the ◯ and

$\bigtriangledown$ markers shown in Fig. 5.11, the signals collected at locations A and B are always above 30 dB. However, the performance of the individual receiver gradually decreases at locations C-F as the SNR decreases with the increasing distance. Specifically, as the $*$ and $\triangleright$ markers shown in Fig. 5.11, the SNRs of signals collected at locations C and D are between 15 dB and 30 dB, leading to an accuracy of around 60% for each individual receiver. At locations E and F, represented by $\square$ and $\lozenge$, respectively, the distance between the DUT and all the receivers is above 20 meters, which makes the SNRs of the received signals low to 10 dB and reduces the accuracy for each individual receiver to about 40%.

The collaborative inference is an effective approach to improve performance with higher accuracy than any individual receiver. As shown in Fig. 5.12, the improvement is relatively limited at locations A and B, because the accuracy of each individual receiver is already high. In contrast, the enhancement of collaborative inference is particularly significant when the SNR is lower. Specifically, the classification accuracy can be improved by over 10% at locations C and D. This is consistent with the controlled experiments shown in Fig. 5.8, when the SNR is between 15 dB and 30 dB the improvement of collaborative inference is most significant. As depicted in Fig. 5.11, the signal SNRs at N210-1, N210-2, and N210-3 are not much different when DUTs are placed at the same location. In other words, there is no extremely imbalanced SNR scenario described in Section 5.4.4. Therefore, the soft fusion and adaptive soft fusion schemes achieve nearly the same accuracy.

5.6 Conclusion

In this chapter, a receiver-agnostic and collaborative RFFI scheme is proposed, and LoRa/LoRaWAN is used as a case study for experimental evaluation. Experiments were conducted with 10 COTS LoRa DUTs and 20 SDR receivers in both residential and office building environments. The results show that conventional deep learning-based RFFI systems are seriously affected by the changes in receiver hardware characteristics. We experimentally found that the performance of an RFFI system implemented with low-cost SDR receivers (RTL-SDR) drops by up to 40%

over four continuous days. This may be due to the unstable characteristics of the hardware components in the RTL-SDR. We also found that changing a new SDR for signal collection results in a sharp decline in identification accuracy, up to 70% in some cases. We leverage a receiver-agnostic training approach. More specifically, a gradient reversal layer is employed to guide the NN to learn receiver-independent features. We evaluate the receiver-agnostic NN with 20 different SDR receivers and the identification performance is always maintained above 75%. Fine-tuning can be done by slightly adjusting the parameters of the NN using a few collected packets, which can further improve the performance of the receiver-agnostic NN. The experimental results show that fine-tuning can lead to up to 40% accuracy improvement. Collaborative RFFI with multiple receivers can enhance identification performance. The predictions made by individual receivers can be fused by weighted averaging. The results show that the collaborative RFFI can increase the identification accuracy by up to 20%. Finally, a more realistic experiment is conducted by deploying three USRP N210 SDRs in an office building. The receiver-agnostic NN performs well on these SDRs and the collaborative inference can improve the identification accuracy by 10%.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

This thesis focuses on solving the key challenges in DL-driven RFFI systems. We take LoRa as a case study and build a LoRa-RFFI experimental platform using COTS LoRa and SDR devices. The detailed contributions are summarized below:

- Chapter 3 proposes a stable and efficient LoRa-RFFI system. We experimentally found that the CFO of the received signal is not constant over time and can seriously compromise the identification accuracy. Incorporating CFO compensation as a pre-processing step in the RFFI system can reduce performance degradation.

  The distortion caused by transmitter impairments is often hard to observe in the time-domain waveform, i.e., IQ samples, and designing suitable signal representations as NN inputs can significantly improve identification performance. Three signal representations are investigated in this chapter, namely IQ samples, FFT results, and spectrograms. Various deep learning models, i.e., MLP, CNN, and LSTM, are used to process the designed signal representations. The experimental results demonstrate that the spectrogram-CNN combination achieves the highest accuracy of 95.35% with the least complexity.

  The CFO is a discriminative hardware attribute, however, it must be com-

pensated from the received signals otherwise the system stability will be compromised. In response to this, a hybrid classifier is designed to calibrate the NN outputs using a pre-built CFO database, which can effectively improve identification accuracy by over 20% while maintaining system stability.

- Chapter 4 designs a scalable/openset and channel-robust LoRa-RFFI system. The DL-RFFI is typically defined as a closed-set classification problem. However, this eliminates the RFFI system's scalability and capability to detect rogue devices. This is mainly because the number of neurons at the last softmax layer is fixed in classification NNs. Alternately, a three-stage RFFI framework comprising training, enrolment, and authentication is proposed, which uses triplet loss to train the NN as a feature extractor rather than a classifier. Then the well-known kNN algorithm is employed for rogue device detection and classification.

  A huge challenge in RFFI is that the characteristics of the received signals are affected by the wireless channel, resulting in incorrect identification when the test signals are collected in a channel condition different from that during training. Firstly, a wireless channel simulator was used to emulate more channel conditions and apply them to the training signals, namely data augmentation. Experimental results indicate both multipath and Doppler effects should be considered for optimal performance. Secondly, a signal representation called channel independent spectrogram as the input to the NN is developed, which is based on the fact that the wireless channel does not undergo rapid changes in low-speed scenarios. Extensive experiments are done to evaluate the channel robustness of the RFFI system, and the proposed mitigation strategies are shown to be effective.

- Chapter 5 presents a receiver-agnostic and collaborative RFFI protocol. The received signal is not only affected by the transmitter's impairments but also by the receiver's. The changes in the receiver hardware characteristics can severely degrade the identification accuracy. An approach is proposed for training a receiver-agnostic NN, which is experimentally demonstrated to perform well on

receivers not present during training. Additionally, a fine-tuning technique is also proposed to further increase NN performance by up to 40% while requiring only a few labeled packets.

The broadcast nature of wireless communication enables a signal to be captured by all the receivers within range. It is therefore possible to design a collaborative RFFI protocol to enhance the RFFI performance, particularly in cases with low SNR. The experimental results indicate that the collaborative RFFI can improve identification precision by up to 20%.

6.2 Future Work

The DL-driven RFFI is undergoing rapid development, and there are still problems to be resolved. The suggestions for future work are summarized below:

- **Channel Effects Mitigation for Wide-Band RFFI Systems**:

  The impact of the wireless channel is a huge challenge in RFFI development, which has been discussed in Chapter 4. In this thesis, we propose data augmentation and a channel-independent spectrogram to combat channel effects, which are experimentally shown to be effective. However, the LoRa case study is a narrow-band system and the channel effects are not strong due to the limited bandwidth. How to mitigate channel effects, i.e., severe frequency-selective fading, in wide-band RFFI systems requires further investigation.

- **RFFI in Low SNR Scenarios**: The distortions caused by transmitter impairments are weak and it is still challenging to extract RFFs when the SNR of the received signal is extremely low. Although Chapter 5 proposes a collaborative RFFI that can effectively improve identification accuracy in low SNR scenarios, its performance still cannot be compared with that in high SNR scenarios. The RFFI systems are currently often evaluated in high SNR scenarios where the transceivers are placed close, while the experiments conducted over long distances are still lacking, e.g., LoRa operates in the sub-GHz frequency

band in Europe and can transmit over several kilometers. It is essentially necessary to create and evaluate an RFFI protocol that can operate well in low SNR conditions.

- **RFFI Stability under Large Temperature Variations**

  The RFFI technique relies on the uniqueness of the transmitter hardware impairments. However, it is inevitable that the hardware characteristics can change with temperature and thus compromising system performance. Although Chapter 3 experimentally finds that the oscillator is the primary source of performance degradation and CFO compensation is effective, the experiments are conducted in controlled indoor environments where the room temperature does not change dramatically. It is necessary to design a bespoke testbed to evaluate the RFFI performance in large temperature variations.

- **High-Speed and Energy-Efficient RFFI System**

  The RFFI systems are desired to perform high-speed and energy-efficient identification. Latency is a crucial evaluation metric in a communication system. The identification time must not be much longer than the decoding time to prevent an increase in network latency. Meanwhile, the RFFI systems should not be power-consuming otherwise the cost of receivers will increase. Current experimental testbeds often use a PC to run the RFFI system in an offline manner, and real-time RFFI evaluation is rarely studied. Future work could attempt to apply RFFI to real RF chips, which would greatly accelerate the possible commercialization of RFFI technology.

In summary, the DL-driven RFFI system is still in the laboratory testing stage. Its further development requires contributions from both wireless communication and artificial intelligence experts, both in academia and industry.

Bibliography

[1] M. Giordani, M. Polese, M. Mezzavilla, S. Rangan, and M. Zorzi, "Toward 6G networks: Use cases and technologies," *IEEE Commun. Mag.*, vol. 58, no. 3, pp. 55–61, 2020.

[2] K. Lasse, Lueth, "State of the IoT 2020: 12 billion IoT connections, surpassing non-IoT for the first time," Tech. Rep., Nov. 2020. [Online]. Available: https://iot-analytics.com/state-of-the-iot-2020-12-billion-iot-connections-surpassing-non-iot-for-the-first-time/

[3] G. Zheng, R. Shankaran, M. A. Orgun, L. Qiao, and K. Saleem, "Ideas and challenges for securing wireless implantable medical devices: A review," *IEEE Sensors J.*, vol. 17, no. 3, pp. 562–576, 2016.

[4] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis *et al.*, "Understanding the mirai botnet," in *26th {USENIX} security symposium ({USENIX} Security 17)*, 2017, pp. 1093–1110.

[5] V. Brik, S. Banerjee, M. Gruteser, and S. Oh, "Wireless device identification with radiometric signatures," in *Proc. Int. Conf. Mobile Comput. Netw. (MobiCom)*, San Francisco, CA, USA, Sep. 2008, pp. 116–127.

[6] Y. Shi and M. A. Jensen, "Improved radiometric identification of wireless devices using MIMO transmission," *IEEE Trans. Inf. Forensics Security*, vol. 6, no. 4, pp. 1346–1354, 2011.

[7] P. Liu, P. Yang, W.-Z. Song, Y. Yan, and X.-Y. Li, "Real-time identification of rogue WiFi connections using environment-independent physical features," in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, Paris, France, Apr. 2019, pp. 190–198.

[8] K. Joo, W. Choi, and D. H. Lee, "Hold the door! fingerprinting your car key to prevent keyless entry car theft," in *Proc. Netw. Distrib. Syst. Security Symposium (NDSS)*, Virtual Conference, Feb. 2020.

[9] J. Hua, H. Sun, Z. Shen, Z. Qian, and S. Zhong, "Accurate and efficient wireless device fingerprinting using channel state information," in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, Honolulu, HI, USA, Apr. 2018, pp. 1700–1708.

[10] A. C. Polak and D. L. Goeckel, "Wireless device identification based on RF oscillator imperfections," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 12, pp. 2492–2501, 2015.

[11] L. Peng, A. Hu, J. Zhang, Y. Jiang, J. Yu, and Y. Yan, "Design of a hybrid RF fingerprint extraction and device classification scheme," *IEEE Internet Things J.*, vol. 6, no. 1, pp. 349–360, 2018.

[12] A. C. Polak, S. Dolatshahi, and D. L. Goeckel, "Identifying wireless users via transmitter imperfections," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 7, pp. 1469–1479, 2011.

[13] Y. Li, Y. Ding, J. Zhang, G. Goussetis, and S. K. Podilchak, "Radio frequency fingerprinting exploiting non-linear memory effect," *IEEE Trans. on Cogn. Commun. Netw.*, 2022.

[14] S. Balakrishnan, S. Gupta, A. Bhuyan, P. Wang, D. Koutsonikolas, and Z. Sun, "Physical layer identification based on spatial–temporal beam features for millimeter-wave wireless networks," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 1831–1845, 2019.

[15] N. Wang, W. Li, L. Jiao, A. Alipour-Fanid, T. Xiang, and K. Zeng, "Orientation and channel-independent RF fingerprinting for 5g IEEE 802.11 ad devices," *IEEE Internet Things J.*, vol. 9, no. 11, pp. 9036–9048, 2021.

[16] P. Robyns, E. Marin, W. Lamotte, P. Quax, D. Singelée, and B. Preneel, "Physical-layer fingerprinting of LoRa devices using supervised and zero-shot learning," in *Proc. ACM Conf. Security Privacy Wireless Mobile Netw. (WiSec)*, 2017, pp. 58–63.

[17] L. Peng, J. Zhang, M. Liu, and A. Hu, "Deep learning based RF fingerprint identification using differential constellation trace figure," *IEEE Trans. Veh. Technol.*, vol. 69, no. 1, pp. 1091–1095, 2019.

[18] A. Al-Shawabka, F. Restuccia, S. D'Oro, T. Jian, B. C. Rendon, N. Soltani, J. Dy, K. Chowdhury, S. Ioannidis, and T. Melodia, "Exposing the fingerprint: Dissecting the impact of the wireless channel on radio fingerprinting," in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, Jul. 2020, pp. 646–655.

[19] G. Shen, J. Zhang, A. Marshall, L. Peng, and X. Wang, "Radio frequency fingerprint identification for LoRa using deep learning," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 8, pp. 2604–2616, 2021.

[20] J. Zhang, R. Woods, M. Sandell, M. Valkama, A. Marshall, and J. Cavallaro, "Radio frequency fingerprint identification for narrowband systems, modelling and classification," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 3974 – 3987, 2021.

[21] G. Shen, J. Zhang, A. Marshall, L. Peng, and X. Wang, "Radio frequency fingerprint identification for LoRa using spectrogram and CNN," in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, Virtual Conference, May 2021, pp. 1–10.

[22] G. Shen, J. Zhang, A. Marshall, and J. R. Cavallaro, "Towards scalable and channel-robust radio frequency fingerprint identification for LoRa," *IEEE Trans. Inf. Forensics Security*, vol. 17, pp. 774–787, 2022.

[23] D. Roy, T. Mukherjee, M. Chatterjee, E. Blasch, and E. Pasiliao, "RFAL: Adversarial learning for RF transmitter identification and classification," *IEEE Trans. on Cogn. Commun. Netw.*, vol. 6, no. 2, pp. 783–801, 2019.

[24] M. Cekic, S. Gopalakrishnan, and U. Madhow, "Wireless fingerprinting via deep learning: The impact of confounding factors," in *Proc. Asilomar Conf. Signals, Systems, and Computers*, 2021, pp. 677–684.

[25] J. Yu, A. Hu, G. Li, and L. Peng, "A robust RF fingerprinting approach using multisampling convolutional neural network," *IEEE Internet Things J.*, vol. 6, no. 4, pp. 6786–6799, 2019.

[26] T. Jian, Y. Gong, Z. Zhan, R. Shi, N. Soltani, Z. Wang, J. G. Dy, K. R. Chowdhury, Y. Wang, and S. Ioannidis, "Radio frequency fingerprinting on the edge," *IEEE Trans. Mobile Comput.*, 2021.

[27] N. Soltani, K. Sankhe, J. Dy, S. Ioannidis, and K. Chowdhury, "More is better: Data augmentation for channel-resilient RF fingerprinting," *IEEE Commun. Mag.*, vol. 58, no. 10, pp. 66–72, 2020.

[28] N. Soltani, G. Reus-Muns, B. Salehihikouei, J. Dy, S. Ioannidis, and K. Chowdhury, "RF fingerprinting unmanned aerial vehicles with non-standard transmitter waveforms," *IEEE Trans. Veh. Technol.*, vol. 69, no. 12, pp. 15 518–15 531, 2020.

[29] Y. Qian, J. Qi, X. Kuai, G. Han, H. Sun, and S. Hong, "Specific emitter identification based on multi-level sparse representation in automatic identification system," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 2872–2884, 2021.

[30] A. Al-Shawabka, P. Pietraski, S. B Pattar, F. Restuccia, and T. Melodia, "Deeplora: Fingerprinting LoRa devices at scale through deep learning and data augmentation," in *Proc. ACM Int. Symposium Mob. Ad Hoc Netw. Comput. (MobiHoc)*, Shanghai, China, Jul. 2021.

[31] M. Piva, G. Maselli, and F. Restuccia, "The tags are alright: Robust large-scale rfid clone detection through federated data-augmented radio fingerprinting," in *Proc. ACM Int. Symposium Mob. Ad Hoc Netw. Comput. (MobiHoc)*, Shanghai, China, Jul. 2021.

[32] K. Merchant, S. Revay, G. Stantchev, and B. Nousain, "Deep learning for RF device fingerprinting in cognitive communication networks," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 1, pp. 160–167, 2018.

[33] A. Elmaghbub and B. Hamdaoui, "LoRa device fingerprinting in the wild: Disclosing RF data-driven fingerprint sensitivity to deployment variability," *IEEE Access*, vol. 9, pp. 142 893–142 909, 2021.

[34] S. Hanna, S. Karunaratne, and D. Cabric, "WiSig: A large-scale WiFi signal dataset for receiver and channel agnostic RF fingerprinting," *IEEE Access*, vol. 10, pp. 22 808–22 818, 2022.

[35] R. Xie, W. Xu, Y. Chen, J. Yu, A. Hu, D. W. K. Ng, and A. L. Swindlehurst, "A generalizable model-and-data driven approach for open-set RFF authentication," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 4435–4450, 2021.

[36] S. Rajendran and Z. Sun, "RF impairment model-based IoT physical-layer identification for enhanced domain generalization," *IEEE Trans. Inf. Forensics Security*, vol. 17, pp. 1285–1299, 2022.

[37] L. Ding, S. Wang, F. Wang, and W. Zhang, "Specific emitter identification via convolutional neural networks," *IEEE Commun. Lett.*, vol. 22, no. 12, pp. 2591–2594, 2018.

[38] R. Das, A. Gadre, S. Zhang, S. Kumar, and J. M. Moura, "A deep learning approach to IoT authentication," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2018, pp. 1–6.

[39] B. He and F. Wang, "Cooperative specific emitter identification via multiple distorted receivers," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 3791–3806, 2020.

[40] G. Shen, J. Zhang, A. Marshall, M. Valkama, and J. Cavallaro, "Radio frequency fingerprint identification for security in low-cost IoT devices," in *Proc. Asilomar Conf. Signals, Systems, and Computers*, 2021, pp. 309–313.

[41] Y. Xu, M. Liu, L. Peng, J. Zhang, and Y. Zheng, "Colluding RF fingerprint impersonation attack based on generative adversarial network," in *Proc. IEEE Int. Conf. Commun. (ICC)*. IEEE, 2022, pp. 3220–3225.

[42] K. Merchant and B. Nousain, "Securing IoT RF fingerprinting systems with generative adversarial networks," in *Proc. IEEE Mil. Commun. Conf. (MILCOM)*. IEEE, 2019, pp. 584–589.

[43] Z. Chen, L. Peng, A. Hu, and H. Fu, "Generative adversarial network-based rogue device identification using differential constellation trace figure," *EURASIP Journal on Wireless Communications and Networking*, vol. 2021, no. 1, pp. 1–27, 2021.

[44] H. Xu and X. Xu, "A transformer based approach for open set specific emitter identification," in *Proc. Int. Conf. Comput. Commun. (ICCC)*. IEEE, 2021, pp. 1420–1425.

[45] G. Shen, J. Zhang, A. Marshall, M. Valkama, and J. Cavallaro, "Towards length-versatile and noise-robust radio frequency fingerprint identification," *arXiv preprint arXiv:2207.03001*, 2022.

[46] J. Robinson and S. Kuzdeba, "Riftnet: Radio frequency classification for large populations," in *Proc. IEEE Consumer Commun. Netw. Conf. (CCNC)*. IEEE, 2021, pp. 1–6.

[47] S. Kuzdeba, J. Robinson, and J. Carmack, "Transfer learning with radio frequency signals," in *Proc. IEEE Consumer Commun. Netw. Conf. (CCNC)*. IEEE, 2021, pp. 1–9.

[48] J. Robinson and S. Kuzdeba, "Novel device detection using RF fingerprints," in *Proc. IEEE Computing Commu. Workshop Conf. (CCWC)*. IEEE, 2021, pp. 0648–0654.

[49] G. Li, J. Yu, Y. Xing, and A. Hu, "Location-invariant physical layer identification approach for wifi devices," *IEEE Access*, vol. 7, pp. 106 974–106 986, 2019.

[50] H. Jafari, O. Omotere, D. Adesina, H.-H. Wu, and L. Qian, "IoT devices fingerprinting using deep learning," in *Proc. IEEE Mil. Commun. Conf. (MILCOM)*. IEEE, 2018, pp. 1–9.

[51] K. Merchant and B. Nousain, "Toward receiver-agnostic RF fingerprint verification," in *Proc. IEEE Globecom Workshops (GC Wkshps)*. IEEE, 2019, pp. 1–6.

[52] ——, "Enhanced RF fingerprinting for IoT devices with recurrent neural networks," in *Proc. IEEE Mil. Commun. Conf. (MILCOM)*, 2019, pp. 590–597.

[53] X. Zhou, A. Hu, G. Li, L. Peng, Y. Xing, and J. Yu, "A robust radio frequency fingerprint extraction scheme for practical device recognition," *IEEE Internet Things J.*, 2021.

[54] Z. Zhang, A. Hu, W. Xu, J. Yu, and Y. Yang, "An artificial radio frequency fingerprint embedding scheme for device identification," *IEEE Commun. Lett.*, vol. 26, no. 5, pp. 974–978, 2022.

[55] Y. Xing, A. Hu, J. Zhang, L. Peng, and G. Li, "On radio frequency fingerprint identification for DSSS systems in low SNR scenarios," *IEEE Commun. Lett.*, vol. 22, no. 11, pp. 2326–2329, 2018.

[56] X. Zhou, A. Hu, G. Li, L. Peng, Y. Xing, and J. Yu, "Design of a robust RF fingerprint generation and classification scheme for practical device identification," in *Proc. IEEE Conf. Commun. Netw. Security (CNS)*. IEEE, 2019, pp. 196–204.

[57] B. Hamdaoui and A. Elmaghbub, "Deep-learning-based device fingerprinting for increased LoRa-IoT security: Sensitivity to network deployment changes," *IEEE Network*, vol. 36, no. 3, pp. 204–210, 2022.

[58] P. Yin, L. Peng, J. Zhang, M. Liu, H. Fu, and A. Hu, "LTE device identification based on RF fingerprint with multi-channel convolutional neural network," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, 2021, pp. 1–6.

[59] Q. Pan, Z. An, X. Yang, X. Zhao, and L. Yang, "RF-DNA: large-scale physical-layer identifications of RFIDs via dual natural attributes," in *Proc. Int. Conf. Mobile Comput. Netw. (MobiCom)*, 2022, pp. 419–431.

[60] W. Lee, S. Y. Baek, and S. H. Kim, "Deep-learning-aided RF fingerprinting for NFC security," *IEEE Commun. Mag.*, vol. 59, no. 5, pp. 96–101, 2021.

[61] M. Liu, X. Han, N. Liu, and L. Peng, "Bidirectional IoT device identification based on radio frequency fingerprint reciprocity," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2021, pp. 1–6.

[62] G. Shen, J. Zhang, A. Marshall, R. Woods, J. Cavallaro, and L. Chen, "Towards receiver-agnostic and collaborative radio frequency fingerprint identification," *arXiv preprint arXiv:2207.02999*, 2022.

[63] C.-S. Shieh and C.-T. Lin, "A vector neural network for emitter identification," *IEEE Trans. Antennas Propag.*, vol. 50, no. 8, pp. 1120–1127, 2002.

[64] G. B. Willson, "Radar classification using a neural network," in *Applications of Artificial Neural Networks*, vol. 1294. SPIE, 1990, pp. 200–210.

[65] Y. Xing, A. Hu, J. Zhang, J. Yu, G. Li, and T. Wang, "Design of a robust radio-frequency fingerprint identification scheme for multimode LFM radar," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 10 581–10 593, 2020.

[66] G. Gok, Y. K. Alp, and O. Arikan, "A new method for specific emitter identification with results on real radar measurements," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 3335–3346, 2020.

[67] M. Zhu, X. Zhang, Y. Qi, and H. Ji, "Compressed sensing mask feature in time-frequency domain for civil flight radar emitter recognition," in *IEEE Proc. Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 2146–2150.

[68] Y. Xing, A. Hu, J. Yu, G. Li, L. Peng, and F. Zhou, "A robust radio frequency fingerprint identification scheme for LFM pulse radars," in *Proc. Int. Conf. Wireless Mobile Comp. Netw. Commun. (WiMob)*. IEEE, 2019, pp. 1–6.

[69] H. Ruotsalainen, G. Shen, J. Zhang, and R. Fujdiak, "LoRaWAN physical layer-based attacks and countermeasures, a review," *Sensors*, vol. 22, no. 9, p. 3127, 2022.

[70] M. Liu, J. Wang, N. Zhao, Y. Chen, H. Song, and F. R. Yu, "Radio frequency fingerprint collaborative intelligent identification using incremental learning," *IEEE Trans. Netw. Sci. Eng.*, vol. 9, no. 5, pp. 3222–3233, 2021.

[71] Y. Liu, J. Wang, J. Li, S. Niu, and H. Song, "Class-incremental learning for wireless device identification in iot," *IEEE Internet Things J.*, vol. 8, no. 23, pp. 17 227–17 235, 2021.

[72] Y. Liu, J. Wang, J. Li, H. Song, T. Yang, S. Niu, and Z. Ming, "Zero-bias deep learning for accurate identification of Internet-of-Things (IoT) devices," *IEEE Internet Things J.*, vol. 8, no. 4, pp. 2627–2634, 2020.

[73] H. Zha, Q. Tian, and Y. Lin, "Real-world ADS-B signal recognition based on radio frequency fingerprinting," in *Proc. IEEE Int. Conf. Netw. Protocols (ICNP)*, 2020, pp. 1–6.

[74] G. Oligeri, S. Sciancalepore, S. Raponi, and R. Di Pietro, "Past-ai: Physical-layer authentication of satellite transmitters via deep learning," 2022.

[75] Q. Jiang and J. Sha, "Rf fingerprinting identification based on spiking neural network for leo-mimo systems," *IEEE Wireless Commun. Lett.*, 2022.

[76] S. D. Andrews, "Extensions to radio frequency fingerprinting," Ph.D. dissertation, Virginia Tech, 2019.

[77] J. Gong, X. Xu, and Y. Lei, "Unsupervised specific emitter identification method using radio-frequency fingerprint embedded InfoGAN," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 2898–2913, 2020.

[78] W. Wang, Z. Sun, S. Piao, B. Zhu, and K. Ren, "Wireless physical-layer identification: Modeling and validation," *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 9, pp. 2091–2106, 2016.

[79] B. Danev, T. S. Heydt-Benjamin, and S. Capkun, "Physical-layer identification of RFID devices." in *Proc. USENIX Security Symposium*, 2009, pp. 199–214.

[80] Y. Lin, J. Jia, S. Wang, B. Ge, and S. Mao, "Wireless device identification based on radio frequency fingerprint features," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2020, pp. 1–6.

[81] J. Zhang, F. Wang, O. A. Dobre, and Z. Zhong, "Specific emitter identification via Hilbert–Huang transform in single-hop and relaying scenarios," *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 6, pp. 1192–1205, 2016.

[82] U. Satija, N. Trivedi, G. Biswal, and B. Ramkumar, "Specific emitter identification based on variational mode decomposition and spectral features in single hop and relaying scenarios," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 3, pp. 581–591, 2018.

[83] Y. Jiang, L. Peng, A. Hu, S. Wang, Y. Huang, and L. Zhang, "Physical layer identification of LoRa devices using constellation trace figure," *EURASIP J. Wireless Communications and Networking*, vol. 2019, no. 1, p. 223, 2019.

[84] K. Sankhe, M. Belgiovine, F. Zhou, L. Angioloni, F. Restuccia, S. D'Oro, T. Melodia, S. Ioannidis, and K. Chowdhury, "No radio left behind: Radio fingerprinting through deep learning of physical-layer hardware impairments," *IEEE Trans. on Cogn. Commun. Netw.*, vol. 6, no. 1, pp. 165–178, 2019.

[85] S. Chen, S. Zheng, L. Yang, and X. Yang, "Deep learning for large-scale real-world ACARS and ADS-B radio signal classification," *IEEE Access*, vol. 7, pp. 89 256–89 264, 2019.

[86] S. Hanna, S. Karunaratne, and D. Cabric, "Open set wireless transmitter authorization: Deep learning approaches and dataset considerations," *IEEE Trans. on Cogn. Commun. Netw.*, vol. 7, no. 1, pp. 59–72, 2020.

[87] A. Gritsenko, Z. Wang, T. Jian, J. Dy, K. Chowdhury, and S. Ioannidis, "Finding a 'new' needle in the haystack: Unseen radio detection in large populations using deep learning," in *Proc. IEEE Int. Symposium Dynamic Spectr. Access Netw. (DySPAN)*, Newark, NJ, USA, 2019, pp. 1–10.

[88] K. Sankhe, M. Belgiovine, F. Zhou, S. Riyaz, S. Ioannidis, and K. Chowdhury, "ORACLE: Optimized radio classification through convolutional neural networks," in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, Paris, France, 2019, pp. 370–378.

[89] C. Morin, L. Cardoso, J. Hoydis, and J.-M. Gorce, "Deep Learning-based Transmitter identification on the physical layer," Dec. 2020, working paper or preprint. [Online]. Available: https://hal.inria.fr/hal-03117090

[90] C. Morin, L. S. Cardoso, J. Hoydis, J.-M. Gorce, and T. Vial, "Transmitter classification with supervised deep learning," in *Proc. Int. Conf. Cogn. Radio Oriented Wireless Netw.* Springer, 2019, pp. 73–86.

[91] S. Andrews, R. M. Gerdes, and M. Li, "Crowdsourced measurements for device fingerprinting," in *Proc. ACM Conf. Security Privacy Wireless Mobile Netw. (WiSec)*, 2019, pp. 72–82.

[92] O. B. Seller and N. Sornin, "Low power long range transmitter," U.S. Patent 9,252,834, Feb., 2016.

[93] "LoRaWAN® Regional Parameters," Semtech, Tech. Rep., Feb. 2020, accessed on 14 Dec., 2020. [Online]. Available: https://lora-alliance.org/sites/default/files/2020-06/rp_2-1.0.1.pdf

[94] T. M. Schmidl and D. C. Cox, "Robust frequency and timing synchronization for OFDM," *IEEE Trans. Commun.*, vol. 45, no. 12, pp. 1613–1621, 1997.

[95] P. Robyns, P. Quax, W. Lamotte, and W. Thenaers, "A multi-channel software decoder for the LoRa modulation scheme," in *Proc. Int. Conf. Internet Things, Big Data Security (IoTBDS)*, Mar. 2018, pp. 41–51.

[96] "LoRa Modulation Crystal Oscillator Guidance," Semtech, Tech. Rep. AN1200.14, Jul. 2019, accessed on 14 Dec., 2020. [Online]. Available: https://lora-developers.semtech.com/library/product-documents/

[97] 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.

[98] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vision Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[99] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[100] I. Agadakos, N. Agadakos, J. Polakis, and M. R. Amer, "Chameleons' oblivion: Complex-valued deep neural networks for protocol-agnostic RF device finger-

printing," in *Proc. IEEE European Symposium Security Privacy (EuroS&P)*, Virtual Conference, 2020, pp. 322–338.

[101] R. Ranjan, C. D. Castillo, and R. Chellappa, "L2-constrained softmax loss for discriminative face verification," *arXiv preprint arXiv:1703.09507*, 2017.

[102] K. Musgrave, S. Belongie, and S.-N. Lim, "A metric learning reality check," in *Proc. European Conf. Computer Vision (ECCV)*, 2020, pp. 681–699.

[103] K. Kobs, M. Steininger, A. Dulny, and A. Hotho, "Do different deep metric learning losses lead to similar learned features?" in *Proc. Int. Conf. on Computer Vision (ICCV)*, 2021, pp. 10 644–10 654.

[104] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proc. IEEE Conf. Comput. Vision Pattern Recognition (CVPR)*, 2015, pp. 815–823.

[105] B. Danev and S. Capkun, "Transient-based identification of wireless sensor nodes," in *Proc. ACM/IEEE Int. Conf. Inf. Process. Sensor Netw. (IPSN)*, NW Washington, DC, USA, 2009, pp. 25–36.

[106] C. Li, H. Guo, S. Tong, X. Zeng, Z. Cao, M. Zhang, Q. Yan, L. Xiao, J. Wang, and Y. Liu, "NELoRa: Towards ultra-low SNR LoRa communication with neural-enhanced demodulation," in *Proc. ACM Conf. Embedded Netw. Sensor Systems (SenSys)*, 2021, pp. 56–68.

[107] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *Proc. Int. Conf. Mach. Learn. (ICML)*, Lille, France, July 2015, pp. 1180–1189.