RadioNet: Robust Deep-Learning Based Radio Fingerprinting

Haipeng Li[†], Kaustubh Gupta[§], Chenggang Wang[†], Nirnimesh Ghose[§], Boyang Wang[†]
[†]University of Cincinnati, [§]University of Nebraska-Lincoln
{*li2hp, wang2c9*}*@mail.uc.edu, kgupta97@huskers.unl.edu, nghose@unl.edu, boyang.wang@uc.edu*

***Abstract*—Radio fingerprinting identifies wireless devices by leveraging hardware imperfections embedded in radio frequency (RF) signals. While neural networks have been applied to radio fingerprinting to improve accuracy, existing studies are not robust due to two major reasons. First, there is a lack of informative parameter selections in pre-processing over RF signals. Second, deep-learning-based radio fingerprinting derives poor performance against temporal variations in the cross-day scenario.**

In this paper, we enhance the robustness of deep-learning-based radio fingerprinting from three aspects, including parameter selection in pre-processing, learning methods, and evaluation metrics. First, we conduct extensive experiments to demonstrate that careless selections of parameters in pre-processing can lead to over-optimistic conclusions regarding the performance of radio fingerprinting. Second, we leverage adversarial domain adaptation to improve the performance of radio fingerprinting in the cross-day scenario. Our results show that adversarial domain adaptation can improve the performance of radio fingerprinting in the cross-day scenario without the need of recollecting large-scale RF signals across days. Third, we introduce device rank as an additional metric to measure the performance of radio fingerprinting compared to using accuracy alone. Our results show that pursuing extremely high accuracy is not always necessary in radio fingerprinting. An accuracy that is reasonably greater than random guess could lead to successful authentication within a second when we measure with device rank.

I. Introduction

Radio fingerprinting, which authenticates wireless devices over radio frequency (RF) signals at the physical layer, is a critical component for the security and trust of wireless networks. It can complement the authentication of wireless devices when traditional methods, such as cryptography, are not available or difficult to deploy. Radio fingerprinting is feasible as RF signals carry non-linear hardware imperfections of radio frequency circuitry due to variations in the manufacturing process [1], [2]. These hardware imperfections carried in RF signals offer opportunities for a receiver to distinguish transmitters. Recent studies [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15] show that leveraging deep learning over RF signals can significantly improve the accuracy of radio fingerprinting.

Despite enormous efforts, two major limitations remain open for the ***robustness*** of deep-learning-based radio fingerprinting. ***First***, there is a lack of consistent pre-processing for preparing the inputs of deep learning from RF signals. Specifically, given a sequence of I/Q samples from RF signals, *slicing windows* [6], [7], [8], [11] are often utilized to extract I/Q traces, which serve as the inputs for a neural network. However, it remains unclear what parameters should be used for extracting I/Q traces such that a neural network primarily learns hardware imperfections rather than other dominating factors, such as channel conditions. Careless selections of parameters in pre-processing could affect the performance and mislead our understanding on radio fingerprinting.

Second, deep-learning-based radio fingerprinting is not robust against temporal variations in the cross-day scenario. For instance, a neural network trained with RF signals from Day 1 derives much lower accuracy if it is tested with RF signals from Day 2. Several methods, such as applying multi-day training [9], [10], performing data augmentation [8], or adding additional component on transmitters [6], [7], have been proposed to tackle this problem. However, multi-day training requires significant amounts of labeled training data across multiple days, which is not always available. Data augmentation could lead to data bias – the augmented data causes distribution shifts compared to the original data, which could derive sub-optimal performance. Adding additional components requires changes to transmitters, which is difficult to apply for devices that are already in service.

In this paper, we investigate and enhance the robustness of radio fingerprinting from three aspects, including *parameter selection in pre-processing*, *learning methods*, and *evaluation metrics*. We conduct comprehensive experiments on one public dataset [6] (collected from 20 USRPs) and our new dataset (collected from 10 HackRF Ones). We examine neural networks over I/Q data represented in the frequency domain and time-frequency domain, respectively. Our main research findings are summarized below:

- We demonstrate that different values of parameters in pre-processing over RF signals can significantly affect the performance of radio fingerprinting. Our experiments show that given the same dataset and same neural network but different values of parameters in pre-processing, the accuracy of radio fingerprinting in the same-day scenario can vary dramatically from the level of extreme optimism (e.g., 99.74%) to the level of hardly distinguishing 20 transmitters (e.g., 9.69%). This indicates that a neural network may achieve a high accuracy but the accuracy may not necessarily reflect its capability in distinguishing hardware imperfections of transmitters. Based on our observations, we suggest steps that can be carried out to mitigate this issue.
- We leverage transfer learning, specifically, *adversarial*

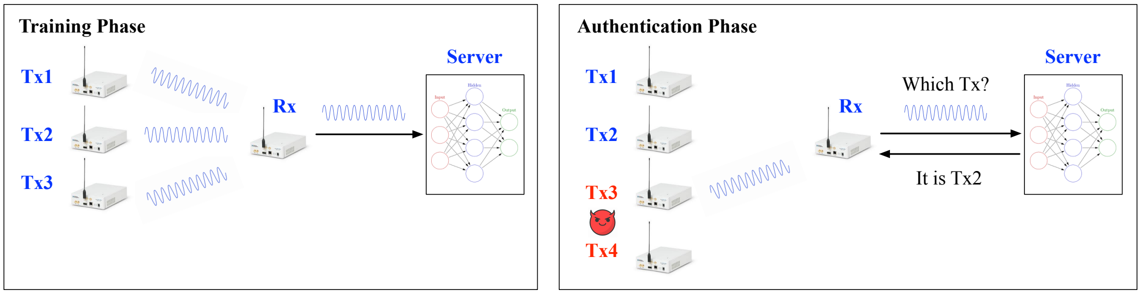

Fig. 1: The system model and threat model of radio fingerprinting. In the authentication phase, Tx3 or Tx4 is considered as an attacker who may impersonate Tx1 or Tx2.

  domain adaptation [16], [17], to promote the robustness of radio fingerprinting against temporal variations in the cross-day scenario. By leveraging adversarial domain adaptation, our method can tune a trained neural network from Day 1 with *a small amount* of labeled data from Day 2 to automatically adapt to temporal variations and maintain a reasonable level of accuracy in the cross-day scenario. For instance, adversarial domain adaptation can improve the cross-day accuracy from 8.41% to 43.17% over NEU dataset and 25.98% to 65.24% over our HackRF dataset.
- We introduce a new metric, *device rank*, to complement the evaluation of radio fingerprinting. Device rank reports the rank of the true transmitter among all the candidates based on aggregated scores over multiple I/Q traces. Device rank is inspired by the use of *key rank* (or *guess entropy*) in the evaluation of side-channel attacks on AES encryption [18], [19]. Our experiments show that even the accuracy is low (but still reasonably higher than random guess), it is feasible to distinguish transmitters within a very short time with device rank. For instance, even the accuracy is only 14.24% over NEU dataset, a classifier can still distinguish which transmitter it is after 35 I/Q traces (around 37 milliseconds in practice) when we measure with device rank.

Reproducibility. The source code and datasets of this study are publicly available at [20].

II. Background

System Model. The system model is described in Fig. 1. It consists of multiple transmitters (Txs), a receiver (Rx), and a server (optional). Upon receiving RF signals, the receiver aims to perform *radio fingerprinting – authenticating which transmitter it is based on RF signals from each transmitter*. The process of radio fingerprinting includes (1) the training phase and (2) the authentication phase (a.k.a test phase).

In the training phase, each transmitter sends RF signals to the receiver. These RF signals, more concretely, I/Q (In-phase and Quadrature) data, are leveraged as training data. These I/Q data can be presented in the *time domain*, i.e., before applying Fast Fourier Transform (FFT) [11]. In addition, I/Q data can also be presented in the *frequency domain* (i.e., after applying FFT) or the *time-frequency domain* (i.e., spectrogram by applying Short-Time Fourier Transform) [13], [21], [22], [23], [24]. Next, the receiver forwards the training data to the server to train a classifier. In the authentication phase, the receiver queries the trained classifier when it receives RF signals. The classifier predicts which transmitter it is.

The server is *optional*, where the prediction (or even the training) could be carried out locally on the receiver side depending on the receiver's computation resources. Whether the server is included or not, it does not affect the threat model.

Threat Model. We assume that all the parties in the training phase are trusted. However, we assume that there are *attackers* in the authentication phase, where an attacker aims to impersonate other transmitters by modifying RF signals. In the threat model, an attacker could be a transmitter that participated the training phase (e.g., Tx3 in Fig. 1 impersonates Tx1 or Tx2) or a transmitter that did not participate the training phase (e.g., Tx4 in Fig. 1 impersonates Tx1, Tx2, or Tx3).

We assume that an attacker can modify I/Q data in RF signals, e.g., changing MAC addresses in packets or adding noise to RF signals, to force the classifier to predict incorrectly. We assume that an attacker is a black-box attacker who does not have access to classifier information, including architectures, hyperparameters, and weights.

Effects of Non-Linear Hardware Imperfections on RF Signals. Non-Linear hardware imperfections exist due to manufacturing variations, such as variations in digital-to-analog converters and power amplifiers even for transmitters manufactured with the same process [9]. In addition, variations of transistors, resistors, inductors, and capacitors can also contribute to non-linearities. The aggregation of these variations is believed to be unique across transmitters, and therefore, form a transmitter's hardware signature. Moreover, these hardware variations lead to non-linear effects in the processing of RF signals and offer opportunities to identify hardware signatures over RF signals. The non-linear effects mainly include I/Q imbalance, differential non-linearity due to digital-to-analog converters, power amplifier non-linearity, and others.

Notations. Given RF signals received on the receiver side, pre-processing needs to be applied to extract I/Q traces from RF signals, where I/Q traces will be used as inputs for a classifier. Details of pre-processing will be discussed later.

An I/Q trace is denoted as x, which contains of a sequence of consecutive I/Q samples $x = (x[1], ..., x[L])$, where $x[i]$

is the i-th I/Q sample and L is trace length. We use $x[i]^R$ and $x[i]^I$ to denote the real part and imaginary part of an IQ sample $x[i]$. A set X consisting of n I/Q traces is denoted as $X = (x_1, ..., x_n)$. We use $\mathcal{Y} = \{y_1^*, ..., y_{|\mathcal{Y}|}^*\}$ to denote a set of transmitters, where y_i^* is a transmitter and $|\mathcal{Y}|$ is the total number of transmitters.

Evaluation Metrics. The performance of radio fingerprinting will be evaluated with two metrics, including *accuracy* and *device rank*. Accuracy of radio fingerprinting is the same concept as accuracy of a machine learning classifier, in which the metric measures how many I/Q traces are predicted with correct labels by a classifier given test I/Q traces. Device rank, on the other hand, measures the rank of the true transmitter among all the candidates based on aggregated scores obtained from a classifier given test traces of a transmitter. *Existing studies mainly utilize accuracy only and we introduce device rank as an additional metric to complement accuracy.*

Accuracy. Given training data $D_{train} = \{X, Y\} = \{(x_1, y_1), ..., (x_n, y_n)\}$, where label $y_i \in \mathcal{Y}$ for $1 \leq i \leq n$, a classifier F is trained in the training phase. In the authentication phase, given a I/Q trace x'_i of test data $D_{test} = \{X', Y'\} = \{(x'_1, y'_1), ..., (x'_{n'}, y'_{n'})\}$, where $y'_i \in \mathcal{Y}$, classifier F outputs a score vector $(s_{i,1}, ..., s_{i,|\mathcal{Y}|})$, where $s_{i,j}$ is the confidence of classifier F on transmitter y_j^* over I/Q trace x'_i. If $y'_i == y_j^*$ and $s_{i,j}$ is the highest score among $(s_{i,1}, ..., s_{i,|\mathcal{Y}|})$, the prediction of classifier F over trace x'_i is correct. If there are m' traces are predicted correctly among all the n' traces, *accuracy* is computed as m'/n'.

Device Rank. Given a subset $D_{test}^{y_j^*} = \{(x'_1, y_j^*), ..., (x'_{n'}, y_j^*)\}$ of test dataset D_{test}, where all the traces of this subset are from one transmitter $y_j^* \in \mathcal{Y}$, the aggregated score vector $(s_1, ..., s_{|\mathcal{Y}|})$ over these n' traces are computed as

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

where $s_{i,k}$ is the score of transmitter y_k^* given trace x'_i. The aggregated scores are further sorted as $(s_1^*, ..., s_{|\mathcal{Y}|}^*)$ in descending order, where $s_j^* \geq s_{j+1}^*$ for $1 \leq j \leq |\mathcal{Y}| - 1$. *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 suggests that classifier F ranks device y_j^* correctly as the top candidate after n' traces. Otherwise, the authentication for device y_j^* is incorrect over the n' test traces. If device rank converges to 1 with a less number of traces, it indicates the authentication is more effective.

Average Device Rank. Given the device ranks $\{r_1, ..., r_{|\mathcal{Y}|}\}$ of all the $\mathcal{Y}$ transmitters in a dataset, where the dataset is balanced and each device rank is computed with the same number of I/Q traces, average device rank can be computed as $r_{avg} = \frac{\sum_{i=1}^{|\mathcal{Y}|} r_i}{|\mathcal{Y}|}$. Average device rank measures the performance of authentication across all the transmitters in a dataset.

Why Device Rank is Helpful? Hardware imperfections are difficult to separate, where scores of candidates in a classifier are often close. If a metric measures accuracy alone, it only keeps top-1 candidates in predictions while the scores of other candidates are ignored. Device rank aggregates the scores across multiple traces to make a more informed decision.

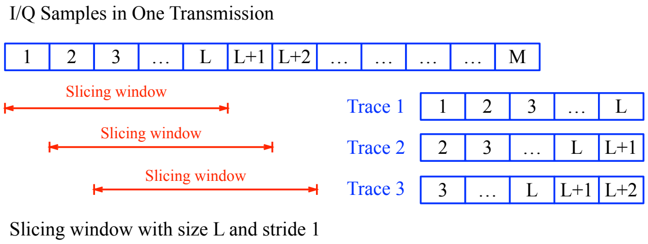

Fig. 2: An example of slicing window method over M I/Q samples given trace length L and stride is 1.

Pre-Processing of I/Q Samples. *Slicing* [6], [7], [8], [11] is a common pre-processing method to extract I/Q samples from RF signals and prepare I/Q traces for a classifier. Specifically, given a sequence of M I/Q samples and a sliding window with length L, where $L << M$, slicing extracts the L I/Q samples within the sliding window as an I/Q trace. Next, the window slides to the right with a stride of s and extracts the L I/Q samples from the updated window as the next I/Q trace. The process is repeated until a certain number of I/Q traces is reached. Trace length L and stride s are two important parameters of this pre-processing. L is fixed within a dataset as a classifier expects the same length of I/Q traces. Stride s can be fixed or dynamic depending on how pre-processing is designed. An example of slicing is described in Fig. 2. Slicing can be applied to I/Q samples in the time domain and also in the frequency domain. These traces can be considered as time-series data with 2 channels, where one channel is In-phase and the other is Quadrature.

Time-Frequency Domain. I/Q samples can also be represented in the time-frequency domain. Specifically, when we collect RF signals at the receiver side, I/Q samples can be collected before and after Fast Fourier Transform (FFT). I/Q samples before FFT are represented in the time domain. To obtain I/Q data in time-frequency domain, also referred to as *spectrogram*, Short-Time Fourier Transform (STFT) is applied to I/Q samples in the time domain. The Discrete STFT can be described as below

$$S(m, w) = \sum_{n=-\infty}^{\infty} x[n]w[n-m]e^{-j\omega n} \tag{2}$$

where $x[n]$ is the signal (i.e., I/Q samples in the time domain), $w[m]$ is a window function of STFT, m is the STFT window size, and ω is the frequency. We implement STFT with package `scipy.signal` in Python.

Given a number of M I/Q samples in the time domain, we can generate a *spectrogram* with length $\lfloor \frac{2M}{m} \rfloor$ and width m. After obtaining the spectrogram, slicing with trace length L is applied to split it into smaller segments with length L and width m, where these segments are used as inputs of a classifier. Put differently, I/Q traces in the time-frequency domain can be considered as time-series data with m channels, where m is STFT window size.

III. Radio Fingerprinting in Cross-Day Scenario

Radio fingerprinting suffers significant performance drop against temporal variations in the cross-day scenario, where a classifier is trained with I/Q traces on Day 1 but is tested with I/Q traces on Day 2. This is mainly because the changes in wireless channel conditions lead to distribution shifts of I/Q traces, which affect the capability of a classifier to distinguish hardware imperfections. A straightforward solution is to recollect a large amount of I/Q traces on Day 2 and retrain a classifier. However, recollecting a large amount of I/Q traces could be time-consuming. To better address this problem, we propose to leverage transfer learning, more specifically, adversarial domain adaptation [16].

Our main idea with adversarial domain adaptation is to train a classifier with a *large amount* of I/Q traces on Day 1 and a *small amount* of I/Q traces on Day 2. Then, this classifier is further tuned with the small amount of I/Q traces on Day 2, such that the tuned classifier can still achieve good performance to authenticate transmitters on Day 2. The reason that only a small amount of I/Q traces on Day 2 is needed is because the knowledge (i.e., weights and feature space) about hardware imperfections can be learned from Day 1 and transferred to the classification task on Day 2. With our method, there is no need to recollect a large amount of I/Q traces on Day 2, which can significantly reduce the overhead in data collection for cross-day radio fingerprinting.

Adversary Domain Adaptation. Transfer learning can transfer knowledge (e.g., weights and hyperparameters) learned from one dataset (referred to as a *source dataset*) to a new dataset (referred to as a *target dataset*), such that a classifier can still perform well over the new dataset. It is one of the major approaches for machine learning classifiers to overcome distribution shifts across datasets. Transfer learning typically can include three steps, including *training*, *tuning*, and *testing*. Target data is split into target training data and target test data, where target training data is used in training (optional) while tuning and target test data is used in testing.

Adversarial Domain Adaptation (ADA) [16], [17], [25] leverages a *domain adversarial network* to learn a domain-invariant feature space. Specifically, assume there is a source dataset and a target dataset, the structure of a domain adversarial network consists of a Feature Extractor F, a Domain Discriminator D, and a Source Classifier C as shown in Fig. 3. The Feature Extractor, Domain Discriminator, or Source Classifier, in essence, is a neural network. The parameters of the Feature Extractor, Domain Discriminator and Source Classifier can be represented as θ_F, θ_D, and θ_C respectively.

During the training of a domain adversarial network, the Feature Extractor takes source data and target training data as inputs and aims to output domain-invariant features, which are difficult for the Domain Discriminator to distinguish. The Domain Discriminator aims to distinguish whether an output of the Feature Extractor is produced by data from the source or the target. The Source Classifier aims to minimize its loss on predicting the correct class label of source data with the outputs produced by the Feature Extractor. The loss function $\mathcal{L}$ can be computed as

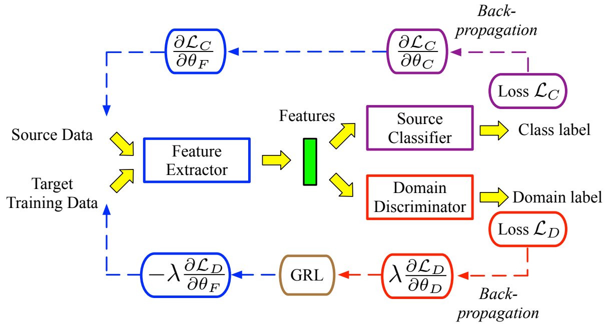
Back-propagation
$\frac{\partial \mathcal{L}_C}{\partial \theta_F}$
$\frac{\partial \mathcal{L}_C}{\partial \theta_C}$
Loss $\mathcal{L}_C$
Source Data
Feature Extractor
Features
Source Classifier
Class label
Target Training Data
Domain Discriminator
Domain label
Loss $\mathcal{L}_D$
$-\lambda \frac{\partial \mathcal{L}_D}{\partial \theta_F}$
GRL
$\lambda \frac{\partial \mathcal{L}_D}{\partial \theta_D}$
Back-propagation

Fig. 3: The structure of a domain adversarial network [16]. GRL stands for Gradient Reversal Layer.

$$\mathcal{L}(\theta_F, \theta_D, \theta_C) = \mathcal{L}_C(\theta_F, \theta_C) - \lambda \mathcal{L}_D(\theta_F, \theta_D) \tag{3}$$

where $\mathcal{L}_C$ is the loss function of the Source Classifier, $\mathcal{L}_D$ is the loss function of the Domain Discriminator, and λ is a pre-defined trade-off parameter shaping features during learning [16]. The parameters of the entire network are updated through back-propagation. After the training, the Feature Extractor F and the Source Classifier C can be extracted out and used to perform classifications over target data.

RadioNet: Our Proposed Method. For cross-day radio fingerprinting, we take I/Q traces on Day 1 as source data and I/Q traces on Day 2 as target data in the context of transfer learning. The standard adversarial domain adaptation described in the previous subsection works well for both the source and target data. To further enhance the performance on Day 2, *we propose to include a tuning step built upon adversarial domain adaptation.*

Specifically, after the training step, our proposed method extracts Feature Extractor F and attach a k-NN classifier to the end of it instead of using Source Classifier C. Then, the parameters of k-NN classifier are tuned with target training data while the weights of Feature Extractor are frozen in tuning. After the tuning, Feature Extractor F and k-NN classifier are used together as a target classifier to perform authenticating transmitters over target test data. The training and tuning steps are highlighted in Fig. 4.

IV. Datasets and Neural Networks

NEU dataset. Al-Shawabka et at. [11] collected radio frequency datasets by leveraging a testbed with 1 USRP as a receiver and 20 USRPs as transmitters. We select one dataset with the setup "Setup A-In-the-Wild, Different Antennas" and refer it as NEU dataset in this paper. RF signals of each transmitter consist of 10 transmissions and each transmission lasts for 30 seconds. Samples are streamed at 2.432 GHz with a sampling rate of 20 million samples per second and BPSK 1/2 as modulation. About 8.05×10^6 I/Q samples on average

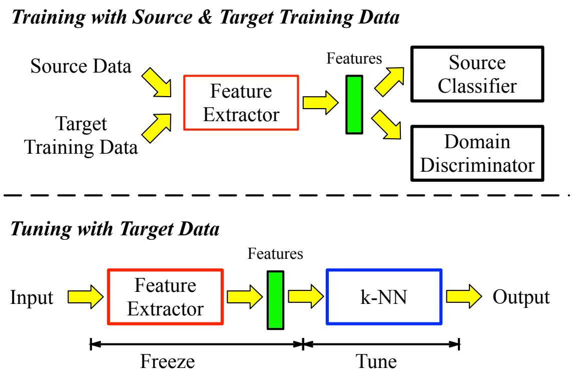

Fig. 4: The training phase and tuning phase in RadioNet.

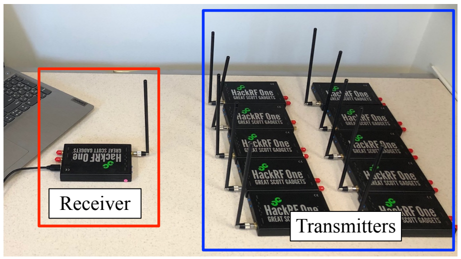

Fig. 5: Our testbed with 1 receiver and 10 transmitters. Each device is a HackRF One running GNU Radio.

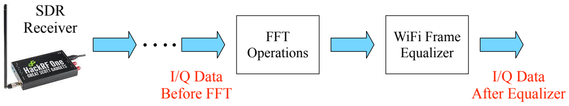

Fig. 6: I/Q data collection on the receiver side.

are observed in each transmission. I/Q samples are collected after WiFi Frame Equalizer in GNU Radio.

HackRF-10 Dataset. We build a testbed with 11 HackRF Ones, where 1 HackRF One is utilized as receiver and 10 HackRF Ones are served as transmitters. This is expanded from our previous textbed in [26]. Each HackRF One is equipped with 1 ANT500 antenna. We use open-source GNU Radio code [27] to establish the WiFi transmissions (IEEE 802.11 a/g) with BPSK 1/2 modulation. We capture I/Q samples at 2.45 GHz center frequency with 2 MHz bandwidth and 2 MHz sampling rate. All the devices are static during the data collection. The receiver is around 3 feet away from a transmitter. We collect I/Q data for two days in our lab. In each day, we collect 3 transmissions and each transmission lasts for 30 seconds. There are 15 seconds between two transmissions. Due to package loss, about 3.26×10^6 I/Q samples are successfully collected in each transmission. We collect the I/Q samples on the receiver side before FFT and after WiFi Frame Equalizer respectively.

Architectures of Neural Networks. In our evaluation, we examine two Convolutional Neural Networks, including Homegrown [11] and Deep Fingerprinting (DF) [28]. Homegrown is a shallow CNN, which consists of 2 convolutional layers and 2 dense layers. It was used in [11] and obtained high accuracy in radio fingerprinting. DF is a deeper CNN which consists of 4 blocks, where each block includes 2 convolutional layers, 1 pooling layer, and 1 dropout. DF was designed in [28] and achieved high accuracy over time-series data, more specifically, side-channel information of encrypted network traffic in website fingerprinting.

V. Evaluation

In this section, we investigate two research questions below through a set of comprehensive experiments.

RQ1: Will parameters of pre-processing cause significant impacts on the performance of radio fingerprinting?

RQ2: Can adversarial domain adaptation improve the performance of radio fingerprinting in the cross-day scenario?

A. Experiments for Investigating RQ1

Changing the parameters of pre-processing, such as stride and trace length, could affect how I/Q traces are sampled from given RF signals, to what degree these traces are overlapped, and to what degree temporal variations may happen. Ideally, if a neural network indeed learns hardware imperfections of transmitters rather than other patterns, such as package content or/and channel conditions, the performance of radio fingerprinting should be relatively stable when the parameters of pre-processing change. To validate this, we perform multiple experiments below.

Experiment A.1: Impacts of stride in the pre-processing of I/Q data in the frequency domain. We evaluate and compare the impact of stride on the performance of radio fingerprinting over I/Q data in the frequency domain. Specifically, given trace length $L = 288$ (we defer the discussion on the impact of trace length in a later experiment), we examine the cases with stride $s = \{1, 0.5L, L, 2L\}$ over NEU dataset and HackRF-10 dataset. We also examine the case where stride s is randomly generated on the fly per I/Q trace, where $L \leq s \leq \frac{W}{L}$ and W is the number of I/Q samples in one transmission.

For NEU dataset, we extract 10,000 I/Q traces per transmitter across 10 transmissions in Day 1, where 1,000 I/Q traces per transmitter are selected from each transmission. In other words, given stride s and trace length L, we prepare an I/Q dataset of 200,000 I/Q traces. For HackRF-10 dataset, we extract 5,000 I/Q traces per transmitter across 3 transmissions in Day 1, where about 1,667 I/Q traces per transmitter are selected from each transmission. In other words, given stride s and trace length L, we prepare an I/Q dataset of 50,000 I/Q traces from HackRF-10 dataset. Given an I/Q trace dataset, we choose 64% traces for training, 16% for validation, and 20% traces for testing. Given a stride and a trace length, we repeat the I/Q trace extraction, its training, and testing for 5 trails.

We use Homegrown and DF and measure the accuracy and device rank of each I/Q dataset over each CNN. We first examine the performance of radio fingerprinting in the same-day scenario with I/Q datasets on Day 1, where both training and testing were performed over I/Q traces on Day 1. In addition, we also repeat the same process to obtain I/Q datasets on Day 2 for NEU dataset and HackRF-10 dataset respectively.

TABLE I: Impacts of stride on accuracy (%) over I/Q data in the frequency domain. Both the same-day scenario and cross-day scenario are evaluated. The results are reported in mean ± standard deviation. The trace length is $L = 288$.

Stride s	CNN	NEU dataset (random guess 5%)		HackRF-10 dataset (random guess 10%)	
		Same-Day	Cross-Day	Same-Day	Cross-Day
s=1	Homegrown	99.74±0.23	6.26±1.95	99.76±0.26	20.40±5.08
	DF	99.95±0.17	6.08±2.44	99.99±0.22	21.85±6.55
s=0.5L=144	Homegrown	26.47±11.21	7.59±0.54	59.31±2.68	24.75±1.94
	DF	50.02±25.10	6.90±1.44	68.63±1.58	26.90±2.92
s=L=288	Homegrown	16.96±4.00	8.72±0.51	52.13±1.85	25.83±3.53
	DF	14.24±6.67	7.31±2.11	60.47±1.25	27.80±1.48
s=2L=576	Homegrown	11.61±1.44	8.68±0.25	45.93±2.48	26.23±2.11
	DF	5.88±1.96	5.70±1.57	47.30±1.18	26.86±1,25
s=random	Homegrown	9.69±0.14	8.70±0.19	44.56±3.04	28.18±0.69
	DF	5.78±1.43	5.76±1.29	39.93±16.49	24.44±8.05

We evaluate the performance of radio fingerprinting in the cross-day scenario, where training I/Q traces are on Day 1 but testing traces are on Day 2.

Observation A.1.1. As shown in Table I, given a trace length, the stride can cause significant impacts on the accuracy in the same-day scenario. Specifically, given trace length is $L = 288$, when stride is only 1, a classifier (either Homegrown or DF) can achieve extremely high accuracy (>99%) on a dataset. On the other hand, when stride increases, the accuracy drops significantly, especially when stride is greater than or equal to trace length (i.e. when there are no overlaps among I/Q traces). This observation is consistent across NEU dataset and HackRF-10 dataset. *Our results suggest that when we choose the parameter of stride, we should choose this parameter such that there will be no or almost no overlaps among I/Q traces. Otherwise, a neural network tends to learn the content of I/Q traces rather than hardware imperfections.*

Observation A.1.2. We observe that device rank would be a more informative metric to understand the performance of radio fingerprinting than accuracy. Specifically, we generate device ranks of the same-day scenario with DF model over I/Q datasets extracted from NEU dataset given trace length is $L = 288$. As presented in Fig. 7, when stride $s = 288$, although the accuracy is only 14.24%, average device rank over 20 transmitters converges to 1 after 35 traces. This suggests that the DF model can correctly identify which transmitter it is after 35 traces, which is about 37 milliseconds of RF signals (i.e., $35 * 288/8.05 \times 10^6$ seconds).

For device ranks over HackRF-10 dataset, we also observe that even the accuracy is low (e.g., 39.93%) given trace length $L = 288$ and stride is random, average device rank over 10 transmitters converges to 1 after 10 traces. Device ranks generated by different strides are similar in Fig. 10 as their accuracy do not drop significantly (e.g., 68% to 39%) and remain much higher than random guess (i.e., 10%). *Our results suggest that pursuing extremely high accuracy may not be necessary for radio fingerprinting. Achieving an accuracy that is reasonably higher than random guess would be sufficient for authenticating a transmitter correctly (i.e., device rank converges to 1) within a short time of RF signals.*

Observation A.1.3. When we examine the accuracy in the cross-day scenario, we find that no matter which stride was used, the accuracy drops significantly to (or close to) the level of random guess. This is consistent with findings in previous studies [11], [9].

Different from previous studies, we further examine the cross-day scenario with device ranks. Specifically, given stride $s = 288$ and trace length $L = 288$ over NEU dataset, the average device rank in the cross-day does not converge to 1 as shown in Fig. 8. We have similar observation over HackRF-10 dataset in Fig. 11.

Experiment A.2: Impacts of trace length in the pre-processing of I/Q data in the frequency domain. We examine the impact of trace length L on the accuracy over I/Q data in the frequency domain. We examine $L = \{144, 288, 576, 864\}$. For each L, we set stride $s = L$ to avoid overlaps across I/Q traces. We evaluate NEU dataset and HackRF-10 dataset with Homegrown and DF. Given stride and trace length, we repeat the I/Q trace extraction, training, and testing for 5 trails as in Experiment A.1. Both the same-day scenario and cross-day scenario are examined.

Observation A.2.1. As shown in Table II, trace length L can affect the accuracy in the same-day scenario, where a smaller trace length derives a higher accuracy. This is expected as a greater trace length suggests the I/Q traces carry greater temporal variations, which lead to greater variations across I/Q traces and therefore accuracy drops.

When we increase trace length over HackRF-10 dataset, the two CNNs can still achieve reasonably high accuracy, although accuracy drops are also observed. However, for NEU dataset, the accuracy of DF drops to the level of random guess when $L \geq 576$ while Homegrown maintains a slightly higher accuracy (12%). *Our results suggest that a greater trace length should be used whenever it is possible, otherwise the classifier could derive over-optimistic accuracy due to the lack of temporal variations within an I/Q dataset. On the other hand, when the trace length is too long, a classifier may fail to authenticate transmitters effectively. We recommend performing experiments over multiple values of trace length and comparing results to decide proper values of trace length for a dataset.* The cross-day scenario offers poor performance regardless which trace length is selected.

Experiment A.3: Impacts of STFT window size in the pre-processing of I/Q data in the time-frequency domain.

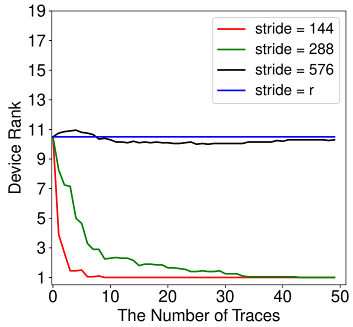

Fig. 7: Average device rank (frequency domain, same-day, NEU, DF, $L = 288$)

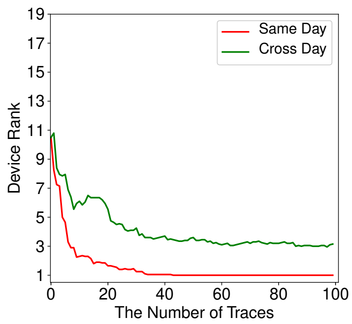

Fig. 8: Average device rank (frequency domain, cross-day, NEU, DF, $L = 288$, and $s = 288$)

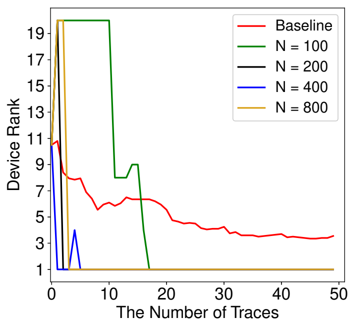

Fig. 9: Average device rank (frequency domain, cross-day, NEU, ADA, $L = 288$, $s = 288$)

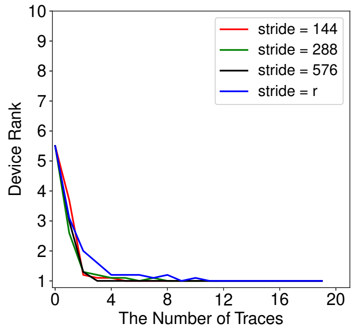

Fig. 10: Average device rank (frequency domain, same-day, HackRF-10, DF, $L = 288$)

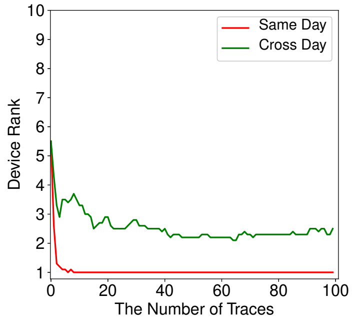

Fig. 11: Average device rank (frequency domain, cross-day, HackRF-10, DF, $L = 288$, $s = 288$)

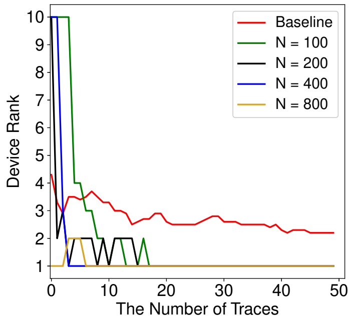

Fig. 12: Average device rank (time-frequency domain, cross-device, HackRF-10, ADA, $L = 288$, $s = 288$)

TABLE II: Impacts of trace length on accuracy (%) over I/Q data in the frequency domain. Both the same-day scenario and cross-day scenario are evaluated. The results are reported in mean ± standard deviation. Stride is $s = L$.

Trace length L	CNN	NEU dataset (random guess 5%)		HackRF-10 dataset (random guess 10%)	
		Same-Day	Cross-Day	Same-Day	Cross-Day
$L = 144$	Homegrown	22.94±8.09	7.13±0.35	52.59±1.46	25.15±3.20
	DF	55.16±9.16	7.33±1.58	64.43±0.70	27.57±2.82
$L = 288$	Homegrown	16.96±4.00	8.72±0.51	52.13±1.85	25.83±3.53
	DF	14.24±6.67	7.31±2.11	60.47±1.25	27.80±1.48
$L = 576$	Homegrown	13.29±1.78	9.26±0.72	46.27±4.03	26.49±2.30
	DF	6.56±2.84	5.82±2.15	57.78±0.92	27.90±0.53
$L = 864$	Homegrown	12.08±2.63	9.67±1.61	44.66±1.11	28.32±2.85
	DF	7.99±4.09	7.32±3.18	57.50±1.92	29.51±0.76

We examine the impact of STFT window size on the accuracy of radio fingerprinting over I/Q data in the time-frequency domain. STFT Window size m decides the size of the STFT window function when transforming I/Q data of the time domain to I/Q data of the time-frequency domain. Performing STFT with a given window size is one additional step in the pre-processing for I/Q data in the time-frequency domain.

We examine HackRF-10 dataset only. We could not generate I/Q data in the time-freqnecy domain of NEU dataset as I/Q data of the time domain are not released. With HackRF-10 dataset, we examine STFT window size $m = \{64, 128, 256\}$. We set stride as $s = 288$ and trace length as $L = 288$ to obtain traces (i.e., segments of spectrogram) after applying STFT.

We select 1,000 traces per transmitter across all the three transmission. Overall, 10,000 traces from 10 transmitters are extracted and used for the evaluation of a classifier. We use 64% for training, 16% for validation, and 16% for testing. We repeat the training and testing for 5 trails.

Observation A.3.1. We observe that increasing STFT window size does not have a critical impact as shown in Table III.

B. Experiments for Investigating RQ2

We leverage adversarial domain adaptation to improve the performance of radio fingerprinting in the cross-day scenario. We also compare it with fine-tuning, which was utilized for radio fingerprinting in [26]. We investigate I/Q data in the frequency domain over NEU dataset and HackRF-10 dataset and I/Q data in the time-frequency domain over HackRF-10 dataset.

TABLE III: Impacts of STFT window size on accuracy (%) over I/Q data in the time-frequency domain. The results are reported in mean ± standard deviation. Stride $s = 288$ and trace length $L = 288$.

STFT window size	CNN	HackRF-10 dataset (random guess 10%)	
		Same-Day	Cross-Day
$m = 64$	Homegrown	30.95±1.65	13.25±3.45
	DF	47.48±0.67	16.10±0.88
$m = 128$	Homegrown	33.62±1.91	11.61±1.31
	DF	49.00±2.01	13.03±0.66
$m = 256$	Homegrown	36.63±1.15	11.49±0.05
	DF	51.85±3.89	15.69±2.53

Implementation of Adversarial Domain Adaptation. We use DF as Feature Extractor. The Source Classifier consists of 1 convolutional layer, 1 pooling layer, and 1 fully connected layer with softmax as the activation function. The Domain Discriminator contains 2 convolutional layers, 2 pooling layers, and 1 fully connected layer with softmax as the activation function. For the training phase of adversarial domain adaptation, we leverage source data and target training data, and train for 50 epochs. For fine-tuning, we train DF with source data and tune the last layer of DF using target training data.

Experiment B.1: Cross-day radio fingerprinting with adversarial domain adaptation (frequency domain). We examine cross-day performance with adversarial domain adaptation over I/Q data in the frequency domain. We select trace length $L = 288$ and stride $s = 288$. Given NEU dataset, we extract 200,000 I/Q traces on Day 1 and 200,000 I/Q traces on Day 2. The I/Q traces from Day 1 serve as source data and I/Q traces from Day 2 serve as target data.

We utilize 64% of source data in the training phase of adversarial domain adaptation, and only a small number of N traces per transmitter in target data are utilized as target training data, where we examine $N = \{100, 200, 400, 800\}$. The target training data is also used in the tuning phase. 20% of target data (40,000 traces from Day 2) are used in testing. For the k-NN classifier in the tuning phase, we choose $k = |\mathcal{Y}| - 1$, where $|\mathcal{Y}|$ is the number of transmitters.

As presented in Table IV, our method based on adversarial domain adaptation can effectively improve cross-day accuracy. When we increase N, the number of traces per transmitter in target training data, cross-day accuracy increases. For instance, given $N = 800$, the accuracy can increase to 43.17% compared to 8.41% of the baseline. Baseline accuracy is the accuracy obtained from training DF with Day 1 and testing with Day 2 directly. As shown in Fig. 9, we observe that device rank can converge to 1 within 20 traces when we use ADA in the cross day scenario while the device rank of baseline does not converge to 1. We also observe that fine-tuning can also improve the accuracy but not significantly over NEU dataset.

We also examine the performance of adversarial domain adaptation over HackRF-10 dataset in the frequency domain. Given trace length $L = 288$ and stride $s = 288$, we extract 50,000 I/Q traces on Day 1 and 50,000 I/Q traces on Day 2. We set $k = 9$ for the k-NN classifier. We have similar observations that adversarial domain adaptation can significantly improve cross-day accuracy and outperform fine-tuning.

Experiment B.2. Cross-day radio fingerprinting with adversarial domain adaptation (time-frequency domain). We examine cross-day radio fingerprinting with adversarial domain adaptation over I/Q data in the time-frequency domain. We choose trace length $L = 288$, stride $s = 288$, and STFT window size $m = 64$. Only HackRF-10 dataset is evaluated. We extract 10,000 I/Q traces in Day 1 and 10,000 I/Q traces in Day 2. Other details are the same as in the previous experiment. As shown in Table V, adversarial domain adaptation can also improve cross-day accuracy when I/Q data are represented in the time-frequency domain. In addition, it can outperform fine-tuning when $N \geq 200$.

VI. Related Work

Due to space limitation, we only discuss deep-learning-based radio fingerprinting. More comprehensive surveys on radio fingerprinting can be found in [29].

To simplify pre-processing steps on RF signals and distinguish transmitters of the same type, recent studies [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15] perform deep learning directly on I/Q samples to fingerprint hardware imperfections of transmitters. Different architectures of neural networks, such as CNNs, LSTM (Long-Short Term Memory), and MLPs (Multi-Layer Perceptrons), have been examined [3], where CNN often outperforms other networks. Complex-valued neural networks [9], [10] can achieve higher accuracy than real-valued neural networks. Existing studies [13], [14] also shown that I/Q data represented in the frequency domain and time-frequency domain can achieve better performance than I/Q data in the time domain. Al-Shawabka et al. [11] established, at the time of writing, the largest public dataset (NEU dataset) that can be utilized to analyze radio fingerprinting. Several studies [3], [14], [13] examined deep-learning-based radio fingerprinting over I/Q data, particularly for Long Range (LoRa) protocols. Identifying Unmanned Aerial Vehicles with radio fingerprinting was studied in [12].

Existing Studies in the Cross-Day Scenario. Cekic et al. [9] utilized CFO (Carrier Frequency Offset) compensation and model-driven augmentation – adding noise to received RF signals – to mitigate the impacts of CFO and multipath before passing RF signals to a complex-valued neural network. The authors also show that multi-day training – training models with RF signals from multiple days – can improve the robustness of radio fingerprinting. Agadakos et al. [10] investigated protocol-agnostic radio fingerprinting with complex-valued neural networks. Pre-processing with decimation and randomized cropping. They demonstrated that multi-day training can increase the robustness of radio fingerprinting. Restuccia et al. [6] proposed to add Finite Input Response Filters on the transmitter side to mitigate the impact of temporal variations across different days. The extensions of their study utilized model-driven data augmentation [8] or channel estimation on the receiver side [7] to mitigate the impact of distribution shifts of RF signals across different days.

TABLE IV: Accuracy (%) in the cross-day scenario (frequency domain).

		N: No. of traces per transmitter in target training data				
		Baseline	100	200	400	800
NEU	Fine-tuning	8.41	9.72±0.98	11.05±0.31	12.35±0.29	13.99±0.08
	ADA		7.50±0.13	10.13±0.80	23.33±0.92	43.17±0.90
HackRF-10	Fine-tuning	25.98	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

TABLE V: Accuracy (%) in the cross-day scenario (time-frequency domain).

		N: No. of traces per transmitter in target training data				
		Baseline	100	200	400	800
HackRF-10	Fine-tuning	17.19	45.60±3.76	50.85±0.21	53.62±1.29	56.04±0.67
	ADA		42.42±0.89	54.41±0.93	58.30±0.10	64.22±0.93

VII. Conclusion

We enhance the robustness of radio fingerprinting in this study. Our experimental results over real-world datasets suggest that adversarial domain adaptation is a promising way to improve the performance of fingerprinting in cross-day scenario. Also, device rank can be used as an important metric in the evaluation of radio fingerprinting.

Acknowledgements

The authors would like to thank the anonymous reviewers for their comments and suggestions. The UC authors were partially supported by National Science Foundation (CNS-1947913).

References

[1] E. Costa, M. Midrio, and S. Pupolin, "Impact of amplifier nonlinearities on ofdm transmission system performance," *IEEE Communications Letters*, vol. 3, no. 2, pp. 37–39, 1999.

[2] S. Merchan, A. G. Armada, and J. L. Garcia, "OFDM performance in amplifier nonlinearity," *IEEE Transactions on Broadcasting*, vol. 44, no. 1, pp. 106–114, 1998.

[3] P. Robyns, E. Marin, W. Lamotte, P. Quax, D. Singelee, and B. Preneel, "Physical-Layer Fingerprinting of LoRa devices using Supervised and Zero-Shot Learning," in *Proc. of ACM WiSec'17*, 2017.

[4] S. Riyaz, K. Sankhe, S. Ioannidis, and K. Chowdhury, "Deep Learning Convolutional Neural Networks for Radio Identification," *IEEE Communications Magazine*, 2018.

[5] H. Jafari, O. Omotere, D. Adesina, H. Wu, and L. Qian, "IoT Devices Fingerprinting using Deep Learning," in *Proc. of Milcom 2018*, 2018.

[6] F. Restuccia, S. D'Oro, A. Al-Shawabka, M. Belgiovine, L. Angioloni, S. Ioannidis, K. Chowdhury, and T. Melodia, "DeepRadioID: Real-Time Channel-Resilient Optimization of Deep Learning-based Radio Fingerprinting Algorithms," in *Proc. of ACM MobiHoc'19*, 2019.

[7] K. Sankhe, M. Belgiovine, F. Zhou, S. Riyaz, S. Ioannidis, and K. Chowdhury, "ORACLE: Optimized Radio Classification through Convolutional Neural Networks," in *Proc. of IEEE INFOCOM'19*, 2019.

[8] N. Soltani, K. Sankhe, J. Dy, S. Ioannidis, and K. Chowdhury, "More is Better: Data Augmentation for Channel-Resilient RF Fingerprinting," *IEEE Communications Magazine*, 2020.

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

[10] I. Agadakos, N. Agadakos, J. Polakis, and M. R. Amer, "Chameleons' Oblivion: Complex-Valued Deep Naural Networks for Protocol-Agnostic RF Device Fingerprinting," in *Proc. of IEEE Euro S&P'20*, 2020.

[11] 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'20*, 2020.

[12] N. Soltani, G. Reus-Muns, B. Salehi, J. Dy, S. Ioannidis, and K. Chowdhury, "RF Fingerprinting Unmanned Aerial Vehicles with Non-Standard Transmitter Waveforms," in *IEEE Transactions on Vehicular Technology*, 2020.

[13] G. Shen, J. Zhang, A. Marshall, L. Peng, and X. Wang, "Radio frequency fingerprint identification for lora using spectrogram and CNN," in *Proc. of IEEE Conference on Computer Communications*. IEEE, 2021, pp. 1–10.

[14] A. Al-Shawabka, P. Pietraski, S. B. Pattar, F. Restuccia, and T. Melodia, "DeepLoRa: Fingerprinting LoRa Devices ar Scale Through Deep Learning and Data Augmentation," in *Proc. of ACM MobiHoc'21*, 2021.

[15] 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 Transactions on Mobile Computing*, 2021.

[16] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," *The journal of machine learning research*, 2016.

[17] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial Discriminative Domain Adaptation," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[18] R. Benadjila, E. Prouff, R. Strullu, E. Cagli, and C. Dumas, "Study of Deep Learning Techniques for Side-Channel Analysis and Introduction to ASCAD Database," *Journal of Cryptographic Engineering*, vol. 10, no. 2, 2020.

[19] S. Bhasin, A. Chattopadhyay, A. Heuser, D. Jap, S. Picek, and R. R. Shrivastwa, "Mind the Portability: A Warriors Guide through Realistic Profiled Side-channel Analysis," in *Proc. of NDSS'20*, 2020.

[20] RadioNet. https://github.com/UCdasec/RadioNet.

[21] N. Soltanieh, Y. Norouzi, Y. Yang, and N. C. Karmakar, "A review of radio frequency fingerprinting techniques," *IEEE Journal of Radio Frequency Identification*, vol. 4, no. 3, pp. 222–233, 2020.

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

[23] 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 Conference on Computer Communications*. IEEE, 2018, pp. 1700–1708.

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

[25] C. Wang, J. Dani, X. Li, X. Jia, and B. Wang, "Adaptive Fingerprinting: Website Fingerprinting over Few Encrypted Traffic," in *Proc. of ACM CODASPY'21*, 2021.

[26] H. Li, C. Wang, N. Ghose, and B. Wang, "POSTER: Robust Deep-learning-based Radio Fingerprinting with Fine-Tuning," in *Proc. ACM WiSec'21*, 2021.

[27] B. Bloessl, M. Segata, C. Sommer, and F. Dressler, "Performance assessment of ieee 802.11p with an open source sdr-based prototype," *IEEE Transactions on Mobile Computing*, vol. 17, no. 5, pp. 1162–1175, 2018.

[28] P. Sirinam, M. Imani, M. Juarez, and M. Wright, "Deep Fingerprinting: Understanding Website Fingerprinting Defenses with Deep Learning," in *Proc. of ACM CCS'18*, 2018.

[29] N. Soltanieh, Y. Norouzi, Y. Yang, and N. C. Karmakar, "A Review of Radio Frequency Fingerprinting Techniques," in *IEEE Journal of Radio Frequency Identification*, 2020.