Fall 7-2023

Spatial & Temporal Agnostic Deep-Learning Based Radio Fingerprinting

Fahmida Afrin
University of Nebraska-Lincoln, fafrin2@huskers.unl.edu

SPATIAL & TEMPORAL AGNOSTIC DEEP-LEARNING BASED RADIO FINGERPRINTING

by

Fahmida Afrin

A THESIS

Presented to the Faculty of

The Graduate College at the University of Nebraska

In Partial Fulfilment of Requirements

For the Degree of Master of Science

Major: Computer Science

Under the Supervision of Professor Nirnimesh Ghose

Lincoln, Nebraska

August, 2023

SPATIAL & TEMPORAL AGNOSTIC DEEP-LEARNING BASED RADIO FINGERPRINTING

Fahmida Afrin, M.S.

University of Nebraska, 2023

Adviser: Nirnimesh Ghose

Radio fingerprinting is a technique that validates wireless devices based on their unique radio frequency (RF) signals. This method is highly feasible because RF signals carry distinct hardware variations introduced during manufacturing. The security and trustworthiness of current and future wireless networks heavily rely on radio fingerprinting. In addition to identifying individual devices, it can also differentiate mission-critical targets. Despite significant efforts in the literature, existing radio fingerprinting methods require improved robustness, scalability, and resilience. This study focuses on the challenges of spatial-temporal variations in the wireless environment. Many prior approaches overlook the complex numerical structure of the in-phase and quadrature (I/Q) data by treating real and imaginary components separately. This approach results in the loss of essential information encoded in the signal's phase and amplitude, leading to lower accuracy. This thesis proposes several enhancements. First, we treat the entire complex structure of the I/Q data as a single input to a complex-valued convolutional neural network (CVNN), thereby improving the model's accuracy. Second, conduct extensive experiments to determine optimal pre-processing parameters, ensuring that over-optimistic conclusions about RF fingerprinting performance are avoided. Third, we compare various activation functions and transfer learning-based fine-tuning and a triplet network to address the variations the wireless environment introduces in scenarios involving different loca-

tions and times. We use the concept of a "device rank" metric to perform device identification with certainty based on RF fingerprinting. Our work concretely proves that CVNN outperforms CNN for radio fingerprinting. Concatenated Rectified Linear Units (CReLU) activation function and fine-tuning-based transfer learning perform the best for cross-location and time device fingerprinting.

DEDICATION

To my parents for always supporting me and making it possible for me to chase my dreams.

ACKNOWLEDGMENTS

First of all, I would like to express my sincere gratitude and appreciation to my academic advisor Dr. Nirnimesh Ghose for his advice, support, motivation, and patience throughout the last two years. Through his vast knowledge and experience, he taught me where to explore, how to think while performing research, and how to present my work. I am extremely fortunate to have him as my advisor and as my mentor. This thesis would have been impossible without his guidance.

I would like to thank Dr. Byrav Ramamurthy and Dr. Lisong Xu for serving on my master's committee and providing insightful and valuable input that helped me improve this thesis.

I am extremely grateful to my mother (Kohinoor Parvin), and my father (Anisur Rahman) for their love, prayers, support, and guidance throughout my life; you have always been my source of inspiration, confidence, and success. Your love, care, and support give me the strength to overcome any difficulties that I may face.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Traditional authentication methods based on cryptography, such as passwords or digital certificates, have limitations when identifying wireless devices. This is primarily due to the characteristics of wireless communication, including high device mobility, the broadcasting nature of wireless transmissions, multiple protocols, the lack of a trusted root authority, and an expanded attack surface [7]. To address these challenges, researchers and practitioners have explored alternative authentication methods, and one promising approach is *Radio Fingerprinting*. Radio fingerprinting involves authenticating wireless devices to analyze radio frequency (RF) signals at the physical communication layer. By examining the unique hardware variations present in RF signals, a receiver can differentiate between different transmitters, even if they are of the same type.

These hardware variations stem from non-linearities introduced during the manufacturing process of radio frequency circuitry. Factors such as I/Q (in-phase and quadrature) imbalance, non-linearity of digital-to-analog converters, and non-linearity of power amplification contribute to the distinct characteristics of RF signals emitted by individual devices [15]. This provides an opportunity to create a unique fingerprint for each device, which can be used for authentication. Radio fingerprinting has demonstrated its potential for enhancing the security and trust of wireless networks.

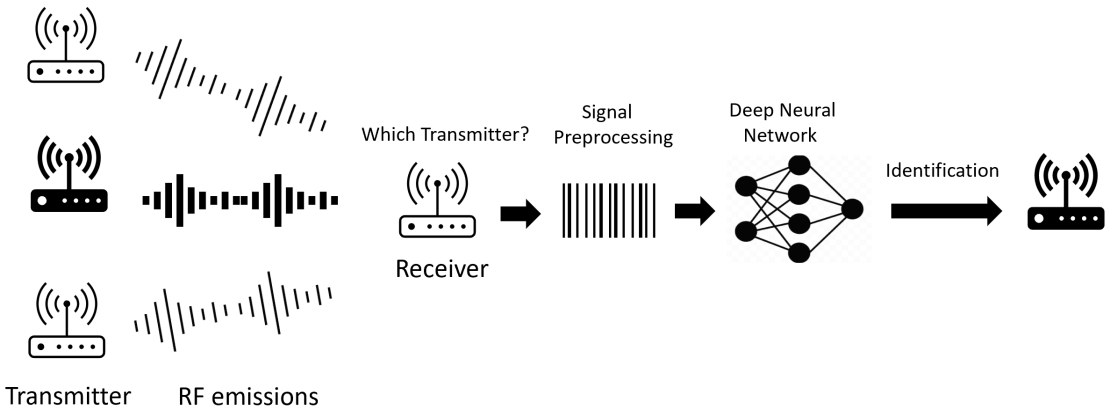


Figure 1.1: Thesis Overview

It can be applied not only to authenticate individual devices like smartphones [22] and Internet of Things (IoT) [18] devices but also to identify mission-critical targets such as Unmanned Aerial Vehicles (UAVs) [30].

1.1 Motivation

CNNs have been widely used for radio fingerprinting to differentiate between different transmitters [2, 11, 14, 15, 19, 21, 24, 33, 34]. The work most closely related to ours is [15], where the authors improved the robustness of deep-learning-based radio fingerprinting by selecting parameters in the pre-processing stage and later employed adversarial domain adaptation and device rank to enhance performance in cross-day scenarios in the frequency and time-frequency domains. However, in all *these previous works, researchers treated the real and imaginary parts of complex numbers separately.* This separation may hinder the model's ability to learn their relationship, potentially leading to sub-optimal performance.

Several studies have explored complex-valued convolutional neural network-based radio fingerprinting over I/Q data, including [6, 10, 16, 29]. The complex-valued rep-

resentation captures the RF signals' amplitude and phase information, providing a richer description than traditional real-valued representations. By leveraging the complex-valued I/Q data, we aim to enhance the discriminative power of the fingerprints extracted by deep learning models. The complex representation allows the neural networks to learn more nuanced relationships and patterns in the RF signals, potentially improving the accuracy of device identification proved through extensive experimentation. While these studies demonstrated the superior performance of CVNNs compared to CNNs, they have *yet to offer a robust solution, particularly one that allows training the fingerprinting at a different time and location from where the transmitter detection must be performed.*

The objective of this thesis is to present a model that incorporates the following key-important elements, in order to progress the prior work done to improve the authentication of wireless devices in a network:

- When incorporating the I/Q data into deep learning models, it is essential to preserve the complex arrangements. The real and imaginary components should be integrated into a one-dimensional input for the model while keeping their intricate structures, as opposed to being handled as separate two-dimensional pieces of data.

- Even in situations like constantly changing wireless channels and other influencing factors, the deep learning model should be capable of categorizing devices based on location and time data.

- By using radio fingerprinting, the proposed strategy should offer security, protecting the network from any potential attacks. The architecture must have robust security features to prevent unauthorized access.

1.2 Contribution

Our work addresses the challenges of fast-changing wireless channel variations using fine-tuning and triplet networks [17], [26]. To enhance performance, we propose integrating Complex-Valued Neural Networks (CVNNs) with transfer learning, as shown in Fig. 1.1. Transfer learning is a technique where knowledge acquired from one task or domain is applied to a related but different task or domain. In our approach, we leverage a pre-trained model, which has been trained on a large dataset from a source task. This pre-trained model serves as a starting point for the target task, as it has already learned valuable features and representations that can benefit it. Additionally, we employ a triplet network consisting of three identical deep neural networks, referred to as sub-networks, sharing the same weights and hyperparameters. This network architecture allows us to exploit the benefits of transfer learning within the triplet network framework.

Our method involves three phases: the training phase, the tuning phase, and the testing phase. The training phase utilizes the entire source dataset, typically a large-scale dataset. The target dataset, typically a smaller-scale dataset, is divided into target tuning data and target testing data. This division enables us to fine-tune the pre-trained model specifically for the target task and adapt it to different environments. By combining transfer learning and the triplet network approach, our method effectively tackles the spatiotemporal variations in the wireless channel. It enables the swift fine-tuning of already fingerprinted devices for different environments, improving performance in fast-changing wireless channel conditions.

In contrast to prior art in radio-fingerprinting [2,6,10,11,14–16,19,21,24,29,33,34]:

- We present a two-pronged enhancement approach: firstly, employing a CVNN to retain the information of complex I/Q data, and secondly, combining it with

transfer learning to account for spatio-temporal variations of the wireless channel.

- We conduct data collection on a B-205mini-i USRP testbed, which consists of six transmitters and one receiver, at different times in four locations. The data collection includes three indoor environments located in two separate buildings and one outdoor environment.

- We conducted a comparison of different activation functions (Cart Leaky ReLU, Complex Cardioid, and CReLU). The results reveal that CReLU outperforms the other activation functions when CVNN is trained and tested with real-world I/Q data.

- We conducted a further comparison of two different algorithms for transfer learning: fine-tuning and triplet network. The results indicate that fine-tuning performs the best with CVNNs, especially when there is a large amount of labeled I/Q data available for wireless data. Additionally, our results provide evidence supporting our hypothesis that CVNNs improve accuracy compared to the prior art [15].

- We also introduced a novel evaluation metric called "device ranking," which addresses variations in accuracy when identifying devices with certainty.

Chapter 2

Related Work

2.1 Deep Learning-based Authentication

Several studies have utilized CNNs for radio fingerprinting to distinguish between different transmitters [2, 11, 14, 15, 19, 21, 24, 33, 34]. AlShawabka *et al.* conducted extensive data collection (20 USRPs as transmitters and 1 USRP as a receiver) to evaluate the impact of the wireless channel on CNN-based radio fingerprinting algorithms and improve fingerprinting accuracy based on the current channel conditions [2]. They employed three CNN architectures: homegrown, baseline, and ResNet-50-1D. Wang *et al.* proposed RF device identification using transfer learning and long short-term memory. It demonstrates that an LSTM model based on transfer learning can demonstrate greater identification than an LSTM [33]. Hanna *et al.* developed a nonlinear model generator and evaluated RF fingerprinting performance under various conditions and types of transmitted data using CNN with the magnitude of the frequency. And shows that the best performance comes from a CNN with a magnitude of the frequency representation of the data.

[11]. Ying *et al.* proposed a novel technique using a sophisticated convolutional neural network to individually recognize communication radiation sources individually, outperforming other methods with 99.7% accuracy, successfully capturing non-

linearity, and requiring less data than constellation-based algorithms [34]. Ohtsuji *et al.* demonstrated that the classification performance of CNNs for identification decreases in low SNR environments. To address this issue, they proposed a CNN-based classification method using logarithmic power spectral density for radio identification, enabling the distinction of WLAN devices [19].

2.2 CVNN Based Radio Fingerprinting

The work closest to ours is [15], where the authors enhanced the robustness of deep-learning-based radio fingerprinting through parameter selection in pre-processing and later used adversarial domain adaptation and device rank to improve performance in cross-day scenarios in the frequency and time-frequency domains. Furthermore, Soltanieh *et al.* reviewed physical-layer identification and state-of-the-art RF fingerprinting for wireless devices. They also compared the performance of transient-based signal extraction algorithms, as detecting transients affects identification success [28]. However, in all these prior works, researchers considered the real and imaginary parts of complex numbers separately. If we separate the real and imaginary parts, the model may require assistance learning their relationship, potentially leading to sub-optimal performance.

This was attested by prior studies that have looked into complex-valued convolutional neural network-based radio fingerprinting over I/Q data, including [6,10,16,29]. Chen *et al.*, who conducted a detailed analysis of CVNN and demonstrated its superiority over real-valued neural networks for four different datasets. They employed the complex activation function "CReLU". However, we have shown more than one complex activation function in our work [6]. Liang *et al.* proposed a model employing a one-dimensional complex-valued neural network to enhance the automatic

modulation recognition accuracy of I/Q data [16]. Gopalakrishnan *et al.* investigated the performance of CVNN for two distinct wireless protocol datasets, ADS-B and WiFi, using complex-valued CNN architectures and complex activation functions (ModReLU and CReLU) [10]. Additionally, Stankowicz *et al.* explored input/output representations and the treatment of complex values in fully connected neural networks, highlighting the advantages of complex-valued inputs and spectro-temporal representations [29].

While these works demonstrated the improved performance of CVNNs compared to CNNs, they did not offer a comprehensive solution. In our work, we enhance the prior art by combining CVNNs with transfer learning to develop a radio fingerprinting technique agnostic to space and time. This enables us to train models in a different environment where the transmitters are recognized. Additionally, we compare the effectiveness of three different activation functions. Furthermore, in contrast to previous CVNN approaches, we collect data on the B-205mini-i USRP testbed to evaluate the performance of our model thoroughly.

Chapter 3

Background

3.1 System Model

Our implementation of spatiotemporal agnostic device fingerprinting utilizes a system model consisting of six transmitters and one receiver. As depicted in Figure 3.1, the primary objective of the receiver is to carry out spatiotemporal agnostic deep-learning-based radio fingerprinting, which involves authenticating the identity of each transmitter based on the RF signals received, irrespective of the wireless environmental conditions. The radio fingerprinting process encompasses two main phases: the training phase and the authentication phase (also known as the test phase).

During the training phase, each transmitter in our system transmits radio frequency (RF) signals to the receiver. These RF signals, specifically represented as in-phase and quadrature (I/Q) data, are collected and utilized as training data. The I/Q data can be expressed in different domains, such as the time domain (before applying the Fast Fourier Transform, FFT), the frequency domain (after applying FFT), or the time-frequency domain (using techniques like spectrogram through Short-Time Fourier Transform) [2]. These representations capture different characteristics of the RF signals and have been explored in related works [11, 13, 23, 27, 28].

After receiving the *training data*, the receiver transmits it to a server responsible

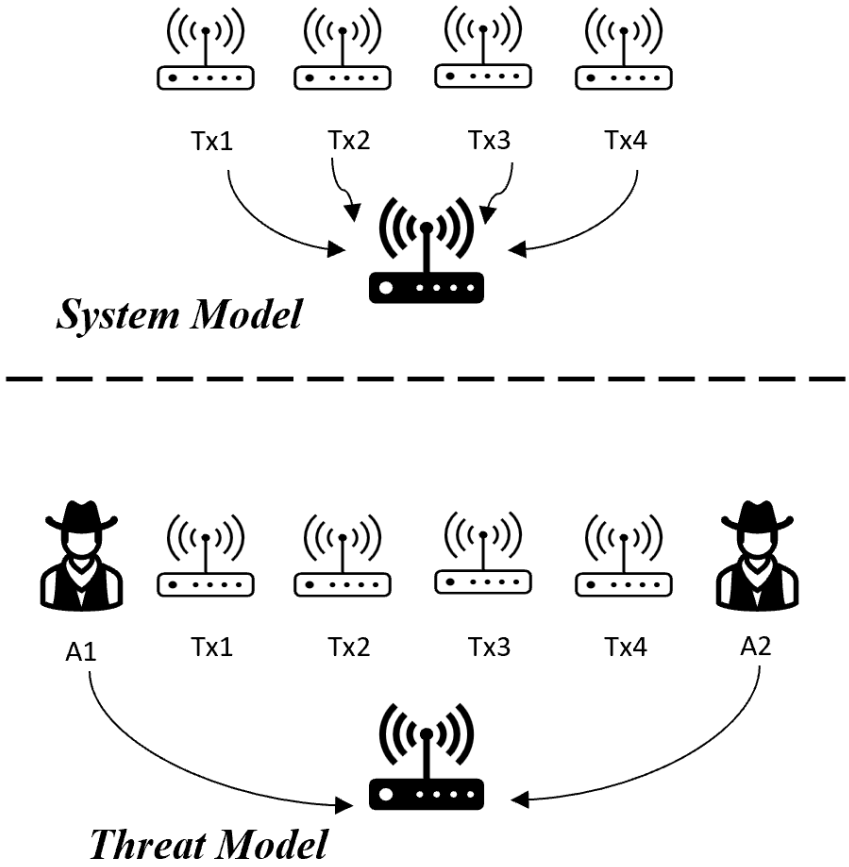


Figure 3.1: The system model and threat model of radio fingerprinting. In the authentication phase, A1 or A2 are regarded as potential attackers who could impersonate Tx1, Tx2, Tx3, or Tx4.

for training a classifier. The server leverages this training data to build a model to classify and distinguish the RF signals from different transmitters. The presence of a server in the system responsible for training the classifier and carrying out predictions is optional. Depending on the computational resources available on the receiver side, the prediction (and even the training) can be performed locally without relying on a server. Whether the server is included or not does not impact the threat model or the system's overall functioning. The decision to include a server or perform all computations locally depends on the receiver's capabilities and requirements.

In the *authentication phase*, when the receiver receives RF signals during operation, it queries the trained classifier to predict the transmitter's identity. The classifier determines which transmitter is transmitting at a given time by comparing the received RF signals with the learned patterns from the training phase.

3.2 Threat Model

The proposed threat model assumes that all parties involved in the training phase are trusted and have no intention of deceiving or manipulating the system. However, during the authentication phase, potential attackers may exist to impersonate other transmitters by modifying the RF signals. These attackers can fall into two categories: those who participated in the training phase and those who did not.

Attackers who participated in the training phase, such as Tx3 impersonating Tx1 or Tx2 in Figure 3.1, can modify the I/Q data in the RF signals. They can manipulate the RF signals by pre-processing the signals before transmission or by introducing noise. The objective of these modifications is to mislead the classifier and induce incorrect predictions. Furthermore, attackers who did not participate in the training phase, such as Tx4 impersonating Tx1, Tx2, or Tx3 in Figure 3.1, can also modify the I/Q data in the RF signals. They can manipulate the signals to deceive the classifier and generate incorrect identification results.

3.3 Fingerprinting Non-Linear Hardware Manufacturing Variations on RF Signals.

Non-linear deviations are inherent in wireless transmitters due to hardware's manufacturing variations, even for devices manufactured using the same process [5]. These imperfections arise from variations in components such as digital-to-analog converters,

power amplifiers, transistors, resistors, inductors, and capacitors. The collective effect of these variations is considered unique to each transmitter and forms a distinctive hardware signature.

These hardware variations introduce non-linearities in the processing of radio frequency (RF) signals, presenting opportunities to identify transmitter hardware signatures based on these RF signals. The non-linear effects manifest in various forms, including I/Q imbalance, differential non-linearity caused by digital-to-analog converters, power amplifier non-linearity, and other related phenomena.

For example, I/Q imbalance refers to mismatches in RF signals' in-phase and quadrature-phase components, which can occur due to manufacturing imperfections. Differential non-linearity, arising from the manufacturing variations of digital-to-analog converters, leads to distortions in the analog output. Power amplifier non-linearity refers to the deviation of the amplifier's behavior from the ideal linear response, resulting in signal distortions. These non-linear effects caused by hardware imperfections contribute to each transmitter's RF signal's uniqueness. Analyzing and exploiting these non-linearities makes it possible to identify and differentiate transmitters based on their distinct hardware signatures, even when they belong to the same type or model.

3.4 Evaluation Metrics and Notations

The evaluation of radio fingerprinting involves two metrics: *accuracy* and *device rank*. Accuracy measures the proportion of correctly labeled I/Q traces by a classifier concerning the total number of test traces, similar to the concept of accuracy in machine learning classifiers. Device rank, on the other hand, determines the position or rank of the true transmitter among all potential candidates based on scores obtained from

the classifier using the test traces of that transmitter. This provides additional insights into the relative performance of identifying the true transmitter compared to others.

Notations: An I/Q trace is represented as x, which consists of a sequential series of consecutive I/Q samples denoted as $x = (x[1], ..., x[L])$. Here, $x[i]$ refers to the i-th I/Q sample, L represents the length of the trace. To distinguish between the real and imaginary parts of an I/Q sample $x[i]$, we use $x[i]^R$ and $x[i]^I$ to represent the real and imaginary parts, respectively.

A set X is defined as a collection of n I/Q traces, denoted as $X = (x_1, ..., x_n)$. Additionally, we use $\mathcal{Y} = \{y_1^*, ..., y_{|\mathcal{Y}|}^*\}$ to represent a set of transmitters, where y_i^* represents a specific transmitter, and $|\mathcal{Y}|$ denotes the total number of transmitters in the set. Please note that further information regarding the pre-processing steps will be provided in subsequent discussions.

3.4.1 Accuracy

In the training phase, a classifier F is trained using the provided training data $D_{\text{train}} = \{X, Y\} = \{(x_1, y_1), ..., (x_n, y_n)\}$. Here, each sample (x_i, y_i) consists of an I/Q trace x_i and its corresponding label y_i, where $y_i \in \mathcal{Y}$. This phase aims to train the classifier F to make accurate predictions based on the provided training data.

In the authentication phase, given a test dataset $D_{\text{test}} = \{X', Y'\} = \{(x'_1, y'_1), ..., (x'_{n'}, y'_{n'})\}$, where each test sample (x'_i, y'_i) consists of an I/Q trace x'_i and its corresponding label y'_i, with $y'_i \in \mathcal{Y}$, the trained classifier F produces a score vector $(s_{i,1}, ..., s_{i,|\mathcal{Y}|})$ for each test sample. Here, $s_{i,j}$ represents the confidence score assigned by the classifier F to the transmitter y_j^* for the I/Q trace x'_i.

To evaluate the correctness of the classifier's prediction, we check if y'_i matches y_j^*, where j corresponds to the index of the highest score among $(s_{i,1}, ..., s_{i,|\mathcal{Y}|})$. If y'_i

matches y_j^* and $s_{i,j}$ is the highest score, the classifier F correctly predicts the label for the I/Q trace x_i'.

To calculate the accuracy metric, we count the number of correctly predicted traces, denoted as m', among the total number of traces in the test dataset, indicated as n'. The accuracy is then computed as the ratio of m' to n', representing the proportion of correctly predicted traces over the total number of traces.

3.4.2 Device Rank

Given a subset $D_{\text{test}}^{y_j^*} = \{(x_1', y_j^*), ..., (x_{n'}', y_j^*)\}$ from the test dataset D_{test}, where all traces in this subset belong to a single transmitter $y_j^* \in \mathcal{Y}$, we compute the aggregated score vector $(s_1, ..., s_{|\mathcal{Y}|})$ over these n' traces as follows:

$$s_k = \sum_{i=1}^{n'} s_{i,k}, \quad \text{for } 1 \leq k \leq |\mathcal{Y}|, \tag{3.1}$$

where $s_{i,k}$ represents the score assigned by the classifier F to transmitter y_k^* for the trace x_i'. The aggregated scores are then sorted in descending order as $(s_1^*, ..., s_{|\mathcal{Y}|}^*)$, where $s_j^* \geq s_{j+1}^*$ for $1 \leq j \leq |\mathcal{Y}| - 1$.

The concept of device rank is assigned as r, where $r \in [1, |\mathcal{Y}|]$, if the aggregated score of transmitter y_j^* is ranked as the r-th among all the aggregated scores $(s_1^*, ..., s_{|\mathcal{Y}|}^*)$. A device rank of 1 over n' traces indicates that classifier F correctly ranks device y_j^* as the top candidate after considering n' traces. On the other hand, if the device rank is different from 1, it implies that the authentication for device y_j^* is incorrect based on the n' test traces.

A lower device rank value suggests a better authentication performance, as it indicates that the true device is ranked higher among all the potential candidates. Moreover, the device rank converges to 1 with fewer traces. In that case, the authen-

tication process is more effective and can correctly identify the device using fewer traces.

Average Device Ranking: In a balanced dataset, where each transmitter in the set $\mathcal{Y}$ has the same number of I/Q traces, the device ranks $\{r_1, ..., r_{|\mathcal{Y}|}\}$ for all $\mathcal{Y}$ transmitters can be obtained.

The average device rank metric measures the overall authentication performance across all the transmitters in the dataset. By calculating the average of the individual device ranks, we gain insight into the typical ranking of the actual transmitter compared to other candidates in the authentication process. A lower average device rank indicates better performance, suggesting that the valid transmitter tends to be ranked higher on average among the potential candidates.

3.4.2.1 Why is Device Rank Helpful?

Hardware manufacturing variations in wireless devices can lead to subtle variations that make distinguishing between candidates in a classifier is challenging. In such cases, the scores assigned to different candidates by the classifier are often close in value. If accuracy alone is used as a metric, it only considers the top-ranked candidate in the predictions, disregarding the scores of the other candidates.

This accuracy limitation can be overcome by utilizing the device rank metric, which considers the scores aggregated across multiple traces. Device rank provides a more comprehensive assessment by considering the collective information from the scores assigned to each candidate. By aggregating the scores, the device rank metric allows for a more informed decision-making process.

3.5 Pre-Processing of I/Q Samples

The slicing process, a commonly used pre-processing method, is designed to extract In-phase (I) and Quadrature (Q) samples from RF signals and prepare them as input for a classifier. This process consists of the following steps:

Input: We use a sequence of M I/Q samples as the input. In the Sliding Window approach, we define a sliding window of length L, where L is much smaller than M. This sliding window is a moving window that selects a subset of L I/Q samples from the sequence. Next, in the Slicing process, we collect the L I/Q samples within the sliding window to form an I/Q trace slice. We slide the window to the right with a stride of s (fixed or dynamic) to implement the slicing process. This updates the window to encompass the next set of L I/Q samples. We continue this process until we achieve a specific number of I/Q traces. This slicing approach generates multiple I/Q traces from the input sequence. Each I/Q trace represents a subset of L I/Q samples from the original sequence. The resulting I/Q traces are treated as time-series data with a single channel, representing both the In-phase and the Quadrature components. These traces can then be utilized as input for training a classifier or analyzing the RF signals.

3.6 Frequency Domain

We can represent I/Q samples in the time-frequency domain by collecting RF signals at the receiver side and capturing I/Q samples before and after the Fast Fourier Transform (FFT). I/Q samples before the FFT are represented in the time domain, while those after the FFT are represented in the frequency domain. The FFT operation is mathematically described as follows:

$$X_k = \sum_{n=0}^{N-1} x_n e^{-i2\pi kn/N} \qquad k = 0, \ldots, N-1; \tag{3.2}$$

where X_K and x_n represent the frequency and time domain respectively, and $e^{i2\pi/N}$ is the primitive N^{th} root of 1. The FFT was implemented using Python on GNURadio.

Given a total of M I/Q samples in the time domain, we can generate a spectral representation with a length of $\lfloor {}^{2M}/_{m} \rfloor$ and width of m, where m is the FFT window size. After obtaining the spectral representation, we apply slicing with a trace length of L to split it into smaller segments with length L and width m. These segments are then used as inputs for a classifier. In other words, I/Q traces in the frequency domain can be considered as time-series data with m channels, where m is the FFT window size.

Chapter 4

Preliminaries

4.1 Complex-Valued Neural Network

CVNN uses complex numbers for both inputs and weights, as opposed to real-valued neural networks that rely on real numbers. The key technical distinction between CVNNs and real-valued neural networks centers on their number representation and employed mathematical operations [12]. The CVNN's inputs and weights are represented using complex numbers, which consist of a real part and an imaginary part. For example, a complex number can be written as $z = a + bi$, where a represents the real part, and b denotes the imaginary part. Furthermore, CVNNs utilize complex-valued operations, enabling them to perform mathematical operations on complex numbers, such as complex multiplication, complex addition, complex subtraction, and more. Another notable difference is that CVNNs have a larger parameter space compared to real-valued neural networks. This results from the use of complex numbers, which introduces two parameters for each weight (one for the real part and one for the imaginary part). CVNNs may also include specialized complex-valued layers, like Complex Convolutional Layers and Complex Dense Layers, specifically designed to process complex-valued inputs and weights. In contrast, real-valued neural networks use conventional layers such as Convolutional Layers and Dense Layers, which only

work with real-valued inputs and weights.

In the context of I/Q data, the representation of complex numbers facilitates capturing a signal's amplitude and phase information at a given moment. By unifying the amplitude and phase components into a single complex representation, CVNNs offer advantages over handling separate I and Q channels. This has been demonstrated in prior studies [6, 10, 16, 29], and our results in Table 7.1 confirm that CVNN-based STADL-RF outperforms the CNN-based approach [15].

4.2 Complex Convolution

CVNNs use complex activation functions, which operate on complex numbers and preserve their complex nature. A complex filter is typically used in the convolutional layers. Unlike real-valued filters used in traditional convolutional neural networks (CNNs), complex filters in CVNNs have complex-valued weights that enable them to handle complex-valued inputs and perform operations on both the real and imaginary parts of the input data. Complex filters are particularly useful when dealing with signals or data that have both amplitude and phase components, as is often the case in radio frequency (RF) and wireless communication systems, where signals are represented as In-phase and Quadrature (I/Q) data. By using complex filters in CVNNs, the model can effectively capture the complex relationships and dynamics present in such signals, leading to improved accuracy and performance in tasks like radio fingerprinting, modulation recognition, and wireless device identification.

Suppose we integrate complex data $z = a + ib$, which has a real component, a, and an imaginary component, b, with a complex filter matrix $W = X + iY$, where X and Y are real-valued filters, to conduct complex-valued convolution. Instead of separating real and imaginary components, we do the real-valued convolution with

complex values.

$$W * z = (X + iY)(a + ib) = (X * a - Y * b) + i(Y * a + X * b)$$

may be calculated using the distributive property of convolution. These convolutions have the matrix form [8] .

$$\begin{bmatrix} \text{Re}(W \cdot z) \\ \text{Im}(W \cdot z) \end{bmatrix} = \begin{bmatrix} X & -Y \\ Y & X \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix}$$

4.3 Complex-Valued Activation Functions

There are several complex activation functions already proposed. Most of the proposed complex activation functions are extensions of real-valued activation functions. Cart Leaky ReLU [9], Complex Cardioid [9, 31], and CReLU [25] are three types of complex-valued activation functions used in our research.

4.3.1 Cart Leaky ReLU

Leaky Rectified Linear Unit (Leaky ReLU) is applied to the real and imag parts of z. The Leaky ReLU function is defined as:

$$\text{LeakyReLU}(z) = \begin{cases} z, & \text{if } z \geq 0 \\ \text{negative\_slope} \times z, & \text{otherwise.} \end{cases}$$

Here, negative_slope Controls the angle of the negative slope. Cart Leaky ReLU is defined as:

$$\text{Cart Leaky ReLU}(z) = \text{LeakyReLU}(\text{Re}(z)) + i \times \text{LeakyReLU}(\text{Im}(z)).$$

4.3.2 Complex Cardioid

By reducing the magnitude in accordance with the phase itself, this function preserves the phase information. It reduces to the ReLU for actual inputs.

$$\mathrm{f}(z) = \frac{1}{2}(1 + \cos(\angle z))z.$$

4.3.3 CReLU

Rectified Linear Unit is applied to the real and imaginary parts of z. The ReLU function defined as:

$$\mathrm{ReLU}(z) = \begin{cases} z, & \text{if } z \geq 0 \\ 0, & \text{otherwise.} \end{cases}$$

The ReLU function returns element-wise max(x, 0) with default values. Hence, CReLU can be defined as:

$$\mathrm{CReLU}(z) = \mathrm{ReLU}(\mathrm{Re}(z)) + i \times \mathrm{ReLU}(\mathrm{Im}(z)).$$

Chapter 5

RADIO FINGERPRINTING

In cross-location and different times radio fingerprinting, we collect I/Q traces from one location as the source data, and I/Q traces from different locations at another time as the target data in transfer learning. The transfer learning discussed next performs effectively for both the source and target data. Before we dive into the details, we first describe the training and tuning steps as illustrated in Fig. 5.2 and Fig. 5.3, emphasizing the process involved in fine-tuning and the triplet network for transfer learning.

5.1 Transfer Learning

In machine learning, we usually assume that both the training data $D_{\text{train}} = \{X, Y\} = \{(x_1, y_1), ..., (x_n, y_n)\}$ and test data $D_{\text{test}} = \{X', Y'\} = \{(x'_1, y'_1), ..., (x'_{n'}, y'_{n'})\}$ belong to the same feature space and follow the same distribution. However, traditional statistical models often require starting the learning process with new training data in distribution shifts. Unfortunately, gathering the necessary training data and rebuilding the models in many real-world scenarios can be expensive or even unfeasible. Hence, minimizing the need and effort of collecting new training data would be beneficial. In such situations, transfer learning or knowledge transfer between different

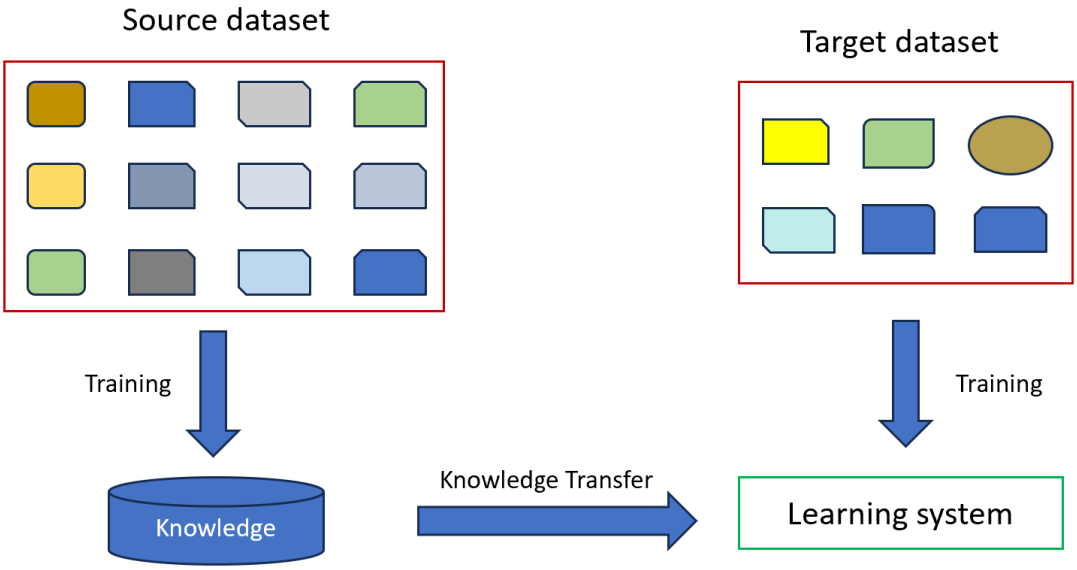


Figure 5.1: Learning Process of Transfer Learning

task domains becomes highly valuable. A basic learning procedure for the transfer learning approach is shown in Fig. 5.1 If the source dataset has a large amount of labeled data, the model can learn from training and apply this information to solve new, unrelated, but related problems. As a result, even if the target dataset is sparse in data, it can still use the prior knowledge to complete the new tasks.

Transfer learning enables models to leverage knowledge and insights gained from related domains. Rather than commencing the learning process anew, transfer learning allows models to apply what they have learned from a source domain to a target domain. By employing transfer learning, models can benefit from features, representations, or parameters learned in a source domain and transfer that knowledge to a target domain, even if the feature space or distribution differs between the two domains. This approach can save time, computational resources, and extensive data collection needs [20].

Transfer learning typically encompasses three steps: pre-training, training, and

testing.

Pre-training: The initial step involves training a model on a source dataset, which is usually large and diverse, to learn general feature representations. This pre-training phase allows the model to grasp fundamental patterns and knowledge from the source data.

Training: In this step, the knowledge acquired during pre-training is transferred to the target task or dataset. The model is then fine-tuned using the target dataset, adapting its learned representations to the specific characteristics of the new data. Tuning steps help the model to specialize for the target task while retaining the previously acquired knowledge.

Testing: The fine-tuned model is evaluated on the test data from the target dataset. The transferred model's classifier is used to make predictions, and the performance of the transfer learning method is reported based on the model's accuracy and other relevant metrics [32].

5.1.1 Fine-Tuning

Fine-tuning is a straightforward method utilized in transfer learning. Initially, this method involves training a neural network on a source dataset during pre-training, as shown in Fig. 5.2. In the subsequent training step, most of the layers in the pre-trained network are frozen, meaning their weights remain unchanged. However, the hyperparameters of the last few layers are adjusted using the training data from the target dataset. By fine-tuning the last layers of the network, the model can adapt to the specific characteristics of the target dataset. Finally, the method utilizes the modified neural network to generate results by evaluating the model on the test data from the target dataset.

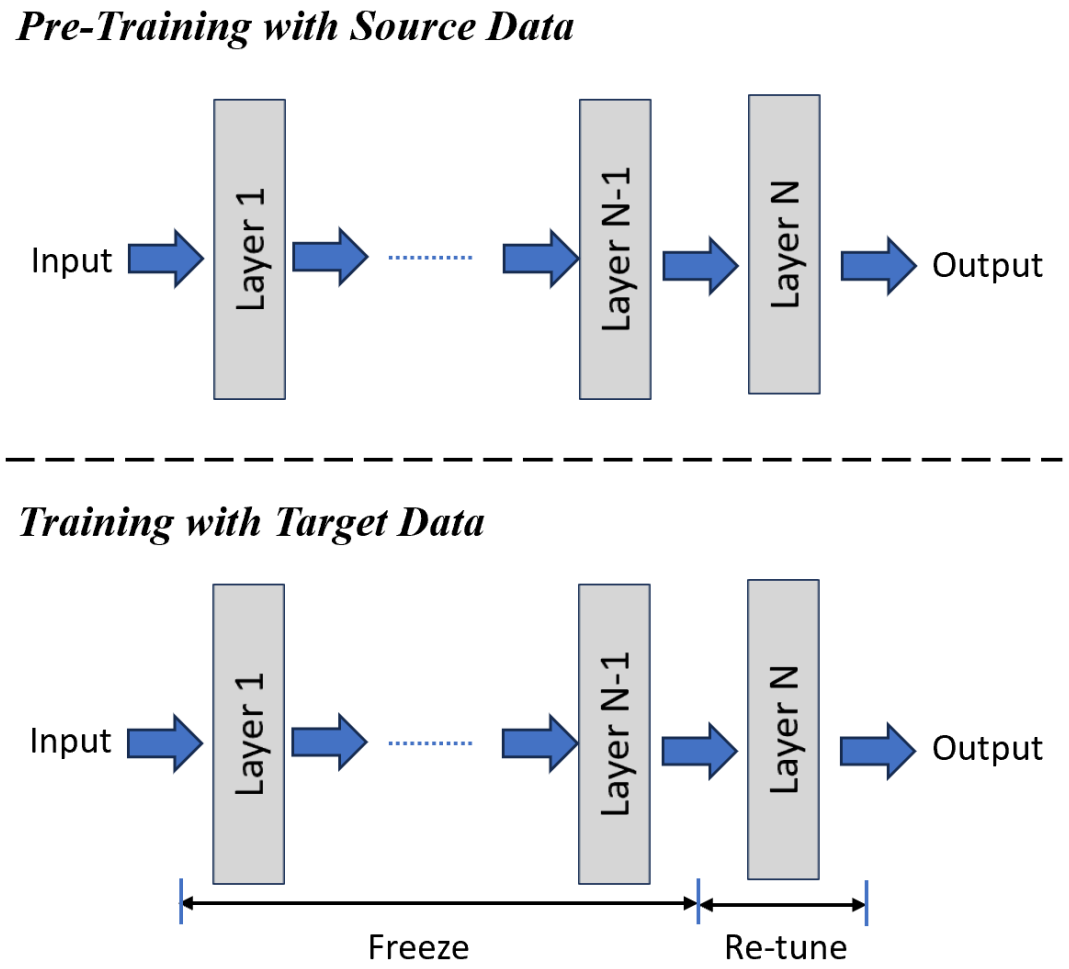


Figure 5.2: Transfer learning with fine-tuning

5.1.2 Triplet Network

The triplet network, which was modeled after the Siamese network, consists of three parallel sub-networks with identical weights and hyperparameters. A triplet is used to represent the input to a triplet network. A triplet is made up of an anchor sample, a positive sample, and a negative sample. The triplet network's sub-networks are each in charge of processing a certain kind of sample.

In this architecture, one sub-network takes only anchor samples as inputs. It fo-

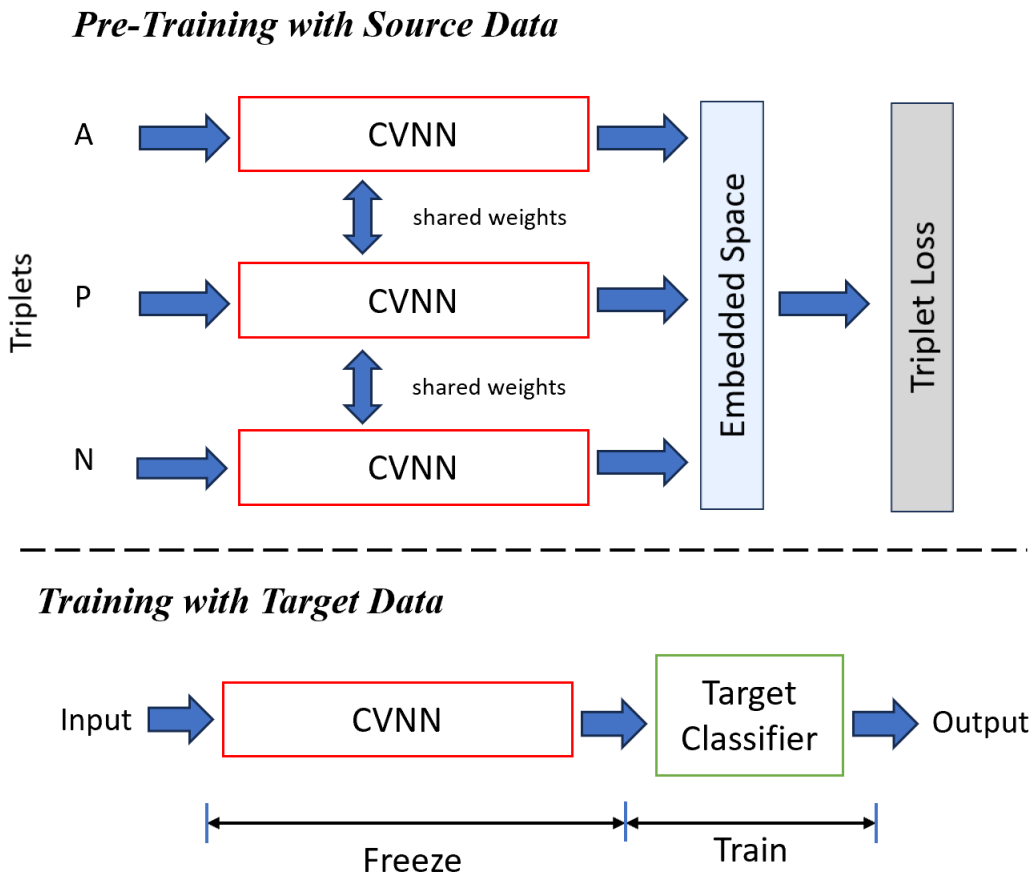


Figure 5.3: Transfer learning with a triplet network

cuses solely on processing the anchor samples and does not include any positive or negative samples in its input. The other two sub-networks are dedicated to processing positive and negative samples, respectively. The triplets used in training the network are selected from the source dataset, either randomly or using specific mining strategies. The goal of selecting triplets is to provide meaningful and informative training examples that contribute to the learning process. Various techniques, such as hard negative mining or semi-hard negative mining, can be employed to select the triplets. By using triplets as input and training the network with a suitable loss function, the triplet network aims to learn representations where the distance between anchor

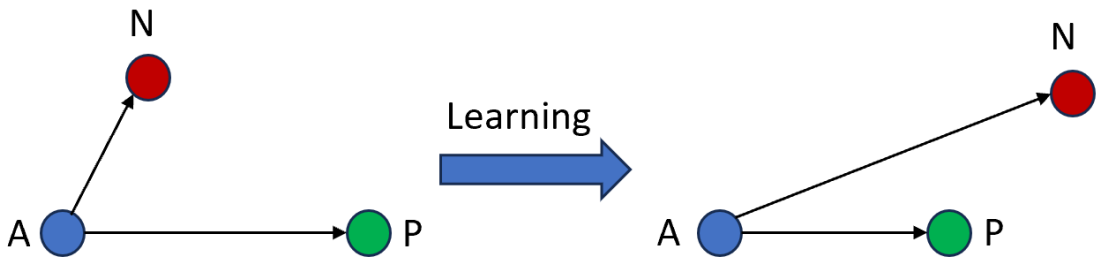


Figure 5.4: The objective of Triplet Loss

and positive samples is minimized, while the distance between anchor and negative samples is maximized, Fig. 5.4 provides an illustration of this. This way, the network learns to discriminate between samples of different classes or categories [32].

Chapter 6

Implementation

In this chapter, we focus on assessing the performance of our method in diverse scenarios. Where the model is trained and tested across different locations and at other times. The objective is to evaluate the generalization capabilities of our method and its robustness in varying environmental conditions. Before we delve into the experiments to evaluate these scenarios, we provide a comprehensive description of the dataset used for examining the performance of our method.

6.1 USRP B205mini-i Dataset (USRPB205)

We conducted our experiments using a testbed consisting of seven USRP B205-mini-i devices, with one device as the receiver and the remaining six as transmitters, as illustrated in Fig. 6.1. Each USRPB205-mini-i is equipped with a VERT2450 antenna. To establish WiFi transmissions (IEEE 802.11 a/g) with BPSK 1/2 modulation, we employed the free and open-source GNU Radio code [4]. The transmissions were carried out at a center frequency of 2.45 GHz, with a bandwidth of 2 MHz and a sampling rate of 5 MS/S, resulting in recording I/Q samples.

All devices were kept stationary during data collection, with the transmitter and receiver separated by approximately 2 feet. I/Q data was gathered at one outdoor lo-

Figure 6.1: Testbed: One receiver and six transmitters

cation and three distinct indoor locations, which we designated as Location 1 (indoor), Location 2 (outdoor), Location 3 (indoor), and Location 4 (indoor), respectively, as shown in Fig. 6.2.

For each transmitter, three transmissions were recorded, each lasting for 30 seconds, with a 15-second gap between consecutive transmissions. Despite some package loss, approximately 1.50×10^8 samples were successfully collected in each transmission. Each I/Q trace comprises 288 consecutive I/Q samples.

For neural network training, we created a dataset by randomly selecting 1000 I/Q traces from each transmitter, resulting in a total of 6,000 I/Q traces for the entire dataset. On the receiver side, I/Q samples were gathered before, after the FFT, and after the WiFi Frame Equalizer, but we only used the post-FFT I/Q samples in our experiments.

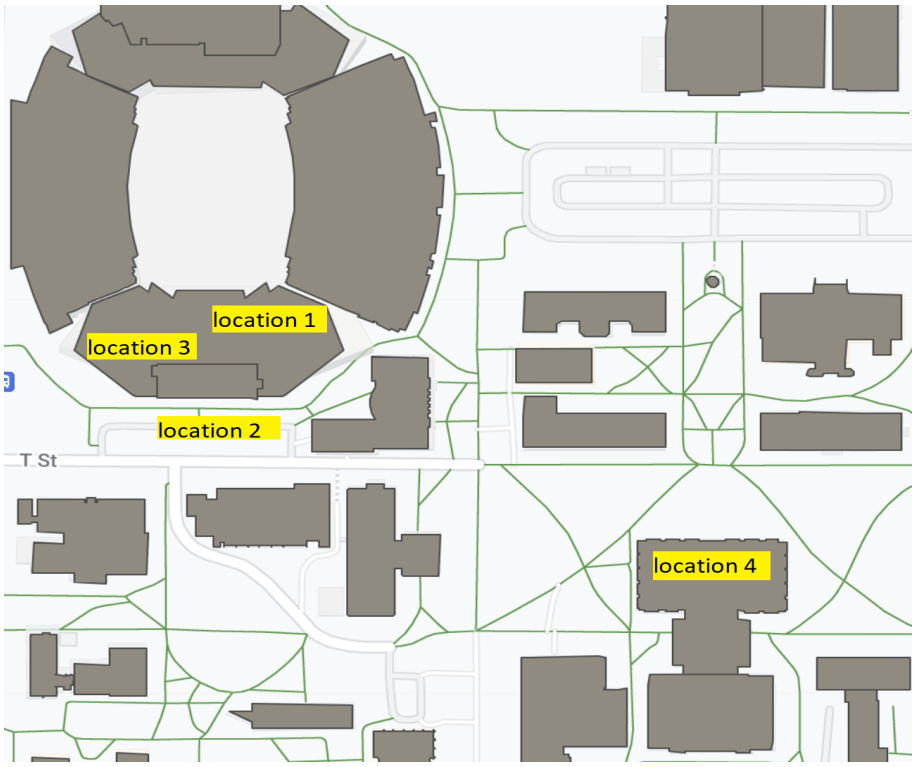


Figure 6.2: Data collection location

6.2 Experimental Settings

The code runs on Holland Computing Center [1] with a Tesla V100-PCIE-32GB GPU, which supports cuDNN version 8.6.0.

For data splitting, we use 72% of the data for training, 8% for validation, and 20% for testing. We employ `RMSprop` as the optimization algorithm with a learning rate of 0.1, and the loss function used is `ComplexAverageCrossEntropy` from the CVNN package. Each model is trained for 100 epochs, and we apply early stopping with patience of 10 to prevent overfitting.

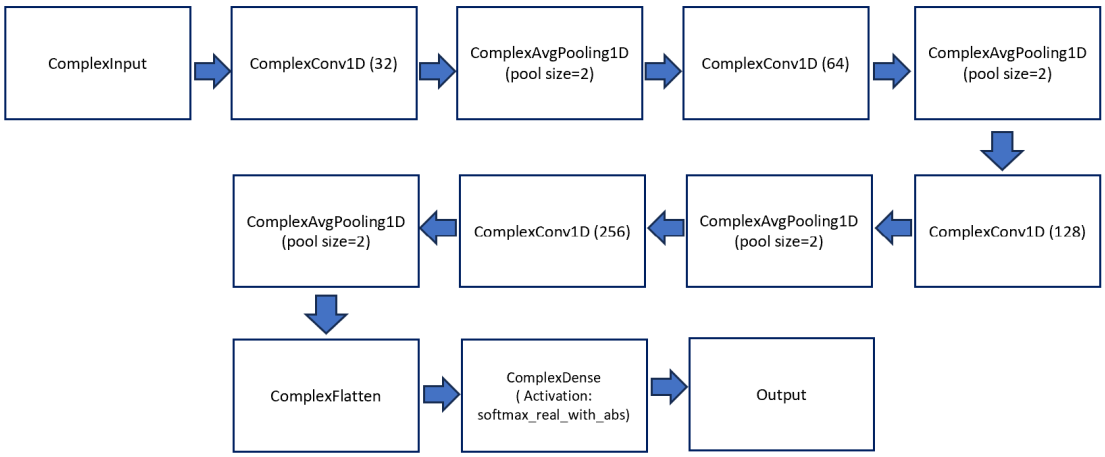


Figure 6.3: Architecture of complex-valued convolutional neural network

6.3 Architectures of Neural Networks

Since Tensorflow hardly supports complex numbers, we use CVNN library [3] to handle complex values. The only distinction is the use of *cvnn.layers* instead of *tf.keras.layers*. Various complex layers and complex activation functions are supported by this library. Our CVNN architecture is made up of the following layers: ComplexInput, 3 ComplexConv1D, 4 ComplexAvgPooling1D, 1 ComplexFlatten, and 1 ComplexDense, as shown in Fig. 6.3.

We tested three different activation functions: Cart Leaky ReLU, Complex Cardioid, and CReLU. Complex numbers can be entered into the layer with the name ComplexInput. The ComplexConv1D layers, on the other hand, are made up of 32, 64, 128, and 256 filters, with a filter size of 2 and the same padding, to accommodate the complex activation functions indicated above. The ComplexFlatten layer flattens the output, while the ComplexAvgPooling1D layers employ average pooling with a pool size of 2. If linear activation is not necessary, the final layer, ComplexDense, employs the "softmax real with abs" activation function for classification. The model's

loss function is ComplexAverageCrossEntropy, and it is built using the RMSprop optimizer.

6.4 Transfer Learning

6.4.1 Fine-Tuning

The CVNN model is pre-trained using a source dataset before tuning its hyperparameters in the training phase. During the training phase, we only allow the last three layers to be modified. The weights and properties of the remaining layers are frozen, so they are frozen.

6.4.2 Triplet Network

Our triplet network uses the CVNN paradigm as its sub-network. To calculate the triplet loss, we use the cosine distance as the metric. In addition, we employ a semi-hard negative mining strategy to choose triplets from a source dataset. For the Triplet Fingerprinting k-nearest neighbor classifier, we choose k=N, where N is the number of traces in the target training set for each class. We train k-nearest neighbors using the Mean Embedded Vector of N samples for the closed-world evaluation.

Table 6.1: Model Parameters

Parameter	**Selected Value**
Base Model	CVNN
Similarity Metrics	Cosine
Mining Strategy	Semi-Hard-Negative
Classifier	k-nearest neighbor
Optimizer	Adam
Batch Size	128
Embedded Vectors Size	64

As previously mentioned, greater accuracy can be achieved by using the Mean

Embedded Vector rather than the number of samples. The parameters we used for Triplet Network is shown in table Table 6.1 presents the hyperparameter for the triplet model.

Chapter 7

Evaluation

This chapter focuses on the outcomes of the complex-valued convolutional neural network for transmitter device identification using three complex activation functions: Cart Leaky ReLU, Complex Cardioid, and CReLU. We trained the Finetuning and Triplet Network models in order to increase the categorization accuracy of wireless devices. These models were assessed for different strides and trace lengths to find their best performance. Accuracy is used to assess the performance of each model. In addition, we used four cross-location frequency domain I/Q data sets to examine the performance of the three complex activation functions across both classification techniques.

7.1 Research Questions

We investigate two research questions to evaluate our methods below through a set of comprehensive experiments:

1. Which activation function performs the best with CVNNs?
2. Can transfer learning (fine tuning or triplet network) improve the performance of radio fingerprinting agnostic to space and time?

Table 7.1: CVNN based RF improves the performance as compared to CNN based approach.

	Transfer Learning Method	N=100	N=200	N=400	N=800
CNN based [15]	Fine-tuning	46.45 ± 0.71	49.98 ± 0.53	52.56 ± 1.12	55.17 ± 0.11
	ADA	47.82 ± 0.37	53.95 ± 0.75	59.94 ± 1.02	65.24 ± 1.42
CVNN based–STADL-RF	Fine-tuning	51.73 ± 2.25	56.42 ± 1.02	61.01 ± 0.40	66.99 ± 2.18
	Triplet Network	65.08 ± 0.33	65.50 ± 0.16	65.95 ± 0.58	66.05 ± 0.17

7.2 Performance Comparison with Existing Study

Before delving into the details of evaluating different scenarios, we first conducted experiments to validate the claim that CVNN outperforms CNN for radio fingerprinting. We set the stride $s = 288$ and trace length $L = 288$ to specific values in these experiments. Table 7.1 presents the results compared to [15], where a 2-channel real-valued input was used for I/Q data. Here, we used 10% Random Guess and data following WiFi Frame Equalizer. The cross-day accuracy of our method, which is based on complex-valued Fine-tuning (FT), significantly improves and increases cross-day accuracy from 55.17% to 66.99%.

Accuracy increases from 65.24% to 66.05% when using Triplet Network (TN), which performs marginally better than Adversarial Domain Adaptation (ADA). From the table, it is evident that for both fine-tuning and triplet network approaches, CVNN-based radio fingerprinting achieves superior performance.

7.3 Performance of Complex Activation Functions with CVNN

Our evaluation investigates the influence of three activation functions: Cart-Leaky-ReLU, Complex-Cardioid, and CReLU with CVNN. We kept the trace length fixed at $L = 288$, and varied the stride $s = \{1, 144, 288\}$. We conducted evaluations for both same-location scenarios, where the model was trained and tested using Location 1

data, and cross-location scenarios, where the model was trained with Location 1 data and tested with Location 2 data. We intentionally avoided implementing transfer learning to observe the performance of different activation functions solely. Table 7.2 shows the accuracy for our experiment.

Table 7.2: Comparison of various complex activation functions without transfer learning for trace length $L = 288$.

Stride s	Same Location			Cross Location		
	Cart-Leaky-ReLU	Complex-Cardioid	CReLU	Cart-Leaky-ReLU	Complex-Cardioid	CReLU
1	99.97 ± 0.05	99.87 ± 0.17	99.85 ±0.16	36.22 ± 5.45	27.43 ± 8.39	38.56 ±9.99
144	65.50 ± 5.08	61.52 ± 2.12	81.78 ±3.61	24.13 ± 1.68	23.68 ± 4.48	32.83 ±4.26
288	67.82 ± 2.70	68.73 ± 5.13	85.88 ±1.89	33.42 ± 3.60	25.77 ± 6.05	38.44 ±2.15

Observation 1: We found that CReLU outperformed the other two activation functions.

Observation 2: Without using transfer learning, we observed that the cross-location accuracy was a little bit better than random guessing (16.67%) but still not the best result.

7.4 Impacts of Transfer Learning with Cart-Leaky-RelU

In this experiment, we investigate CVNN based transfer learning method with the Cart-Leaky-ReLU activation function when trained with Location 1 data and tested with Location 2 data. We fixed the trace length $L = 288$, while varying the stride. We implemented both Fine-Tuning and Triplet network methods with different numbers of N = {100, 200, 400, 800}. The results of this experiment are presented in Table 7.3, showing the accuracy achieved with different strides. And we noted a few observations for Cart-Leaky-ReLU activation functions with transfer learning methods.

Observation 3: We found that Fine-Tuning outperforms Triplet Network in all the cases.

Observation 4: Increasing the value of N also leads to an improvement in accuracy. But there are few exceptions in specific cells.

Table 7.3: Cross location accuracy (random guess = 16.67%) for Cart-Leaky-Relu activation function implemented with transfer learning for trace length $L = 288$.

Stride s	Transfer Learning Method	N=100	N=200	N=400	N=800
1	Fine-tuning	85.74 ± 0.75	96.73 ± 0.30	99.87 ± 0.03	99.99 ± 0.00
	Triplet Network	50.09 ± 2.74	56.48 ± 2.32	57.84 ± 1.74	68.05 ± 1.23
144	Fine-tuning	67.20 ± 0.89	73.12 ± 1.67	57.36 ± 2.30	60.50 ± 2.12
	Triplet Network	43.90 ± 0.08	49.87 ± 1.25	48.25 ± 0.69	48.96 ± 0.27
288	Fine-tuning	81.82 ± 1.50	84.52 ± 1.26	56.98 ± 0.87	60.12 ± 1.10
	Triplet Network	44.75 ± 1.53	44.92 ± 4.24	54.81 ± 2.70	56.88 ± 0.06

7.5 Impacts of transfer learning with Complex-Cardioid

We explore CVNN with the Complex-Cardioid activation function when training it with Location 1 data and testing it with Location 2 data. The trace length was fixed at $L = 288$, while we varied the stride and we implemented both Fine-Tuning and Triplet network methods with different numbers of N = {100, 200, 400, 800}. The accuracy achieved with different strides is presented in Table 7.4.

Observation 5: We found that Fine-Tuning outperforms Triplet networks in all the cases.

Observation 6: Increasing the value of N also leads to an improvement in accuracy. But there are few exceptions in specific cells.

Observation 7: The performance of Complex-Cardioid Activation is inferior to Cart-Leaky-ReLU.

Table 7.4: Cross location accuracy (random guess = 16.67%) for Complex-Cardioid activation function implemented with transfer learning for trace length $L = 288$.

Stride s	Transfer Learning Method	N=100	N=200	N=400	N=800
1	Fine-tuning	78.52 ± 1.07	90.72 ± 0.96	98.65± 0.71	99.97 ± 0.05
	Triplet Network	38.09 ± 2.74	38.75 ± 10.67	34.18 ± 4.10	35.22 ± 0.41
144	Fine-tuning	53.24 ± 0.71	54.38 ± 2.17	57.15 ± 0.32	58.47 ± 1.65
	Triplet Network	26.65 ± 1.21	30.77 ± 0.09	28.89 ± 2.20	29.53 ± 0.28
288	Fine-tuning	51.18 ± 2.33	54.71 ± 0.58	56.55 ± 2.37	49.38 ± 1.34
	Triplet Network	47.33 ± 3.63	44.57 ± 0.05	45.03 ± 8.14	50.30 ± 0.24

7.6 Impacts of Transfer Learning with CReLU

In this experiment, we investigate CVNN with the CReLU activation function while training it with Location 1 data and testing it with Location 2 data. We keep the trace length fixed at $L = 288$ and set, varying the stride and N size we evaluate the performance of Fine-Tuning and Triplet Network. Table 7.5 presents the accuracy achieved with different strides.

Observation 8: We found that Fine-Tuning outperforms Triplet networks in all the cases.

Observation 9: Increasing the value of N also leads to an improvement in accuracy. But there are few exceptions in specific cells.

Observation 10: Compared to Cart-Leaky-ReLU and Complex-Cardioid, CReLU Activation performs better.

Table 7.5: Cross location accuracy (random guess = 16.67%) for CReLU activation function implemented with transfer learning for trace length $L = 288$.

Stride s	Transfer Learning Method	N=100	N=200	N=400	N=800
1	Fine-tuning	82.82 ± 2.20	92.91 ± 1.69	99.67 ± 0.15	99.99 ± 0.00
	Triplet Network	38.45 ± 0.44	44.31 ± 0.30	47.02 ± 0.27	47.67 ± 0.08
144	Fine-tuning	64.60 ± 0.48	66.41 ± 0.39	67.20 ± 0.39	71.63 ± 2.52
	Triplet Network	47.47 ± 0.40	48.03 ± 0.58	48.44 ± 0.31	48.47 ± 0.53
288	Fine-tuning	84.90 ± 2.80	87.77 ± 0.31	88.59 ± 0.60	89.27 ± 2.50
	Triplet Network	57.00 ± 4.14	58.33 ± 1.43	66.63 ± 2.02	67.76 ± 0.63

7.7 CReLU Performance with Different Datasets

In this experiment, we explore CVNN with the CReLU activation function while training it with Location 1 data and testing it with outdoor (Location 2) data. Again training with indoor (Location 3) and testing with other indoor data (Location 4) and we select a trace length of $L = 288$ and a stride of $s = 288$.

From the usrpb200 dataset, we extract 6,000 I/Q traces from Location 3 and 6,000 I/Q traces from a different location. In the training phase of fine-tuning, we utilize 64% of the source data for each case, and only a small number of N traces per transmitter in the target data are used as target training data, where we examine $N = \{100, 200, 400, 800\}$. The tuning step also makes use of the goal training data, where 8,000 traces from Location 4 representing 20% of the target data are used for testing. In the tuning phase, we select $k = |Y| - 1$ for the $k - NN$ classifier, where $|Y|$ is the total number of transmitters. The accuracy achieved with different strides is presented in Table 7.6.

Table 7.6: Cross location accuracy (random guess = 16.67%) for CReLU activation function tested for various locations $s = 288$ and $L = 288$.

Locations (Training & Testing)	Transfer Learning Method	N=100	N=200	N=400	N=800
Location 1 & Location 2	Fine-tuning	84.90 ± 2.80	87.77 ± 0.31	88.59 ± 0.60	89.27 ± 2.50
	Triplet Network	57.00 ± 4.14	58.33 ± 1.43	66.63 ± 2.02	67.76 ± 0.63
Location 3 & Location 4	Fine-tuning	71.85 ± 3.70	91.64 ± 0.75	99.88 ± 0.08	99.00 ± 0.03
	Triplet Network	46.79 ± 4.86	43.95 ± 3.06	48.04 ± 1.41	52.45 ± 0.08

Our two research problems have now been resolved. The activation functions of CVNNs and CReLUs complement one another well. And in response to our second question, transfer learning can indeed enhance RF performance that is independent of both space and time.

7.8 Device Ranking

In this experiment, we explore CVNN with the CReLu activation function while training it with Location 1 (indoor) data and testing it with Location 2 (outdoor). Next, we verified training and testing with indoor data from two different buildings (Location 3 and Location 4). We select a trace length of $L = 288$ and a stride of $s = 144$. From the usrpb200 dataset, we extract 6,000 I/Q traces from Location 1 and 6,000 I/Q traces from a different location. The I/Q traces from Location 1 serve as source data, and the I/Q traces from the other location serve as target data. In the training phase of fine-tuning, we utilize 64% of the source data, and only a small number of N traces per transmitter in the target data are used as target training data, where we examine $N = \{100, 200, 400, 800\}$. The tuning step also makes use of the goal training data, where 8,000 traces from Location 2 representing 20% of the target data are used for testing. In the tuning phase, we select $k = |Y| - 1$ for the $k - NN$ classifier, where $|Y|$ is the total number of transmitters. The accuracy achieved with different strides is presented in Table 7.6. Figure 7.1 and Figure 7.2 shows the plot for device ranking for various parameters.

Observation 11: We find that device ranking converges to 1 more quickly for fine-tuning than the triplet network. This further supports our earlier finding that fine-tuning performs better than the triplet network.

7.9 Reproducibility

The source code of the implementation is available at STADL-RF_GitHub and the dataset is available at usrp_b205.

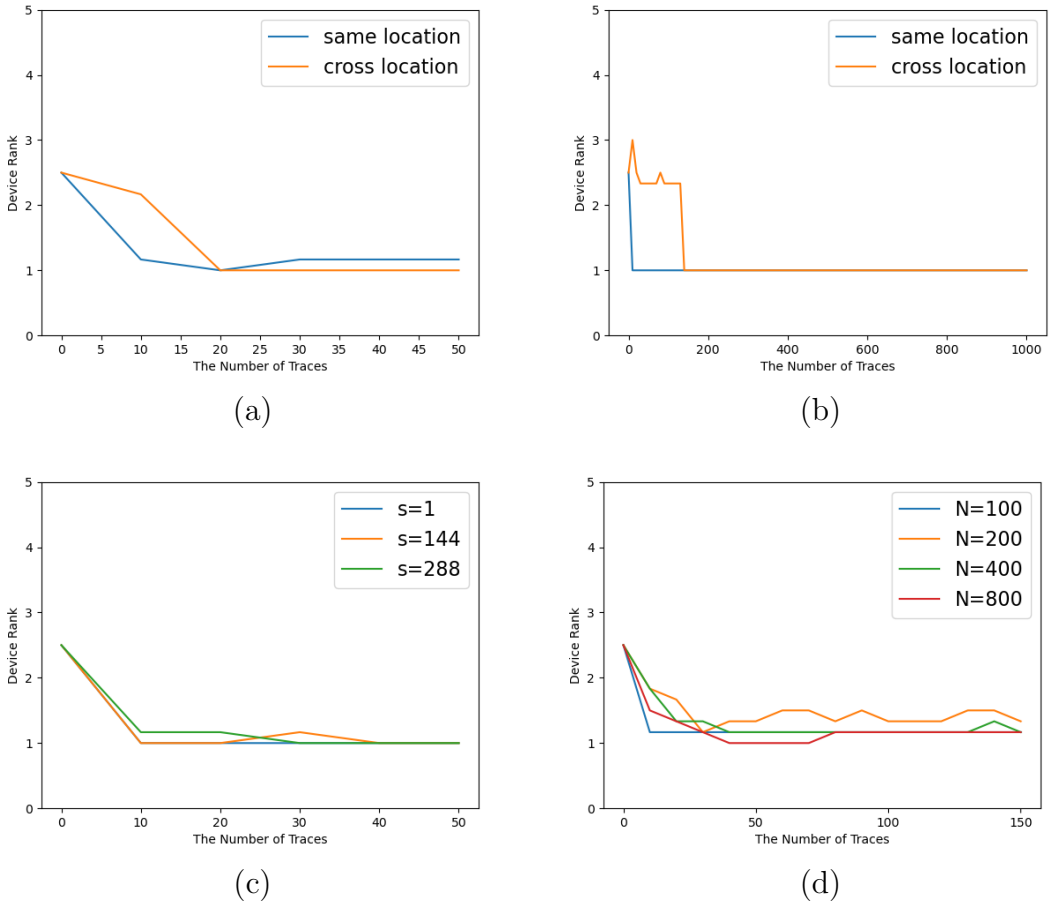
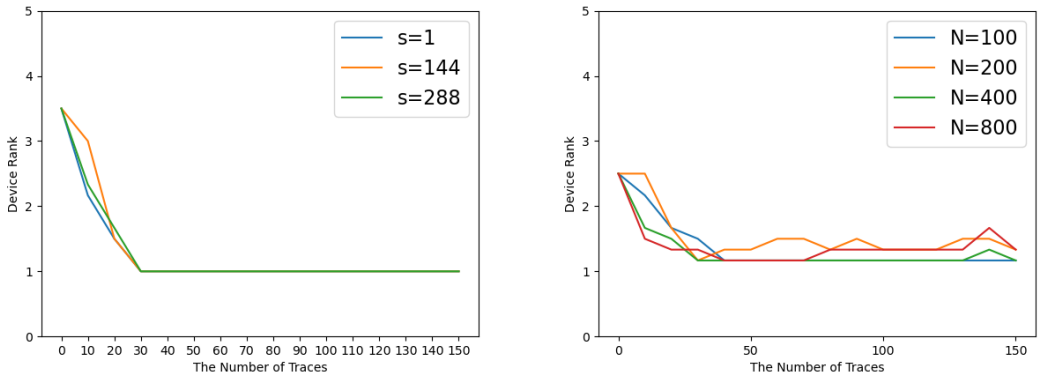


Figure 7.1: Average device rank for frequency domain with CReLU, usrpb205 dataset, $L = 288$ (a) cross-location (indoor and outdoor), $s = 288$, dataset: Location 1 and Location 2, (b) cross-location (different buildings), $s = 288$, train set: Location 3, and test set: Location 4, (c) cross-location (indoor and outdoor), fine tuning with various stride, and (d) cross-location (indoor and outdoor), fine tuning with various N.



(a) (b)

Figure 7.2: Average device rank for frequency domain with CReLU, usrpb205 dataset, $L = 288$ (a) cross-location (indoor and outdoor), triplet network with various stride, and (b) cross-location (indoor and outdoor), $s = 288$, triplet network with various N.

Chapter 8

Conclusion and Future Work

In this thesis, we propose a novel approach to enhance the performance of radio fingerprinting, a process involving training and authentication phases, by utilizing a complex-valued convolutional neural network. Our method addresses potential vulnerabilities that adversaries may exploit during both stages. We conducted experiments on four cross-location scenarios using neural networks applied to I/Q data in the frequency domain. We employed two classifiers for device type level classification and physical-layer authentication and evaluated their performance based on accuracy and device rank metrics. To handle complex numbers in a one-dimensional architecture, we utilized the CVNN as a base model, incorporating different complex activation functions and parameters like stride and trace length. Comparing the two models, we found that the fine-tuning approach achieved the highest accuracy score, exceeding 90%. Furthermore, our results indicate that the CReLU complex activation function performed well among the three considered parts. Lastly, we demonstrated the effectiveness of device ranking in various scenarios for transmitter identification. Overall, our research presents a comprehensive analysis of complex-valued convolutional neural networks and their application to radio fingerprinting, showcasing improved performance and the suitability of certain activation functions and techniques for this task. Although we have proposed a solution to enhance radio

frequency fingerprinting (RFF), other crucial issues still require attention. So, we're planning to train classification models for various more complex activation functions in our future work. Moreover, we aim to gather more data and incorporate supplementary features derived from the signal-to-noise ratio specific to each device type. Furthermore, in addition to the machine learning classifiers employed in this thesis, we plan to explore cross-layer classification techniques that encompass not only the physical layer authentication but also extend to the network layer. We will leverage several other classifiers and apply more advanced transfer learning techniques to enhance their performance. By incorporating these future enhancements, we aim to comprehensively address the remaining challenges in RFF and provide more robust solutions.

8.1 Future Work

Although we have proposed a solution to enhance radio frequency fingerprinting, other crucial issues still require attention. We plan to train classification models for various, more complex activation functions in our future work. Moreover, we aim to gather more data and incorporate supplementary features derived from the signal-to-noise ratio specific to each device type. Furthermore, in addition to the machine learning classifiers employed in this thesis, we plan to explore cross-layer classification techniques that encompass not only the physical layer authentication but also extend to the network layer. We will leverage several other classifiers and apply more advanced transfer learning techniques to enhance their performance. By incorporating these future enhancements, we aim to comprehensively address the remaining challenges in radio frequency fingerprinting and provide more robust solutions.

Bibliography

[1] Holland computing center. `https://hcc.unl.edu/`. [Online; accessed 24-July-2023].

[2] A. Al-Shawabka, F. Restuccia, S. D'Oro, T. Jian, B. C. Rendon, N. Soltani, J. Dy, S. Ioannidis, K. Chowdhury, and T. Melodia. Exposing the fingerprint: Dissecting the impact of the wireless channel on radio fingerprinting. In *Proc. of IEEE INFOCOM 2020-IEEE Conference on Computer Communications*, pages 646–655. IEEE, 2020.

[3] J. A. Barrachina. Negu93/cvnn: Complex-valued neural networks, Nov. 2022.

[4] B. Bloessl. IEEE 802.11 a/g/p transceiver. `https://github.com/bastibl/gr-{IEEE}802-11`. [Online; accessed 22-Aug-2022].

[5] M. Cekic, S. Gopalakrishnan, and U. Madhow. Robust wireless fingerprinting: Generalizing across space and time. *arXiv preprint arXiv:2002.10791*, 2020.

[6] J. Chen, W.-K. Wong, B. Hamdaoui, A. Elmaghbub, K. Sivanesan, R. Dorrance, and L. L. Yang. An analysis of complex-valued CNNs for RF data-driven wireless device classification. In *Proc. of ICC 2022-IEEE International Conference on Communications*, pages 4318–4323. IEEE, 2022.

[7] Y. Chen, H. Wen, H. Song, S. Chen, F. Xie, Q. Yang, and L. Hu. Lightweight one-time password authentication scheme based on radio-frequency fingerprinting. *IET communications*, 12(12):1477–1484, 2018.

[8] E. K. Cole, J. Y. Cheng, J. M. Pauly, and S. S. Vasanawala. Analysis of deep complex-valued convolutional neural networks for MRI reconstruction. *arXiv preprint arXiv:2004.01738*, 2020.

[9] A. K. Dubey and V. Jain. Comparative study of convolution neural network's relu and leaky-relu activation functions. In *Proc. of MARC 2018*, pages 873–880. Springer, 2019.

[10] S. Gopalakrishnan, M. Cekic, and U. Madhow. Robust wireless fingerprinting via complex-valued neural networks. In *Proc. of 2019 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6. IEEE, 2019.

[11] S. S. Hanna and D. Cabric. Deep learning based transmitter identification using power amplifier nonlinearity. In *Proc. of 2019 International Conference on Computing, Networking and Communications (ICNC)*, pages 674–680. IEEE, 2019.

[12] A. Hirose. *Complex-valued neural networks*, volume 400. Springer Science & Business Media, 2012.

[13] J. Hua, H. Sun, Z. Shen, Z. Qian, and S. Zhong. Accurate and efficient wireless device fingerprinting using channel state information. In *Proc. of IEEE INFOCOM 2018-IEEE Conference on Computer Communications*, pages 1700–1708. IEEE, 2018.

[14] T. Jian, B. C. Rendon, E. Ojuba, N. Soltani, Z. Wang, K. Sankhe, A. Gritsenko, J. Dy, K. Chowdhury, and S. Ioannidis. Deep learning for RF fingerprinting:

A massive experimental study. *IEEE Internet of Things Magazine*, 3(1):50–57, 2020.

[15] H. Li, K. Gupta, C. Wang, N. Ghose, and B. Wang. RadioNet: Robust deep-learning based radio fingerprinting. In *Proc. of 2022 IEEE Conference on Communications and Network Security (CNS)*, pages 190–198. IEEE, 2022.

[16] Z. Liang, M. Tao, J. Xie, X. Yang, and L. Wang. A radio signal recognition approach based on complex-valued cnn and self-attention mechanism. *IEEE Transactions on Cognitive Communications and Networking*, 8(3):1358–1373, 2022.

[17] L. Morge-Rollet, F. Le Roy, D. Le Jeune, and R. Gautier. Siamese network on I/Q signal for RF fingerprinting. In *Proc. of Conference on Artificial Intelligence for Defense (CAID) 2020*, 2020.

[18] D. Nouichi, M. Abdelsalam, Q. Nasir, and S. Abbas. IoT devices security using RF fingerprinting. In *Proc. of 2019 Advances in Science and Engineering Technology International Conferences (ASET)*, pages 1–7. IEEE, 2019.

[19] T. Ohtsuji, T. Takeuchi, T. Soma, and M. Kitsunezuka. Noise-tolerant, deep-learning-based radio identification with logarithmic power spectrum. In *ICC 2019-2019 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2019.

[20] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[21] G. Qing, H. Wang, and T. Zhang. Radio frequency fingerprinting identification for zigbee via lightweight cnn. *Physical Communication*, 44:101250, 2021.

[22] K. Ren, Z. Qin, and Z. Ba. Toward hardware-rooted smartphone authentication. *IEEE Wireless Communications*, 26(1):114–119, 2019.

[23] Y. Ren, L. Peng, W. Bai, and J. Yu. A practical study of channel influence on radio frequency fingerprint features. In *Proc. of 2018 IEEE International Conference on Electronics and Communication Engineering (ICECE)*, pages 1–7. IEEE, 2018.

[24] S. Riyaz, K. Sankhe, S. Ioannidis, and K. Chowdhury. Deep learning convolutional neural networks for radio identification. *IEEE Communications Magazine*, 56(9):146–152, 2018.

[25] W. Shang, K. Sohn, D. Almeida, and H. Lee. Understanding and improving convolutional neural networks via concatenated rectified linear units. In *Proc. of international conference on machine learning*, pages 2217–2225. PMLR, 2016.

[26] G. Shen, J. Zhang, A. Marshall, and J. R. Cavallaro. Towards scalable and channel-robust radio frequency fingerprint identification for lora. *IEEE Transactions on Information Forensics and Security*, 17:774–787, 2022.

[27] G. Shen, J. Zhang, A. Marshall, L. Peng, and X. Wang. Radio frequency fingerprint identification for lora using deep learning. *IEEE Journal on Selected Areas in Communications*, 39(8):2604–2616, 2021.

[28] N. Soltanieh, Y. Norouzi, Y. Yang, and N. C. Karmakar. A review of radio frequency fingerprinting techniques. *IEEE Journal of Radio Frequency Identification*, 4(3):222–233, 2020.

[29] J. Stankowicz, J. Robinson, J. M. Carmack, and S. Kuzdeba. Complex neural networks for radio frequency fingerprinting. In *Proc. of 2019 IEEE Western New*

York Image and Signal Processing Workshop (WNYISPW), pages 1–5. IEEE, 2019.

[30] C. J. Swinney and J. C. Woods. Unmanned aerial vehicle flight mode classification using convolutional neural network and transfer learning. In *Proc. of 2020 16th International Computer Engineering Conference (ICENCO)*, pages 83–87. IEEE, 2020.

[31] P. Virtue, X. Y. Stella, and M. Lustig. Better than real: Complex-valued neural nets for MRI fingerprinting. In *Proc. of 2017 IEEE international conference on image processing (ICIP)*, pages 3953–3957. IEEE, 2017.

[32] C. Wang, J. Dani, X. Li, X. Jia, and B. Wang. Adaptive fingerprinting: website fingerprinting over few encrypted traffic. In *Proc. of the Eleventh ACM Conference on Data and Application Security and Privacy*, pages 149–160, 2021.

[33] X. Wang, Y. Zhang, H. Zhang, Y. Li, and X. Wei. Radio frequency signal identification using transfer learning based on LSTM. *Circuits, Systems, and Signal Processing*, 39:5514–5528, 2020.

[34] L. Ying, J. Li, and B. Zhang. Differential complex-valued convolutional neural network-based individual recognition of communication radiation sources. *IEEE Access*, 9:132533–132540, 2021.