Chameleons' Oblivion: Complex-Valued Deep Neural Networks for Protocol-Agnostic RF Device Fingerprinting

Ioannis Agadakos*⊗, Nikolaos Agadakos*†‡, Jason Polakis‡, Mohamed R. Amer†◇
⊗*SRI International*
‡*University of Illinois at Chicago*
◇*Robust AI*

***Abstract*—Prior work has demonstrated techniques for fingerprinting devices based on their network traffic or transmitted signals, which use software artifacts or characteristics of the underlying protocol. However these approaches are not robust or applicable in many real-world scenarios. In this paper we explore the feasibility of device fingerprinting under challenging realistic settings, by identifying artifacts in the transmitted signals caused by devices' unique hardware "imperfections". We develop RF-DCN, a novel Deep Complex-valued Neural Network (DCN) that operates on *raw* RF signals and is *completely agnostic of the underlying applications and protocols*. We introduce two DCN variations: a retrofitted Convolutional DCN (CDCN) originally created for acoustic signals, and a novel Recurrent DCN (RDCN) for modeling time series. Our work demonstrates the feasibility of operating on raw I/Q data collected within a narrowband spectrum from open air captures across vastly different modulation schemes. In contrast to prior work, we do not utilize knowledge of the modulation scheme or protocol intricacies such as carrier frequencies. We conduct an extensive experimental evaluation on large and diverse datasets as part of a DARPA red team evaluation, and investigate the effects of different environmental factors as well as neural network architectures and hyperparameters on our system's performance. Our novel RDCN consistently outperforms all baseline neural network architectures, is robust to noise, and can identify a target device even when numerous devices are concurrently transmitting within the band of interest under the same or different protocols. While our experiments demonstrate the applicability of our techniques under challenging conditions where other neural network architectures break down, we identify additional challenges in signal-based fingerprinting and provide guidelines for future explorations.**

1. Introduction

In recent years, the feasibility of identifying devices through browser or device fingerprints has garnered significant attention [1], [2], [3], [4], [5], [6], [7], [8]. Such techniques are not restricted to device characteristics and specifications but can also fingerprint devices based on imperfections inherent in the device's hardware [9], [10]. In an alternative line of research, techniques have been proposed for fingerprinting devices based on unique characteristics of the device's hardware that are exposed in their transmissions. A remote network-based fingerprinting technique was presented in the seminal work by Kohno et al. [11], but could be prevented by software-level defenses [12]. Brik et al. [12] proposed a more robust but geographically-localized technique that focused on transmitted radio frequency (RF) signals. However, their approach also relied on protocol-specific information, and crucial practical factors were not considered in the experimentation and evaluation. While both approaches pose significant contributions, the diversity of wireless protocols in the wild (which differ considerably across versions or implementations [13]), coupled with the emergence of new protocols and standards, hinders their applicability in many real-world scenarios. Even the more generalizable techniques by Danev et al. [14], [15] require, at least, some knowledge of higher-level but protocol-defined information, such as the frequencies of the carrier or sub-carrier signals or the bandwidth of the transmission channel. Inspired by the extensive prior research in the area, we explore novel methods to ameliorate current limitations and study the feasibility of fingerprinting without any prior knowledge of even the central carrier frequencies or modulation schemes involved.

Furthermore, while RF signals are inherently complex-valued, limitations of both theoretical frameworks and available development tools have constrained RF-based research to real-valued conventional networks that severely limit the representational power of generated models. However, recent breakthroughs laid the theoretical foundations for complex-valued networks [16] and paved the road for a new strain of deep learning models which are able to work *natively* with complex data. These networks have higher representational capacity and thus more degrees of freedom, allowing them to locate appropriate manifolds that are out of the reach of conventional real-valued networks. In this work, we leverage the power of complex-valued deep learning architectures, study their ability to operate on raw complex RF data, and devise methodologies for training these novel networks. Our models are trained on *raw* I/Q data of signals captured under challenging environmental conditions in *real* settings, to show their applicability in practical and large scale deployments. In fact, RF-DCN is able to fingerprint devices using open air captures without any protocol preprocessing, and without knowledge of the underlying

*Equal contribution joint first authors.
†Part of this work was done while working at SRI International.

protocol or what carrier frequencies to look for.

Specifically, our system guides the deep neural network towards identifying the artifacts in signals that are left by the unavoidable minor hardware variations or impairments (hereby referred to as "imperfections") that occur during the manufacturing process. To train our models and demonstrate their performance under different conditions, we utilize a plethora of datasets with as many as 1,000 devices. With each sample corresponding to a captured transmission of only 64 μseconds, our total training set requires only *6.4-51.2 milliseconds* worth of captured signals per device. We show that when using only a 64 μs long signal, RF-DCN can correctly fingerprint the device 72% and 100% of the time for WiFi and ADS-B transmitters respectively. We also show that our system is able to train a *single* model for *both* types and obtain an accuracy of 82% despite the drastic differences between the two signals (i.e., protocols, transmission frequencies, and modulation schemes).

We show that RF-DCN is completely protocol-agnostic at the different spectrum ranges of our dataset and can handle devices that operate at multiple ranges (e.g., a device that transmits WiFi at both 2.4GHz *and* 5GHz) without any prior knowledge, without relying on protocol implementation flaws like predictable sequence numbers [17], exposed device identifiers [18], [19], [20] or software-level data patterns (our dataset includes only control packets). Our system can be potentially deployed in a wide range of different scenarios, including identifying devices within a given area, device-authentication for authorizing access to resources (e.g., a cell tower or access point), and unmasking spoofed transmissions (e.g., relay attacks or attacks using compromised certificates). Nonetheless, our experimental evaluation is designed to abstract away the specificities of the deployment scenario and focus on the task of identifying devices based on their transmitted RF signals.

Overall, our research presents new techniques for handling raw RF I/Q data in their native complex format, highlights the challenges of RF-based fingerprinting in *real-world* settings, and demonstrates novel directions for passively fingerprinting devices without requiring any knowledge of the underlying protocols. We believe that our system presents a significant advancement towards the deployability of device fingerprinting under realistic and extremely challenging conditions. Our evaluation explores numerous critical dimensions, yet many interesting future directions remain unexplored. We hope that the methodologies, network architectures, and experimental findings presented in this work will spearhead more research in the area. In summary, our research contributions are:

- We present a novel device fingerprinting technique that works on complex I/Q data representing *raw* RF wireless signals. We build a novel deep complex-valued network that is *protocol-agnostic*, designed to identify devices based on learnable characteristics from passively captured transmitted wireless signals. This is the first, to our knowledge, system able to fingerprint devices using unprocessed raw signals in a wide range of frequencies.
- We detail a series of neural network architectures and configurations, and explore their suitability for device fingerprinting. We derive appropriate insights and highlight the inherent limitations of certain architectures, which can help guide future research.
- We provide an extensive experimental evaluation using datasets generated as part of a DARPA red team evaluation. Our experiments investigate multiple dimensions of signal-based device fingerprinting and assess the impact of multiple factors under different environmental settings and real-world conditions that previous work has not explored. We demonstrate the effectiveness of our system's fingerprinting capabilities under challenging and realistic scenarios, using real-world, multi-device, open air captures of RF signals.

2. Background and Deployment Predicates

2.1. RF Primer

We first introduce background information that facilitates comprehension of the following sections. This includes basic telecommunication terms and notions and technical details on the two signal families that we use in this paper. As the methodologies involved in signal processing are sufficiently complicated, we refer interested readers to [21] for a more complete overview.

Baseband signals. Signals are encountered in this form before being transmitted or after the receiver removes the carrier signal, during the early stages of demodulation, which centers the received signal around 0Hz.

Modulation is the process by which a transmitter encodes information so that it can be effectively and efficiently transmitted. The main idea is to encode the information from a *baseband* signal by utilizing a *carrier* signal of higher frequency. Information is encoded by modifying one or more of the carrier's characteristics: amplitude, phase, and frequency.

I/Q data. Signals are commonly represented in a format known as I/Q form [22]. This is based on a methodology of representing a signal using a vector of complex numbers. `I` stands for the in-phase component whereas `Q` stands for the quadrature or out-of-phase component. This can be represented by the following equation:

$$S(t) = A\cos\theta t + B\sin\theta t \rightarrow S(t) = I + Qj$$

In essence, any signal can be generated by a combination of in-phase and out-of-phase components and, hence, it is widely used in signal processing and telecommunications.

SDR. Software Defined Radios revolutionized the field of telecommunications as they allowed for the reception and transmission of RF signals with any modulation scheme and at any frequency, while offering the convenience of defining everything through software while using the same underlying hardware. One of the key components behind this technology is the quadrature mixer. The quadrature mixer can synthesize any signal using two signals with a phase difference of 90 degrees, and these quadrature signals can be conveniently expressed in analytical form as I/Q data. Figure 11 in Appendix A depicts the process of transmuting I/Q data to a modulated signal using a quadrature modulator.

Channel: The medium through which the signal is transmitted, which can greatly alter signals in transmission and have a detrimental effect. A channel's characteristics are often ephemeral as they change due to a variety of environmental conditions. As such, the same signal

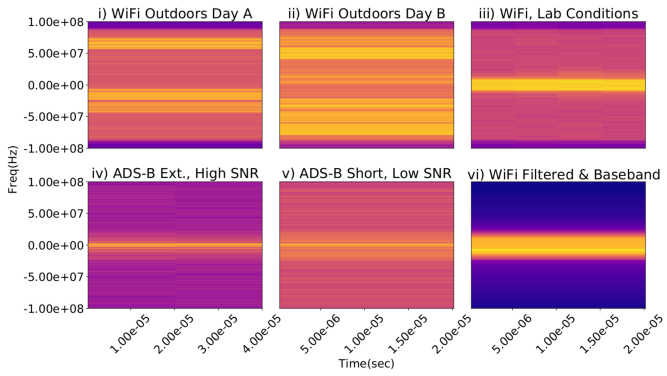


Figure 1: Spectrograms depicting WiFi and ADS-B signals captured under different channel environments and SNR. The outdoors WiFi signals depict open air captures.

transmitted through the same medium at different times can differ drastically. Tackling this inherent challenge is crucial, and we explore it in depth during our evaluation.

SNR. Due to the transient nature of a channel's conditions, the Signal-to-Noise Ratio metric is useful in denoting the quality of the received signal by quantifying how much of a received signal is noise.

WiFi. WiFi signals are part of the 802.11 IEEE protocol family. While many variations exist, the most frequently encountered versions are 802.11b and 802.11g (which use a central frequency of 2.4GhZ) and the newer 802.11n (which utilizes the 5GhZ band). WiFi signals utilize the Orthogonal Frequency Division Multiplexing (OFDM) modulation which encodes information in *multiple* sub-carrier signals, where each sub-carrier can be modulated using amplitude, phase or frequency modulation schemes such as 16QAM, 16QPSK etc. The information content in a WiFi signal also vary significantly in size. We direct the interested reader to [23] for an excellent resource with more information.

ADS-B. A protocol of choice used by air crafts that is scheduled to replace most RADAR systems according to the Federal Aviation Agency [24]. It is encountered in two forms, either short or extended (64 or 120 μs). Apart from the header and the CRC, short messages only contain a unique aircraft identifier (24bits), whereas the extended version carries additional information on the altitude, position, heading, horizontal and vertical velocity. ADS-B transmits at 1090MHz with 50KHz bandwidth or at 978MHz with 1.3MHz bandwidth. The modulation employed in this protocol is referred to as Pulse Position Modulation (PPM), which encodes data by increasing or decreasing the width of the carrier pulses according to the value of the modulating signal at a given time. It is a much simpler modulation scheme than the one used in WiFi signals and the messages transmitted are either 5 or 10 bytes. An important characteristic of this protocol is that it is very robust to multipath effects and environmental noise. A more detailed overview and description can be found in [25]. While ADS-B packets contain a unique device identifier, certain transceivers have the option for anonymous mode where a randomized ID is sent [26]. Prior work has discussed how ADS-B data can easily be spoofed to disrupt air travel safety [27], [28]. As such, the ability to fingerprint and identify transmitters without relying on identifiers has significant operational value.

Spectrogram. Captured transmissions can be visually represented as spectrograms. To obtain a spectrogram from raw I/Q data, a Short Term Fourier Transform (STFT) [29] is performed over the entire captured trace. The SFTF outcome depicts the spectral content of the signal through time and one can easily identify drastic changes, the effect of noise, or different parts of the transmission. Figure 1 shows a collection of spectrograms for both WiFi and ADS-B signals captured under different conditions to highlight the aforementioned effects. Plots (i) and (ii) show a WiFi signal captured from the *same* device over two different days and (vi) depicts the signal after a basebanding and low pass filtering operation. Since the capture was conducted outdoors, the interference from other channels/devices is evident in the first two plots. In (iii) we show an unprocessed WiFi signal that was captured indoors where the SDR basebanded the signal without using an intermediate frequency (IF); one can notice that noise is still present since the signal has not been filtered. Plots (iv), (v) show both versions of ADS-B transmissions under different SNR (the x-axes are different for the two signals as the short version is almost half the transmission).

2.2. Deep Learning Primer

This section introduces the basic concepts required for following the deep learning networks and methodologies discussed in this paper. Readers familiar with the field may advance to the next section. A detailed presentation of foundational concepts can be found in [30].

CNN. Convolutional Neural Networks [31] perform exceptionally well at detecting structural properties and spatial points of interest, and variants have been used in a plethora of applications with great success, such as face and gender detection [32], semantic image segmentation [33], speech recognition [34] and stock prediction [35]. The basic idea behind a CNN is to perform a convolution operation between a weight vector of a given size (often called a kernel) and a subset of the input data, with the length of each convolution being the size of the kernel. Convolutional layers are almost always used in conjunction with pooling layers to achieve dimensionality reduction without loss of information.

RNN. Recurrent Neural Networks are networks that excel at modeling sequential or time series data. The core concepts behind them are the context and temporal relations between data points, potentially in different parts of the input. Two advanced forms of RNNs enhance the concept of context by introducing well-defined memory properties, namely Long Short Term Memory Models (LSTM) and Gated Recurrent Units (GRU). These advanced models allow the recurrent network to "choose" and "memorize" parts of the context that are significant for the model, thus, emulating a form of memory. While the mathematical models behind each variant are different, both networks have been successfully used in practice [36], [37], [38], [39].

Back Propagation Through Time. As recurrent networks capture sequential properties and try to find temporal relations, the gradient descent step during the back

propagation phase essentially attempts to link the current state with previous states in time. This is referred to as Back Propagation Through Time (BPTT) [40]. RNNs face the problem of vanishing or exploding gradients for lengthy time sequences; while LSTMs are known to be more robust they can still suffer from the same problems albeit at much longer sequences [41].

Complex-valued inputs. Most networks currently in use operate on real-valued data. However, many signals (e.g., RF) are naturally expressed as complex numbers. Down-converting complex numbers to either the real or imaginary part limits the available information and introduces a miss-match between the model and the task at hand. To introduce complex-valued inputs and, thus, create complex-valued networks that theoretically have higher representational power (as they can express solutions in complex-valued manifolds) one needs to address many theoretical challenges. This includes providing alternatives to elementary units such as activation units, or properly initializing complex-valued weights in hidden layers.

Complex batch normalization. Data normalization is commonly used to efficiently train most types of deep networks. However, when the underlying problem lies in the complex domain, data requires special handling as outlined in [16]. Briefly, one cannot perform two-way independent normalization in the real and imaginary part of a complex number, as the relation of the real and imaginary axes also incorporates information. This is becomes apparent if we consider the polar representation of a complex number $z = re^{\iota\phi}$, where ϕ is the angle the number forms in the complex plane with the positive real axis, and is given by $\phi = atan2(IM(z), Re(z))$. If one operates on either the real or imaginary part independently, thus disregarding any correlation, an affine transformation is introduced which is not equivalent to normalization. To properly handle the normalization process, we treat the complex numbers as 2D vectors and the process as 2D whitening. We scale the $\mathbf{0}$ centered data by the inverse square root of their covariance matrix $\mathbf{V}$ (the existence of the inverse is guaranteed by the Tikhonov regularization; see Appendix A.4): $\tilde{x} = (\mathbf{V})^{-\frac{1}{2}}(\mathbf{x} - E[\mathbf{x}])$, where $\mathbf{V}$ is the 2x2 covariance matrix:

$$\mathbf{V} = \begin{pmatrix} V_{rr} & V_{ri} \\ V_{ri} & V_{ii} \end{pmatrix} = \begin{pmatrix} Cov(\Re(x), \Re(x)) & Cov(\Re(x), \Im(x)) \\ Cov(\Re(x), \Im(x)) & Cov(\Re(x), \Re(x)) \end{pmatrix}$$

The complex batch normalization layer is defined as:

$$BN(\tilde{\mathbf{x}}) = \gamma\tilde{\mathbf{x}} + \beta, \; where\, \gamma = \begin{pmatrix} \gamma_{rr} & \gamma ri \\ \gamma_{ri} & \gamma_{ii} \end{pmatrix}$$

Complex activation units. As in the normalization case, activation functions must also be adapted to handle complex-valued numbers. Several activation units have been proposed such as ModRelu [42], $\mathbb{C}$ReLU [16] and zReLU [43]. These functions preserve differentiability, at least in part, to maintain compatibility with back-propagation. As shown in [44], full complex differentiable functions are not strictly required. While many activation functions have been proposed, we evaluate the following complex activation functions:

$$\mathbb{C}ReLU = ReLU(\Re(z)) + \iota ReLU(\Im(z))$$

$$zReLU = \begin{cases} z & if\theta_z \in [0, \pi/2], \\ 0 & elsewhere \end{cases}$$

Complex weight initialization. Appropriate weight initialization is often considered helpful for the training process of networks. It can help avoid the risk of vanishing or exploding gradients and facilitate a faster convergence. In the complex case, as derived in [16], the variance of the weight vectors is given by Var(W) = $2\sigma^2$, where σ is Rayleigh distribution's only parameter; σ can be selected according to any initialization criterion such as those described in [45], [46]. Note that as the variance is not affected by phase, one can initialize the phase ϕ of W as $\phi_W \sim U[-\pi, \pi]$.

2.3. Deployment Predicates

The techniques that we present can be applied to a wide range of scenarios. Nonetheless, for ease of presentation, we will assume that role of an entity aiming to identify the sources of wireless signals (be it smartphones or air crafts). We assume that this entity passively collects wireless signals within a certain area, without interacting with or actively probing the devices. Similarly we do not require the user to interact with a specific resource. The only assumption is that the device is transmitting some form of wireless signal; in our experimental evaluation we use WiFi and ADS-B traces, but this could be applied to other types as well (e.g., Bluetooth, GSM, 4G, etc). Depending on the scenario, the application can include the deployment of multiple antennas within a larger physical area (e.g., for identifying the devices within a given area) or may focus on a specific small space (e.g., for detecting rogue access points using stolen certificates).

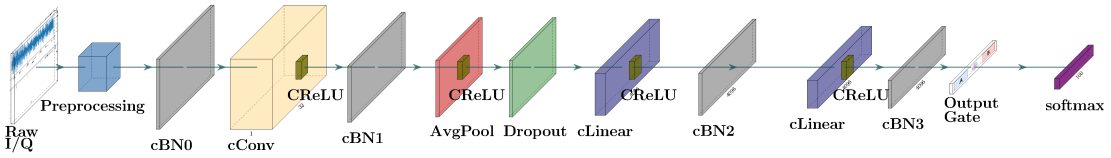
3. System Design and Implementation

Here we present an overview of RF-DCN, and discuss the motivation behind our design and implementation.

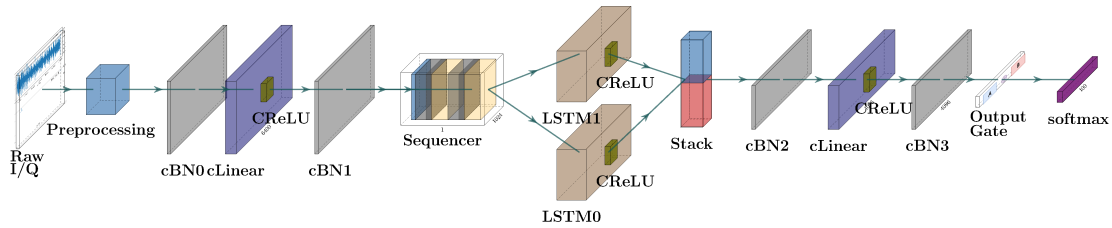
Design constraints. The main goal of RF-DCN is to operate on raw I/Q data so as to decouple the fingerprinting capabilities of our system from protocol or software specificities and implementation flaws. It also removes the burden of manual analysis that would otherwise be required if new or custom protocols were encountered. To that end, we limit all data preprocessing to *generic* RF methodologies (i.e., applying low or band pass filtering) and data augmentation techniques that can be applied to *any* RF signal regardless of protocol (i.e., decimation).

CDCN. We first develop a complex-valued DCN network (Figure 2a) similar to the shallow MusicNet proposed by Trabelsi et al. [16], which was successfully applied to acoustic data. While acoustic and RF signals are different in nature (one is mechanical and the other electromagnetic) they share many similarities. Both can be modeled as a 1D complex vector of samples, and many of the techniques, methodologies and signal processing theories are applicable to both in practice (low/bandpass filtering, decimation, etc). We use this as a comparison baseline for our novel LSTM-based approach.

RDCN. Since RF signals can be expressed as a time series and have temporal coherency, we hypothesize that networks that capture temporal features will perform well at analyzing RF data. Consider, for instance, the case of amplitude modulation, where voltage changes from a higher value to a lower value pass through intermediate



(a) Convolutional Deep Complex Network (CDCN) architecture.



(b) Recurrent Deep Complex Network (RDCN) architecture.

Figure 2: Convolutional Deep Complex Network (CDCN): the Complex Convolutional Layer combines both I/Q pairs in two mixed channels **(top)**. The Recurrent Deep Complex Network (RDCN) is based on the powerful LSTM variant; while initially the I/Q pairs are in two different channels the Complex Linear layer binds them together, and its output contains two mixed channels **(bottom)**. The schematic was produced with PlotNeuralNet [47].

values (albeit ephemerally). The precise nature of this transition is based on the device's electronics, which we aim to capture. Our second complex network elevates LSTM layers, and employs a custom build layer we call the "Sequencer", as shown in Figure 2b. BPTT is a known problem for RNNs; as RF signals can expand to extremely large time series ranging to several thousands of samples, operating on a point-to-point basis is infeasible. While LSTM's memory mechanisms help alleviate the problem, by providing alternative ways for the gradient to flow back through time, they still break down under sufficiently large signals [41]. Drawing from the conclusions of Gers et al. [41], we design the Sequencer layer to tackle this problem. This layer splits the signal, which is modeled as a 1D time sequence of I/Q pairs, into a number of smaller vectors. The generated vectors are sequentially served as input to the LSTM in the proper temporal order, as shown in Figure 4, allowing the LSTM to "see" the same amount of data but over less timesteps. This transmutation of a longer vector into a collection of smaller vectors changes the nature of the signal by transforming it into different dimensions. As such, the size and number of the newly generated vectors are critical as they might not preserve the qualities of the original signal.

3.1. Real, Imaginary and Mixed Data Channels

Due to limitations of the underlying frameworks, the fundamental data-holding structure (i.e., *tensors*) cannot contain complex numbers. To overcome this, our tensors contain the real ("I") component in one dimension and the imaginary ("Q") component in another, akin to [16]. Initially, our input channels are clearly separated between real and imaginary components containing real-valued float numbers. This separation ceases once the data is pushed through the first layer. The linear, convolutional layers operate on both channels utilizing the equations described previously, and the output channels exiting these layers contain two channels with data containing information from both the real and imaginary parts.

These output channels no longer solely represent the real and imaginary parts of the number, but are instead mixed information-wise. To differentiate from the prior "pure" input channels we refer to the new channels as "A" and "B". To illustrate the qualitative difference between these sets, consider the output of the complex linear layer:

$$L_W^{\mathbb{C}}(\mathbf{h}) = \begin{bmatrix} \Re(\mathbf{W}\cdot\mathbf{h}) \\ \Im(\mathbf{W}\cdot\mathbf{h})) \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{-B} \\ \mathbf{B} & \mathbf{A} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} \tag{1}$$

The outcome of this equation contains values that are qualitatively mixed and stores them as:

- channel $\boldsymbol{A}= R*R-I*I$
- channel $\boldsymbol{B}= R*I+I*R$

This operation's outcome is still real-valued in one channel and imaginary in the other. However, information-wise, the channels contain mixed data since the imaginary values are used in tandem with the real valued parts.

Output channels. The output of the network is contained in two mixed channels $\boldsymbol{A}$ and $\boldsymbol{B}$. Trabelsi et al. [16] utilize the output of channel $\boldsymbol{A}$, and discard the output of channel $\boldsymbol{B}$ in the shallow version of their proposed network, but utilize both channels through a linear dense output layer in the deep version of their model. Since we are working with novel RF data, we explore three possible configurations of the outputs so as to identify the optimal way for leveraging the mixed output channels: (i) only the output of channel A (per prior work), (ii) we introduce parameters G_a and G_b and sum the channels, allowing the network to modify the parameters, and (iii) we employ a convolutional layer for joining the two channels into one output channel of identical dimensions. It is important to note that while Wolter et al. [48] followed a similar scheme for the gates that control the memory of a Complex GRU, that should not be confused with our parameters which act as a weighting factor for the two output channels. Our results show that both channels are actually used in practice, and the network keeps both parameters in close proximity.

3.2. Layers

Complex layers. We use the complex batch normalization, complex convolutional, and complex linear layers proposed by Trabelsi et al.[16]. The input of the initial convolutional and linear layers is split in two channels that contain the (purely) imaginary and real part in each channel. However, as aforementioned, the output of these layers contains mixed information from both the imaginary and real parts of the signal.

Activation units. To identify the optimal activation unit we evaluate both ZReLU and CReLu. We empirically find that ZReLU, which activates on the first quadrant of the Cartesian plane, is not optimal for the mixed nature of the data and CReLU provides better results.

LSTM. Our LSTM layers are unmodified real-valued LSTMs, which receive *mixed* input. We utilize two LSTMs in parallel, which receive the $\boldsymbol{A}$, $\boldsymbol{B}$ channels as input. The output used for classification is the context from each LSTM during the *final* step. The internal context is preserved for the entire epoch and reinitialized to a clean context at the start of each epoch. The computational graph is preserved but detached within each minibatch, yet the context *values* are still preserved across minibatches for avoiding backpropagation. During our experiments we found that the RDCN architecture was significantly harder to train, corroborating the findings of [49].

Preprocessing step. Our data preprocessing is minimal and protocol agnostic, comprised of: (i) basebanding the signals [21], and (ii) applying a 3^{rd} order Butterworth bandpass filter [50] with cutoffs at 25KHz and 20MHz. Our insight behind the lower order Butterworth filter is to allow a smooth cutoff and include potential discriminating effects that emanate from the imperfections inherent to the transmitter's filters. The precise lower value was calibrated experimentally. No other data preprocessing or other protocol-specific treatment takes place.

3.3. Data Augmentation

Existing methodologies for processing image or audio data contain a multitude of data manipulation techniques for creating new synthetic instances by increasing robustness and reducing generalization errors. For instance, a well established approach for creating synthetic image training data is to rotate the image, scale it or add/remove noise. Prior work has shown that such practices improve the network [51], [52], [53]. We adapt common data augmentation practices and evaluate their applicability and limitations in the realm of RF data.

Decimation. A widely used methodology for reducing the computational load in signal processing, is to keep every M_{th} sample of a signal, where M is called the decimation factor. To avoid aliasing, a low pass filter is applied before the decimation operation. For neural networks, decimation reduces the computational load and the number of network parameters, since the input signal is reduced according to M. In our work decimation has the additional role of augmenting the dataset, since the decimated signals are used as extra samples.

The decimation factor must be an integer value and obey the lower limit of the Nyquist theorem [54] to be able to correctly reconstruct the signal. Decimation can be seen as reducing the rate of sampling by a ratio equal to the decimation factor M, thus, the maximum ratio must be the number that allows a sampling rate with at least $F_s \geq 2F_{max}$, where F_{max} is the maximum frequency required. For example, in the case of WiFi where the bandwidth is 20MHz the minimum required sample rate is 40MSPS (Mega Samples Per Second). Since our sample rate is 100MSPS per channel and the factor M must be an integer, the only allowed value is two. Increasing the decimation factor beyond that limit reduces the highest frequency one may reconstruct, leading to potentially missed information. To establish whether this constrain also applies to the problem at hand (which is to fingerprint the transmitter rather than faithfully reconstruct the signal) we add this parameter to our hyperparameter exploration. Figure 4 (left) shows an illustration of decimating a signal composed of 8 data samples with a factor of two. We also adapted other common data augmentation techniques but did not find them to be useful (see Appendix B).

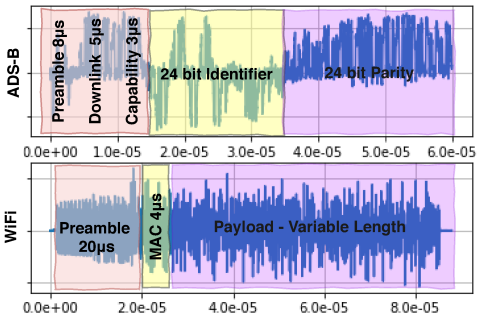


Figure 3: Signal breakdown and unique identifier position for the "I" component over time.

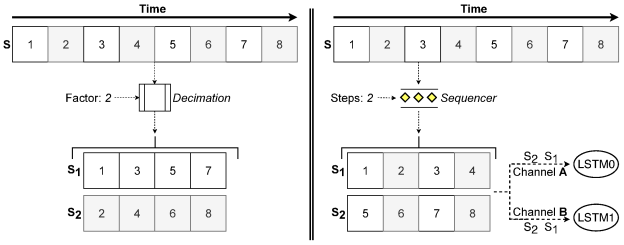


Figure 4: Decimation with a factor of 2 generates two new samples half the original length **(left)**. The Sequencer converts the signal into a series of vectors (here a vector of 2 vectors) sequentially entered in the LSTM in proper temporal order, i.e, samples are seen in the *same* order as if the original signal was provided unprocessed **(right)**.

Randomized Cropping. Our main design goal is for RF-DCN to be protocol-agnostic and to not learn software-level characteristics. To guide the network away from latching onto protocol identifiers, we crop each sample in every minibatch into N parts and randomly choose, without replacement, one part to train on. This approach forces the network towards selecting features that are common in different parts of the signal, since it has to minimize the error for every part it encounters during training. The same procedure is followed for testing as well, where each test signal is fragmented into N parts and a random piece is chosen for classification.

Both the WiFi and ADS-B signals used in our evaluation contain identifiers that exist in predetermined positions and have different sizes (see Figure 3 in Appendix A). In our current experiments we find that $N = 6$ provides a good balance between having a sufficient part of the signal and making sure that the majority of the generated crops do not contain any unique identifier. Our technique is protocol-agnostic as we do not make assumptions on the precise location of the unique identifier in the signal, and choose a random part when training or testing. In other words, during our experiments 83.3% of each testing and training signal is randomly *discarded*, which prevents the system from latching on to identifiers. We do not reveal information to the system about the position of the identifier within a specific type of packet (as that would implicitly teach the system information about each protocol). By following this scheme, we force the network to identify the device without learning superficial features that correspond to identifiers, and it does so regardless of where an identifier is located in the signal. Indeed, our experimental evaluation verifies our hypothesis that RF-DCN captures the essential characteristics of each device; if the networks had learned to predominantly discern a signal through its identifier, the accuracy results reported with random signal cropping would arguably be significantly lower than those actually achieved by our system.

4. Dataset Description

We provide details on the datasets used for our experiments, which are comprised of signal traces captured under a variety of conditions over a period of 13 days. This was part of a DARPA red team evaluation (they were in charge of creating the dataset – performers were not part of IRB-related processes). Data is stored in the SigMF format, which we describe in Appendix A.2.

The RF signals were captured from 96 different Tektronix RSA5106B each equipped with a HG2458-08LP antenna, and four USRP B210 SDRs. Signals were captured at 100MSPS[1] which is well beyond the minimum sample rate required to reconstruct the most demanding signal category in our dataset, i.e., WiFi. The captures were conducted with both vertical (90%) and horizontal (10%) antenna polarization, and 5% of the signals are captured in both forms. The captured signals have variable lengths according to the protocol and the message type. ADS-B signals have a length of 6,400 (64μs short) or 12,000 (120μs extended) whereas for WiFi length is extremely variable (as per specification), ranging from a few thousand I/Q pairs to 272K I/Q pairs in our data. As our goal is to create a system without any knowledge of the underlying protocols, we perform our experiments with vectors of 6,400 samples (I/Q pairs) for both WiFi and ADS-B. We do not differentiate between protocols by adding more data points for WiFi signals, as that would increase the informational content the system sees for one type and could guide it to separate signals due to superficial characteristics (e.g., different signal lengths). At our sample rate 6,400 complex data points correspond to 64μs of signal capture. Our datasets include only control packets (e.g., WiFi control frames but not data frames).

WiFi-1. This dataset spans all 13 days and data was captured in both indoor and outdoor settings. It includes 4TB of signals that have been collected from 53,853 unique devices from a variety of manufactures. The majority of devices are Apple iPhones ($\sim$30%) and the rest are from different manufacturers and types (smartphones, laptops, tablets, drones). The signals captured are mainly from 2.4Ghz but a significant portion ($\geq$ 20%) is also from 5Ghz emitting devices; some devices transmit in both bands. Device labeling was done using the MAC addresses, as MAC address randomization was disabled to facilitate the labelling process of such a massive and diverse set of devices. The real-world capturing process has resulted in additional interference and "noise" in the captured signals from other protocols that transmit at those ranges (e.g., Bluetooth), which poses an additional challenge for protocol-agnostic fingerprinting. The open air capturing was designed to eliminate or minimize environmental sources of bias that could assist the fingerprinting systems. Specifically, while the deep learning models could potentially learn environmental factors or location-specific artifacts, this was counteracted by: (i) the weather and other environmental factors (e.g., physical obstacles that alter signals) changing over the course of 13 days, (ii) using mobile devices that are not stationary (a common limitation of prior studies) during normal operation where they change multiple locations per day while transmitting their signals, and (iii) the antennas being deployed in different locations which also changed through time. As such, this dataset allows for the evaluation of our proposed models across different capturing conditions and for large-scale deployments where many devices need to be tracked.

WiFi-2. This dataset is comprised of 19 identical devices with the same firmware version installed. The signals were collected in a lab (indoors) and the devices were set up to transmit the same data and configured with the same MAC address. The dataset includes $\sim$200K signals and requires 17GB of storage. Labeling was done manually; traffic was captured from one device and labelled, then the next device was used and labelled, etc. It is important to note that while all devices have the same MAC address, and devices transmitted the same data, ephemeral identifiers in the traffic will differ (e.g., timestamps). The main purpose of this dataset is to demonstrate that RF-DCN does not learn specific identifiers and is unaffected by other software or content-related artifacts, but is instead capable of identifying the artifacts left by a device's unique hardware imperfections in a transmitted signal.

ADSB-1. This was collected over 10 days and contains roughly the same amount of short and extended signals. The transmitters were airplanes and the collected signals are from 10,404 unique sources. The captured data contains a plethora of different airplane positions, velocities, altitude and SNR values. Labeling was done using the unique 24bit identifier encoded in each message. The dataset includes 3.5M signals and is approximately 7TB. The outdoors collection took place at a single location.

Realistic environmental conditions. Numerous factors affect the quality of a recorded signal; channel effects (e.g., multipath, hysterisis), environmental effects (e.g., outdoors vs indoors), the type of transmitter (e.g.,

1. The overall sampling rate is 200MSPS split between the "I" and "Q" components, leading to 100 MSPS per channel.

cellphones, UAV), capture conditions (e.g., antenna polarization). All these will vary in a realistic deployment setting. Our dataset was generated so as to contain signals representing *all* of the above conditions. We believe that this plethora of different capture parameters is a critical requirement for evaluating a system's ability to fingerprint devices *in practice*, as ideal indoor-lab settings alone are not indicative of actual deployment settings. Furthermore, the signal captures contain multiple concurrent transmissions from different devices, thus, introducing additional noise. Another important feature is that our training and testing data are not from a contiguous time window (apart from WiFi-2), further increasing the realism of our testing scenarios. Prior studies have evaluated their proposed systems using datasets that reflected some but not all of the aforementioned parameters. To our knowledge, we are the first to present a study that experimentally evaluates a system using a dataset that contains signals captured in such diverse conditions *and* quantifies their effect.

5. Experimental Evaluation

We experimentally evaluate RF-DCN's performance under different environmental factors and across multiple dimensions of our model's design and implementation. The combinations of factors and the specific details (e.g., number of training and testing samples) and outcome metrics in each experiment were dictated by the DARPA red team evaluation. All the experiments presented, unless otherwise noted, were performed on captured signals that correspond to 6400 I/Q pairs (with our sampling rate that is precisely 64μs per sample), represented by 1D vectors of 6,400 complex numbers. In all experiments the testing and training data are from different transmissions.

Evaluation metric. We evaluate all models using *precision*, denoting the percentage of test signals that were identified correctly out of all the test signals. In practice, due to the realistic conditions of the dataset creation, where "interference" from concurrent transmissions is common, our reported precision presents a lower bound of the actual precision of our models. For instance, consider an example where a test signal capture for device A also contains background transmissions from a device B which is also a device used in the experiment. If our system labels the test signal as "Device B" it is counted as a misclassification, despite Device B being one of the model classes and also being present in the test signal.

Hardware specifications. Experiments were performed on a Supermicro SuperServer 1029GQ-TVRT equipped with 2 Intel Xeon Bronze 3104 (6 cores at 1.7GHz each) with 96GB of DDR4 RAM and two Nvidia Tesla V100 SXM2 GPUs with 16GB of VRAM each. All of the architectures occupied a maximum of 8GB on the GPU; the multiple units enabled the concurrent training of different models and reduced the time required for the hyperparameter exploration.

Models. We present two real-valued conventional networks, namely ANN and CNN, and two complex-valued networks, namely CDCN and RDCN shown in Figure 2. We use the real-valued networks as a baseline for evaluating the complex-valued variants and using all the information present in a signal (properly bound using the covariance matrices, and the usage of complex weights and complex activation units).

Machine learning configuration. The networks were trained over 100 epochs with a batch size of 32. We found that adaptive learning rate annealing is helpful and, thus, reduce the learning rate by $1e^{-1}$ after every 10 epochs where the network fails to improve its validation precision. We also allow *early stopping* once the learning rate reaches $1e^{-7}$. The results we report in this section represent the best generalization precision over 100 epochs (or fewer if the early stopping criterion is met). This falls in line with evaluation approaches in the deep learning literature when handling large and complicated neural networks (e.g., [55], [56], [57], [58]). This is standard practice for obtaining unbiased error estimations when facing significant computational requirements for training such networks and exploring different architectures, configurations, and datasets, as opposed to k-fold validation approaches that are common for other machine learning techniques. We evaluated both Cross Entropy and Binary Cross Entropy and found that the later improves the convergence of the model and produces better results.

We evaluated both the Adam optimizer and SGD in a subset of the experiments and found that Adam with `amsgrad` disabled is superior. Real-valued layers are initialized using uniform distribution based on the work by Glorot et al. [45], while complex-valued layers leverage complex weight initialization. There was no normalization or standardization of the data during preprocessing – we relied on the first Batch Normalization layer that performs a similar computation. The hyperparameters we explored include several variables such as the optimal decimation factor, the optimal sequencer time step and the number of hidden units in the RDCN layer. The hyperparameter space was explored using Spearmint [59], and the optimal values obtained corresponded to the best precision using a validation dataset comprised of both WiFi and ADS-B devices. Our methodology and details on the obtained hyperparameters can be found in Appendix C.

Hyperparameter exploration takeaways. While our experiments show that the hyperparameters greatly affect the learning capabilities and precision of each architecture, our exploration reveals that the RDCN network is the most "sensitive" to hyperparameters. Its performance can range from not learning anything (i.e., getting results equivalent to randomly guessing the source of the signal) to obtaining highly accurate results. For RDCN we find that the most crucial parameter, besides the learning rate, is the number of time steps produced by the sequencer; this indicates that restructuring the 1D time series vector into a tensor of vectors directly influences the representation of the signal, effectively changing an N-length one dimensional signal to an N/M vector length of M-dimensional vectors. The best results were obtained for a length that corresponded to approximately 1μs worth of data, given our sampling rate that corresponds to 100 values per timestep (or 50 when a factor 2 decimation is used).

5.1. Signal Preprocessing

We explore how signal preprocessing affects precision. We compare the precision obtained when the different neural architectures are given completely unfiltered data

TABLE 1: Precision in regards to temporal relation of the collected signals, using the WiFi-1 dataset.

T-train	T-test	Environment	Classes	Samples (Train)	Samples (Test)	ANN	CNN	CDCN	RDCN	RDCN (top5)
Day-1	Day-1	Indoors	100	800	200	39%	66%	74%	72%	90%
Day-1	Day-1	Outdoors	100	800	200	38%	52%	66%	63%	89%
Day-1	Day-1	Both	100	800	200	32%	61%	71%	70%	91%
Day-1	Day-2	Both	50	100	25	2%	3%	7%	10%	30%
Day-1,2	Day-3	Both	65	218	54	4%	10%	12%	21%	44%

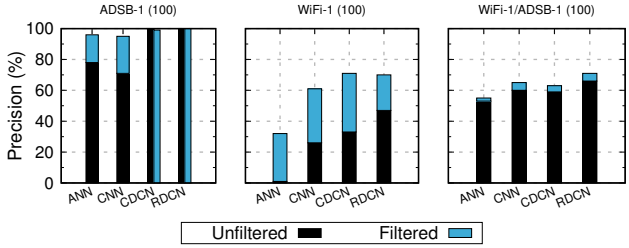


Figure 5: Comparison of our system's precision on open capture signals across different modulation schemes without any form of basebanding or filtering, versus when generic filtering techniques are applied.

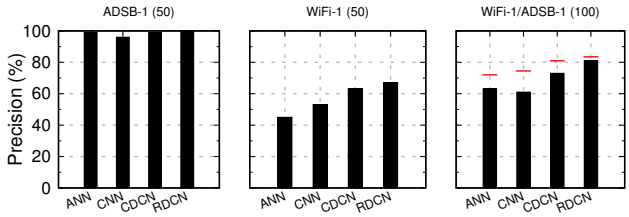


Figure 6: Precision of two stand-alone single-protocol networks versus a protocol-agnostic network for multiple signal types. The red lines denote the "ideal" precision, which is the average of the two stand-alone networks.

as opposed to data that has undergone *generic* telecommunications preprocessing, namely baseband and passband filtering. Figure 5 shows that filtering and basebanding are crucial for the WiFi signals. Apart from being more impacted by outdoor conditions and channel effects, the inherent ability of WiFi transmitters to broadcast in different communication channels mandates the use of basebanding. Interestingly, we find that the complex-valued networks are able to "see through" the clutter induced by concurrent WiFi transmissions and ambient noise. Indeed, they locate the intended signal even in the raw pre-processed signal without basebanding, where the training and testing signals may even be in *different* central frequencies. Nonetheless, it is evident that applying basic (i.e., not specific to a given type or protocol) signal processing techniques can greatly improve precision, especially for the weaker real-valued networks.

5.2. Multi-protocol networks

Apart from the positive effect of filtering, Figure 5 demonstrates the feasibility of building a network for signals from multiple protocols. When the network is trained on a mixture of two signal types, performance is lower than the average precision of the two stand-alone networks (which could be seen as the "optimal" performance). To further explore this effect we consider the following experiment. We assume a scenario where 50 WiFi and 50 ADS-B sources exist, and want to quantify the performance impact of a protocol-agnostic approach where no explicit information is provided to the network for differentiating between them. Specifically, we quantify the precision of two stand-alone versions of RF-DCN, each trained and tested on 50 classes of a single protocol, and compare it to a protocol-agnostic version that has to handle all 100 classes. As can be seen in Figure 6, the multi-protocol network is "weighed down" by the WiFi protocol and is not able to reach the average precision of the two standalone networks. While the increase in the number of classes from 50 to 100 can affect the performance, the presence of multiple types of protocols has a dominant impact (as in Figure 5). We find that the RDCN architecture not only achieves the highest precision, but also presents the smallest precision loss for the multi-protocol system, further highlighting its effectiveness and suitability for challenging settings where multiple types of signals are emitted from sources.

5.3. Channel effect: spatial implications

Due to the transient nature of the channel through which a signal is being transmitted, we explore how RF-DCN's precision changes based on the spatial conditions of the different signals. We break down our experiments on the WiFi-1 dataset in the first two rows of Table 1. We start by using indoor data that was captured on the same day, for 100 different devices. Our RDCN network is able to correctly identify the device in 72% of the cases, while the correct device is in the top five for 92% of the tests. When we replicate this experiment using signals that were captured outdoors, precision drops by 4-9%, due to the impact environmental factors have on the transmitted signals. When training and testing using a mixture of indoor/outdoor signals RF-DCN's performance is roughly in the middle between the indoor and outdoor settings. Once again, RDCN vastly outperforms real-valued networks, highlighting our system's robustness against environmental factors. This demonstrates that our system does not learn environmental or location-specific artifacts – if our model relied on such features we would have observed a much larger precision loss in this experiment.

5.4. Channel effect: temporal implications

Prevalent environmental differences greatly affect the morphology of each signal, as shown in Figure 1. Apart from the effect of natural environmental changes (e.g., weather), ephemeral factors such as vehicles or obstacles may induce multipath or other effects. Moreover, some protocols like WiFi allow for transmission in different communication sub-channels according to conditions in

TABLE 2: Precision when using different subsets of ADSB-1, with varying levels of SNR ratios.

Train	Test	ANN	CNN	CDCN	RDCN	RDCN (top5)
Low	Med	1%	98%	100%	100%	100%
Low	High	75%	99%	99%	100%	100%
Med	Low	85%	85%	98%	99%	100%
Med	High	99%	100%	99%	100%	100%
High	Low	27%	26%	99%	99%	100%
High	Med	78%	78%	99%	100%	100%

the RF spectrum, e.g., if more than one device broadcasts in the same channel. As such, we also experiment with a more challenging and realistic scenario where the training data is collected during one day and the testing data is from a different day, using a smaller number of classes. The last two rows in Table 1 contain the experiments used to estimate the effect of these conditions. When we train on one day and test on a different one, all the aforementioned challenges are reflected in our results. Even the RDCN obtains only a 10% precision, with the correct answer being in the top five 30% of the time. The real-valued networks learn nothing, with precision equivalent to random selection. When we train on two days, the network appears to learn and "ignore" the channel effects and create more robust signal features; testing on a different day now achieves an increase in precision with the RDCN rising to 21%. While the other architectures also improve, they remain considerably less accurate.

It is important to note that while we refer to Days 1,2,3 for ease of presentation, the data is not from three specific days but spans the entire 13 days. For example, the data for one device might be from the first, sixth and tenth day, while for a different device it might be from the second, third, and seventh day of the collection period. This allows us to truly stress test our system under different environmental settings. In our opinion, this experiment highlights perhaps the toughest challenge in operating with raw I/Q data. Previous work that relied on protocol-specific techniques by extracting attributes (e.g., constellation phase-shift) obfuscated this obstacle since sophisticated protocol-specific signal processing techniques can rectify these effects. The fact that the RDCN network seems to be able to learn the channel effects within such a complicated domain, using signals captured from different days is an encouraging result for the feasibility of protocol-agnostic fingerprinting in realistic settings.

5.5. Channel effect: signal-to-noise implications

If the target devices are outdoors, the varying environmental factors can result in different levels of noise. In this set of experiments we aim to investigate how precision is affected when using datasets with different signal-to-noise ratios. We train our models with 100 classes using an equal number of training and testing samples (525) that were captured over a mix of different days. We break down our data into three different levels of SNR (*Low* $\leq$2dB, 2dB$<$*Med*$<$5dB and *High* $\geq$5dB). To make conditions more challenging, we experiment with testing and training combinations that have different SNR levels, as shown in Table 2. The complex-valued networks both achieve precision higher than 98% and our results show that they are unaffected by different SNRs. The

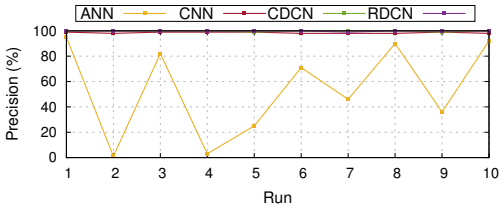


Figure 7: Consistency of neural architectures over ten runs on the same testing and training datasets.

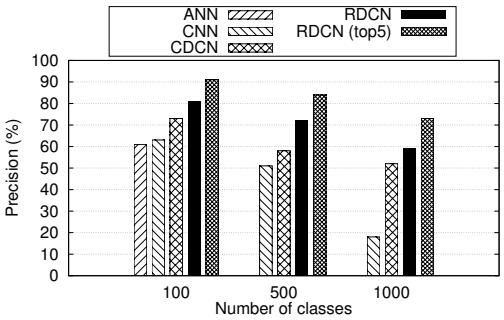


Figure 8: Performance impact of number of classes.

RDCN network achieves almost 100% precision in all settings. On the contrary, the real-valued networks perform relatively better when the testing data is of higher quality than the training, but suffer when attempting to recognize signals of *lower* quality when trained on signals of *higher* quality. The most indicative case is when the training signals have *High* SNR and the testing have *Low* SNR, with ANN and CNN both achieving a precision of less than 27%. We observe an interesting relation between this experiment and the one comparing filtered and raw signals; in both cases the complex-valued networks appear to be able to "see through" the noise and extract robust and meaningful features, whereas the real-valued networks are more susceptible to superficial effects induced by noise.

Model Reliability. In this experiment we encountered a discrepancy in the ANN model, shown in the first row of Table 2. To further investigate this, we performed 10 runs for each architecture, depicted in Figure 7. We observe that the ANN network has a mean value of 54.1% with a standard deviation of 36.7 while the other networks remain consistent. In the runs where the ANN performed poorly it suffered from the first epoch, indicating that this issue stems from weight initialization. The limitations of this simplistic, yet parameter-rich, model are also evident when the number of devices increases, as we discuss next.

5.6. Class size effect

We explore how our system fares when the number of classes increases. As Figure 8 shows, we run three sets of experiments where RF-DCN is required to differentiate between 100, 500, and 1000 sources. These experiments evaluate our system using an equal mix of data from both the WiFi-1 and ADSB-1 datasets; in all setups we use 13.95ms and 3.45ms worth of signal captures for training and testing respectively. One can easily observe that precision diminishes as the number of classes increase (which is expected [60]). Interestingly it does so in a non-linear fashion, with the RDCN achieving a precision of

TABLE 3: Precision when using random signal cropping to prevent models from learning software-level identifiers.

Dataset	Classes	ANN	CNN	CDCN	RDCN	RDCN (t5)
WiFi-2	19	21%	26%	27%	99%	100%
ADSB-1	100	1%	15%	37%	98%	100%
WiFi-1	100	2%	26%	24%	69%	90%
Mix	100	15%	34%	36%	80%	93%

about 82% for 100 devices and dropping to 72% and 62% for 500 and 1000 classes respectively. It is also evident that the real-valued networks fail to produce models with sufficient descriptive power to follow the increase in cardinality, even though they have more parameters than the CDCN. Finally, we observe a complete breakdown of the ANN and a more rapid decline of the real-valued CNN compared to its corresponding complex-valued CDCN.

5.7. Learning hardware imperfections

If the data processing methodology is not designed correctly, then the models tend to latch onto (software-level) identifiers instead of the transmitter's hardware characteristics. To further demonstrate the feasibility of creating fingerprints from the unique hardware imperfections that arise during a device's manufacturing process, our next experiment demonstrates our system's robustness to software-level spoofing. Specifically, we leverage the *WiFi-2* dataset obtained from 19 identical smartphones flashed with the same firmware version, configured with the same link-layer identifier (i.e., MAC address) and transmitting the same data. We present the results in Table 3. To avoid latching on to either ephemeral or unique identifiers we cut each signal in 6 parts and perform classification using one randomly selected segment. Each training sample is a random segment that is less than 17% of the original signal (i.e., we *discard* 83.3%). This fragmentation prevents latching onto ephemeral or unique identifiers wherever they may be located in the signal, without requiring prior knowledge of their position in the signal (i.e., even the cropping process is protocol-agnostic). This leads to two primary benefits: it increases the diversity of the dataset, and guides the network towards finding features that identify the device based on the signal itself. As a result, any network that simply latches onto identifiers will achieve poor precision in this experiment. In our current setup, random segments are only $\sim 10\mu$s worth of transmission.

As Table 3 shows, this has a detrimental effect on all the networks except our RDCN, as its precision remains within 2-4% of when provided with the entire signal. This demonstrates that our cropping methodology prevents the models from using secondary protocol features (e.g., timestamps) as a discerning signal since, if that was the case, all the models would continue to exhibit high precision. We believe that the superiority of our model is due to the existence of sufficient gated memory cells in the RDCN layer and its ability to learn features in *context*. Once the signal is cropped, parts from different transmission phases (i.e., preamble, transient states, payload-data transmission, checksum etc) are used for training and testing. The morphology of the signal in these different phases differs drastically as seen in Fig 3, and

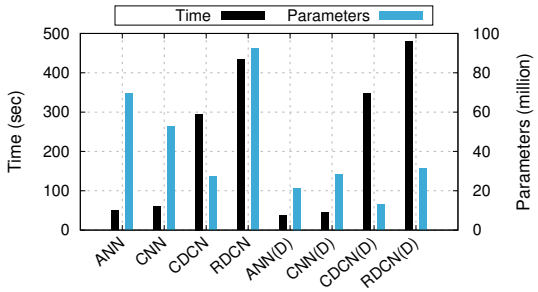


Figure 9: Performance of different architectures. (D) denotes the use of decimation with a factor of 2.

the RDCN network's ability to memorize features and operate *in context* is the crucial difference in comparison to the other architectures. This robustness, derived from capturing the essential transitional imperfections of the original transmitting device, allows our system to *correctly* identify devices despite them sending spoofed identifiers (e.g., MAC address) and bitwise identical data.

5.8. System Performance

Figure 9 depicts the processing time required to train a mixed dataset over a single epoch, and the number of parameters contained in the model. We find that the number of parameters does not always correlate with the time required to process the data; this is evident for the ANN which has almost double the parameters of the CDCN yet requires less than a quarter of the processing time. The computational load also exhibits an inverse correlation to the networks' precision. We include measurements from our decimation experiment, where signals were decimated by a factor of two. Decimation vastly reduces the number of parameters and, even though it leads to a signal half the size, the number of parameters is less than one third of the original. RDCN is the "heaviest" model containing 30M parameters, down from $\sim$92M. Using all the decimated samples instead of just one elevates decimation from a computation-reducing technique to a data-augmentation strategy as well. While this improves precision, it increases the processing time required by the complex-valued networks, indicating the computational mismatch between the real and complex-valued counterparts.

Figure 10 plots RF-DCN's testing duration for each phase, using RDCN and conducting 100 runs on an idle server. Our dataset includes 11,200 signals from 100 classes, amounting to a 1.7G testing dataset. Here the model has already been trained as we are interested in the time required for the different testing phases. It is evident that computation is dominated by the time required to pre-process and then test the data, requiring 23-33 and 29-33 seconds respectively. This experiment reflects a scenario where the attacker monitors an area with 100 signal sources and aims to identify them. In a scenario where RF-DCN is interacting with a single device, e.g., for authenticating a device connecting to an access point, verifying the device would require approximately 0.5 seconds which is a reasonable requirement for bootstrapping authenticated communication between the two endpoints.

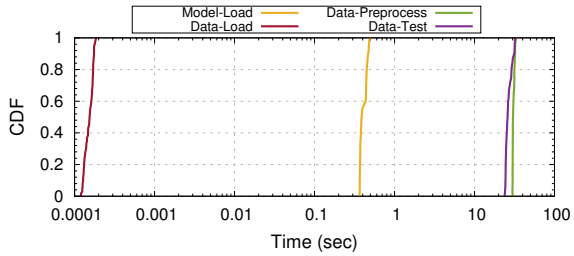


Figure 10: Time required for each test subphase (RDCN).

6. Related Work

Fingerprinting. Numerous prior studies have explored device fingerprinting, with browser fingerprinting attracting much attention due to their ubiquitous presence [1], [7], [6], [4], [61], [3], [62]. Fingerprinting, however, is not exclusive to browsers. Kohno et al. [11] introduced the notion of remotely fingerprinting a physical device over the network, by inspecting packet headers and identifying the clock skew due to the hardware characteristics of a given device. Subsequent studies have explored alternative approaches for detecting clock skew or other protocol-specific features [63], [64], [65], [66], [67]. Brik et al. [12] demonstrated that hardware imperfections can be leveraged as successful fingerprinting features; however their approach relied on protocol-specific information since during the last steps of demodulation one needs to know the utilized modulation scheme to estimate the difference from the ideal expected symbol constellation. Such approaches, while accurate, rely on protocol-specific imperfections that need to be *manually* identified and extracted using handcrafted techniques – this prohibits unknown protocols. Importantly, their data was collected from an indoor testbed with statically placed transmitters – in practice a fingerprinting system would face significantly harsher conditions that affect performance, as we demonstrate. Also, their data was collected using one receiver which can prove problematic in realistic settings. Such limitations are common in this line of research (e.g. [68]).

In contrast to prior work we propose and develop a widely applicable fingerprinting approach, which presents a paradigm shift as it removes many assumptions and practical limitations that affected prior techniques. First, our system is fully passive and does not require any form of interaction with the user or the target device. Second, it is completely protocol-agnostic and does not require any information about the underlying protocols, operating system or applications running. As a result, our proposed system is not affected by differences in software implementations across platforms or versions, changes to the device's software (e.g., updates), or the use of custom protocols. Moreover, our extensive experimentation explores the effect of multiple dimensions of the data collection and model creation process. Finally, our system design guides the neural network towards learning the effect of the device's hardware on the transmitted signal, thus overcoming obstacles of software-level defenses.

Complex-valued networks. Trabelsi et al. [16] laid the theoretical foundations for using complex-valued networks and proposed appropriate building blocks such as complex batch normalization and complex weight initialization to facilitate training. Wolter et al. [48] subsequently introduced and successfully trained a complex-enabled GRU that exhibits stability and competitiveness.

Deep learning in RF. Recent papers [69], [70], [71] highlighted the potential benefits of using deep learning for signal processing as well as tackling fundamental problems like channel estimation [69], [71], modulation classification [70] even directly recovering symbols without demodulation [69]; all these networks rely on real valued networks and treat the naturally complex data as two real valued channels. The authors of [70], [71] stated the lack of complex-valued networks as an open research question and highlighted the mismatch between the common use of complex numbers in almost all signal processing algorithms and the real-valued nature of the networks. O'Shea and Hoydis [70] identified some crucial problems such as the non-holomorphic nature of common activation functions, which was addressed in [16].

In an independent concurrent study, Restuccia et al. [20] have proposed the use of deep learning for fingerprinting devices based on RF signals. However, their study presents multiple significant limitations compared to our work. First, their system relies on protocol-specific processing of the signal, which introduces an important deployment constraint compared to our protocol-agnostic design. Next, their system works with real-valued data, as opposed to our complex-valued data approach, which inherently results in a loss of information during the down-converting process. Moreover, their system is built on top of a CNN which, as demonstrated by our experimental evaluation, is considerably less accurate – our RDCN architecture outperformed the CNN across all experiments. Perhaps the most critical limitation is that they provide the entire signal for training and testing; as we highlighted in our experiments, that guides the CNN towards learning the device identifiers as opposed to actual hardware imperfections. Also, their largest experiment includes up to 500 classes but, as shown in our Figure 8, the CNN's accuracy deteriorates significantly for more devices. Finally, they train their WiFi and ADS-B networks *separately*, whereas we experiment with multi-protocol systems where the networks are fed both types of signals without any prior knowledge of the underlying protocols. Overall, while their work presents similarities, we believe that our paper presents significant advantages for practical deployment, while also demonstrating superior accuracy under more challenging scenarios. We also explore the effect of many practical dimensions of the data collection and network training processes that are omitted in their work.

7. Discussion and Future Work

Deployment. Our techniques can be deployed in numerous scenarios, ranging from passively authenticating devices [72], [73], [74], [75] to detecting unauthorized devices in restricted areas [76] or unmasking spoofed transmissions (e.g., "wormhole" attacks [77], [78]). Another potential application is detecting the illegitimate use of credentials/certificates from unauthorized transmitters.

Agnostic vs generalizable techniques. An interesting related line of work explores the use of mathematical and statistical tools that are broadly applicable across protocols, such as the work by Danev et al. [15], [14]

or Klein et al. [79]. The main difference to our work is that we explore the feasibility of fingerprinting *without knowledge of the modulation or the carrier frequencies*. While our system's precision does benefit from generic preprocessing, our system works in a completely agnostic manner using signals captured within a wide frequency range; it also handles devices that transmit at multiple different frequency ranges without any "guidance" information provided to the system. In contrast, while these prior methods are indeed generalizable and applicable across protocols, they *require* the ability to locate the signal and move it to an IF, while also filtering and clearing the signal sufficiently so as to extract the statistical or mathematical properties used in defining the fingerprint.

Hardware-based bypassing. While our technique is inherently robust to OS and software level attempts of subversion, attackers could try to bypass it using hardware-based approaches and trying to mimic the physical characteristics learned by our network for a specific device. This presents several significant challenges in practice, and would require considerable effort and money [12], if at all feasible. Danev et al. [80] explored two attacks for circumventing physical authentication systems: (i) a feature replication scheme for bypassing modulation-based schemes, and (ii) using a high quality signal generator to faithfully reconstruct a captured signal and deceive low-level physical authentication schemes such as transient-based approaches. The first attack is ineffective against our system since modulation features are irrelevant; the second, more powerful attack could potentially trick our system within lab conditions. In their evaluation they captured and replicated signals in a lab environment with devices placed on a tripod 30cm away from the receiver. In a realistic scenario, capturing a signal in a given location would consist of transmissions that are affected by the given channel and would be re-transmitted in the channel of the new location, which would adversely affect the re-transmitted signal. For a relay-attack to "fool" our system, two key factors would have to be in satisfied. First, the capture and relay hardware must be of such high quality that they minimize (or remove) any artifacts related to their circuitry. Second, the combined channel effects of the two alternate transmission environments (the victim location where the capture takes place and the location of the attack target) would have to be bellow the threshold that would result in our classifier rejecting the test signal. The second requirement is not satisfied under the experimental conditions of [80] where the environment was the same for both the capture and the attack location, and the overall channel conditions were that of a sterilized lab room with the capture done at an extremely short distance. While we consider this as a *potential* threat to our system, it is an open research question and interesting future direction.

Scalability. The limitations on softmax-based classification, also referred to as the "softmax bottleneck", have been studied extensively in the work of Yang et al. [60]. Softmax adaptations like Sigsoftmax [81] might allow for higher cardinalities. Discriminatory schemes such as Siamese networks [82] can facilitate an open set classification strategy, providing robustness in the face of a large number of classes. Our RDCN and CDCN networks can be retrofitted to power a Siamese architecture.

Signal types. Our experimental evaluation uses WiFi and ADS-B signals collected both in the wild and in laboratory settings. As our datasets were collected as part of a DARPA red team evaluation, we were constrained to these two signal families. While creating datasets of other types requires significant effort, we plan to experiment with other prevalent signal types in the future. Nonetheless, since RF-DCN works on raw data and is agnostic of the underlying protocols we believe that our techniques can be directly applied to a wide variety of wireless signals.

Model transferability. Similarly to how the transmitters leave a unique fingerprint on signals, the receiving hardware may also uniquely affect the signal. As such, in scenarios where the training and testing data have been collected by different devices a model's accuracy could be impacted. In the datasets used in our experiments the testing and training data was collected by multiple receivers and antennas. However, as the datasets do not have labels regarding the capturing device/antenna, we were unable to quantify this effect. We consider the in-depth exploration of the transferability of models across different monitoring devices as part of our future work. Given our current results we believe that multi-antenna or multi-device monitoring setups, where signals are collected from several receivers with different polarizations, can lead to RF-DCN learning the imperfections of the transmitting devices while "ignoring" the receiving hardware's effect.

8. Conclusion

We studied the effectiveness of novel complex-valued networks with raw I/Q data for device fingerprinting, without using additional protocol specific information such as modulation schemes or any data layer artifacts such as unique identifiers. We found that these networks outperform their real valued counterparts, producing higher precision with less than one third of the parameters. We introduced a novel RDCN architecture able to fingerprint devices from different parts of the transmitted signal, and compared it with recent complex valued networks from the literature. Our work is the first, to our knowledge, that explores the feasibility of using unbasebanded and unfiltered data; our findings show that this is indeed feasible. In our extensive exploration we identified additional challenges, such as channel effects that adversely affect learning and the dangers of latching onto superficial artifacts prevalent in signals, and evaluated methodologies to ameliorate them. While our approach is by design generalizable across protocols and effective under realistic conditions, we believe that this is an important exploratory first step within a vast and challenging new space.

Acknowledgments

We thank the anonymous reviewers, and Srdjan Capkun for useful feedback that allowed us to improve our work. We are also thankful to Paul Tilghman, Esko Jaska and Joshua Alspector for their constructive criticism. This research was supported by DARPA under RFMLS program contract N66001-18-C-4047 and NSF under contract CNS-1934597. The views and conclusions expressed herein are those of the authors and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

References

[1] P. Eckersley, "How unique is your web browser?" in *Proceedings of the 10th International Conference on Privacy Enhancing Technologies*, ser. PETS'10, 2010.

[2] P. Laperdrix, W. Rudametkin, and B. Baudry, "Beauty and the beast: Diverting modern web browsers to build unique browser fingerprints," in *2016 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2016, pp. 878–894.

[3] G. Acar, M. Juarez, N. Nikiforakis, C. Diaz, S. Gürses, F. Piessens, and B. Preneel, "Fpdetective: Dusting the web for fingerprinters," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, ser. CCS '13, 2013.

[4] O. Starov and N. Nikiforakis, "Xhound: Quantifying the fingerprintability of browser extensions," in *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2017, pp. 941–956.

[5] A. Sjösten, S. Van Acker, and A. Sabelfeld, "Discovering browser extensions via web accessible resources," in *Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy*, ser. CODASPY '17. New York, NY, USA: ACM, 2017, pp. 329–336. [Online]. Available: http://doi.acm.org/10.1145/3029806.3029820

[6] Y. Cao, S. Li, and E. Wijmans, "(cross-)browser fingerprinting via OS and hardware level features," in *24th Annual Network and Distributed System Security Symposium, NDSS 2017, San Diego, California, USA, February 26 - March 1, 2017*, 2017. [Online]. Available: https://www.ndss-symposium.org/ndss2017/ndss-2017-programme/cross-browser-fingerprinting-os-and-hardware-level-features/

[7] K. Mowery and H. Shacham, "Pixel perfect: Fingerprinting canvas in HTML5," in *Proceedings of W2SP 2012*, May 2012.

[8] S. Karami, P. Ilia, K. Solomos, and J. Polakis, "Carnus: Exploring the privacy threats of browser extension fingerprinting," in *26th Annual Network and Distributed System Security Symposium (NDSS)*. The Internet Society, 2020.

[9] S. Dey, N. Roy, W. Xu, R. R. Choudhury, and S. Nelakuditi, "Accelprint: Imperfections of accelerometers make smartphones trackable." in *NDSS*, 2014.

[10] F. Marcantoni, M. Diamantaris, S. Ioannidis, and J. Polakis, "A large-scale study on the risks of the html5 webapi for mobile sensor-based attacks," in *30th International World Wide Web Conference, WWW '19*. ACM, 2019.

[11] T. Kohno, A. Broido, and K. C. Claffy, "Remote physical device fingerprinting," *IEEE Transactions on Dependable and Secure Computing*, vol. 2, no. 2, pp. 93–108, 2005.

[12] V. Brik, S. Banerjee, M. Gruteser, and S. Oh, "Wireless device identification with radiometric signatures," in *Proceedings of the 14th ACM international conference on Mobile computing and networking*. ACM, 2008, pp. 116–127.

[13] X. Zeng, R. Bagrodia, and M. Gerla, "Glomosim: a library for parallel simulation of large-scale wireless networks," in *Proceedings. Twelfth Workshop on Parallel and Distributed Simulation PADS'98 (Cat. No. 98TB100233)*. IEEE, 1998, pp. 154–161.

[14] B. Danev, D. Zanetti, and S. Capkun, "On physical-layer identification of wireless devices," *ACM Computing Surveys (CSUR)*, vol. 45, no. 1, p. 6, 2012.

[15] B. Danev and S. Capkun, "Transient-based identification of wireless sensor nodes," in *Proceedings of the 2009 International Conference on Information Processing in Sensor Networks*. IEEE Computer Society, 2009, pp. 25–36.

[16] C. Trabelsi, O. Bilaniuk, Y. Zhang, D. Serdyuk, S. Subramanian, J. F. Santos, S. Mehri, N. Rostamzadeh, Y. Bengio, and C. J. Pal, "Deep complex networks," *arXiv preprint arXiv:1705.09792*, 2017.

[17] M. Vanhoef, C. Matte, M. Cunche, L. S. Cardoso, and F. Piessens, "Why mac address randomization is not enough: An analysis of wi-fi network discovery mechanisms," in *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*. ACM, 2016, pp. 413–424.

[18] D. Rupprecht, K. Kohls, T. Holz, and C. Pöpper, "Breaking LTE on layer two," in *IEEE Symposium on Security & Privacy (SP)*. IEEE, May 2019.

[19] A. Musa and J. Eriksson, "Tracking unmodified smartphones using wi-fi monitors," in *Proceedings of the 10th ACM conference on embedded network sensor systems*. ACM, 2012, pp. 281–294.

[20] F. Restuccia, S. D'Oro, A. Al-Shawabka, M. Belgiovine, L. Angioloni, S. Ioannidis, K. R. Chowdhury, and T. Melodia, "Deepradioid: Real-time channel-resilient optimization of deep learning-based radio fingerprinting algorithms," *CoRR*, vol. abs/1904.07623, 2019. [Online]. Available: http://arxiv.org/abs/1904.07623

[21] J. G. Proakis and M. Salehi, *Fundamentals of communication systems*. Pearson Education India, 2007.

[22] "National instruments - i/q data," 2018, http://www.ni.com/tutorial/4805/en/.

[23] "Concepts for orthogonal frequency division modulation," 2019, rfmw.em.keysight.com/wireless/helpfiles/89600B/WebHelp/Subsystems/wlan-ofdm/content/ofdm_basicprinciplesoverview.htm.

[24] "Federal aviation administration," 2018, https://www.faa.gov/nextgen/programs/adsb/faq/.

[25] "Airplane tracking using ads-b signals," 2019, https://www.mathworks.com/help/supportpkg/rtlsdrradio/examples/airplane-tracking-using-ads-b-signals.html.

[26] "Federal aviation administration," 2018, https://www.faa.gov/nextgen/equipadsb/resources/faq/.

[27] A. Atanasov and R. Chenane, "Security vulnerabilities in next generation air transportation system," *Master of Science in Computer Systems and Networks, Chalmers University of Technology*, 2013.

[28] R. Faragher, P. F. MacDoran, and M. B. Mathews, "Spoofing mitigation, robust collision avoidance, and opportunistic receiver localisation using a new signal processing scheme for ads-b or ais," in *Proceedings of the 27th International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS+ 2014)*, 2014.

[29] J. Allen, "Short term spectral analysis, synthesis, and modification by discrete fourier transform," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 25, no. 3, pp. 235–238, 1977.

[30] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[31] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner *et al.*, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[32] R. Ranjan, V. M. Patel, and R. Chellappa, "Hyperface: A deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 1, pp. 121–135, 2019.

[33] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[34] Y. Zhang, W. Chan, and N. Jaitly, "Very deep convolutional networks for end-to-end speech recognition," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 4845–4849.

[35] Z. Zhang, S. Zohren, and S. Roberts, "Deeplob: Deep convolutional neural networks for limit order books," *IEEE Transactions on Signal Processing*, vol. 67, no. 11, pp. 3001–3012, 2019.

[36] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo, "Convolutional lstm network: A machine learning approach for precipitation nowcasting," in *Advances in neural information processing systems*, 2015, pp. 802–810.

[37] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, 2013, pp. 6645–6649.

[38] Z. Che, S. Purushotham, K. Cho, D. Sontag, and Y. Liu, "Recurrent neural networks for multivariate time series with missing values," *Scientific reports*, vol. 8, no. 1, p. 6085, 2018.

[39] Z. Zhao, W. Chen, X. Wu, P. C. Chen, and J. Liu, "Lstm network: a deep learning approach for short-term traffic forecast," *IET Intelligent Transport Systems*, vol. 11, no. 2, pp. 68–75, 2017.

[40] P. J. Werbos, "Generalization of backpropagation with application to a recurrent gas market model," *Neural networks*, vol. 1, no. 4, pp. 339–356, 1988.

[41] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with lstm," 1999.

[42] M. Arjovsky, A. Shah, and Y. Bengio, "Unitary evolution recurrent neural networks," in *International Conference on Machine Learning*, 2016, pp. 1120–1128.

[43] N. Guberman, "On complex valued convolutional neural networks," *arXiv preprint arXiv:1602.09046*, 2016.

[44] A. Hirose and S. Yoshida, "Generalization characteristics of complex-valued feedforward neural networks in relation to signal coherence," *IEEE Transactions on Neural Networks and learning systems*, vol. 23, no. 4, pp. 541–551, 2012.

[45] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 249–256.

[46] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[47] "Plotneuralnet: Latex code for making neural networks diagrams," 2018, https://github.com/HarisIqbal88/PlotNeuralNet.

[48] M. Wolter and A. Yao, "Gated complex recurrent neural networks," *arXiv preprint arXiv:1806.08267*, 2018.

[49] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *International conference on machine learning*, 2013, pp. 1310–1318.

[50] S. Butterworth, "On the theory of filter amplifiers," *Wireless Engineer*, vol. 7, no. 6, pp. 536–541, 1930.

[51] J. Ding, B. Chen, H. Liu, and M. Huang, "Convolutional neural network with data augmentation for sar target recognition," *IEEE Geoscience and remote sensing letters*, vol. 13, no. 3, pp. 364–368, 2016.

[52] J.-J. Lv, X.-H. Shao, J.-S. Huang, X.-D. Zhou, and X. Zhou, "Data augmentation for face recognition," *Neurocomputing*, vol. 230, pp. 184–196, 2017.

[53] H. Salehinejad, S. Valaee, T. Dowdell, and J. Barfett, "Image augmentation using radial transform for training deep neural networks," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 3016–3020.

[54] H. Nyquist, "Certain topics in telegraph transmission theory," *Transactions of the American Institute of Electrical Engineers*, vol. 47, no. 2, pp. 617–644, 1928.

[55] L. Prechelt, "Early stopping-but when?" in *Neural Networks: Tricks of the trade*. Springer, 1998, pp. 55–69.

[56] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[57] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[58] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," *arXiv preprint arXiv:1611.03530*, 2016.

[59] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," in *Advances in neural information processing systems*, 2012.

[60] Z. Yang, Z. Dai, R. Salakhutdinov, and W. W. Cohen, "Breaking the softmax bottleneck: A high-rank rnn language model," *arXiv preprint arXiv:1711.03953*, 2017.

[61] T. Hupperich, D. Maiorca, M. Kührer, T. Holz, and G. Giacinto, "On the robustness of mobile device fingerprinting: Can mobile users escape modern web-tracking mechanisms?" in *Proceedings of the 31st Annual Computer Security Applications Conference*. ACM, 2015, pp. 191–200.

[62] S. Englehardt and A. Narayanan, "Online tracking: A 1-million-site measurement and analysis," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2016, pp. 1388–1401.

[63] C. Arackaparambil, S. Bratus, A. Shubina, and D. Kotz, "On the reliability of wireless fingerprinting using clock skews," in *Proceedings of the third ACM conference on Wireless network security*. ACM, 2010, pp. 169–174.

[64] S. Jana and S. K. Kasera, "On fast and accurate detection of unauthorized wireless access points using clock skews," *IEEE transactions on Mobile Computing*, vol. 9, no. 3, pp. 449–462, 2009.

[65] M. Miettinen, S. Marchal, I. Hafeez, N. Asokan, A.-R. Sadeghi, and S. Tarkoma, "Iot sentinel: Automated device-type identification for security enforcement in iot," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2017, pp. 2177–2184.

[66] S. V. Radhakrishnan, A. S. Uluagac, and R. Beyah, "Gtid: A technique for physical deviceanddevice type fingerprinting," *IEEE Transactions on Dependable and Secure Computing*, vol. 12, no. 5, pp. 519–532, 2014.

[67] R. R. R. Barbosa, R. Sadre, and A. Pras, "Flow whitelisting in scada networks," *International journal of critical infrastructure protection*, vol. 6, no. 3-4, pp. 150–158, 2013.

[68] T. D. Vo-Huu, T. D. Vo-Huu, and G. Noubir, "Fingerprinting wi-fi devices using software defined radios," in *Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks*, ser. WiSec '16. New York, NY, USA: ACM, 2016, pp. 3–14. [Online]. Available: http://doi.acm.org/10.1145/2939918.2939936

[69] H. Ye, G. Y. Li, and B.-H. Juang, "Power of deep learning for channel estimation and signal detection in ofdm systems," *IEEE Wireless Communications Letters*, vol. 7, no. 1, pp. 114–117, 2018.

[70] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, no. 4, pp. 563–575, 2017.

[71] S. Rajendran, W. Meert, D. Giustiniano, V. Lenders, and S. Pollin, "Deep learning models for wireless signal classification with distributed low-cost spectrum sensors," *IEEE Transactions on Cognitive Communications and Networking*.

[72] B. Kroon, S. Bergin, I. O. Kennedy, and G. O. Zamora, "Steady state rf fingerprinting for identity verification: one class classifier versus customized ensemble," in *Irish Conference on Artificial Intelligence and Cognitive Science*. Springer, 2009, pp. 198–206.

[73] K. I. Talbot, P. R. Duley, and M. H. Hyatt, "Specific emitter identification and verification," *Technology Review*, vol. 113, 2003.

[74] M. J. Riezenman, "Cellular security: better, but foes still lurk," *IEEE Spectrum*, vol. 37, no. 6, pp. 39–42, 2000.

[75] P. Scanlon, I. O. Kennedy, and Y. Liu, "Feature extraction approaches to rf fingerprinting for device identification in femtocells," *Bell Labs Technical Journal*, vol. 15, no. 3, pp. 141–151, 2010.

[76] I. Bisio, C. Garibotto, F. Lavagetto, A. Sciarrone, and S. Zappatore, "Unauthorized amateur uav detection based on wifi statistical fingerprint analysis," *IEEE Communications Magazine*, vol. 56, no. 4, pp. 106–111, 2018.

[77] Y.-C. Hu, A. Perrig, and D. B. Johnson, "Packet leashes: A defense against wormhole attacks in wireless ad hoc networks," in *Proceedings of INFOCOM*, vol. 2003, 2003.

[78] I. Polakis, S. Volanis, E. Athanasopoulos, and E. P. Markatos, "The man who *was* there: validating check-ins in location-based services," in *Annual Computer Security Applications Conference (ACSAC '13)*, 2013, pp. 19–28.

[79] R. W. Klein, M. A. Temple, and M. J. Mendenhall, "Application of wavelet-based rf fingerprinting to enhance wireless network security," *Journal of Communications and Networks*, vol. 11, no. 6, pp. 544–555, 2009.

[80] B. Danev, H. Luecken, S. Capkun, and K. El Defrawy, "Attacks on physical-layer identification," in *Proceedings of the third ACM conference on Wireless network security*. ACM, 2010, pp. 89–98.

[81] S. Kanai, Y. Fujiwara, Y. Yamanaka, and S. Adachi, "Sigsoftmax: Reanalysis of the softmax bottleneck," in *Advances in Neural Information Processing Systems*, 2018.

[82] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a" siamese" time delay neural network," in *Advances in neural information processing systems*, 1994, pp. 737–744.

[83] P. Werbos, "Beyond regression:" new tools for prediction and analysis in the behavioral sciences," *Ph. D. dissertation, Harvard University*, 1974.

[84] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[85] M. D. Zeiler, "Adadelta: an adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.

[86] "Sigmf specification," 2017, https://github.com/gnuradio/SigMF/blob/master/sigmf-spec.md.

[87] A. N. Tikhonov, A. Goncharsky, V. Stepanov, and A. G. Yagola, *Numerical methods for the solution of ill-posed problems*. Springer Science & Business Media, 2013, vol. 328.

[88] "Commpy: Digital communication with python," 2015, https://github.com/veeresht/CommPy.

Appendix

1. Background

1.1. General training and backpropagation.. In the general case, we use a subset of available data as training samples that the networks attempt to best describe by adjusting their parameters accordingly. This adjustment step is usually done by passing a number of samples trough the network, acquiring the output and combining it with a function that is usually referred to as a loss function or cost module. This loss function outputs a value, usually scalar, that is a metric representing the error or distance of the network's output relative to the ground truth. Driving the loss towards zero is a desirable goal, as it tends to lead to a better performing network; this is done by adjusting the parameters of the network in such a way that if we passed the same sample we used to obtain the loss for this iteration the new output's loss will be smaller. This is generally performed with a method known as *backpropagation* (proven to be able to train deep networks [83]), which essentially uses the chain differentiation rule to discover the contribution of each of the network's parameters to the computed loss. The algorithm starts computing the expressions from the final layers of the network (closest to the output), towards the start, computing the error contributions of each layer's representation based on the previous layer's representation of the input. The general algorithm employed for the actual adjustment step, i.e, the magnitude and direction of change, is given by a gradient descent-based algorithm such as Stochastic Gradient Descent (SGD) or variants like Adam [84] and AdaDelta [85].

1.2. SigMF: storing I/Q data.. Storing and sharing RF data had been a long standing problem. Following the DARPA RF initiative, the SigMF specification was established in 2017 to tackle this issue and allow for a flexible yet well-specified standard for sharing RF data in an efficient manner. The raw I/Q data is specified in an interleaved form in a binary file where the precise location of each signal in the data file is denoted in an accompanying metadata file in a JSON format. The metadata files also contain a host of additional information such as capture details, sampling rate and may optionally be extended to support new custom headers. All the datasets used in this work utilize the SigMF [86] specification.

1.3. Quadrature Modulation. Figure 11 depicts a schematic of an ideal quadrature modulator. The output signal is composed of a linear combination of signals in-phase or "I" component and the quadrature or "Q" component. Besides its apparent simplicity, quadrature modulators can synthesize **any** real signal.

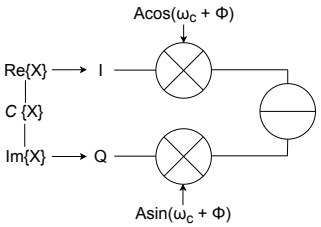


Figure 11: Quadrature Modulation Overview and I/Q data. An inverse procedure is followed on the receiver side in order to obtain the I/Q pair from the modulated signal.

1.4. Inverse Square Root of Matrix V. To guarantee the existence of the inverse square root we leverage the Tikhonov regularization [87] that is performed by: $\epsilon\,\mathbf{I} + \mathbf{V}$. Mean subtraction and multiplication by the inverse square root of the variance guarantees a standard complex distribution with mean $\mu = 0$, covariance $\Gamma = 1$ and relation $C = 0$. As in the real case batch normalization procedure, two learnable parameters are used: the shift parameter β and scale parameter γ. These translate and scale a layer's input along new learned principal components to achieve the desired variance.

2. Data Augmentation using Rotation, Scaling and Noise

While our experiments using decimation revealed that it can have a positive impact, we found that for our particular task these transformations actually have a negative impact on our system's performance. Below we provide more details for these augmentations.

2.1. Rotation and scaling.. Similarly to their 2D counterparts (images), signals can be rotated (phase shift) and scaled (magnitude scaling). This lead to an expansion of the amount of available data by introducing reversible transformations.

We induce rotation by adding a phase shift and scaling by altering the magnitude of each complex data point. To calculate the phase and magnitude we convert each point to polar form, perform the necessary calculations, and recover the I/Q components using Euler's equations.

Formulae 2 describe the procedure for rotation, scaling follows the same principle.

$$
\begin{aligned}
&Original\,Signal = I + Qj \\
&Polar\,Form = \rho e^{\phi}, \rho = \sqrt{I^2 + Q^2}, \phi = \arctan(I, Q) \\
&Rotation(\theta) = \rho e^{\iota\phi} e^{\iota\theta} = \rho e^{\iota(\phi+\theta)} \\
&I' = \rho\cos(\phi+\theta), Q' = \rho\sin(\phi+\theta)j, \\
&Rotated\,Signal = I' + Q'j
\end{aligned}
\tag{2}
$$

2.2. Adding noise - channel effects.. Perhaps the most prominent problem regarding RF data is the effect the channel has to captured signals under different channel conditions. As shown in Figure 1 signals from the same device captured over different days can differ drastically. The fundamental way the signal mutates under different channel conditions introduces a mismatch between the data used for training and signals captured for testing at a later time when the ambient channel conditions deviate significantly between them. We evaluated two methodologies and their efficacy in improving the model's generalization in regards with being able to classify signals from a day not part of the training set: (i) Augmenting the dataset by introducing samples from different days in the training data (ii) Augmenting captured signals synthetically by introducing Rician or Rayleigh fading using a SISO channel model, based on [88].

3. Hyperparameters

The hyperparameters used for each architecture are obtained using Spearmint [59], thus allowing 500 runs for each (complex) architecture. The input data used in Spearmint for optimizing the hyperparameters is a balanced dataset that It is important to note that even though using Spearmint and its underlying Bayesian optimization engine requires less tries than conventional searches such as grid search, we were still unable to exhaustively explore all options as that would require a prohibitive amount of time and used the best values after 500 tests. We included the following hyperparameters in the Spearmint exploration for each architecture:

CDCN: Learning rate, kernel/stride/padding of the convolutional layer, the size of the output dense layer.

RDCN: Learning rate, weight decay, the timestep (μs) used by the sequencer to split the signal,the size of the hidden layer of the LSTM, the number of layers in the LSTM unit and whether it should be bidirectional or not.

For both architectures the Spearmint search configurations also include the factor of decimation factor M and explore whether the following layers should be part of the architecture or not: (i) batch normalization for the input layer,(ii) batch normalization for each layer independently. For the decimation factor our exploration range lies between two and four, given our sampling rate factors above 2 violate Nyquist criterion, and the original signal (for the WiFi category) can no longer be faithfully reconstructed. Since our goal in this work is rather to fingerprint the devices and *not* to reconstruct the original signal we evaluated the impact of using higher decimation factors, even if that would violate Nyquist's theorem. We identified that indeed the best results where obtained with a decimation factor of 2 (which obeys the theorem). As a final note, while the non-conforming ($M > 2$) decimation factors produced worst results they also led to less computations and less parameters which in some applications might be a wanted outcome.

Optimized hyperparameters. Our exploration revealed the following hyperparameters and network configurations: *CDCN:* lr=1e-4, weight decay=9e-5, kernel size=32, stride=1, padding=1, dense layer=4096.

RDCN lr=1e-4, weight decay=9e-5, single layer unidirectional LSTM with 1024 hidden units and sequencer time step set to 1μ leading to 64 vectors of 100 data points for the baseline setting.

For both networks the batch normalization layers were found to be of critical importance and the best performing activation unit was found to be $\mathbb{C}$ReLU. We believe that since our channels are mixed channels containing information that is not entirely real or imaginary, the effect of ZReLU that constricts activations to the first quadrant is over-constraining the network, whereas the outcome of $\mathbb{C}$ReLU that allows activation on positive values for both channels shows better results.

It is important to note that these hyperparameters are optimized for our design goals, i.e., operating on raw I/Q data and with minimal pre-processing being performed. Alternate data representations or extra preprocessing steps would potentially render these hyperparameters invalid and a re-exploration using Spearmint (or a conventional, albeit slower, grid search) to obtain the optimal parameters would be required.