Learning Robust Radio Frequency Fingerprints Using Deep Convolutional Neural Networks

DISSERTATION

José A. Gutiérrez del Arroyo Pérez, Major, USAF

AFIT-ENG-DS-22-S-019

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT-ENG-DS-22-S-019

LEARNING ROBUST RADIO FREQUENCY FINGERPRINTS USING DEEP CONVOLUTIONAL NEURAL NETWORKS

DISSERTATION

Presented to the Faculty

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Doctor of Philosophy

José A. Gutiérrez del Arroyo Pérez, B.S.E.C.E., M.S.Comp.E.

Major, USAF

September 2022

AFIT-ENG-DS-22-S-019

LEARNING ROBUST RADIO FREQUENCY FINGERPRINTS USING DEEP CONVOLUTIONAL NEURAL NETWORKS

DISSERTATION

José A. Gutiérrez del Arroyo Pérez, B.S.E.C.E., M.S.Comp.E.
Major, USAF

Committee Membership:

Brett J. Borghetti, PhD
Chair

Michael A. Temple, PhD
Member

Matthew C. Fickus, PhD
Member

ADEDJI B. BADIRU, PhD

Dean, Graduate School of Engineering and Management

Abstract

Radio Frequency Fingerprinting (RFF) is the attribution of uniquely identifiable signal distortions to emitters via Machine Learning (ML) classifiers. RFF is often proposed as an authentication mechanism for wireless device security, but techniques are limited by fingerprint variability under different operational conditions. First, this work studies the effect of frequency channel for typical RFF techniques, which have previously only been evaluated using bursts from a single frequency channel without considering the effects of multi-channel operation. Performance characterization using the multi-class Matthews Correlation Coefficient (MCC) revealed that operating on frequency channels other than those used to train the models can lead to a deterioration in performance from MCC > 0.9 (excellent) down to MCC < 0.05 (random guess), indicating that single-channel models should not be expected to maintain performance in realistic multi-channel operation. A training data selection technique was proposed to create multi-channel models which outperform single-channel models, improving the cross-channel average MCC from 0.657 to 0.957 and achieving frequency channel-agnostic performance. Second, this work introduced, developed, and demonstrated the Fingerprint Extraction through Distortion Reconstruction (FEDR) process, a neural network-based approach for quantifying signal distortions. Coupled with a simple Dense network, FEDR fingerprints were evaluated against four common RFF techniques for $N_c = \{5, 10, 15, 25, 50, 100\}$ unseen classes. The Dense network with FEDR fingerprints achieved best performance across all values of N_c with MCC ranging from 0.945 ($N_c = 5$) to 0.746 ($N_c = 100$), using nearly 73% fewer training parameters than the next-best Convolutional Neural Network.

To my wife and family, who support even my most ridiculous ideas, like...

...“I should get a PhD.”

Acknowledgements

This work would not have been possible without the rock-solid support from my wife and family, the camaraderie from my PhD classmates, and the thoughtful guidance from members of my research committee and my research advisor. Thank you all for always giving me words of encouragement and for showing me compassion, especially in my times of doubt. I will also be forever grateful to AFIT leadership, faculty, and staff, who navigated the complexities of a national shutdown amidst the COVID-19 global pandemic and whose resilience, innovation and resourcefulness allowed us to continue to meet the AFIT mission.

José A. Gutiérrez del Arroyo Pérez

Table of Contents

Page

Page

List of Figures

List of Tables

List of Acronyms

AWGN	Additive White Gaussian Noise
CB-DNA	Constellation-Based Distinct Native Attribute
CNN	Convolutional Neural Network
CR	Cognitive Radio
CRC	Cyclic Redundancy Check
DARPA	Defense Advanced Research Projects Agency
EVM	Error Vector Magnitude
FEDR	Feature Extraction through Distortion Reconstruction
IFFT	Inverse Fast Fourier Transform
IMF	Intrinsic Mode Function
IMF-DNA	Instrinsic Mode Function-Distinct Native Attribute
IoT	Internet of Things
ISM	Industrial, Scientific and Medical
MCC	Matthews Correlation Coefficient
MDA/ML	Multiple Discriminant Analysis/Maximum Likelihood
MFA	Multi-Factor Authentication
ML	Machine Learning
MSE	Mean Squared Error
O-QPSK	Offset-Quadrature Phase Shift Keying
OFDM	Orthogonal Frequency Division Multiplexing
OSI	Open Systems Interconnection
PCA	Principal Component Analysis
PEVM	Prediction Error Vector Magnitude
QAM	Quadrature Amplitude Modulation

QDA	Quadratic Discriminant Analysis
ReLU	Rectified Linear Unit
ResNet	Residual Network
RF	Radio Frequency
RFF	Radio Frequency Fingerprinting
RFML	Radio Frequency Machine Learning
RFMLS	Radio Frequency Machine Learning Systems
SDR	Software-Defined Radio
SEI	Specific Emitter Identification
TD-DNA	Time-Domain Distinct Native Attribute
WirelessHART	Wireless Highway Addressable Remote Transducer

LEARNING ROBUST RADIO FREQUENCY FINGERPRINTS USING DEEP CONVOLUTIONAL NEURAL NETWORKS

I. Introduction

The United States Air Force Science and Technology Strategy Document for 2030 and beyond establishes the need to "develop and deliver transformational strategic capabilities" to assure United States dominance in any conflict with its near-peer adversaries [1]. These strategic capabilities include assured information sharing and rapid and effective decision making, driven by resilient and flexible communications platforms. Such assurance requires an ever-present worldwide infrastructure of secure wired and wireless links that connect everything from personal computers and portable electronic devices, to critical information systems, intelligence sensor networks, and military weapons systems.

Wireless security is of particular importance because of the nature of the communications medium, which can be trivially monitored by an adversary. Security hinges on the bit-level cryptographic algorithms, which enable encryption and identity authentication through the use of secret information. For instance, in WPA2, a widely used wireless authentication protocol, a shared secret (e.g., a network password) is leveraged in a 4-way handshake process that authenticates both parties and generates a key used for encrypting traffic [2]. However, any entity with knowledge of this generated key can theoretically pose as the already-authenticated party, breaking the trust. Unfortunately, this single point of failure in cryptography is always present.

One strategy to eliminate the single point of failure is to require multiple forms of authentication. Multi-Factor Authentication (MFA) requires that users provide infor-

mation from at least two out of three categories: i) *something you know*, ii) *something you have*, and iii) *something you are* [3]. For instance, human users trying to log into an information system might be required to present a password (i.e., *know*) and an ID (i.e., *have*), or a password and a code sent to a subscribed mobile device (i.e., *have*). Recent technology like smartphone-based fingerprint scanners (e.g., Samsung Ultrasonic Fingerprint ID) and infrared face mappers (e.g., Apple FaceID), coupled with modern advancements in machine learning, have made it possible to incorporate information from the *something you are* category into authentication mechanisms.

A similar strategy can be extended to wireless communications through the use of Radio Frequency Fingerprinting (RFF) [4]. RFF extracts *something you are* information about a wireless transmitter from its emissions. This physical-layer information can augment bit-level authentication mechanisms, enabling identity validation at multiple levels and effectively bolstering security via defense-in-depth [5]. However, RFF poses many practical implementation challenges, including edge device resource constraints [6] and significant drift and variability in the extracted fingerprints [7].

The following sections provide an introductory overview of RFF and its challenges, followed by an overview of the contributions made by this work to address those challenges.

1.1 Fundamentals of Radio Frequency Fingerprinting

RFF is a technique used to identify an emitter from the characteristics of its emitted signals [8]. This technique is applied directly to the signals themselves, in contrast to cryptographic algorithms which rely on the bit-level interpretation of those signals. The underlying theory is that each emitter is built slightly differently due to inconsistencies in the manufacturing process, even those from the same manufacturer with the same internal components. Those inconsistencies manifest in small but

measurable distortions in the emitted signals, leaving behind a "fingerprint" on the signal. It is possible to find commonalities across multiple fingerprints left behind by the same emitter such that future fingerprints can be attributed back to that emitter.

At its core, RFF relies on a Machine Learning (ML) classifier, where the model is trained with a large number of examples (i.e., fingerprints) from a set of N_c classes (i.e., emitters). When a new, never-before-seen example is input to the trained model, the model can predict which of the N_c classes generated it. Most published RFF techniques comprise two steps: i) fingerprint extraction and ii) classification [8]. Fingerprint extraction processes, like Time-Domain Distinct Native Attribute (TD-DNA) [9] or Constellation-Based Distinct Native Attribute (CB-DNA) [10], are algorithms used to build feature sets that serve as inputs to the classifier. A fingerprint is a measurement of the *signal*, and more specifically of the *distortions* present in the signal. The underlying assumption is that the emitter distorts signals in a consistent manner, and that consequently, a classifier can map distortions back to the emitter.

In more modern RFF approaches, the classifier intakes raw signals directly, integrating fingerprint extraction within the model. These often leverage the power of Convolutional Neural Networks (CNNs) to achieve impressive classification performance [11]. The CNN automatically learns which parts of the signal contribute most to discern the N_c classes and structures itself to extract that information.

1.2 Challenges with Radio Frequency Fingerprinting

Unlike human fingerprints, which remain unchanged throughout a person's lifetime, emitter fingerprints are often influenced by operational conditions, like temperature, channel noise, interference from other emitters, component age, and the type of information being transmitted [7]. Each of these operational conditions applies different types and combinations of distortions to the signals, varying the fingerprints.

This variability makes it difficult to employ practical fingerprint-based authentication mechanisms, as a system that works well under one set of conditions may not work well under others. Operational conditions must therefore be considered when designing RFF techniques.

The fundamental challenges are i) understanding which operational conditions contribute to changes in fingerprints, and ii) understanding how those changes manifest in the fingerprints. The first challenge requires an enumeration of operational conditions, a non-trivial feat given the endless possible environments in which an emitter can operate. Research in that vein focuses on demonstrating that a particular condition affects classification performance and then producing a technique to counter those effects [7, 12]. To that end, Study I of this work provides an examination of the effects of frequency channel to RFF. It studies the performance deterioration stemming from the use of multi-channel data on single-channel models and proposes a data-driven technique to mitigate those effects.

Studying *how* operational conditions manifest in fingerprints poses its own hurdles. First, the definition of what constitutes a fingerprint varies across the RFF community. Typical fingerprint extraction techniques, like TD-DNA [9, 13], CB-DNA [10, 14], Gabor Transform [15, 16], and Intrinsic Mode Function (IMF)-based transforms like the Hilbert-Huang Transform [17] all produce very different types of fingerprints. Each technique makes assumptions about which types of distortions are important for distinguishing emitters, measures signals based on those assumptions, and uses those measurements as fingerprints. Thus, fingerprint variability can only be studied within the context of those assumptions, which provides an incomplete picture as to how fingerprints vary in general.

Another hurdle is that recent RFF techniques leverage complex ML models to do "end-to-end" classification, primarily using deep CNNs [18]. These networks do

not require a fingerprinting step, as they couple fingerprint extraction with classification [8]. Models ingest raw signals, pass them through a series of interconnected non-linear layers, and output a predicted class. By design, these networks hone in on features that differentiate the classes best, and although they tend to yield strong performance, they are heavily biased towards the classes used to train them. As a result, performance variability under different operational conditions can only be expressed with respect to the classes used to train the CNN.

Study II contributes a novel neural network-based fingerprint extraction technique which leverages CNNs to measure the relative distortion between two input signals, namely a distorted signal and an undistorted version of the same signal. The technique, called Feature Extraction through Distortion Reconstruction (FEDR), ignores class information and instead focuses on quantifying the distortions themselves. It improves on previous fingerprint extraction techniques because it makes no assumptions about which distortions are present, and it improves upon end-to-end CNNs because it provides a common fingerprint type that can be studied across any number of classes and use cases. Moreover, the neural network used for fingerprint extraction requires no exposure to the RFF classes, meaning that its training can be performed independently of those classes.

Contributions of the two studies are detailed in the following section.

1.3 Research Questions and Contributions

This section provides an overview of each of the two studies, highlighting primary questions and contributions.

1.3.1 (Study I) Considerations for Radio Frequency Fingerprinting across Multiple Frequency Channels

Study I extends previous work on the effects of operational conditions on RFF by examining how frequency channel (i.e., carrier frequency) affects fingerprinting. Modern communications protocols often employ multiple frequency channels to enable simultaneous users and interference avoidance. For example, WiFi (IEEE 802.11 b/g/n) [2] subdivides the 2.4 GHz ISM band into 11 $\times$ 20 MHz overlapping channels, whereas ZigBee [19] and Wireless Highway Addressable Remote Transducer (WirelessHART) [20] (i.e., IEEE 802.15.4-based protocols) use the same frequency band but divide it into 15 $\times$ 5 MHz non-overlapping channels [21], and Bluetooth [22] uses an even more granular division of 80 $\times$ 1 MHz non-overlapping channels.

Application of existing RFF techniques in multi-channel scenarios is limited because prior models were created and evaluated using bursts from a single frequency channel without considering the effects of multi-channel operation. This study presents an evaluation of the multi-channel performance of four single-channel models with increasing complexity, characterized by the multi-class Matthews Correlation Coefficient (MCC). Models include one fingerprint extractor with simple discriminant analysis and three neural networks. A multi-channel training technique is proposed to improve cross-channel performance, and all models are evaluated in the presence of Additive White Gaussian Noise (AWGN).

Specifically, the research questions and contributions of Study I are:

S1-Q1. How does frequency channel affect Radio Frequency (RF) Fingerprints?

S1-Q2. Does frequency agnostic information exist within the fingerprints? And can that information be leveraged in frequency-agnostic RFF models?

S1-C1. A first-of-its-kind evaluation of the sensitivity of single-channel models to multi-channel datasets. The evaluation suggests that failing to account for frequency channel during RFF model training can lead to a deterioration in performance from MCC > 0.9 (excellent) down to MCC < 0.05 (random guess), indicating that single-channel model performance from previous RFF research should not be expected to extend to the multi-channel case (Experiment A).

S1-C2. A training data selection technique to construct multi-channel models that can outperform single-channel models, with average cross-channel MCC improving from 0.657 to 0.957. The findings indicate that frequency-agnostic variability can be learned from a small subset of channels and can be leveraged to improve the generalizability of RFF models across multiple frequency channels (Experiment B).

S1-C3. An assessment of multi-channel models in the presence of Additive White Gaussian Noise (AWGN), which demonstrated that the advantage of multi-channel models in noise performance depended on model type and noise level. Multi-channel neural networks approximately maintained or surpassed single-channel performance, but multi-channel MDA/ML models were consistently outperformed by their single-channel counterparts (Experiment C).

1.3.2 (Study II) FEDR: A Neural Network-Based Technique for Radio Frequency Fingerprint Extraction

Study II presents a novel fingerprint extraction technique that focuses on quantifying signal distortions irrespective of emitter. FEDR is a neural network-based approach for learning a relative distortion latent space. It relies on the key observation that undistorted communications signals can be generated from their distorted counterparts, as long as the bit-level information can be reliably extracted. Through

a constrained network architecture and a custom regularization loss, FEDR can quantify how much distortion is present between the two signals. The technique makes no assumptions as to which distortion or how much distortion is present in the signal, and quantifies distortions irrespective of class, making it ideal for studying how fingerprints across operational environments.

Moreover, since the FEDR network learns about signal distortions, it can be trained independently of RFF classes and deployed as a static fingerprint extractor, which can be coupled with a low-complexity ML classifier at the endpoint to perform RFF. FEDR was applied to synthetic IQ-imbalanced data and to a real-world IEEE 802.11a/g dataset. Isolation of distortion information and removal of content information yielded fingerprints that, when coupled with a basic classifier, outperformed state-of-the-art end-to-end classification techniques.

Research questions and contributions for Study II included:

S2-Q1. Can a network be trained to learn about the differences between a distorted signal and its corresponding undistorted signal?

S2-Q2. Do those differences provide enough information for the discernment of specific emitters?

S2-C1. A deep learning technique for fingerprint extraction called Fingerprint Extraction through Distortion Reconstruction FEDR. Leveraging distorted and undistorted versions of a received signal, FEDR reconstructs the original distortions using a structurally-constrained and regularized relative distortion latent space. Because FEDR learns about distortions and not devices, the network can be trained independently of the RFF classes.

S2-C2. Introduction, development and demonstration of the FEDR technique using synthetic WiFi data with simulated IQ imbalance. Using a basic dense network, simulated distortion parameters were extracted from the learned FEDR fingerprints, implying that distortion information was quantified by the FEDR technique.

S2-C3. An evaluation of RFF performance of FEDR fingerprints extracted from a real-world WiFi dataset of never-before-seen emitters. The FEDR-based classifier achieved best performance with MCC ranging from 0.945 ($N_c = 5$ classes) to 0.746 ($N_c = 100$ classes), using nearly 73% fewer training parameters than the next-best CNN.

1.4 Document Structure

The document centers around the two keynote studies and their contributions. A brief background is provided in Chapter II to provide technical context that spans both studies. This includes basic information about ML, its applications in Radio Frequency Machine Learning (RFML), and an summary of recent RFF research. Study I and Study II are presented as standalone manuscripts in Chapters III and IV, where each is formatted in line with a journal publication. Each of the two chapters presents its own relevant introduction and related work, as well as relevant background information specific to that corresponding study. Research conclusions and future work are summarized in Chapter V.

II. Background and Related Work

The recent growth of ML, and particularly the resurgence of deep learning, has unlocked new and evolving techniques that can be leveraged in the fields of RFF and RFML. The following sections cover some of the fundamentals of ML and introduce some of the most recent developments. This chapter focuses primarily on machine learning research. A technical introduction to the wireless physical layer and communications protocols is provided in Appendix A.

2.1 Statistical Machine Learning

The primary goal of statistical ML is to use data to estimate a functional mapping between features (i.e., input variables) and responses (i.e., output variables) [23]. Typically, the functional mapping is used to predict responses to new inputs. In general, the process involves model design and selection, data-driven training, validation and evaluation, and operational deployment.

2.1.1 Model Types

Model design and selection is driven by the types of features and responses in the data. When the responses are categorical, the model is a solution to a *classification* problem. Here, the end goal is for the model to output the category (or "class") to which the input variables belong. Common types of classifier models include Logistic Regression, k-Nearest Neighbors, Decision Trees and Random Forests, Support Vector Machines, and Linear and Quadratic Discriminant Analysis. Descriptions of these classifiers can be found in [23] and in [24], which also provides Python-based implementations for easy deployment. One other type, common especially in RFF applications, is an extension to Quadratic Discriminant Analysis called Multiple Dis-

criminant Analysis/Maximum Likelihood (MDA/ML) [5], which includes additional dimensionality reduction through Fisher transform.

Inputs to classifiers can either be raw data or some transformation of those data. Transforming data before providing it to a model is one technique to inject domain expertise into a particular solution. A domain expert might know, for instance, that a particular equation that combines several raw input variables is more representative of the phenomenon being modeled and might choose to input the transformed data rather than the raw data. More complex ML models can inherently extract some of those relationships between variables, without the need for domain expert knowledge—this is largely the appeal of neural networks.

RFF is typically framed as a classification problem. Signals are fed into the ML model, either directly of after some pre-processing, and the job of the model is to predict which emitter generated that signal. When pre-processing is used, the extracted features are typically called "fingerprints." FEDR, one of the key contributions of this work, relies on a regression model for fingerprint extraction, but the output fingerprints eventually become inputs to a classifier model for RFF.

In a *regression* problem, the goal is for the model to output a numeric estimate for a particular parameter or for a set of parameters. Linear Regression (i.e., "best fit line") is one of the simplest and most common model types, but is limited to linear relationships in the model parameters [25]. On the other hand, Neural Networks can model both linear and non-linear relationships and have therefore been well-suited for many regression applications [26].

2.1.2 Training, Validation, and Evaluation Datasets

One fundamental truth in all ML is that models are only as good as the data used to train them. Models learn about the relationships within the data—they cannot

learn about things to which they have not been exposed. That being said, models are typically intended to be used on unseen data for making predictions about new inputs. Thus, the data used for learning relationships must be representative enough of the real phenomenon such that the model can make adequate future predictions.

These data are typically separated into a training dataset, a validation dataset, and an evaluation dataset. The training dataset is used by the model to learn about the fundamental relationships between input features and output responses. However, models can also learn relationships *too* well, to the point where the model performs very well on training data but very poorly on everything else—this unintended phenomenon is called *overfitting*.

A validation dataset can be used to help prevent overfitting. Validation data are representative of unseen data, so performance on the validation dataset can serve as an estimate of how well model performance will generalize to new inputs. Overfitting can be detected when performance improves on the training dataset but deteriorates on the validation dataset. Because validation data are used to make decisions about model training and selection, they are technically considered to be an extension to the training dataset. In fact, there are validation techniques, like k-fold validation [23], where the validation set is more explicitly a subset of the training data. Finally, evaluation data are used only once model training and final selection has been made. Typically, these data are used to evaluate and report on model performance on an unseen dataset.

2.2 Neural Networks

Neural networks have dominated recent ML-based research, largely due to their ability to model highly-diverse, non-linear systems to generate accurate predictions in both classification and regression problems [26]. These ML models are typically

conceptualized as directed acyclic graphs, where data flow from input nodes, through middle *hidden* nodes, to output nodes that generate the desired model response. Often, nodes are grouped into *layers*, defined by their distance in the graph from the input nodes. Each individual node produces a response that depends on a linear combination of its inputs and a non-linear activation function—coefficients in this linear combination are the trainable parameters in the neural network. Model training is the alteration of those coefficients such that the model output more closely matches the desired response.

Non-linear activation functions enable learning of non-linear relationships between input and output nodes and fundamentally differentiate neural networks from linear models. Typical non-linear activation functions include the sigmoid/logistic function, the hyperbolic tangent, and the rectified linear unit (ReLU). Output responses for both the sigmod and the hyperbolic tangent functions are constrained to narrow ranges (0 to 1 for sigmoid and -1 to 1 for tangent), resulting in diminished input sensitivity and high susceptibility to saturation [26]. Those limitations hinder neural network training because they make it difficult to attribute changes in the model output to changes in the node coefficients, particularly when the changes in the output are small. The ReLU, given by

$$\text{ReLU: } f(x) = \max(0, x), \tag{1}$$

is not limited in the same way because its output can be any non-negative number. As a result, ReLU activation functions tend to facilitate model training and are often preferred, especially in CNNs [27].

2.2.1 Convolutional Neural Networks (CNNs)

One common type of neural network is a CNN, which contains at least one *convolutional* layer [26]. A convolutional layer is defined by its *kernels*, which are trainable vectors that alter the inputs to the layer through the mathematical convolution operation. Kernels are akin to digital filters in digital signal processing—they can be trained to isolate or remove many types of information. The sizes of the kernels are typically much smaller that the sizes of the layer inputs, and as a result, CNNs tend to have fewer trainable parameters than fully-connected neural networks with the same number of layers. Outputs of convolutional layers are often *pooled*, meaning that a single summary statistic is selected to represent a small neighborhood of outputs. Pooling enables consistent performance in the presence of slight input translations, which is useful in applications where the detection of a feature is more important than knowledge of its location (e.g., useful for image recognition, but perhaps not image segmentation).

CNNs are heavily used for classification in the image processing and signal processing domains, where the relational information between pixels or samples can be exploited for a number of ML-based applications. Recent studies into the interpretability of CNNs in computer vision have shown that layers closer to the model inputs tend to isolate simple image features, while layers farther from the inputs combine the simple features within their receptive fields to extract more complex features [28]. For example, a CNN trained to classify handwritten symbols might comprise convolutional layers that identify short line segments, which in turn feed into other convolutional layers that identify combinations of lines as the letter *A*, or the number *2*, etc. Complex features are often inputs to fully-connected nodes, which map them to output nodes that represent the specific classes. Though CNNs are often used to solve classification problems, they can be effectively trained to perform a

variety of useful non-classification tasks through the correct loss function and training approach.

2.2.2 Training and Loss Functions

Each weight in each node, to include a biasing parameter for each node, is a trainable parameter of the model [26]. The goal in model training is to find the best set of parameters that maximize the predictive power of the model for the particular problem at hand. Unlike with simpler ML techniques, like linear regression or discriminant analysis, the best set of parameters for a given neural network cannot typically be found using a solvable equation. Instead, a greedy iterative algorithm called *gradient descent* is used to alter each of the weights to gradually minimize a loss function. The gradient of each weight is calculated through an algorithm called backpropagation, which can attribute changes in weights to changes in the loss function.

The loss function is designed and selected such that its minimization accomplishes the desired network objective. For instance, in a typical regression problem, where the objective is to predict some continuous numeric quantity, a common loss function is Mean Squared Error (MSE), calculated between the desired quantity and the predicted quantity. In classification problems, the loss function is commonly Categorical Cross-Entropy [29], which is a comparison between the output of the network to a class vector with "one-hot" encoding (i.e., one vector element is set to one and all others are set to zero). Translating the network output to a predicted class is done by finding the maximum of the predicted class vector. Researchers often design domain-specific loss functions to improve neural network performance for their particular task of interest.

2.3 Radio Frequency Machine Learning

The overarching research area that includes RFF is called RFML. Developments in RFML have been recently brought to the forefront largely due to the interest in cognitive radio [8]. The main idea behind cognitive radio is to leverage Software-Defined Radio (SDR) technology to dynamically access the communications spectrum in the presence of other emitters.This application requires automated signal recognition, which has driven significant efforts in ML-based signal identification [30] and modulation recognition [31, 32]. Moreover, the flexibility promised by cognitive radio has led to automated development of full-blown modulation schemes using deep neural networks, customizing the scheme to a particular RF environment without the need to handcraft or engineer a new protocol [33, 34].

Research in RFF has also leveraged advances in ML to achieve accurate emitter identification. RFF (a.k.a. Specific Emitter Identification (SEI)) techniques have been applied to many communications protocols and emitter types, including Internet of Things (IoT) radios and radar sources [35]. The following section highlights some of the most recent and relevant work.

2.3.1 Radio Frequency Fingerprinting

RFF is typically framed as an ML classification problem, where a set of features are input into an ML model trained to classify the emitters. As with many ML applications, the two overarching approaches to building these models are to either hand-craft features *a priori* using some well-defined transform or process, or to allow the network to learn the best features that discriminate the classes. Some of the most pertinent RFF literature also fall into those two camps.

Table 1 lists much of the recent work related to RFF. One of the more common fingerprinting techniques for RFF in communications is Multiple Discriminant Anal-

ysis/Maximum Likelihood (MDA/ML) with TD-DNA fingerprints [9]. It generally involves extracting statistics from a common region of interest in the time-domain signal (e.g., the preamble) and using those statistics as inputs to a classifier. Other handcrafted feature sets generated by calculating statistics over a region of interest include the Gabor-Based DNA [15, 16], which employs a Gabor or Gabor-Wigner Transform, CB-DNA [9, 10], which first pre-processes signals by projecting them into constellation space, and Instrinsic Mode Function-Distinct Native Attribute (IMF-DNA) [36], which uses signal decomposition techniques to separate the original signal into intrinsic mode functions.

Another common approach, especially for radar applications, is to transform the input signals into a 2D image and then use a typical image classifier architecture to perform RFF. This is an appealing hybrid approach because it can leverage both expert knowledge about radar classification and mature image classification techniques. Some of the typical transforms used include the Hilbert-Huang Transform [17, 35], Synchrosqueezing Transform [37], Pulse Waveform Transform [38], Transient Energy Spectrum Transform [39], and the Contour-Stella Image Transform [40, 41]. In general, these transforms extract time-frequency information that can be displayed in two dimensions via heatmap, which are fed into convolution-based networks, like CNNs.

One final approach, highlighted by Table 2, is to feed the raw signal data directly into the ML model and let the model learn the best features for performing emitter identification. CNNs have been successfully used perform classification on raw input signals for a variety of communications protocols, including Bluetooth [51], WiFi [53, 54], ZigBee [56, 58], and LoRa [47, 59]. As mentioned, these CNNs are generally based on the AlexNet architecture, which combines several convolutional layers with a stack of fully-connected dense layers.

Table 1. Representative compilation of extractor-based RFF works.

Raw Signal → Fingerprint Extractor → Classifier		
[42] IEEE 802.15.4 (ZigBee)	TD-DNA	RndF
[43] IEEE 802.15.4 (ZigBee)	TD-DNA	MDA/ML
[44] IEEE 802.15.4 (ZigBee)	CB-DNA	MDA/ML, RndF
[9] IEEE 802.15.4 (Other)	TD-DNA	MDA/ML
[45] IEEE 802.15.4 (Other)	TD-DNA	MDA/ML
[15] IEEE 802.15.4 (Other)	Gabor-based DNA	MDA/ML
[46] Z-Wave	TD-DNA	GRLVQ-I
[16] WiMax	Gabor-based DNA	MDA/ML
[10] Ethernet (Wired)	CB-DNA	MDA/ML
[47] LoRa	Spectrogram	CNN
[36] Synthetic (LFM, BPSK)	IMF-DNA	SVM
[17] Synthetic (QPSK)	Hilbert-Huang Transform	ResNet
[48] Synthetic (Gaussian)	PA Nonlinearity	ANN, CNN
[35] Radar	Hilbert-Huang Transform	CNN
[37] Radar	Synchrosqueezing Transform	CNN
[38] Radar	Pulse Waveform Images	CNN
[40] Radar	AF-RS, CS-MASK, STFT, MC	CNN
[41] Radar	Contour Stella Image	CNN
[39] IEEE 802.11b	Transient Energy Spectrum	Probabilistic NN
[49] IEEE 802.11	Contour Stella Image	CNN
[50] Synthetic (QAM, PSK)	CNN, IQ Imbalance	Bayes

Table 2. Representative compilation of end-to-end RFF works.

Raw Signal → Classifier	
[51] Bluetooth/BLE	CNN
[52] IEEE 802.11ac (WiFi)	CNN
[53, 54] IEEE 802.11a (WiFi)	CNN
[55] IEEE 802.11a/g (WiFi)	CNN
[6] IEEE 802.11a/g (WiFi)	CNN
[56] IEEE 802.15.4 (ZigBee)	CNN
[57] IEEE 802.15.4 (ZigBee)	CNN
[58] IEEE 802.15.4 (ZigBee)	CNN
[59] LoRa	CNN
[55] ADS-B	CNN
[6] ADS-B	CNN

One of the key contributions of this work is a CNN-based fingerprint extractor called FEDR. Through end-to-end training of its neural network, FEDR learns to isolate distortion information into short fingerprint vectors. Those vectors can then be used as inputs to simpler classifiers (e.g., a small artificial neural network) for highly effective RFF.

2.3.2 Robustness in RFF Fingerprints

One of the key challenges with RFF is that fingerprints tend to change under different operational conditions, and changes in fingerprints can significantly deteriorate classifier performance [60]. As such, practical implementation of RFF techniques requires enhanced robustness in both fingerprint extraction and classification. Recent research has presented several approaches to improve robustness.

A common technique is to study variability stemming from changes in operational environments and specifically account for those changes. Notably, Andrews recently studied fingerprint changes due to fluctuations in temperature [7]. He found that temperature had a significant effect on the local oscillator and consequently on the carrier frequency offset, so he proposed a fingerprinting technique that removes carrier frequency offset. Elmaghbub and Hamdaoui studied fingerprint variability between indoor and outdoor deployments, short and long range transmissions, and varying levels of congestion [59]. To improve robustness, they developed a technique that coupled in-band emissions with unintended out-of-band emissions for a more descriptive fingerprint.

General robustness to the RF channel is another common area of research [18]. For instance, Soltani et al. and Zhou et al. augmented training data by adding many types of synthetic noise, effectively making classifiers more noise-resilient [61, 62]. Conversely, Wang and Gan proposed a technique for smoothing out noise in the

original signal to suppress its effect on the classifier [63]. Chen et al. proposed a dual neural network approach for learning features of signal distortions for a specific set of devices, separating features between those common across devices and those unique to devices [35]. Isolation of relevant distortion information improved resiliency to channel conditions and improved overall RFF performance.

Yet another approach is to add a "watermark" to a signal before transmission that is detectable across many channel conditions [64]. Sankhe et al. used a CNN to detect the presence of phase imbalance, gain imbalance, and DC offset [54]. They leveraged the IQ imbalance calibration tools for the X310 USRP SDR to inject their *own* imbalances. Their CNN classifier, called ORACLE, was designed to recognize and classify their specific injected distortions across a broad RF environment, enabling them to perform practical authentication for their SDR-based transmitters.

Finally, recent research has also focused on reducing the computational overhead of RFF at the endpoint. Rondeau et al. and Mims et al. reduced fingerprint dimensionality by measuring the relevance of each fingerprint element (e.g., through RndF or LVQ relevance), and selecting only the most relevant elements [14, 15]. Computational overhead was reduced because smaller fingerprints can be calculated more quickly and require smaller classifiers. Jian et al. reduced classifier size by pruning trained neural networks [6]. Starting with the very deep ResNet50-1D CNN classifier, they progressively removed nodes that had little impact to classifier performance, effectively reducing deployed network size and slashing network inference times.

III. (Study I) Considerations for Radio Frequency Fingerprinting across Multiple Frequency Channels

[13] *J. A. Gutierrez del Arroyo, B. J. Borghetti, and M. A. Temple, "Considerations for Radio Frequency Fingerprinting across Multiple Frequency Channels," Sensors, vol. 22, p. 2111, 2022. [Online]. Available: https://doi.org/10.3390/s22062111*

This manuscript was published in the *Radio Frequency Machine Learning* special issue of MDPI Sensors on 09 March 2022 and is presented here as published. A full set of experimental results from this study may be found in Appendix B. An on-the-fly burst detection technique was used to capture wideband Wireless Highway Addressable Remote Transducer (WirelessHART) signals with efficient use of hard disk space. Details of this technique are included in Appendix C.

3.1 Abstract

Radio Frequency Fingerprinting (RFF) is often proposed as an authentication mechanism for wireless device security, but application of existing techniques in multi-channel scenarios is limited because prior models were created and evaluated using bursts from a single frequency channel without considering the effects of multi-channel operation. Our research evaluated the multi-channel performance of four single-channel models with increasing complexity, to include a simple discriminant analysis model and three neural networks. Performance characterization using the multi-class Matthews Correlation Coefficient (MCC) revealed that using frequency channels other than those used to train the models can lead to a deterioration in performance from MCC > 0.9 (excellent) down to MCC < 0.05 (random guess), indicating that single-channel models may not maintain performance across all channels used by the transmitter in realistic operation. We proposed a training data

selection technique to create multi-channel models which outperform single-channel models, improving the cross-channel average MCC from 0.657 to 0.957 and achieving frequency channel-agnostic performance. When evaluated in the presence of noise, multi-channel discriminant analysis models showed reduced performance, but multi-channel neural networks maintained or surpassed single-channel neural network model performance, indicating additional robustness of multi-channel neural networks in the presence of noise.

3.2 Introduction

Physical-layer emitter identification, known as RFF or Specific Emitter Identification (SEI), is often proposed as a means to bolster communications security [64]. The underlying theory is that the manufacturing processes used for chip components create hardware imperfections that make each emitter unique, irrespective of brand, model, or serial number. These hardware imperfections are akin to human biometrics (e.g., fingerprints) in that they are distinctive and measureable. Imperfections cause small distortions to the emissions of idealized signals, and those signal distortions can be learned by Machine Learning (ML) models to identify emitters solely from their emissions. This is particularly useful for communications security applications where the reported bit-level identity (e.g., MAC Address, Serial Number) of a device cannot or should not be implicitly trusted. Here, RFF can serve as a secondary out-of-band method for identity verification.

Prior related RFF research has trained ML models to identify devices by using bursts received on a single frequency channel. However, modern communications protocols often employ multiple frequency channels to enable simultaneous users and interference avoidance. For example, WiFi (IEEE 802.11 b/g/n) [2] subdivides the 2.4 GHz ISM band into 11 $\times$ 20 MHz overlapping channels, ZigBee [19] and Wireless

Highway Addressable Remote Transducer (WirelessHART) [20] (i.e., IEEE 802.15.4-based protocols) use the same frequency band but divide it into 15 $\times$ 5 MHz non-overlapping channels [21], and Bluetooth [22] uses an even more granular division of 80 $\times$ 1 MHz non-overlapping channels.

When the channel changes, the carrier frequency used by both the transmitter and receiver shifts to the center of the new channel. This change in carrier frequency affects the signal distortions because radio hardware components such as Phase-Locked Loops (PLLs), amplifiers, and antennas operate irregularly at different frequencies. Furthermore, the Radio Frequency (RF) environment also varies with frequency channel because different sources of interference are present at different frequencies. Since most RFF research predominantly considers bursts received on a single channel, it is not clear whether the performance achieved by those research efforts generally extends to multiple frequency channel operation.

For instance, when researchers in [56, 57, 58] collected ZigBee signals, they configured their receivers to capture a narrow bandwidth centered on a single carrier frequency. They used those collections to train RFF models and tested their models on sequestered data from the same collections. No evidence was provided showing that the authors verified model performance across all channels.

In works providing single channel demonstrations, channel carrier frequency details are often omitted, given that the RF signals are commonly down-converted to accommodate baseband fingerprint generation. For example, ZigBee research in [42, 43, 44, 14], and similarly, WirelessHART research in [9], cited the collection bandwidth but omitted carrier frequency information. Furthermore, from an experimental perspective, it is more time-consuming to collect signals across multiple carrier frequencies, given the narrow RF bandwidth limitations of commonly accessible Software-Defined Radios (SDRs). Therefore, the omission of carrier frequency

information suggests that researchers in [42, 43, 44, 14, 9] did not consider the effects of multi-channel operation.

Finally, there are researchers who implicitly use collections from multiple carrier frequencies but make no explicit declaration in their work. For instance, any researchers using the Defense Advanced Research Projects Agency (DARPA) Radio Frequency Machine Learning (RFML) WiFi dataset, such as [53, 54, 65], have the ability to account for multiple frequency channels because collections for that dataset were performed with a wide bandwidth. It is not clear whether [53, 54, 65] deliberately considered frequency channel when selecting training and evaluation datasets.

To our knowledge, no prior work has considered the sensitivity of RFF performance to different frequency channels; our results suggest that frequency channel must be considered. Signal bursts were collected using a wideband SDR receiver, which captured signals from eight IEEE 802.15.4-based devices communicating across 15 frequency channels. Each individual burst was filtered and categorized based on the frequency channel within which it was received. RFF models were trained and tested using data from different channel combinations to evaluate the effects of frequency channel to performance, which was reported using the multi-class MCC. Machine learning models included a Multiple Discriminant Analysis/Maximum Likelihood (MDA/ML) model with expert-designed features, a shallow fully-connected Artificial Neural Network (ANN), a Low-Capacity Convolutional Neural Network (LC-CNN), and a High-Capacity Convolutional Neural Network (HCCNN).

The key contributions of our work include:

- A first-of-its-kind evaluation of the sensitivity of single-channel models to multi-channel datasets. The evaluation suggests that failing to account for frequency channel during training can lead to a deterioration in performance from MCC > 0.9 (excellent) down to MCC < 0.05 (random guess), indicating that single-

channel model performance from previous RFF research should not be expected to extend to the multi-channel case (Experiment A).

- A training data selection technique to construct multi-channel models that can outperform single-channel models, with average cross-channel MCC improving from 0.657 to 0.957. The findings indicate that frequency-agnostic variability can be learned from a small subset of channels and can be leveraged to improve the generalizability of RFF models across all channels (Experiment B).

- An assessment of multi-channel models against Additive White Gaussian Noise (AWGN) that demonstrated the advantage of multi-channel models in noise performance depended on model type and noise level. Multi-channel neural networks approximately maintained or surpassed single-channel performance, but multi-channel MDA/ML models were consistently outperformed by their single-channel counterparts (Experiment C).

The rest of this paper is structured as follows: an overview of the state-of-the-art in RFF is provided in Section 3.2.1, and assumptions and limitations of this research are covered in Section 3.2.2. Our wideband data collection technique is described in Section 3.3, including how the data were processed for RFF model training. Methodology and results for the three experiments are detailed in Section 3.4, and study conclusions and potential future work are presented in Section 3.5.

3.2.1 Related Work

RFF is fundamentally an ML classification problem, where discriminative features are leveraged to distinguish individual devices. Generation and down-selection of the best features remains an open area of research [11]. For instance, researchers have proposed an energy criterion-based technique [66] and a transient duration-based technique [39] to detect and extract features from the transient region of the signal during power-on. Researchers in [9] extract features from the signal preamble region, focusing on down-selecting statistical features to reduce computational overhead while maintaining classification accuracy. Yet another technique presented in [67] extracts features from a 2-dimensional representation of the time series data. Commonly, researchers avoid feature selection altogether by ingesting time series data directly into Convolutional Neural Networks (CNNs), which can be trained to learn the best feature set to be used for the ML task. CNNs have been recently used to classify a large number of devices across a wide swath of operational conditions [65], to verify claimed identity against a small pool of known devices [58], and to measure identifiable levels of IQ imbalance deliberately injected by the transmitter [54].

Another area of research focuses on bolstering the practicality of deploying RFF mechanisms in operational environments. Notably, [6] explores how neural networks might be pruned to reduce their size and complexity, enabling their deployment to resource-constrained edge devices. Researchers in [68] tackle the need to retrain RFF models whenever a new device is added to the network by employing Siamese Networks, effectively aiding model scalability through one-shot learning. Our work contributes to this effort by comparing performance across four model types with increased levels of complexity, including an expert-feature-based MDA/ML model, a shallow ANN, a low-capacity CNN, and a high-capacity CNN.

Finally, the sensitivity to deployment variability is considered extensively in recent works. For instance, variability stemming from time, location, and receiver configuration are studied extensively in [59], and variability stemming from the RF environment is evaluated in depth by [55, 69]. Often, the goal is to remove or reduce environmental effects to bolster classification accuracy, which can be done through data augmentation [61], transmitter-side pre-filtering [70], deliberate injection of IQ imbalance at the transmitter [54], and through receiver-side channel equalization [65].

Our work extends the research in deployment variability by exploring the impact to model performance from the use of different frequency channels. Consistent with the works by [55, 59], we concluded that evaluating models under conditions that are different from training conditions leads to negative impacts to RFF performance. Like [61], we found that adding more variability in our training made the models more generalizable, even to conditions not seen during training.

3.2.2 Assumptions and Limitations

RFF and SEI are broad areas of research that include everything from the fingerprinting of personal and industrial communications devices, to radar and satellite identification. Our work leverages communications devices, but it is likely that any transmitter which operates across multiple carrier frequencies would exhibit the effects highlighted in this research.

The protocol used in this study was WirelessHART, which implements the PHY-layer in the IEEE 802.15.4 specification. That specification divides the 2.4 GHz Industrial, Scientific and Medical (ISM) band into 15 $\times$ 5 MHz channels, each with a different carrier frequency. It is not clear whether the performance improvements of the multi-channel models shown in this research depend on the bandwidth of the frequency channel. For instance, models for Bluetooth, which employs narrower

80×1 MHz channels, may need more frequency channels or further-spaced channels in the training set to achieve the same levels of performance improvements. The impact of channel bandwidth to multi-channel models is left as future work.

Another key WirelessHART feature is that it allows the use of mesh networking, whereby each device can act as a relay of data for neighboring devices. Mesh networking is also becoming increasingly popular in home automation, particularly with the adoption of new Internet of Things (IoT)-centric protocols such as Thread [71]. The distributed nature of those networks poses a challenge in RFF because there is no centralized endpoint with which all other devices communicate, so there is no ideal centralized location to place the RFF receiver. Our work is limited to the centralized configuration, which assumes that all WirelessHART devices communicate directly with the gateway. Configurations to handle mesh-networking, which could include multi-receiver systems or edge-based RFF, should be explored in future work.

Although our work is limited to preamble-based fingerprinting, in part due to its recent success in classifying WirelessHART devices [14], another highly researched technique is transient-based fingerprinting. Transient-based fingerprinting employs features related to how a device becomes active in preparation for transmission, which has proven fruitful for the purpose of device identification [66, 39]. It is likely that transient-based detection will also be affected by carrier frequency, given that the same radio components persist in the transmit chain. Regardless, the study of the effects of frequency channel to transient-based fingerprinting is an interesting area of future work.

In the wideband collection for this work, only one WirelessHART device was configured to communicate at a time. Under real operational conditions, multiple devices would be able to communicate simultaneously on separate frequency channels, introducing the potential for Adjacent Channel Interference (ACI). ACI occurs when

energy emitted on one frequency channel leaks into adjacent frequency channels. At a minimum, this energy leakage could raise the noise floor, reducing the Signal-to-Noise Ratio (SNR) and potentially degrading model performance. A study of the specific effects of ACI to model performance is left as future work.

In the end, our goal was to demonstrate that frequency channel can have a significant effect on RFF models in the hopes of encouraging future researchers to take it into consideration.

3.3 Data Collection

A multi-channel WirelessHART dataset was collected and validated for the purpose of this study. Precautions were taken to minimize the effects stemming from the RF environment and receiver, as our focus was the study of transmitter effects. This section covers the methodology used for collection and pre-processing and includes a description of the training, validation, and evaluation dataset(s).

3.3.1 WirelessHART Communications Protocol

The WirelessHART communications protocol, used by industrial sensors to transmit stateful information (e.g., temperature, humidity, voltage, etc.) between industrial sensors and to human-machine interfaces, implements the Offset-Quadrature Phase Shift Keying (O-QPSK) physical layer from IEEE 802.15.4, the standard for low-rate personal area networks [21]. This is the same standard employed by ZigBee, another common IoT protocol; many RF chips designed for WirelessHART are also compatible with ZigBee applications. WirelessHART uses the first 15 channels defined by the standard on the 2.4 GHz ISM band and employs a pseudo-random frequency hopping scheme, where subsequent transmissions are sent on different frequency channels. Although the channel numbering given by IEEE 802.15.4 ranges

from 11 through 25, we arbitrarily number our channels 0 through 14. Then, for a given channel $i \in [0, 14]$, its center frequency is $f_c(i) = 2405 + i \times 5$ MHz, and its bandwidth is $(f_c - 2.5, f_c + 2.5)$ MHz.

WirelessHART was selected as the candidate protocol for collection for several reasons. First, the IEEE 802.15.4-based protocol is representative of many of the basic low-power IoT devices being deployed across the globe at an exponentially increasing rate. Second, it operates in a common frequency band using a manageable number of channels. And finally, recent research in [9] employed MDA/ML using the same WirelessHART devices used in our research, providing a baseline for performance comparison.

3.3.2 Collection Technique

Figure 1 envisions how an RFF-based authentication mechanism might be deployed for WirelessHART. In this configuration, a centrally-located wideband SDR passively captures bursts sent between the WirelessHART sensors and the gateway. Those bursts are offloaded to a monitoring application, which performs RFF and validates the claimed identity of the communicating device. Once the burst is validated, the information contained within it is considered trusted. Our collection setup is based on this conceptual configuration.

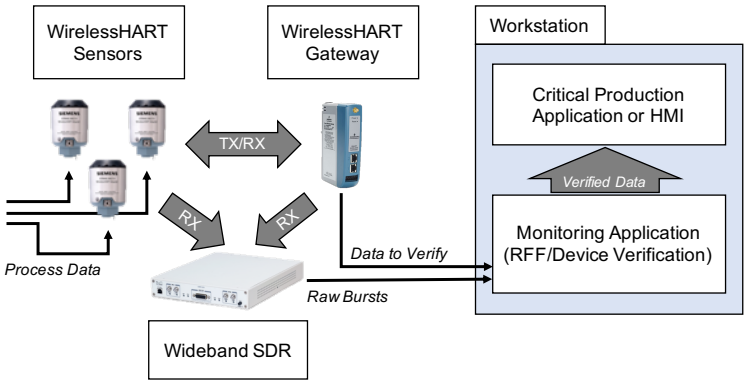


Figure 1. Envisioned use case for RFF-based authentication of WirelessHART devices. Our experimental setup mimicked this configuration.

WirelessHART devices were allowed to communicate directly with the WirelessHART gateway one at a time, while an SDR captured device emissions. During collection, the device was placed 8 ft. from the gateway, and the SDR antenna was positioned 18 in. from the device. The eight devices observed are listed in Table 3 and included four Siemens AW210 [72] and four Pepperl+Fuchs Bullet [73] devices.

Table 3. Serial numbers and source addresses for the eight WirelessHART devices.

Device Number	**Manufacturer**	**Serial Number**	**Hex Source Address (Assigned by Gateway)**
0	Siemens	003095	0002
1	Siemens	003159	0005
2	Siemens	003097	0006
3	Siemens	003150	0003
4	Pepperl+Fuchs	1A32DA	0004
5	Pepperl+Fuchs	1A32B3	0007
6	Pepperl+Fuchs	1A3226	0008
7	Pepperl+Fuchs	1A32A4	0009

Bursts were captured using a USRP X310 with a 100 MHz bandwidth centered at 2.440 GHz, enabling simultaneous collection of all 15 WirelessHART channels. Burst detection for the wideband data was performed "on-the-fly" by thresholding the power of the received signal. This enabled the collector to be efficient with its limited hard disk space. In particular, given a received signal,

$$c[n] = c_I[n] + jc_Q[n], \tag{2}$$

the instantaneous power was calculated as

$$|c[n]|^2 = (c_I[n])^2 + (c_Q[n])^2. \tag{3}$$

A 100-sample moving average was then used to detect the start and end of each burst using empirically-set thresholds. Buffers of 10 K complex IQ samples were added before the start and after the end of the burst to aid in SNR approximation, and each detected burst was saved to a new file.

Three precautions were taken in an attempt to minimize effects from the RF environment. First, collections were done within a ranch-style suburban household, far away from other wireless emitters relative to the distance between emitter and receiver. Second, all collections were performed in the same physical location, meaning that any RF effects due to interference with nearby non-emitters would likely manifest in the same way for all devices. Finally, collections were performed in sets of 10 K bursts, started at random times during the day throughout the course of two weeks. This ensured any potential time-dependent sources of interference were well distributed across devices.

Collecting data in sets of 10 K bursts also forced the WirelessHART gateway to assign a new frequency hopping scheme to each device when it re-established communication with the gateway (per the WirelessHART specification [20]). This resulted in a relatively even distribution of bursts across the 15 frequency channels.

3.3.3 Burst Validation

Our approach for burst detection did not guarantee that the received signal came from the expected WirelessHART device; it could be the case that strong interference triggered the burst detector. Further protocol-specific analysis was performed to ensure the validity of each collected burst. The most straightforward way to do this

was to detect and verify the structure of the preamble, and subsequently read message-level bits to verify that the transmitted source address corresponded to that listed in Table 3. The process of burst validation consisted of Frequency Correction, Low-Pass Filtering, WirelessHART Preamble Detection and Verification, Phase Correction, and Message Parsing and Address Verification.

3.3.3.1 Frequency Correction and Low Pass Filtering

Unlike [65], which leveraged the Center Frequency Offset (CFO) as a discriminable classification feature, we chose to remove the CFO because it depends on the characteristics of the receiver, and we are most interested in understanding the impact of carrier frequency to the transmitter. First, bursts were downconverted to baseband using a energy-based coarse estimation of frequency channel. Then, the algorithm presented in [74] was used to quickly approximate the remaining frequency offset between transmitter and receiver by squaring the detected signal and taking a Fast Fourier Transform (FFT). The squaring created peaks at two different frequencies, and taking the average between the peaks gave an estimate of the CFO. Figure 2 shows how two peaks are present after squaring the burst. To correct the frequency offset, the bursts were shifted in frequency such that the two peaks were centered about 0 MHz.

All frequency correction was achieved through multiplication by complex sinusoid. In particular, for a given burst $c[n]$ with estimated center frequency f_e, the frequency-corrected burst was

$$c_{cor}[n] = c[n]e^{-j2\pi(2440\times10^6 - f_e)n}. \tag{4}$$

After frequency correction, a 2 MHz 4^{th}-order Butterworth low pass filter was applied to each burst for noise suppression outside of baseband.

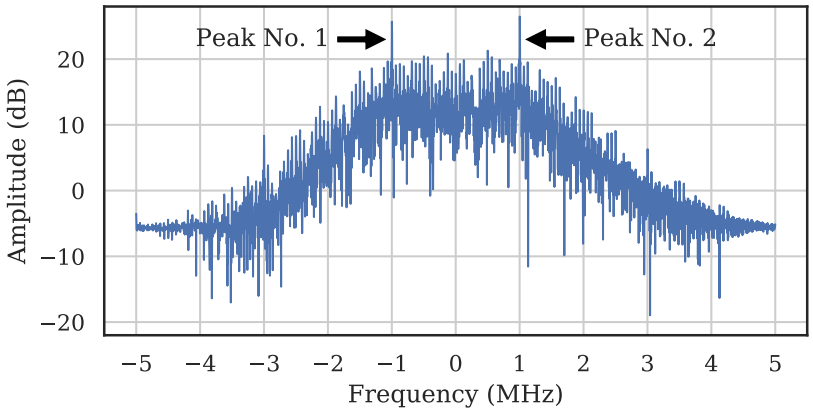


Figure 2. FFT of the square of a WirelessHART burst with yielded peaks. Centering the peaks about 0 MHz center-aligns the carrier and corrects the frequency offset.

3.3.3.2 WirelessHART Preamble Detection and Verification

Preamble detection was done in the time domain by correlating the peak-normalized filtered burst with a generated preamble. The index with maximum-amplitude correlation was assumed to be the starting index of the burst preamble. As a measure of similarity, the corresponding correlation coefficient was also used for WirelessHART preamble verification; if the coefficient was too low, the burst was discarded.

3.3.3.3 Phase Correction

The angle of the correlation coefficient was also an estimate of the phase offset between the burst preamble and the generated preamble. It was used to correct phase through multiplication by complex exponential. Given a burst $c[n]$ with phase offset θ, the phase-corrected burst was

$$c_{cor}[n] = c[n]e^{-j\theta} \tag{5}$$

A note on frequency and phase correction: in operational receivers, both phase and frequency correction are done via PLLs [75]. PLLs continuously tune the receiver cen-

ter frequency to drive the phase offset between transmitter and receiver to zero. This is undesirable for the RFF application because the distortions in phase and frequency, which are suppressed by the PLL, could be leveraged for device discrimination.

3.3.3.4 Message Parsing and Address Verification

The downconverted, phase-corrected burst was demodulated, and the symbols were mapped to message bits. The source address was extracted from the parsed message and compared to the known device addresses listed in Table 3. If the address matched the expected device address, the burst was considered valid.

3.3.4 SNR Estimation

SNR estimation of the validated bursts was necessary to enable model evaluation against noise. All SNR estimates were performed after the 2 MHz low pass filter was applied.

Noise power and signal-plus-noise power were estimated on a per-burst basis directly from the captured burst. The *signal-plus-noise* region from which signal-plus-noise power (P_{S+N}) was estimated was defined as the 1605 samples that made up the preamble. The preceding 160 samples were deemed a *buffer* region, where none of the samples were used for any power estimation. The *noise* region for estimating noise power (P_N) was defined as the 1605 samples immediately preceding the buffer region. Figure 3 shows these three regions for an example burst. For a given complex burst, $c[n] = c_I[n] + jc_Q[n]$, the estimated powers and SNR (in dB) are

$$P_N = \frac{\sum_{n \in \text{noise rgn}} (c_I[n]^2 + c_Q[n]^2)}{1605} \tag{6}$$

$$P_{S+N} = \frac{\sum_{n \in \text{sig+noise rgn}} (c_I[n]^2 + c_Q[n]^2)}{1605} \tag{7}$$

$$\text{SNR (dB)} = 10 \log_{10} \left(\frac{P_S}{P_N} \right) = 10 \log_{10} \left(\frac{P_{S+N} - P_N}{P_N} \right). \tag{8}$$

Figure 4 shows the per-class range and average SNRs estimated on the full set of collected bursts. The average burst SNR across all devices and all channels was 41.4 dB.

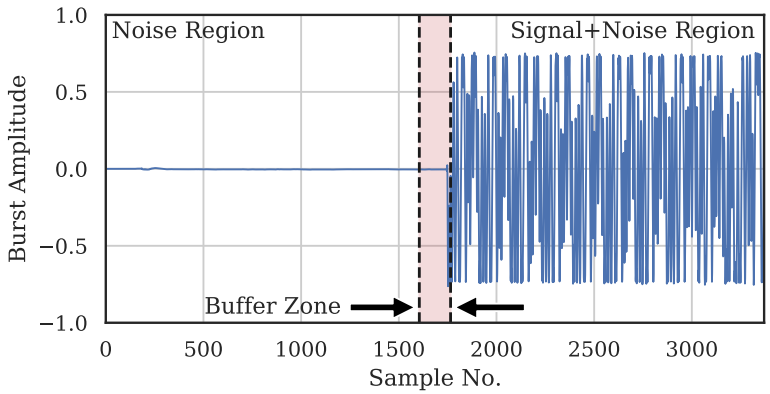


Figure 3. Sample burst depicting the noise region (1605 samples), buffer zone (160 samples) and signal-plus-noise region (1605 samples) used for SNR estimation. For clarity, only the I-component is shown, but the power was calculated using the complex samples.

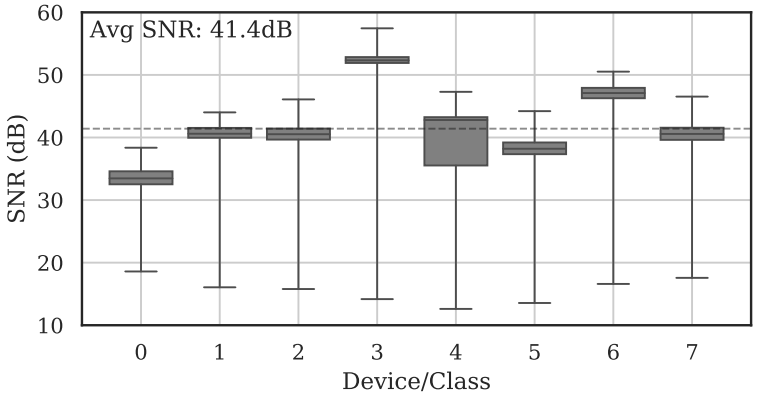


Figure 4. Summary of estimated collection SNRs for all devices across all channels. As denoted by the dotted line, the average SNR was 41.4 dB.

3.3.5 Datasets

Once all bursts were validated, datasets were created for the purpose of model training, validation, and evaluation. The large pool of valid bursts were randomly sampled to create datasets that were balanced with respect to the classes and to the frequency channels. Table 4 outlines the fundamental datasets for our work. The experiments leveraged subsets of these datasets (e.g., by training with data from only one channel), but the delineation between the sequestered training, validation, and evaluation datasets remained.

Table 4. Fundamental datasets used in this work, broken down by class and channel.

No. of Observations	Training Set	Validation Set	Evaluation Set
per Device per Channel	5,000	500	1,000
per Channel	40,000	4,000	8,000
per Class	75,000	7,500	15,000
in Total	600,000	60,000	120,000

3.4 Experiments

Three experiments were designed to (i) assess the sensitivity of single-channel models to a multi-channel dataset, (ii) determine whether carefully-constructed multi-channel models generalized better than single-channel models, and (iii) determine whether multi-channels models gained more resiliency to noise. This section provides a description of the four models used across all tests, followed by individual methodologies, results, and analyses for the three experiments.

3.4.1 RFF Models

Four typical RFF models with varying levels of complexity were selected as candidates for experimentation in this work. All four model types were used in all three experiments. This section describes the structure of each model in detail.

1. Multiple Discriminant Analysis/Maximum Likelihood. This model type leverages the commonly-used Time-Domain Distinct Native Attribute (TD-DNA) feature set [42, 76, 77], as depicted in Figure 5, which was recently used by Rondeau et al. on single-channel WirelessHART bursts [9]. TD-DNA features are statistics calculated for a set of signal subregions (defined by N_R) after a number of signal transforms. The features are dimensionally reduced through Fisher Transform and fed in to a standard Quadratic Discriminant Analysis (QDA) model for classification. Including dimensionality reduction and discriminant analysis, the total number of trainable parameters is 1,757.

2. Fully-Connected Artificial Neural Network. Figure 6 shows the ANN, which operated directly on raw complex I-Q burst data input to the network on two independent data paths (one for I and one for Q). The two-dimensional input was flattened into a single 3210-wide vector which was fed to the first fully-connected layer. Even with 207,848 trainable parameters (10 times that of MDA/ML), this particular ANN was a shallow, low-complexity alternative to the other two neural networks.

3. Low-Capacity Convolutional Neural Network. Like the ANN, the LCCNN presented in Figure 7 operated directly on raw complex IQ burst data. The network comprised eight Conv1D layers that applied a total of 112 digital filters and two fully-connected layers that mapped the filter output to a four-dimensional latent space. The Softmax output layer was used for classification, where class

prediction was determined by the node with largest output value. Because of its convolutional layers, the LCCNN is able to add depth to the ANN with 232,156 trainable parameters.

4. High-Capacity Convolutional Neural Network. This high-capacity model was inspired by ORACLE, a CNN that extracts and classifies injected IQ imbalance [54]. This model carried significantly more capacity, with 14 hidden layers comprising four pairs of 128-filter Conv1D layers that applied a total of 1024 digital filters, and two fully-connected layers that mapped the filter outputs to a 64-dimensional latent space. Like in the LCCNN, the input layer contained two independent input paths, and the output Softmax layer contained one node for each of the eight classes in this study. This is the most complex model, requiring 3,985,544 trainable parameters, approximately 20 times as many as the ANN and LCCNN. Figure 8 depicts the HCCNN.

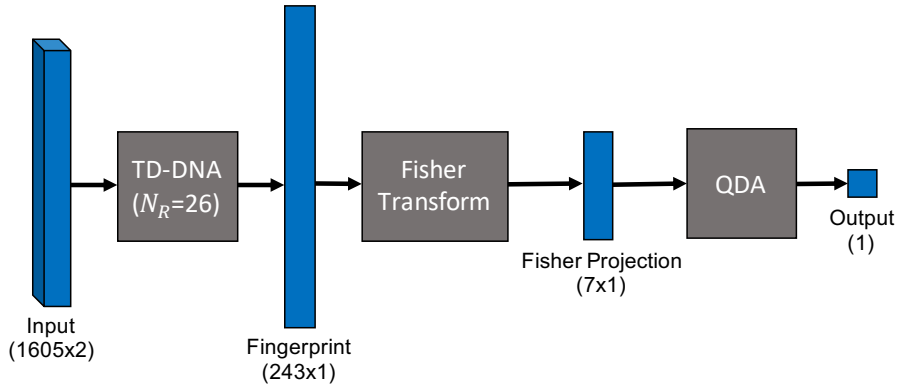


Figure 5. MDA/ML QDA model with TD-DNA feature set with 26 subregions.

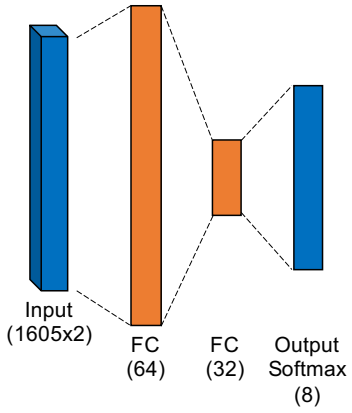


Figure 6. Fully-Connected Artificial Neural Network.

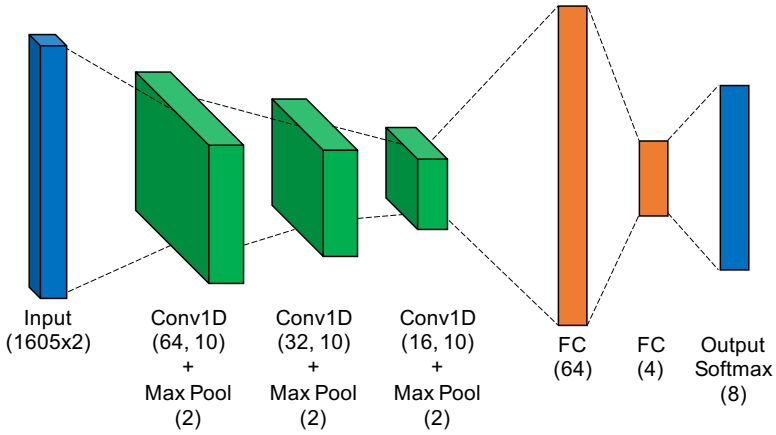


Figure 7. Low-Capacity Convolutional Neural Network.

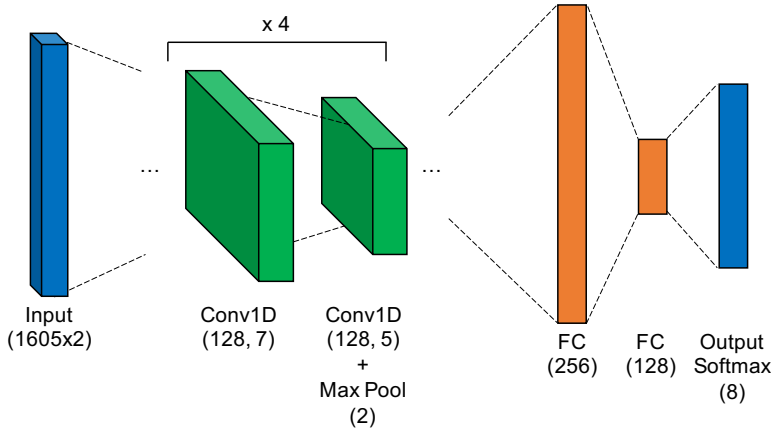


Figure 8. High-Capacity Convolutional Neural Network.

MDA/ML models were trained by fitting a QDA model onto the dimensionally-reduced TD-DNA features. All neural networks were trained using Stochastic Gradient Descent with the Adam optimizer and learning rate of 1×10^{-4}, with batch size of 512 observations. An early stopping callback was used to monitor the performance on the validation set, where training was stopped after 10 epochs of no improvements. The model weights with best validation set performance were restored and this was used as the final trained model.

3.4.2 Performance Metric: Matthews Correlation Coefficient (MCC)

Performance for RFF models is typically reported as per-class classification accuracy, which is the accuracy averaged across all classes. For K classes, per-class classification accuracy degrades to $1/K$ when models perform no better than random guess. It is therefore difficult to grasp model performance without first knowing how many classes were used to train the model. One way to address this problem is to standardize performance by the number of classes through the use of MCC.

MCC is a performance metric first designed for binary class classification models, derived by Matthews as a discrete version of the correlation coefficient [78]. An MCC value of 1.0 indicates perfect correct model performance, 0.0 indicates performance no better than random guess, and -1.0 indicates perfect incorrect model performance. The multi-class case, derived by Gorodkin [79], preserves the -1.0 to 1.0 quantitative performance characteristics. For K classes, MCC is calculated as

$$MCC = \frac{cs - \sum_k p_k t_k}{\sqrt{s^2 - \sum_k (p_k)^2} \sqrt{s^2 - \sum_k (t_k)^2}} \tag{9}$$

where t_k is the number of occurrences for class $k \in [0, 1, 2, \ldots, K-1]$, p_k is the number

of times class k was predicted, c is the total number of correct predictions, and s is the total number of predictions [80]. For our experiments, we use $K = 8$ classes and report all model performance via MCC.

3.4.3 Experiment A: Single-Channel Models

The goal in this experiment was to determine whether the performance of models trained on a single channel (i.e., "single-channel models") extends to other frequency channels. To that end, single-channel models were evaluated against a dataset containing bursts from multiple channels, and the per-channel evaluation performance was reported.

3.4.3.1 Methodology

The training and validation datasets described in Section 3.3.5 were subdivided into 15 subsets, one for each channel, yielding a total of 5 K training observations and 500 validation observations per device in each subset. Then, the four models were trained with each subset to create a total of 60 single-channel RFF models. All models were evaluated on the full evaluation dataset described in Section 3.3.5, which contained bursts from all 15 channels. MCC was calculated using Equation (9).

3.4.3.2 Results and Analysis

Figures 9–11 show the performance for single-channel models trained on data from Channel 0, Channel 7 and Channel 14, respectively. Plots for the remaining single-channel models have been omitted for brevity, given these three figures are representative of the overarching observations for this experiment. Performance was reported individually for each channel in the evaluation set to show how well the models operated outside their training scope. Note that performance always peaked

at the channel on which the models were trained and deteriorated when the evaluation channel was farther away (in frequency) from the training channel.

As expected, none of the models generalized across all channels when trained on data from a single channel, but some models did generalize better than others. In particular, when the models were trained on data from Channel 7, they generalized well across a wide swath of channels (e.g., Channels 5–10), whereas when the models were trained on Channel 14, performance only roughly generalized to Channel 13 and only for MCA/ML and HCCNN. In the worst case, signals from distant channels were classified by the Channel 14 model with success no better than random guess.

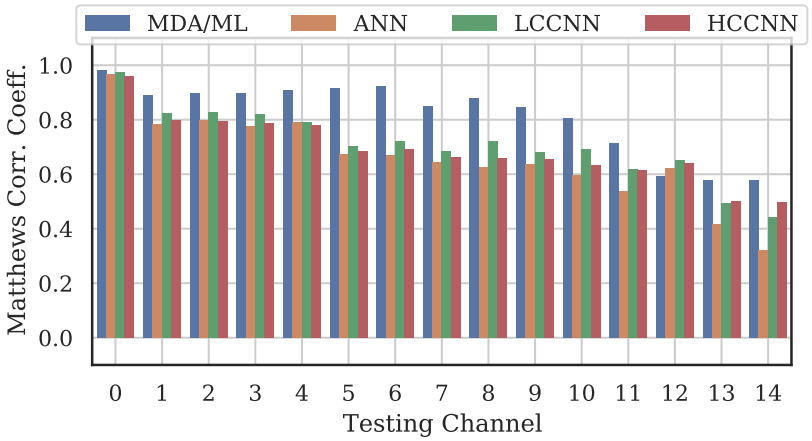


Figure 9. Performance of single-channel models trained with data from Channel 0.

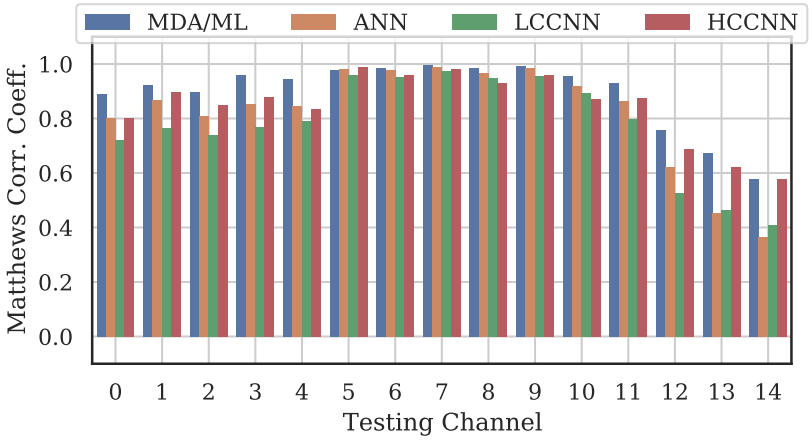


Figure 10. Performance of single-channel models trained with data from Channel 7.

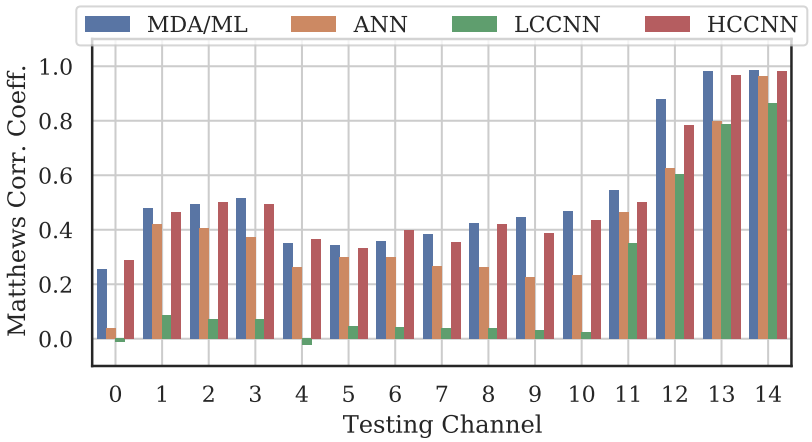


Figure 11. Performance of single-channel models trained with data from Channel 14.

Between the model types, the MDA/ML model and the HCCNN model were the most competitive, with the MDA/ML model performing best most of the time. The LCCNN and ANN performed similarly most of the time, but the LCCNN was especially bad at generalizing when it was trained on Channel 14. These observations suggest that the MDA/ML and HCCNN model were better at inherently targeting variability that existed irrespective of frequency channel.

In general, frequency-dependent signal distortions appeared to be on a spectrum, where signals from nearby channels exhibited similar distortions. However, it is not clear how the channel width might affect those distortions. It could be the case that with narrower channels, like the 1 MHz channels in Bluetooth, performance extends across more channels. Such exploration of the effects of channel width are interesting future work.

Additionally, recall that the transmitter, RF environment, and receiver distortions were not decoupled. Thus, it is possible that a non-trivial part of the variability between frequency channels was imposed by non-transmitter sources. Finding a way to decouple (or at least reduce the coupling) across the three sources would enable more flexible applications (e.g., multi-receiver narrowband systems) and could help

researchers more precisely target the signal alterations imposed by hardware components in the transmitter. Regardless of its source, the variability must be considered to achieve good RFF performance.

A natural follow-up experiment was to include data from multiple channels in the training set to determine if this improved model generalizability. This experiment is covered in the following section.

3.4.4 Experiment B: Multi-Channel Models

The focus of Experiment A was to demonstrate that single-channel models did not always perform well across all frequency channels. In Experiment B, the goal was to determine whether including bursts from multiple channels during training improves performance. During training, we deliberately use data from an increasing number of channels, relatively spread throughout the 80 MHz band to create "multi-channel models." These models were tested against the same evaluation set from Experiment A, which included bursts from all 15 channels.

3.4.4.1 Methodology

The same four models presented in Section 3.4.1 were used in Experiment B. Training was done using 11 datasets assembled from portions of the full training set described in Section 3.3.5, but special consideration was taken with respect to the number of observations in training sets.

Generally, the performance of ML models is influenced by the number of observations provided during training. To enable comparison between multi-channel models from Experiment B and single-channel models from Experiment A, the size of the training datasets was limited to no more than 5,000 observations per device (i.e., the size of training sets in Experiment A). Concretely, two-channel data sub-

sets contained 2,500 observations per device per channel (5,000 observations/device in total), three-channel data subsets contained 1,666 observations/device per channel (4,998 observations/device), four-channel subsets had 1,250 observations per device per channel (5,000 observations/device), and the all-channel (i.e., 15-channel) subset contained only 333 observations per device per channel (4,995 observations/device). Channel combinations were selected to explore how channel coverage impacted performance. The 11 data subsets used for Experiment B are listed in Table 5. Evaluation was done using the same dataset from Experiment A, which contained bursts from all 15 channels.

Table 5. Summary of datasets used for Experiment B.

No. of Chans.	Chans. in Datasets	Observations per Chan. per Device	Observations per Device	Total
2	[0,14], [1,13], [2,12], [3,11]	2,500	5,000	40,000
3	[0,7,14], [1,7,13], [2,7,12], [3,7,11]	1,666	4,998	39,984
4	[0, 4, 10, 14], [1, 5, 9, 13]	1,250	5,000	40,000
All	[0,1,2,. . . ,14]	333	4,995	39,960

3.4.4.2 Results and Analysis

A representative sample of the performance results for the multi-frequency models are depicted in Figures 12–15.

Comparing the Channel 14 single-channel models from Experiment A in Figure 11 with the 2-channel models in Figure 12, it is immediately evident that adding even one more channel to the training set improved generalizability, regardless of model type. With the addition of Channel 0 to the training set, the worst-case performance

across all models improved from MCC = −0.23 to MCC = 0.716. Note that model performance improved across *all* channels, even if the channels were not explicitly included in the training set.

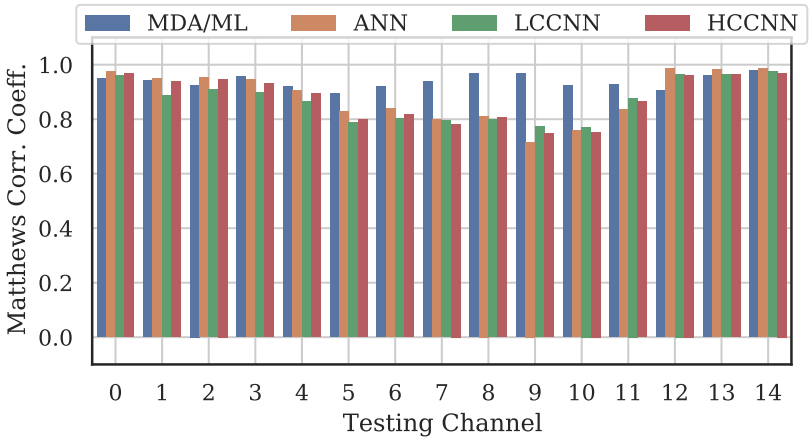


Figure 12. Performance of multi-channel models trained with data from Channels 0 and 14.

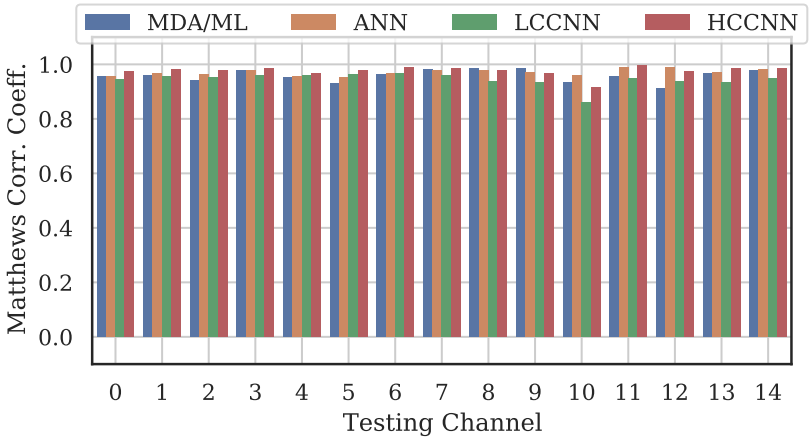


Figure 13. Performance of multi-channel models trained with data from Channels 0, 7 and 14.

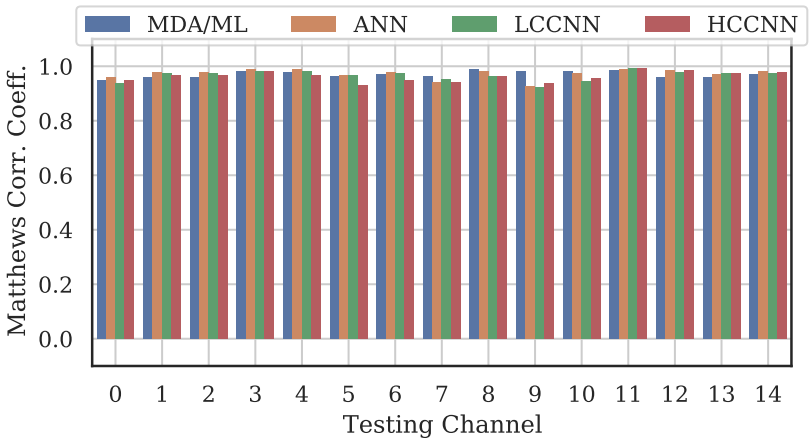


Figure 14. Performance of multi-channel models trained with data from Channels 0, 4, 10 and 14.

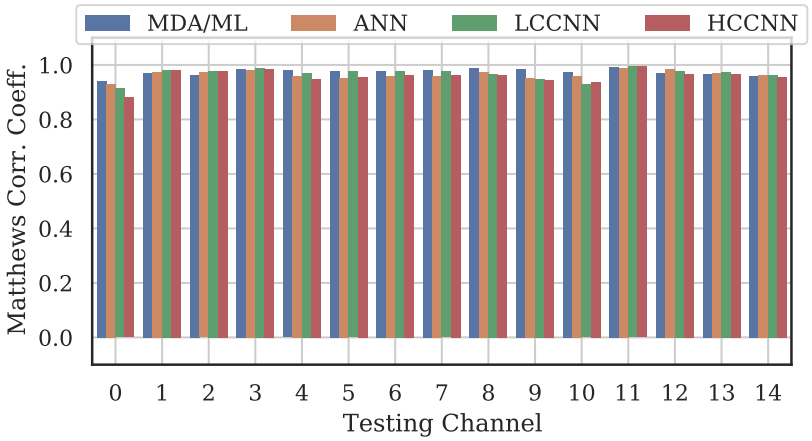


Figure 15. Performance of multi-channel models trained with data from all channels.

Table 6 and Figure 16 summarize the combined performance of models from Experiment A and Experiment B. A single MCC metric was calculated for each trained model by aggregating the per-channel results, and MCC_{avg} was calculated by averaging across all models of the same type that employed the same number of channels. For the 15-channel case, the reported MCC_{avg} is the performance of the sole 15-channel model.

Table 6. Summary of MCC_{avg} for all models from Experiments A and B. In general, MCC improved as the number of channels in the training set were increased.

No. of Channels	MCC_{avg}			
	MDA/ML	ANN	LCCNN	HCCNN
1	0.833	0.723	0.657	0.742
2	0.943	0.857	0.823	0.859
3	0.961	0.957	0.930	0.950
4	0.967	0.970	0.957	0.958
15 (All)	0.974	0.964	0.967	0.958

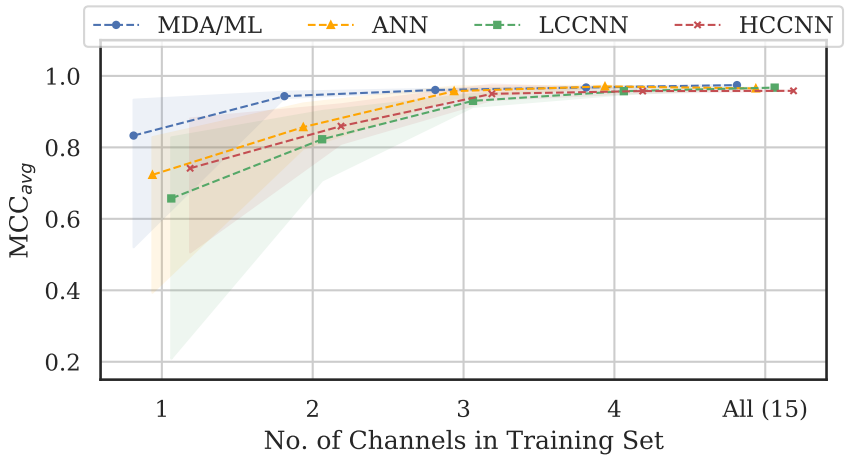


Figure 16. Performance summary for all models as the number of channels in the training was increased. Trend lines represent MCC_{avg}, and color bands represent the range of MCC values at that number of training channels. Notably, all-channel performance was achieved with only four channels included in the training set.

Two trends are evident from these metrics: (i) model performance generally improved when channels were added to the training set, and (ii) the performance of 4-channel models approached the performance of 15-channel models. Note that even the worst performing single-channel model type, i.e., the LCCNN, improved its MCC_{avg} from 0.657 to 0.957 with only three additional channels in the training set. The one

exception was with the ANN, for which the 15-channel MCC_{avg} was 0.006 units lower than the 4-channel MCC_{avg}. This small gap in performance might be attributed to the variability stemming from the randomized initial model weights before training, or from the fact that the 15-channel "average" included only one model—retraining the 15-channel ANN may yield slightly better results[1].

Regardless, these trends again support the existence of frequency-irrespective variability. Multi-channel models were better suited to learn that variability, even with limited (i.e., 4-channel) exposure to the spectrum. Furthermore, the MDA/ML model consistently generalized better than its neural network counterparts, suggesting that the frequency-irrespective variability in our particular experimental setup can be effectively extracted through the TD-DNA fingerprint generation process.

In practice, it is desirable for RFF models to perform well across all frequency channels, but that is not the only requirement. Models should also perform well under the presence of environmental noise. Thus, researchers often report model performance across multiple levels of noise, modeled as AWGN. Experiment C explores whether multi-channel models gain any performance advantages over single-channel models under varying noise conditions.

3.4.5 Experiment C: Gains in Noise Performance

The goal of the final experiment was to determine whether multi-channel models gained any performance advantages over single-channel models in noisy RF environments. Tested models included the Channel 7 single-channel models from Experiment A, and new "all-channel/all-data" multi-channel models built exclusively for this experiment. All models were evaluated with bursts from Channel 7 with varying SNR levels adjusted synthetically through the addition of power-scaled AWGN.

[1]After publication of this manuscript in [13], the 15-channel ANN model was retrained from scratch 10 times, wherein MCC averaged 0.971 and peaked at 0.976. This result is consistent with the expected improvement with the additional training channels.

3.4.5.1 Methodology

Two datasets were used for training: a single-channel set, where the models were only exposed to Channel 7 data (5 K observations/device), and a multi-channel set, which included the full training dataset (75 K observations/device). Performance was captured on a subset of the full evaluation set with only Channel 7 bursts (1 K observations/device). The full training dataset was used to maximize the exposure of multi-channel models to Channel 7 data, enabling them to better compete against the single-channel models. Multi-channel models were at an inherent disadvantage because they had to overcome both channel and noise, whereas single-channel models only had to overcome noise.

Each model was trained and evaluated at the same SNR level, which was adjusted to simulate increasingly harsh operational environments. Individual bursts were adjusted to the desired SNR through the addition of AWGN. For each burst, a noise realization was generated from a normal distribution and filtered using the same 2 MHz low pass filter from burst processing. Then, the power of the noise realization was scaled such that when it was added to the burst, the desired SNR was achieved. Models were re-trained at each SNR level, and MCC was calculated after aggregating the results across 100 noise realizations. Performance was reported for SNR $\in \{5, 6, 7, 8, 9, 10, 12, 14, 16, 18, 25, 25, 30, 35, \text{Max}\}$, where "Max" means no SNR adjustment was made (i.e., capture SNR).

Note that since noise could only be added (not removed), the per-burst SNR could only ever be decreased. As an example, Figure 17 shows the same bursts from Figure 4 after they were adjusted to a maximum SNR of 35 dB, resulting in a mean SNR of 34.8 dB across all data. In that case, bursts that had SNR lower than 35 dB remained unchanged.

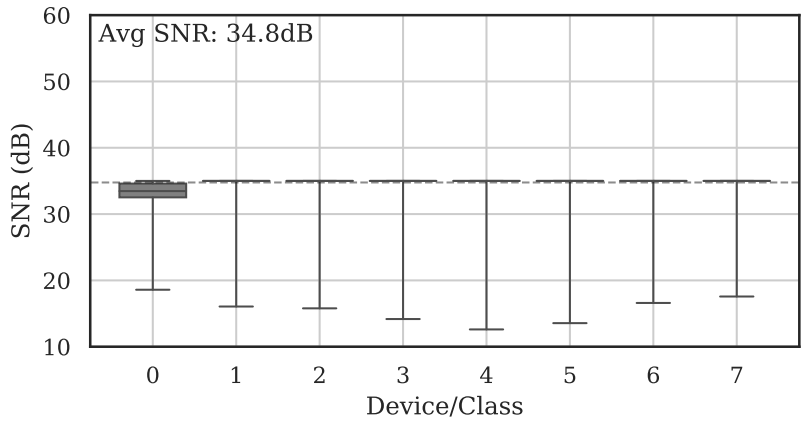


Figure 17. Summary of estimated collection SNRs for all classes across all channels after adjusting SNR to 35 dB. The mean SNR dropped to 34.8 dB, as denoted by the dotted line.

3.4.5.2 Results and Analysis

Figure 18, which depicts noise performance of the LCCNN models, is representative of the results across all model types. As expected, MCCs for both the single- and multi-channel models worsened as SNR was decreased. To help determine whether multi-channel models gained an advantage against noise, we define a new metric, MCC_Δ, as the difference between multi-channel performance and single-channel performance for a given model type, i.e.,

$$\text{MCC}_\Delta = \text{MCC(multi-channel)} - \text{MCC(single-channel)}, \tag{10}$$

for which a positive MCC_Δ implies better multi-channel performance. Figure 19 illustrates MCC_Δ for the four models across varying SNR levels.

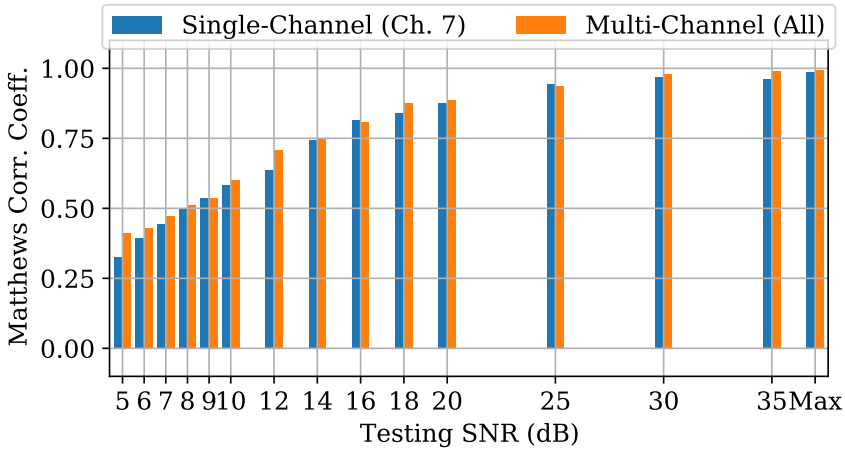


Figure 18. Performance of single-channel and multi-channel LCCNN models at varying SNRs.

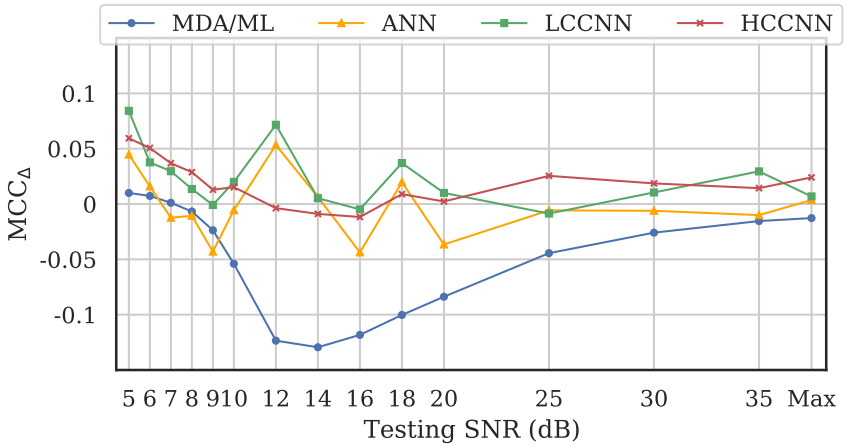


Figure 19. Performance difference between multi-channel models and single-channel models across varying SNR levels, where $MCC_\Delta > 0$ implies better multi-channel performance. Multi-channel neural networks (i.e., CNNs and ANN) approximately maintained or outperformed their single-channel counterparts, but multi-channel MDA/ML models did not.

The advantage of multi-channel models depended on model type and SNR level. At high SNR levels (SNR $>$ 20 dB), MCC_Δ was generally stable for all model types. In that region, the multi-channel CNNs matched or beat single-channel CNNs, but the single-channel ANNs and MDA/ML models bested their multi-channel counterparts. With mid-level SNRs (10 dB $<$ SNR $<$ 20 dB), MCC_Δ for the three neural networks fluctuated between positive and negative, suggesting no clear advantage for

multi-channel models. Notably, single-channel MDA/ML models thrived in this region, surpassing multi-channel models by up to 0.13 units. Finally, at low SNR levels (SNR $<$ 10 dB), multi-channel models for all four model types showed some advantage over single-channel models, though arguably, the performance in this region was already too weak (MCC $\lesssim$ 0.5) to be practical for RFF applications.

For the neural networks (i.e., CNNs and ANN), the frequency-irrespective variability learned by the multi-channel models enabled them to approximately maintain or surpass single-channel model performance. Conversely, the single-channel MDA/ML models consistently outperformed their multi-channel counterparts in the presence of noise. It could be that frequency-specific variability, which was deliberately ignored by multi-channel models, allowed some of the single-channel MDA/ML models to overfit the training channel, giving them an advantage against random noise.

3.5 Conclusions and Future Work

Modern communications protocols often employ multiple frequency channels to enable simultaneous user operation and mitigate adverse interference effects. Although recent RFF research targets devices that implement these protocols, the direct applicability of these proof-of-concept works is generally limited given that they train RFF models using bursts received on a single channel. Because the signal distortions leveraged by RFF models are linked to the radio hardware components, and those components operate irregularly across different frequencies, practical RFF models must account for multiple frequency channels. Using WirelessHART signal bursts collected with a wideband SDR, our work demonstrated that RFF model performance depends on the frequency channels used for model training.

Candidate models, including MDA/ML using expert-aided features, a fully-connected ANN and two CNNs, were evaluated across several training-evaluation channel com-

binations. Performance of single-channel models did not always generalize to all frequency channels. In the most disparate case, one of the single-channel models performed almost perfectly (MCC > 0.9) on its training channel and no better than random guess (MCC < 0.05) on a non-training channel. Often, models performed well on the training channel and relatively well on the adjacent channels, but deteriorated outside of that scope. This suggests that signal distortions were continuous with respect to frequency, i.e., nearby channels exhibit similar distortions.

When data from multiple channels were included in the training set, the multi-channel models generalized better across *all* channels, achieving adequate performance even when just a small subset of channels were included (i.e., four of the 15). In the worst case, the average MCC for LCCNN models improved from 0.657 in the single-channel configuration to 0.957 in the 4-channel configuration, again implying bolstered performance across all channels. This finding suggests that there existed frequency-irrespective variability that could be learned by the models and used for RFF.

The performance advantage of multi-channel models under noisy conditions depended on model type and SNR level. Multi-channel neural networks (i.e., CNNs and ANN) were able to approximately maintain or surpass single-channel model performance across most SNR levels, but multi-channel MDA/ML models were consistently outperformed by their single-channel counterparts. It could be that the frequency-specific variability available to the single-channel MDA/ML models caused them to overfit the training channel, giving them an advantage against random noise.

One interesting area of future work would be to explore how the bandwidth and spectrum location of the frequency channels used in training affects multi-channel performance. Each additional channel included in the training set exposed the RFF models to an additional 5 MHz "chunk" of that spectrum. This additional expo-

sure enabled models to learn frequency-agnostic variability, making them generalize better across all frequencies. We found that 20 MHz (i.e., four WirelessHART channels) of exposure spread throughout the 80 MHz band was enough to achieve frequency channel-agnostic performance. Other common communications protocols employ channels of different sizes; e.g., Bluetooth channels are 1 MHz wide, and typical WiFi channels are up to 20 MHz wide. It could be the case that *more* Bluetooth channels and *fewer* WiFi channels would be needed to achieve generalizable multi-channel model performance because of the difference in channel bandwidth. Further study of the effects of channel bandwidth and spectrum location to frequency-irrespective variability remains an area of future work.

Another area of future work would be to address the radio limitations for practical RFF applications. As discussed, modern IoT protocols enable mesh networking, whereby each endpoint in the network can relay data to and from its neighbors. A practical RFF solution must be able to target all of these data transfers to be useful for security. One solution would be to include RFF capabilities within the individual endpoints, as proposed by researchers in [6]. To that end, our work explored the use of low-complexity models in a multi-channel configuration (e.g., MDA/ML or LCCNN and found them to be generally adequate under most conditions, as long as they were trained using multiple channels. Indeed, this type of deployment is the long-term vision for wireless security, but it does not address the devices that are already deployed and operational.

A stopgap solution would be to deploy more RFF-capable SDRs, forming multi-receiver RFF systems. Multi-receiver systems could also be useful in non-mesh configurations if individual SDRs cannot cover all frequency channels. The key challenge would be to find a way to share RFF models across radios to avoid the tedious collection and training effort that would come with scale. One approach may be to com-

bine bursts collected from multiple receivers, similar to our multi-channel approach, whereby the RFF models could learn receiver-irrespective variability. Notably, this effort would also aid in decoupling signal distortions imposed by the receiver from those imposed by the transmitter and RF environment, further adding to its value as future work.

Finally, with the extension of RFF to multi-channel configurations, the effects of ACI to RFF model performance should be explored. When multiple devices communicate simultaneously on different frequency channels, the potential exists for some of the energy in one channel to leak to adjacent channels. At a minimum, this energy leakage could raise the noise floor, reducing SNR and potentially degrading model performance. Understanding the extent to which ACI can affect model performance will therefore be critical in the deployment of RFF models to real operational environments.

RFF models continue to offer an attractive out-of-band method for wireless device authentication, especially as a component in the defense-in-depth security paradigm. As modern protocols grow in operational complexity, the variability of signal distortions across these expanded modes of operation must be considered to achieve the most effective and generalizable RFF systems.

IV. (Study II) FEDR: A Neural Network-Based Technique for Radio Frequency Fingerprint Extraction

4.1 Abstract

Radio Frequency Fingerprinting (RFF) is the attribution of uniquely identifiable signal distortions to emitters via Machine Learning (ML) classifiers, of which Convolutional Neural Networks (CNNs) attain state-of-the-art performance. This work presents Feature Extraction through Distortion Reconstruction (FEDR), a neural network-based approach for quantifying signal distortions in a latent space which intends to embed distortion parameters, achieved through a constrained network architecture and a custom regularization loss. Because the FEDR network learns about signal distortions, it can be trained independently of RFF classes and deployed as a static fingerprint extractor, which can be coupled with a low-complexity ML classifier at the endpoint to perform RFF. FEDR was evaluated on synthetic IQ-imbalanced IEEE 802.11a/g data, for which it reconstructed original distorted 16-QAM symbols with impressive -41.5 dB error. The learned parameter space was shown to represent IQ imbalance parameters, implying successful distillation of distortion information. On a real-world dataset, FEDR achieved distortion reconstruction with -14.1 dB error. Coupled with a simple Dense network, FEDR fingerprints were evaluated against four common RFF techniques for $N_c = \{5, 10, 15, 25, 50, 100\}$ unseen classes. The Dense network with FEDR fingerprints achieved best performance across all values of N_c with Matthews Correlation Coefficient ranging from 0.945 ($N_c = 5$ classes) to 0.746 ($N_c = 100$ classes), using nearly 73% fewer training parameters than the next-best CNN.

4.2 Introduction

With wireless device interconnectivity at the core of the rapidly expanding Internet of Things, the security of wireless links and endpoints is more critical than ever. Typical endpoint authentication mechanisms rely on bit-level cryptographic algorithms to establish trust between devices and to encrypt communications. These algorithms rely on the intractability of guessing secret information, but when that secret information is compromised, the security of the link and the trust between devices is compromised. Recent research has proposed a physical-layer authentication mechanism called RFF as an augmentation to traditional authentication techniques [11].

RFF relies on unique information incidentally built into every device during the manufacturing process. Regardless of make or model, the components within every device are slightly different because of random manufacturing imperfections. Those imperfections affect how signals are generated and create small but measurable distortions in the Radio Frequency (RF) emissions. When those emissions are captured and the distortions are measured, the measurements (a.k.a. "fingerprints") can be traced back to the device that emitted them. This process enables security systems to identify devices solely from their emissions, which can be leveraged to detect when bit-level device impersonation (i.e., man-in-the-middle) attacks take place.

Tracing the fingerprints back to the emitter is framed as a ML classification problem. Classifiers are trained to recognize fingerprints from a set of emitters—when the classifier is presented a new fingerprint, it predicts which emitter from that set generated the signal. Recent research has focused on techniques for deploying those classifiers to edge Internet of Things (IoT) devices, where compute resources are generally limited. This is typically done through dimensionality reduction on the fingerprints [14, 56], but techniques have been proposed for pruning and simplifying classifiers as well [6, 47]. Model complexity can be especially detremental to the clas-

sifier training process, where models with more training paramters can require more compute resources to train.

Types of classifiers and classifier complexity can vary greatly. They often depend on the overarching RFF approach, which can be described as either: i) fingerprint extraction before classification, and ii) end-to-end classification. Fingerprint extractors typically input received signals and output measured fingerprints, which are in turn used to train the ML classifier. Commonly proposed techniques include Time-Domain Distinct Native Attribute (TD-DNA) [9, 13], Constellation-Based Distinct Native Attribute (CB-DNA) [10, 14], Gabor Transform [15, 16], and Intrinsic Mode Function (IMF)-based transforms like the Hilbert-Huang Transform [17].

Classifiers paired with fingerprint extractors can typically be simple—their complexity depends primarily on the size of the fingerprint, hence why dimensionality reduction can make a big impact. Most of the "heavy lifting" is done by the fingerprint extractors themselves because they are tasked with generating fingerprints that can be discriminated across devices. They make assumptions about how distortions manifest in a signal and leverage those assumptions to perform measurements. Typical extractors inherently discard information that might be useful for classification and are often outperformed by end-to-end techniques, which make no assumptions.

End-to-end techniques typically intake raw signals or portions of raw signals and output a predicted emitter, integrating the fingerprint extraction process within the classifier itself. These techniques, like ORACLE [53, 54] and ResNet-50-1D [65], are typically inspired by a popular image classifier called AlexNet [81]. They achieve state-of-the-art RFF performance by leveraging the predictive power of deep CNNs, which learn the best discriminable features for the given set of emitters. The improved performance comes at the cost of complexity—the number of trainable parameters in neural networks can grow very quickly.

This work presents a best-of-both-worlds fingerprint extraction technique named FEDR, where the extractor is itself based on a trainable neural network. Through the structure of the neural network and a specialized loss metric, the technique isolates distortion information from content information, enabling distortion quantification. Because FEDR learns about distortions and not about specific devices, network training can be offloaded and accomplished *a priori*, using signals from arbitrary sets of devices. A trained FEDR network can subsequently be deployed to edge devices for the purpose of fingerprint extraction through network inference (i.e., network forward pass), which requires less compute than network training. Then, as with other fingerprint extraction techniques, a simpler model can be employed for classification.

The key contributions of this work are:

- A deep learning technique for fingerprint extraction called Fingerprint Extraction through Distortion Reconstruction FEDR. Leveraging distorted and undistorted versions of a received signal, FEDR reconstructs the original distortions using a structurally-constrained and regularized relative distortion latent space. Because FEDR learns about distortions and not devices, the network can be trained independently of the RFF classes.

- An evaluation of the FEDR technique on synthetic WiFi data with simulated IQ imbalance. Using a basic dense network, simulated distortion parameters were extracted from the learned FEDR fingerprints, implying that distortion information was quantified by the FEDR technique.

- An evaluation of RFF performance of FEDR fingerprints extracted from a real-world WiFi dataset of never-before-seen emitters. The FEDR-based classifier achieved best performance with Matthews Correlation Coefficient (MCC) ranging from 0.945 (5 classes) to 0.746 (100 classes), using nearly 73% fewer training parameters than the next-best CNN.

The rest of the paper is structured as follows: Section 4.3 discusses related work in the area of RFF and its parent research area of Radio Frequency Machine Learning (RFML). FEDR is evaluated as a distortion-quantifying technique in Section 4.5 on both synthetic and real-world OFDM data, for which hyperparameter selection is also explored. Section 4.6 provides an evaluation of RFF performance of FEDR fingerprints, as compared to state-of-the-art techniques, and Section 4.7 concludes and provides interesting areas of future work and improvement.

4.3 Related Work

The overarching area of research under which RFF falls is RFML, wherein problems are traditionally solved with engineered techniques are solved effectively with ML. RFML research has experienced a recent boom driven by the performance improvements of CNNs and the interest and promise of Cognitive Radio (CR), which must be spectrum-aware to make decisions with respect to spectrum utilization. Specific RFML research vectors include communications protocol identification [8], signal characteristics recognition [82], automatic modulation recognition [83], and RFF.

RFF has been especially driven by its potential in cybersecurity as an enhanced authentication technique. Within RFF, research has generally focused on making fingerprinting techniques more robust and simultaneously less resource intensive. Recent research identified mitigation techniques for the detrimental effects of frequency channel [13], the effects of carrier offset, driven by temperature fluctuations in the environment [7], and the performance variability stemming from changes in operational deployment time and location [59]. Other researchers have focused on reducing the footprint and resource utilization of RFF models to make them better suited for low-power IoT edge devices and protocols (e.g., Bluetooth Low Energy, LoRa) [6, 59, 84]. A trained FEDR network provides a static fingerprint extraction process with low

compute overhead, since network training can be offloaded to a more capable system. Furthermore, FEDR fingerprints provide a common space in which fingerprint robustness can be evaluated.

Deep neural networks have also been used to learn high-efficiency physical layer designs, which can be individually catered to specific types of communications channels [33, 34]. FEDR takes inspiration from this previous work by leveraging a CNN that is trained end-to-end to reconstruct a signal of interest. An overview of the technique is provided in the following section.

4.4 FEDR Overview

Serving as a fingerprint extractor, the FEDR network is trained to learn about distortions, not specific devices. It can therefore be trained independently of RFF classifiers, using signals from entirely different devices, as long as the types of distortions in the training signals represent the types of distortions in the RFF signals. This section describes FEDR, its machine learning objectives, network architecture and loss functions.

4.4.1 Definition of the FEDR Fingerprint

A key insight of this work is that the original undistorted signal depends only on the content being communicated and on the PHY-layer protocol, e.g., the Orthogonal Frequency Division Multiplexing (OFDM) PHY protocol in IEEE 802.11a/g [2]. As long as the content information can be accurately estimated from the distorted signal, it is possible to construct an undistorted version of the same signal. Fortunately, communications transceivers are already designed to perform both content estimation and signal construction for normal operation.

Given a length-L complex distorted signal, $\boldsymbol{d} \in \mathbb{C}^L$, and corresponding undistorted

signal, $\boldsymbol{u} \in \mathbb{C}^L$, the FEDR fingerprint is defined as a length-P measurement of the distortion of $\boldsymbol{d}$ with respect to $\boldsymbol{u}$, namely

$$\boldsymbol{\delta}(\boldsymbol{d}, \boldsymbol{u}) \in \mathbb{R}^P. \tag{11}$$

The distortion parameter vector $\boldsymbol{\delta}$ quantifies a *relative* distortion and should have the property that

$$\boldsymbol{d} = \boldsymbol{u} \implies \boldsymbol{\delta} = \mathbf{0}^P, \tag{12}$$

i.e., the distortion parameters are zero when no relative distortion is present between the two input signals. Fingerprint extraction is therefore the process of estimating $\boldsymbol{\delta}$, which is not known *a priori*.

As a concrete example, suppose the (only) distortion type applied to a signal was IQ Imbalance by the transmitter. Then, $\boldsymbol{\delta}$ would be some numerical representation of the amount of amplitude imbalance and the amount of phase imbalance applied to the signal. This example is explored in detail Section 4.5 to demonstrate that FEDR indeed learns information about those imbalances.

4.4.2 Machine Learning Objectives

Unlike typical RFF techniques, the goal is not to classify emitters from their distorted signals. Instead, the primary objective is to learn some operator **FE**, i.e., a *Fingerprint Extractor* defined by

$$\mathbf{FE} : (\boldsymbol{u}, \boldsymbol{d}) \to \hat{\boldsymbol{\delta}}, \tag{13}$$

which outputs an estimate of the distortion parameters. The key challenge is that the original $\boldsymbol{\delta}$ is not known for signals received in the real world. Thus, a typical ML supervised learning objective cannot be applied.

To overcome this limitation, a secondary learning objective requires the estimated distortion parameters $\hat{\boldsymbol{\delta}}$ to contain enough information to reapply the original distortions to the undistorted signal. Namely, a *Distortion Reconstructor* operator, **DR**, is defined as

$$\mathbf{DR} : (\boldsymbol{u}, \hat{\boldsymbol{\delta}}) \rightarrow \hat{\boldsymbol{d}}. \tag{14}$$

It provides an estimate of the distorted signal given the undistored signal and the distortion parameters. Note that **DR** should be capable of applying all distortions stemming from the transmitter, channel, and receiver.

The CNN architecture presented in the following section was designed to simultaneously learn both of these objectives.

4.4.3 Network Architecture

Figure 20 depicts the FEDR network architecture. It coupled a traditional deep CNN inspired by ORACLE [53] with a U-Net-like architecture typically used for image segmentation [85]. The ORACLE CNN was shown capable of extracting IQ imbalance information, so its architecture was a good initial candidate for general fingerprint extraction. Changes were made to the dense layers to account for the size of the distortion parameter space. A U-Net-like design was useful for distortion reconstruction because of its ability to generate outputs of similar form to its inputs and because of its feed-forward links. Those links allowed FEDR to pass along content information at multiple levels of abstraction.

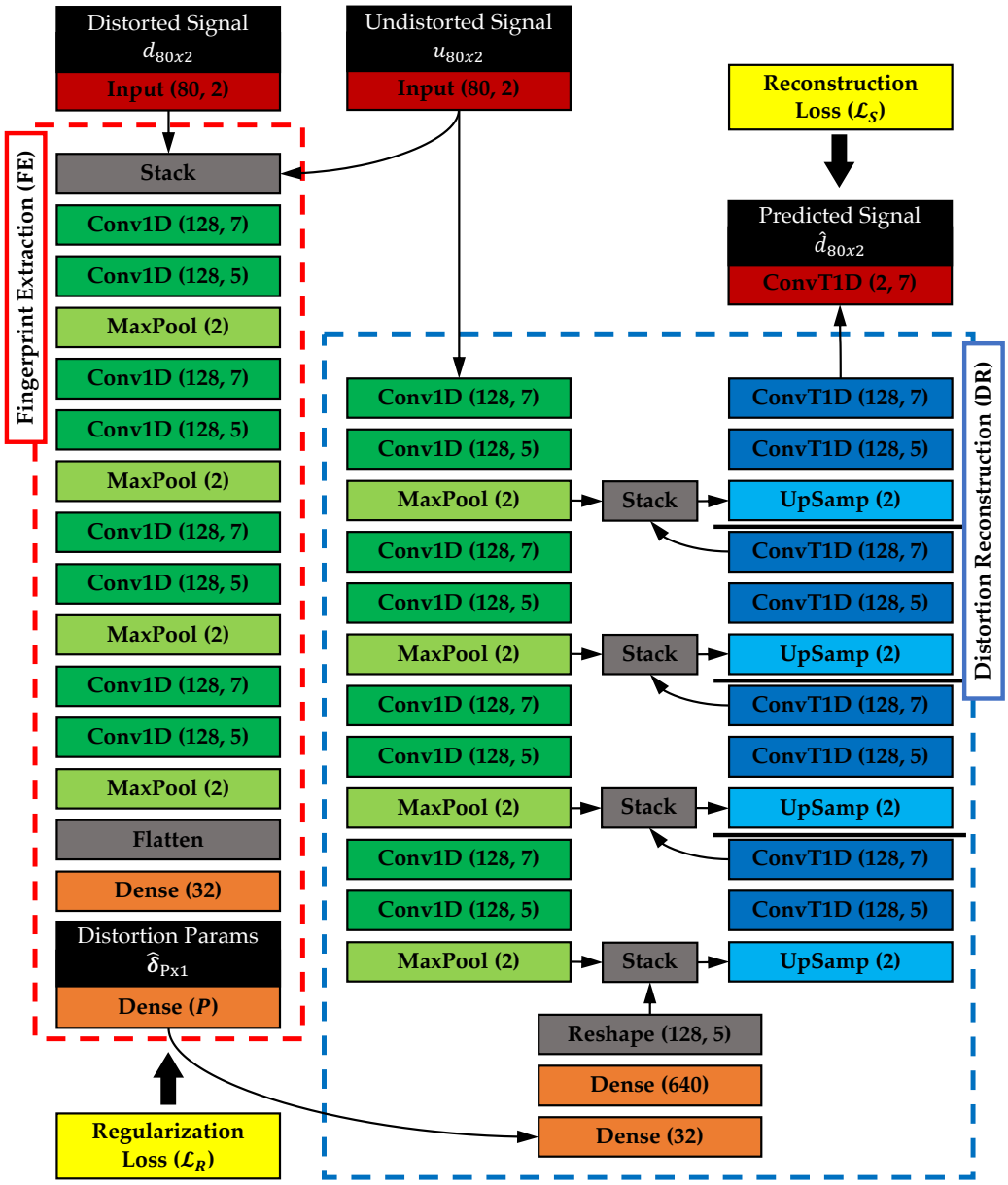


Figure 20. FEDR Network Architecture. The only information passed from the Fingerprint Extractor (FE) to Distortion Reconstructor (DR) are the undistorted signal and the parameters. Model is trained end-to-end to minimize $\mathcal{L}_S$ and $\mathcal{L}_R$.

Portions of the network can generally be described by the objectives they were designed to learn. The leftmost column served as the Fingerprint Extractor and was designed to learn **FE**, inputting the distorted and undistorted signals and outputting and estimate of the distortion parameters. Those distortion parameters were passed

to the right side of the network, which served as the Distortion Reconstructor. It learned **DR** by combining the undistorted signal and distortion parameters to output an estimate of the original distorted signal. The Distortion Reconstructor was inspired by a U-net, allowing abstractions of the undistorted signal to feed forward to the deconvolutional and upsampling layers, which themselves applied the distortions at multiple layers of abstraction.

Another way of describing the network architecture is through the flow of content information and distortion information. Content information comes from the undistorted signal, whereas both content *and* distortion information come from the distorted signal. The goal was to allow only distortion information to pass from the Fingerprint Extractor through the distortion parameters to the Distortion Reconstructor. This was done by limiting the size of the parameter space, P, and by regularizing the parameter space through a custom loss function. Additionally, multiple layers of abstracted content information were made available within the Distortion Reconstructor, suppressing the need for content information to flow from the parameter space.

For this work, the complex inputs were $L = 80$ samples long, i.e., the length of an IEEE 802.11a/g OFDM symbol at 20 MSps. The real and imaginary components of $\boldsymbol{u}$ and $\boldsymbol{d}$ were stacked at the network input, creating input layers of size 80×2. Rectified Linear Unit (ReLU) activation functions were used on all convolutional and dense layers, except for the *Distortion Parameters* layer and the *Predicted Signal* output layer, where no activation was used.

The value of P depended on the breadth of distortions quantified, e.g., $P = 2$ was used for validation and verification on synthetic IQ imbalanced data, and $P = 16$ was used for real-world data. Ideally, P should be selected *just* large enough to be able to represent distortion information in $\hat{\boldsymbol{\delta}}$. Larger parameter spaces run the risk

of also passing through content information, albeit this is discouraged by a proposed regularization loss. A selection strategy for P is presented in Section 4.6. An overview of the loss functions is provided in the following section.

4.4.4 Loss Functions

Two loss functions encouraged the learning of objectives presented in the previous section. The first was a reconstruction loss applied at the output of the Distortion Reconstructor, i.e., the output layer of the network. It comprised a sample-by-sample comparison between the predicted distorted signal $\hat{\boldsymbol{d}}$ and the original distorted signal $\boldsymbol{d}$. Comparison was the squared l^2-norm of the error in 2D space, with the real sample value in one dimension and the imaginary sample value in the other. Concretely, the Signal Reconstruction Loss ($\mathcal{L}_S$) was a sum of squared errors given by

$$\mathcal{L}_S = \sum_{\text{batch}} \sum_{n=0}^{L-1} ||\boldsymbol{d}[n] - \hat{\boldsymbol{d}}[n]||^2, \tag{15}$$

where $n \in \{0, 1, 2, ..., L-1\}$ denotes the signal index in time and $||\cdot||$ denotes the l^2-norm.

A second loss was applied at the output of the Fingerprint Extractor, i.e., the distortion parameter layer. The goal of the second loss was to enforce the property in (12), which regularizes the distortion space such that $\hat{\boldsymbol{\delta}} = \mathbf{0}$ when there is no relative distortion detected between $\boldsymbol{u}$ and $\boldsymbol{d}$.

To do this, the following intermediate predictions were made

$$\hat{\boldsymbol{\delta}}_u = \mathbf{FE}(\boldsymbol{u}, \boldsymbol{u}), \tag{16}$$

$$\hat{\boldsymbol{\delta}}_d = \mathbf{FE}(\boldsymbol{d}, \boldsymbol{d}). \tag{17}$$

The predictions estimated the relative distortion of the undistorted signal to itself and the relative distortion of the distorted signal to itself, both of which should ideally be zero. The Regularization Loss ($\mathcal{L}_R$) measured the l^2-norm of each of these predictions relative to the origin in $\mathbb{R}^P$ and is given by

$$\mathcal{L}_R = \sum_{\text{batch}} (||\hat{\boldsymbol{\delta}}_u||^2 + ||\hat{\boldsymbol{\delta}}_d||^2). \tag{18}$$

Note that in the case where both input signals are undistorted but have different content, the network interprets them as having a non-zero relative distortion between them.

4.5 Evaluation for Quantifying Distortions

The fundamental question in evaluating FEDR was whether the learned parameter space actually quantified distortion information. A synthetic IEEE 802.11a/g OFDM dataset with injected IQ imbalance was generated and used to train a FEDR-2 network (i.e., with $P = 2$). This synthetic dataset also provided truth data for $\boldsymbol{\delta}$, namely the amount of injected amplitude and phase imbalance. Section 4.5.3 shows that with a simple dense network, the parameter space can be mapped back to the original distortion parameters, demonstrating that distortion information was quantified through FEDR technique.

Then, the FEDR technique was applied to a real-world IEEE 802.11a/g OFDM dataset in Section 4.5.4. An evaluation of the impact of the size of the parameter space P is provided. For both synthetic and real-world results, performance is provided as a measure of how well the networks were able to reconstruct the distorted signals. Unless otherwise specified, the symbol modulation type used was 16-QAM.

4.5.1 Prediction Error Vector Magnitude (PEVM)

Use of 16-QAM modulation enables a symbol-based measure of reconstruction quality. Error Vector Magnitude (EVM) is typically used to measure the distance between a reference symbol and a distorted symbol, which is a standard description of error in a symbol-based communications system [86]. The metric of interest for this work is the distance between the distorted symbol and the predicted symbol—the prediction should match the distortion as much as possible. A new metric, Prediction Error Vector Magnitude (PEVM), is proposed to quantify this new error.

Figure 21 depicts EVM and PEVM, given a reference symbol, a distorted symbol, and a predicted symbol. Because this measure is never used during training (i.e., only time-domain and distortion space losses are used), it provides an external perspective into how well FEDR models perform distortion reconstruction.

Given predicted distorted symbol $\boldsymbol{S_p}$ and true distorted symbol $\boldsymbol{S_d}$, PEVM (dB) is defined as

$$\text{PEVM} = 20\log_{10}(||\boldsymbol{S_p} - \boldsymbol{S_d}||). \tag{19}$$

4.5.2 Synthetic Data: Dataset and Training

FEDR was evaluated on an artificial dataset with synthesized IQ imbalance. Here, the goal was to demonstrate that FEDR learned information about distortions. Since the dataset was synthetic, truth values for $\boldsymbol{\delta}$ were available for comparison to the parameter space.

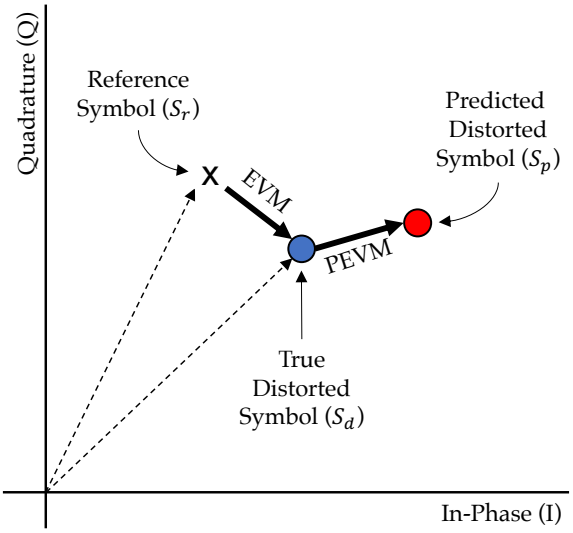


Figure 21. EVM and PEVM. Typically error is measured between the distorted symbol and the undistorted symbol (EVM), but the important error for this application is the distance between the distorted symbol and the prediction of the distorted symbol (PEVM).

4.5.2.1 IQ Imbalance

A typical transmitter architecture is shown in Figure 22. IQ imbalance occurs when there is an amplitude and/or phase imbalance between the in-phase (I) and quadrature (Q) channels of a transmitter or receiver [12].

Given a baseband signal to be transmitted,

$$x_{BB}(t) = x_I(t) + jx_Q(t), \tag{20}$$

IQ imbalance manifests on the signal as individual gains (g_I, g_Q) and a phase offset (2ϕ) between the two channels. Concretely, the transmitted signal is given by

$$s(t) = x_I(t)g_I \cos(\omega_c t + \phi) - x_Q(t)g_Q \sin(\omega_c t - \phi), \tag{21}$$

where g_I and g_Q can be defined through single amplitude imbalance parameter β by

$$g_I = 10^{\frac{0.5\beta}{20}}, \tag{22}$$

$$g_Q = 10^{-\frac{0.5\beta}{20}}. \tag{23}$$

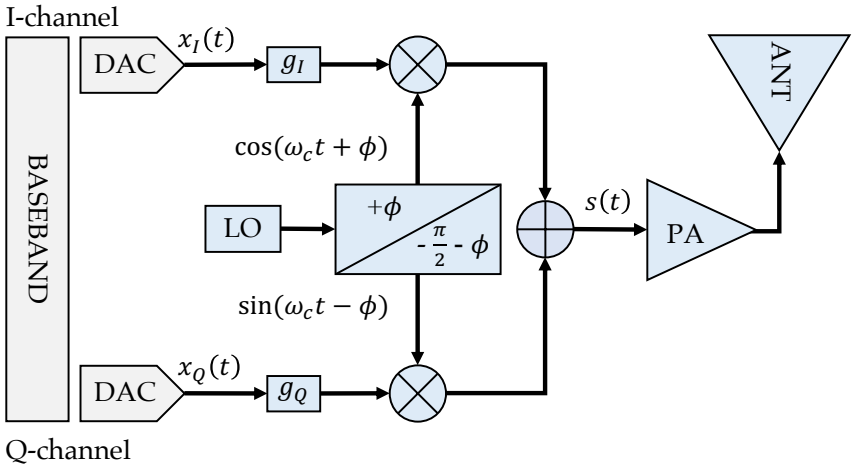


Figure 22. Typical Transmitter Architecture. IQ imbalance comprises amplitude and phase imbalances between the in-phase (I) and quadrature (Q) channels. These imbalances have been shown in recent research to contribute significantly to RFF [50].

The goal of a typical receiver is to reconstruct the original baseband signal from the received signal by individually recovering each channel. It does so by mixing the received signal with the original in-phase and quadrature carriers and low-pass filtering each channel. Assuming the transmitted signal, $s(t)$, is received without further distortion, the resulting signals at the receiver channels are given by

$$\begin{aligned} r_I(t) &= \mathrm{LPF}\{2s(t)\cos(\omega_c t)\} \\ &= x_I(t)g_I\cos(\phi) - x_Q(t)g_Q\sin(\phi) \end{aligned} \tag{24}$$

$$\begin{aligned} r_Q(t) &= \mathrm{LPF}\{2s(t)\sin(\omega_c t)\} \\ &= -x_I(t)g_I\sin(\phi) + x_Q(t)g_Q\cos(\phi) \end{aligned} \tag{25}$$

where LPF$\{\cdot\}$ denotes a low-pass filtering operation. Finally, to reconstruct an estimate of the original baseband signal, the two channels are combined as

$$\begin{aligned}\hat{x}_{BB}(t) &= r_I(t) + jr_Q(t) \\ &= x_I(t)g_I\cos(\phi) - x_Q(t)g_Q\sin(\phi) \\ &- jx_I(t)g_I\sin(\phi) + jx_Q(t)g_Q\cos(\phi). \end{aligned} \tag{26}$$

4.5.2.2 Signal Generation

Each baseband synthetic signal was generated by applying 16-QAM and 48-carrier OFDM to a string of 192 random bits, representing a single OFDM symbol. A cyclic prefix was prepended in accordance with the IEEE 802.11a/g PHY specification. A 20 MSps sample rate was selected such that each synthetic signal comprised 80 complex samples, where the real part described the in-phase channel and the imaginary part described the quadrature channel. Amplitude and phase imbalances were then applied to the synthetic signal, consistent with Equation (26).

Ranges in IQ imbalance parameters were selected such that the EVM did not surpass -16 dB, the published acceptable limit in IEEE 802.11a/g for 16-QAM [2]. Figure 23 shows how EVM deteriorates quickly with IQ imbalance. The target zone for the synthetic data was anywhere in the center dark region, where IQ imbalance does not surpass -16 dB. Amplitude imbalance parameter from $\beta \in [-2.5, 2.5]$ and phase imbalance parameter from $\phi \in [-\frac{\pi}{32}, \frac{\pi}{32}]$ were randomly selected for each synthesized signal. A total of 140,000 signals (100,000 for training, 20,000 for validation, and 20,000 for evaluation) were generated.

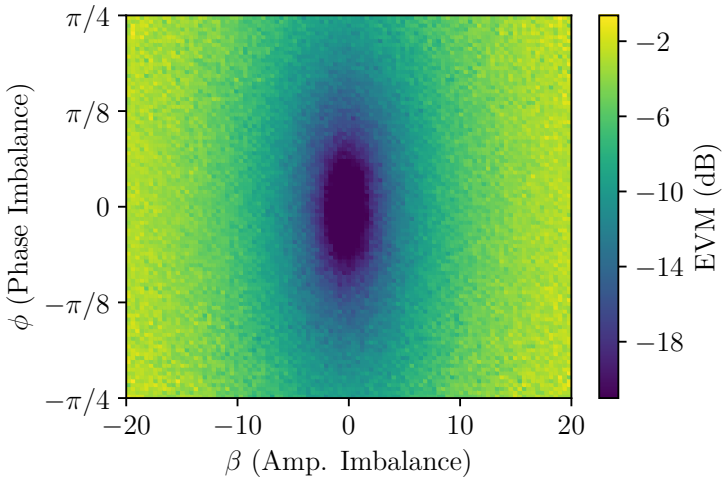


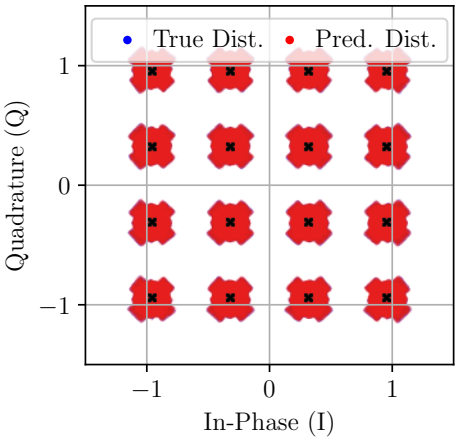
Figure 23. EVM for OFDM+16-QAM with amplitude (β) and phase (ϕ) imbalance. The dark region in the middle is the target zone for the synthetic distortion, where EVM less than -16 dB, within the operational parameters specified by IEEE 802.11a/g.

4.5.2.3 Training

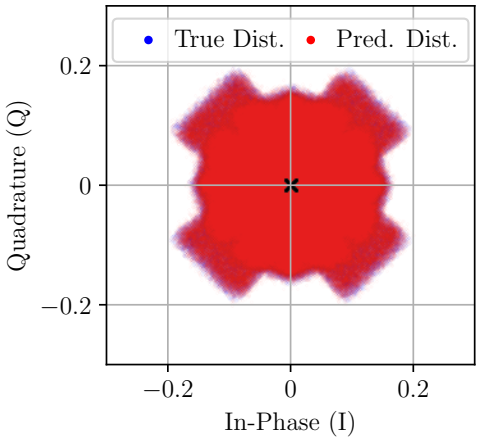
A FEDR model with $P = 2$ was used, since the distortions were originally represented by two parameters (i.e., β, ϕ). Network training was accomplished using the Adam optimizer with learning rate of 1×10^{-4}, stopping training after 5 epochs of no performance improvement on validation data. Training for all FEDR models was done in an end-to-end fashion, meaning that both the Fingerprint Extractor and Distortion Reconstructor were simultaneously trained.

4.5.3 Synthetic Data: Results

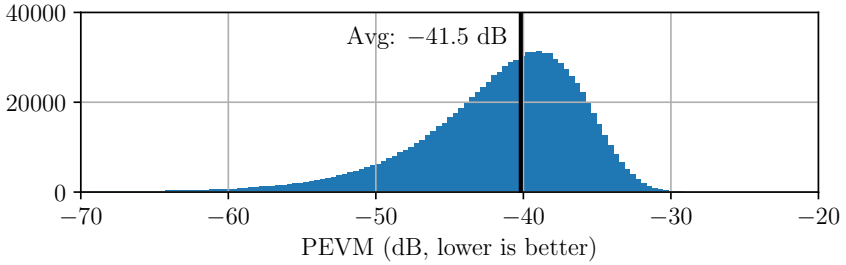
Figure 24 shows the evaluation results for the FEDR-2 network trained on synthetic data. The network was able to effectively reconstruct the distorted signals and their corresponding symbols, as demonstrated by the coverage of the predicted distortions over the true distortions in the symbol diagrams in Figures 24a and 24b. The peculiar "X"-like shape in those figures stems from the limits imposed on β and ϕ when randomly selecting IQ imbalance parameters. A histogram of PEVMs



(a) True distortions and predicted distortions, per 16-QAM Symbol.



(b) True distortions and predicted distortions, origin-centered.

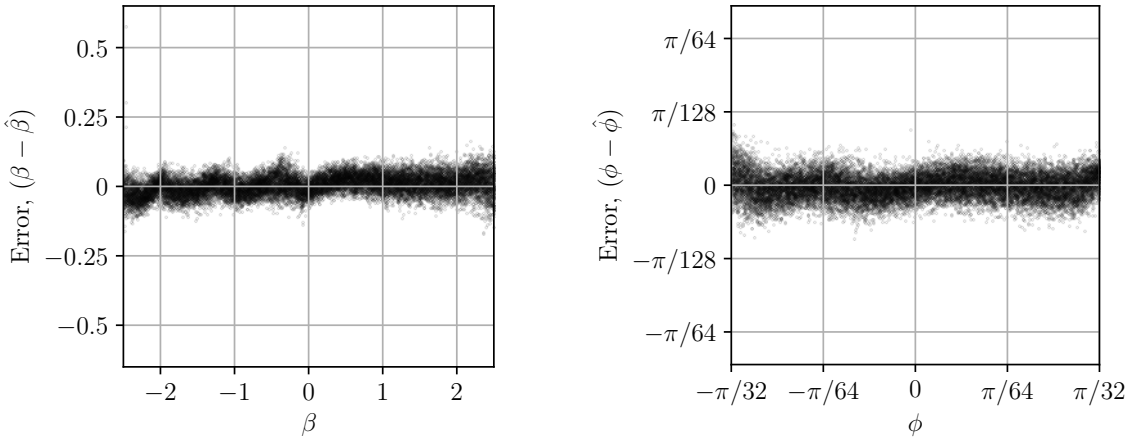


(c) Histogram of distances between predicted distortions and true distortions shown as PEVMs.

Figure 24. FEDR Training Results on Synthetic Data. Black markers in (a) and (b) represent the locations of the 16-QAM reference symbols. The technique was highly successful in reconstructing distortions, demonstrated by the coverage of predicted distortions over true distortions in (a) and (b), and by the average -41.5 dB in PEVM from (c).

is provided in Figure 24c, showing that the distances between predicted distortions and true distortions is very small, with an average of -41.5 dB. Of note, original distortions were already small, within -16 dB of the origin.

Next, $\hat{\boldsymbol{\delta}}$ were mapped back to the original β and ϕ using a Dense network comprising a single hidden layer with 32 nodes. Amplitude and phase prediction errors presented in Figure 25 demonstrate that the original parameters can be estimated from the learned parameter space, implying that FEDR learned and distilled information *about* the original distortions.



(a) Prediction error in estimated *amplitude* imbalance, yielding $\sigma \approx 0.355$ across all β considered.

(b) Prediction error estimated *phase* imbalance, yielding $\sigma \approx 0.0043$ across all ϕ considered.

Figure 25. Through a basic dense network with a single hidden layer, $\hat{\boldsymbol{\delta}}$ was successfully mapped back to the original distortion parameters, implying that β and ϕ were encoded within and recoverable from the learned parameter space.

4.5.4 Real-World Data: Dataset and Training

Next, FEDR was applied to a dataset containing real-world collections (i.e., the "Extractor Training Dataset"). This Extractor Training Dataset is a subset of the Defense Advanced Research Projects Agency (DARPA) IEEE 802.11a/g data obtained through the Radio Frequency Machine Learning Systems (RFMLS) program [87]. Communications employing 16-QAM modulation were selected for direct comparison with synthetic results. Signals were randomly selected from 2,000 devices, albeit the class information was never used for FEDR training.

4.5.4.1 Pre-Processing

All real-world data were pre-processed as described in Figure 26. Specifically, the original signals were time-aligned and frequency-corrected using the well-established Schmidl-Cox technique [88] and low-pass filtered with an 8^{th}-order Butterworth filter—this was the only pre-processing done to the distorted signals. Then, the bit-

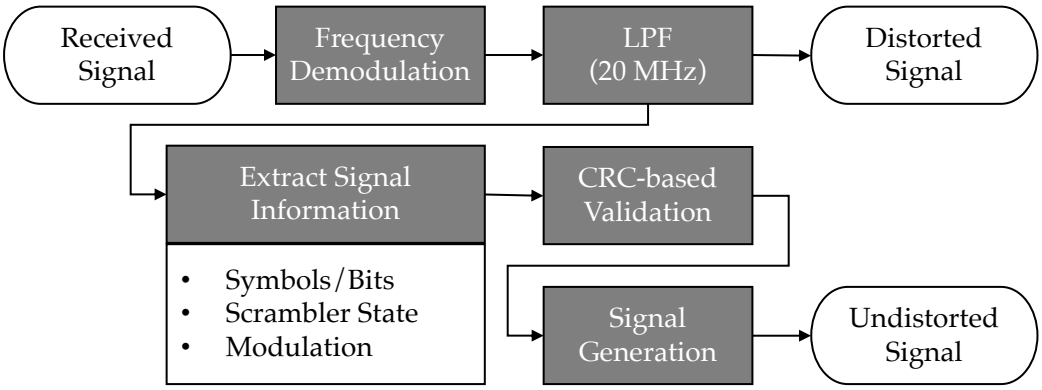


Figure 26. Overview of real-world signal pre-processing in preparation for machine learning. Parameters required for signal generation are the original symbols/bits, the initial scrambler state, and the OFDM symbol modulation type [2].

level information was estimated from the distorted signals and validated using Cyclic Redundancy Check (CRC), and an undistorted version of the original signal was generated. Each tuple of distorted-undistorted signals was sliced into OFDM symbols (similar to the structure of the synthetic data), comprising 80 complex samples apiece.

Of those 80-sample segments, 1,200,000 segments were randomly selected for training, 100,000 segments were selected for validation, and 100,000 segments were selected for evaluation.

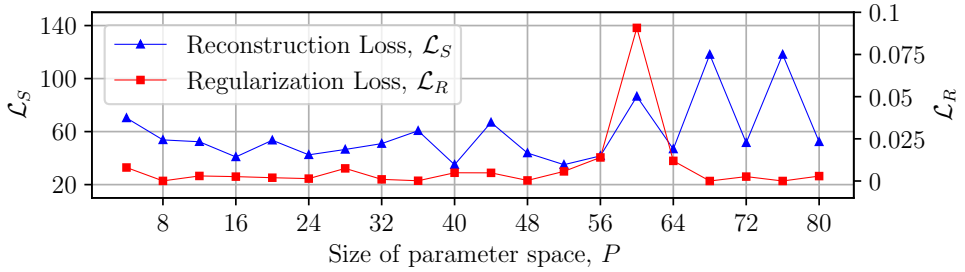
4.5.4.2 Training

Network training parameters were the same as with the synthetic data. Several choices of P were tested to balance reconstruction loss with regularization of the distortion parameter space. The following section describes how P was tuned for the dataset.

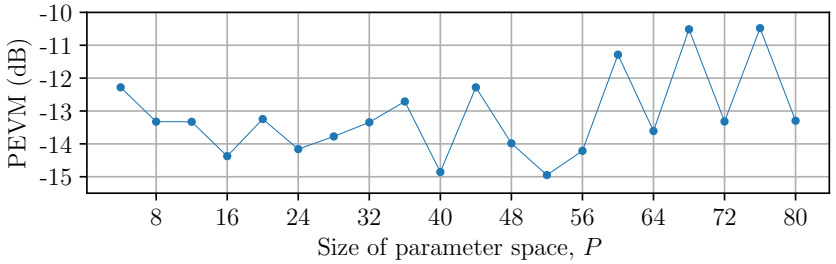
4.5.5 Real-World Data: Hyperparameter Tuning

Since the distortions were not known *a priori* in the real-world data, the parameter space could not be engineered to fit known distortions. Instead, an empirical approach was used, where loss information was collected for varying lengths of latent vector

embedding of the distortion (various values of P). Five FEDR networks were trained for each level of P, and the network with best validation data performance was selected as the best candidate for that level. Figure 27 shows how reconstruction and regularization loss vary with P for the validation subset in the Fingerprint Extraction Dataset.



(a) FEDR losses on validation data



(b) PEVM of validation data

Figure 27. Resulting losses and PEVM for varying sizes of P. Good candidates for P are 16 and 40, as both have local minimum PEVM.

The goal was to select P to be as small as possible to prevent too much content information from leaking through, while simultaneously being able to sufficiently represent the distortions in the data. In light of that goal, three choices for P that emerged from Figure 27 were 16, 40 and 52. All three had local PEVM minima in Figure 27b, implying better content representation than nearby choices of P, and all three yielded near-zero regularization loss in Figure 27a, indicating adequate implementation of the property in (12).

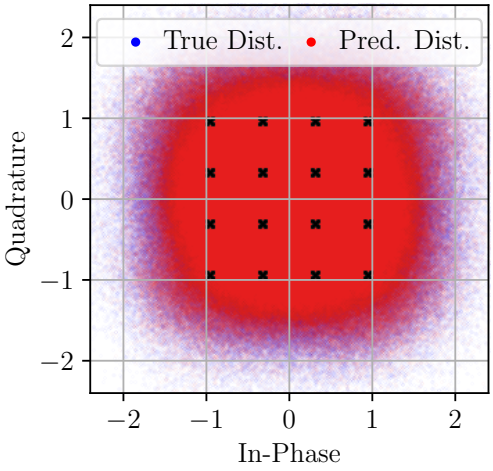
Interestingly, higher levels of P yielded worse PEVM performance. Conceivably, a larger parameter space would allow more of the original distorted signal to pass through, making it easier to reconstruct, although the regularization loss was designed to suppress that effect. Regardless, it is interesting that performance deteriorated instead of remaining steady. It could be that the increased size of the parameter space added too much sparsity to the distortion information, making it more difficult for the network to apply that information during reconstruction. A more in-depth study of hyperparameter selection is left as future work.

Ultimately, a value of $P = 16$ was selected for the rest of this work to prioritize content suppression in the relative distortion space as much as possible. Henceforth, the $P = 16$ version of the FEDR network will be referred to as FEDR-16.

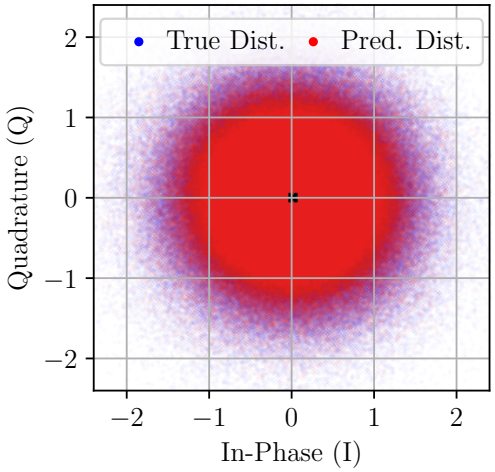
4.5.6 Real-World Data: Results

Figure 28 shows the results of training FEDR-16 with the real-world data. Notably, the distortions in the real-world data were more drastic that in the synthetic data, given the wider spread in Figure 28b versus Figure 24b. FEDR-16 was still able to account for much of the distortion, but some of the most drastic examples (i.e., the outermost visible blue cloud) were not well quantified. As a result of both the more drastic distortion and the lack of full coverage, the mean PEVM was only -14.1 dB, almost 30 dB worse than in the synthetic case. It could be that adding capacity (e.g., depth) to the Extractor and Distorter could help bolster how much distortion can be quantified, but further exploration is left as future work.

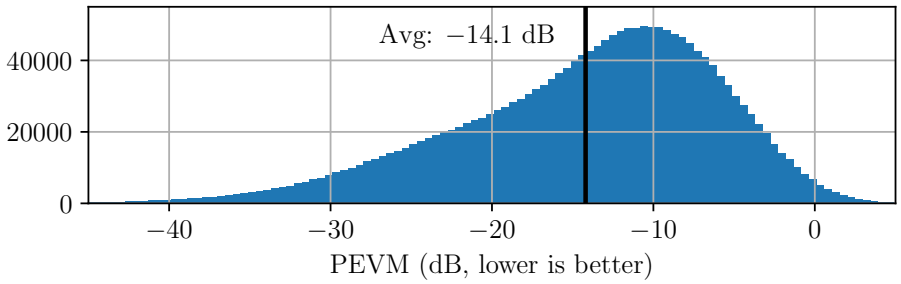
Once trained with real data, the FEDR-16 network was used to extract fingerprints from a set of classes never before seen by the network. Those fingerprints were used for RFF and compared against other techniques.



(a) True distortions and predicted distortions, per 16-QAM Symbol.



(b) True distortions and predicted distortions, origin-centered.



(c) Histogram of distances between predicted distortions and true distortions.

Figure 28. FEDR-16 Training Results on Real-World Data. Black marks in (a) and (b) represent the locations of the 16-QAM reference symbols. Because distortions in the real-world data are more drastic that in synthetic data, FEDR does not cover all distortions in (a) and (b). The resulting mean PEVM from (c) is -14.1 dB, nearly 30 dB worse than synthetic data results in Figure 24c.

4.6 Evaluation for Radio Frequency Fingerprinting

This section presents an evaluation of fingerprints generated by FEDR-16 in an RFF application. Three other fingerprinting techniques were evaluated, as well as a popular end-to-end neural network technique. When coupled with a dense network with single hidden layer, the FEDR-16 fingerprints outperformed all other evaluated techniques with MCC ranging from 0.945 (5 classes) to 0.746 (100 classes), using nearly 73% fewer training parameters than the next-best CNN.

4.6.1 RFF Dataset

Similar to the Fingerprint Extractor Training Dataset, the "RFF Dataset" is a subset of the DARPA IEEE 802.11a/g data described in Section 4.5.4. However, an entirely *new* set of 100 classes were selected for inclusion in the RFF Dataset—no data from RFF classes were used for FEDR-16 training.

The same pre-processing described in Figure 26 was performed. After pre-processing, the number of examples per class in the RFF Dataset ranged from 2020-8038, with a median class size of 4338. Performance evaluation was done with subsets of the RFF dataset containing $N_c \in \{5, 10, 15, 25, 50, 100\}$ classes, produced by selecting the N_c largest classes in the 100-class RFF Dataset.

4.6.2 Models

The following techniques were evaluated on the RFF Dataset:

1. FEDR-16 fingerprints with Dense-2048. A Dense network with a 2048-node single hidden layer was used. Like other neural network classifiers, an N_c-wide softmax layer was used as the output.

2. End-to-end CNN. Inspired by AlexNet [81], a convolutional block/dense block network structure is often used in RFF research for baseline evaluation of classification performance [84, 47, 59]. The model used here comprised four convolutional blocks, each with two convolutional layers and one max pooling layer, and one dense block with two fully connected dense layers. This type of network is typically representative of state-of-the-art performance.

3. TD-DNA fingerprints with MDA/ML classifier [13, 9, 45]. One of the more common fingerprinting techniques, the TD-DNA process divides input signals into subregions and calculates statistics on each. The statistics are used as

features for a Quadratic Discriminant Analysis (QDA) classifier after dimensionality reduction through Fisher transform. The number of subregions used was 15, yielding 144-feature fingerprints.

4. Hilbert-Huang Transform fingerprints with ResNet image classifier [17]. This fingerprint extraction technique develops a new basis for each signal using IMFs and applies Hilbert spectral analysis to extract time-frequency information. The resulting 80×80 2D matrix representations are fed into a Residual Network (ResNet) image classifier for classification. This technique is representative of the many time-frequency-based fingerprinting techniques recently proposed.

5. CB-DNA-like fingerprints with QDA classifier [14, 89]. CB-DNA calculates statistics related to how much drift is present in symbol space (i.e., constellation space). Typically, CB-DNA statistics are calculated separately for each symbol, but because OFDM was used, the symbol space was not guaranteed to be fully covered by each OFDM symbol. Instead, statistics were calculated across frequency carriers, yielding compact CB-DNA-like 9-feature fingerprints.

Models were implemented on a Kubernetes-containerized 16-core CPU with 128 GB RAM and a dedicated Quadro RTX-6000 GPU with 24 GB GB of VRAM. All neural network models used an Adam optimizer with a learning rate of 1×10^{-4}, and training was halted after 10 epochs of no improvement on the validation loss.

4.6.3 Matthews Correlation Coefficient

Performance was reported using the multi-class MCC [79]. MCC was selected over the typical per-class classification accuracy because the metric is standardized by the number of classes, enabling comparison across models with different N_c. An MCC value of 1.0 corresponds to perfect correct model performance, -1.0 indicates

perfect incorrect model performance, and 0.0 indicates performance no better than random guess. The multi-class MCC is defined as

$$MCC = \frac{cs - \sum_k p_k t_k}{\sqrt{s^2 - \sum_k (p_k)^2} \sqrt{s^2 - \sum_k (t_k)^2}} \tag{27}$$

where t_k is the number of occurrences for class $k \in \{0, 1, 2, \ldots, N_c - 1\}$, p_k is the number of times class k was predicted, c is the total number of correct predictions across all classes, and s is the total number of predictions across all classes[80].

4.6.4 Results

Classification results are shown in Table 7. Notably, the FEDR-based classifier outperformed all other techniques under evaluation conditions, to include the end-to-end CNN classifier, with MCC ranging from 0.945 ($N_c = 5$) to 0.746 ($N_c = 100$) and up to a 17% improvement over the next-best CNN. This is primarily because the FEDR-16 relative distortion parameter space distills distortion information and discards content information. Thus, the classifier using FEDR-16 fingerprints did not have the additional burden of learning to be content-agnostic.

Another advantage was that FEDR-16 could be trained offline without wasting edge device resources, and without requiring exposure to the RFF classes. This also offloads any added network complexity—it is possible to add depth to FEDR with minimal impact to the edge device running the RFF inferences. Consequently, the classifier trained by the edge device could be much smaller. Table 7 also lists the total number of training parameters for the evaluated models when trained for the $N_c = 100$ case. The Dense-2048 classifier with FEDR-16 fingerprints outperformed the baseline CNN using 72% fewer training parameters.

Table 7. RFF classification results as multi-class MCC.

No. of Classes (N_c)	FEDR-16 Dense-2048	Baseline CNN	TD-DNA MDA/ML	Hilbert-Huang ResNet	CB-DNA-like QDA
5	**0.945**	0.907	0.644	0.58	0.389
10	**0.902**	0.834	0.547	0.551	0.261
15	**0.848**	0.721	0.477	0.541	0.177
25	**0.816**	0.729	0.454	0.538	0.140
50	**0.806**	0.726	0.461	0.547	0.111
100	**0.746**	0.738	0.441	0.513	0.076
No. of Params ($N_c = 100$)	239716	884452	524349	176772	5445

4.6.5 FEDR Parameter Space

Recall that the FEDR-16 parameter space represents relative distortions between the two input signals—it does not necessarily inherently represent the optimal space for separation of classes. As an example, Figure 29 plots the first and second Principal Component Analysis (PCA) components for FEDR-16 fingerprints from the $N_c = 5$ case. Even though the Dense-2048 classifier outperformed all other techniques for this case, only minimal clustering is visible within these high variance components, e.g., Dev1 and Dev2 being the least clustered.

More likely, the latent space represents similarities in the types and amounts of relative distortions observed in the input signals and may therefore provide a common lens through which to study and express distortions themselves. By comparing fingerprints across use cases, it may be possible to describe how much distortion can be attributed to specific device components, the RF channel and other operational conditions. This exciting research potential is proposed as an area of future work.

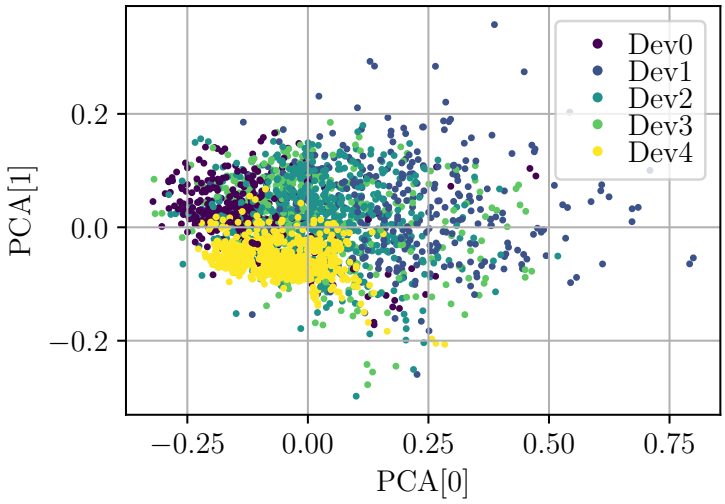


Figure 29. First and second PCA elements of FEDR-16 parameter space for 5-class RFF. Minor clustering is visible for some of the classes, but most of the hyperspace separability is nonlinear.

4.7 Conclusion and Future Work

This work presented FEDR, a neural network based RFF fingerprint extraction technique. Leveraging a constrained neural network architecture and a latent space regularization loss, the network was trained to learn information about signal distortions, the presence of which can be used to identify specific emitters. When coupled with a dense network with a single 2048-node hidden layer, FEDR fingerprinting achieved best performance across all values of N_c with Matthews Correlation Coefficient ranging from 0.945 ($N_c = 5$) to 0.746 ($N_c = 100$), using nearly 73% fewer training parameters than the next-best CNN.

FEDR held at least two clear advantages over previous techniques. First, by using ML to quantify signal distortions, it also inherently separated content information from distortion information. As a result, the learned relative distortion space was more descriptive and better targeted than other fingerprint extraction techniques. Second, the FEDR fingerprint extractor was trained fully offline, without exposure to any of the RFF classes. Thus, any future extensions to model complexity or depth can

be offloaded to a resource-intensive machine before deployment to the edge device, a key advantage over other edge-based CNNs.

Although $P = 16$ was sufficient for the presented use case, further exploration of the effects of the size of the parameter space remain an area of future work. For this work, the space was strategically kept small to help prevent content information from leaking into the Distorter, but the regularization loss may generally do well enough that any added dimensionality only increases fingerprint sparsity. It would also be interesting to study whether adding depth to the Extractor and Distorter would help account for more distortion information and whether that would necessitate a larger parameter space.

Another area of future work is in the application of FEDR for fingerprint attribution to particular types of distortion. FEDR can provide a common lens through which distortions can be studied, making it possible to compare fingerprints across different conditions, like devices, RF channels, and operational environments. Further analysis might yield insight into whether portions of the fingerprint can be attributed to specific conditions and whether those conditions can be identified from the fingerprints. This in-depth knowledge of the operational environment is a capability that the spectrum-aware radios of the future will undoubtedly need.

V. Conclusions and Future Work

This work presented two studies designed to learn more robust fingerprints for use in RFF. Learning robust fingerprints requires studying whether and how fingerprints vary across operational conditions. This section presents conclusions of this work with respect to the original research questions presented in Chapter I.

S1-Q1. How does frequency channel affect RF Fingerprints?

Though emitter components are designed to operate within a specified frequency range, distortions are known to vary across that range. Thus, when typical single-channel RFF models were evaluated against a multi-channel dataset, per-channel performance ranged from almost perfect (MCC > 0.9) to random guess (MCC < 0.05). These results indicate that single-channel models should not be applied to other frequency channels and more generally, that frequency can have a significant detrimental effect on typical RFF techniques.

S1-Q2. Does frequency agnostic information exist within the fingerprints? And can that information be leveraged in frequency-agnostic RFF models?

RFF training data were augmented with signals stemming from multiple frequency channels. Multi-channel models generalized better across *all* channels, achieving adequate performance even when just a small subset of channels were included (i.e., 4-of-15). For the worst-performing model, the average MCC improved from 0.657 in the single-channel configuration to 0.957 in the 4-channel configuration, implying bolstered performance across all channels. This finding suggests that there exists frequency-agnostic information within the fingerprints that can be learned by the models and used for RFF.

S2-Q1. Can a network be trained to learn about the differences between a distorted signal and its corresponding undistorted signal?

RFF fingerprints are measurements of the signal distortions left behind by emitter components. A neural network-based technique called FEDR was presented, which learned to express the difference between distorted and undistorted signals through a relative distortion parameter space. When the technique was applied to synthetic IQ-imbalanced data, the distortion parameter space was shown to represent the original IQ amplitude and phase imbalance parameters. This result indicates that FEDR learns about and quantifies the differences between distorted and undistorted signals.

S2-Q2. Do those differences provide enough information for the discernment of specific emitters?

FEDR was trained for fingerprint extraction on a real-world dataset. When coupled with a dense network with a single 2048-node hidden layer, FEDR fingerprinting achieved best performance across all values of N_c with Matthews Correlation Coefficient ranging from 0.945 ($N_c = 5$) to 0.746 ($N_c = 100$), using nearly 73% fewer training parameters than the next-best CNN. Thus, FEDR fingerprints provided enough information for the discernment of specific emitters. Moreover, FEDR outperformed all other evaluated techniques, likely because the classifier network was not burdened by content information – it could focus explicitly on the distortions.

5.1 Future Work

Several interesting areas of future work arose from the two studies and from the work as a whole:

- **Effects of channel bandwidth to multi-channel performance.** Each additional channel included in the multi-channel training set exposed the RFF models to an additional 5 MHz "chunk" of the spectrum. This additional exposure enabled models to learn frequency-agnostic variability, making them generalize better across all frequencies. Study I found that a total of 20 MHz (i.e., four WirelessHART channels) of exposure spread throughout the 80 MHz band was enough to achieve frequency channel-agnostic performance. Other common communications protocols employ channels of different sizes; e.g., Bluetooth channels are 1 MHz wide, and typical WiFi channels are up to 20 MHz wide. More Bluetooth channels and/or fewer WiFi channels might be needed to achieve generalizable multi-channel model performance because of the difference in channel bandwidth. Future work should explore how much exposure to the spectrum is required for effective frequency-agnostic performance.

- **Edge device limitations for practical RFF applications.** RFF must necessarily take place after signal reception, so the most practical location to do this is at the edge device itself. However, training complex classifiers like deep neural networks can be resource intensive, and typical IoT receivers may not be well-equipped due to their small form factors and strict power constraints. FEDR helps offload some of this training, and other researchers are actively finding ways to prune complex networks to reduce the resource overhead [6], but further work remains in this area to enable more practical model deployment. Novel techniques might consider other ways to compress models to reduce computational requirements and inference times, or tiered classifier structures, where complex classifiers are only used after simpler classifiers fail.

- **Effects of the size of the FEDR parameter space.** For this work, the latent space of distortion embedding was strategically kept small (i.e., $P = 16$) to help prevent content information from leaking into the Distorter, but the regularization loss may generally do well enough that any added dimensionality only increases fingerprint sparsity. It would be interesting to study whether adding depth to the Extractor and Distorter would help account for more distortion information and whether that would necessitate a larger parameter space.

- **FEDR fingerprint attribution to operational conditions.** FEDR can provide a common lens through which distortions can be studied, making it possible to compare fingerprints across different conditions, like devices, RF channels, and operational environments. Further analysis might yield insight into whether portions of the fingerprint can be attributed to specific conditions and whether those conditions can be identified from the fingerprints. For instance, the effects of frequency channel could be studied under the lens of FEDR, which could provide further insight into which parts of the fingerprints are frequency-agnostic.

- **FEDR generalization across communications protocols.** There are several image classifiers that have been trained to recognize a large corpus of images, making them useful starting points for other image recongition tasks (e.g., VGG-19 [90]). A FEDR network should be trained in a similar fashion, across many protocols, modulation schemes, and operational conditions, for exposure to myriad types of distortions. Then, a trained version of FEDR could be made publicly available for the purpose of standardizing the study of fingerprints across operational conditions, as well as for providing a baseline for future advancements in fingerprinting research.

RFF systems continue to offer an attractive out-of-band method for wireless device authentication, especially as integrated components in the defense-in-depth security paradigm. As modern communications grow in operational complexity, the variability of signal distortions across operational conditions must be considered to achieve the best and most generalizable performance. This work presented novel techniques for studying and understanding that variability and for developing the robust RFF systems of the future.

Appendix A. Physical Layer in Wireless Communications Systems

Although the information in digital communications is represented as ones and zeros, its transmission relies on real analog signals sent over communications channels. Communications protocols are targeted towards specific layers of the Open Systems Interconnection (OSI) model, depicted in Figure A.1, which delineates protocol responsibilities and enables implementation flexibility through an isolated service-based approach. Each layer provides a service to the layer above it, which it can do without knowledge of implementation details about the layer above or below it.

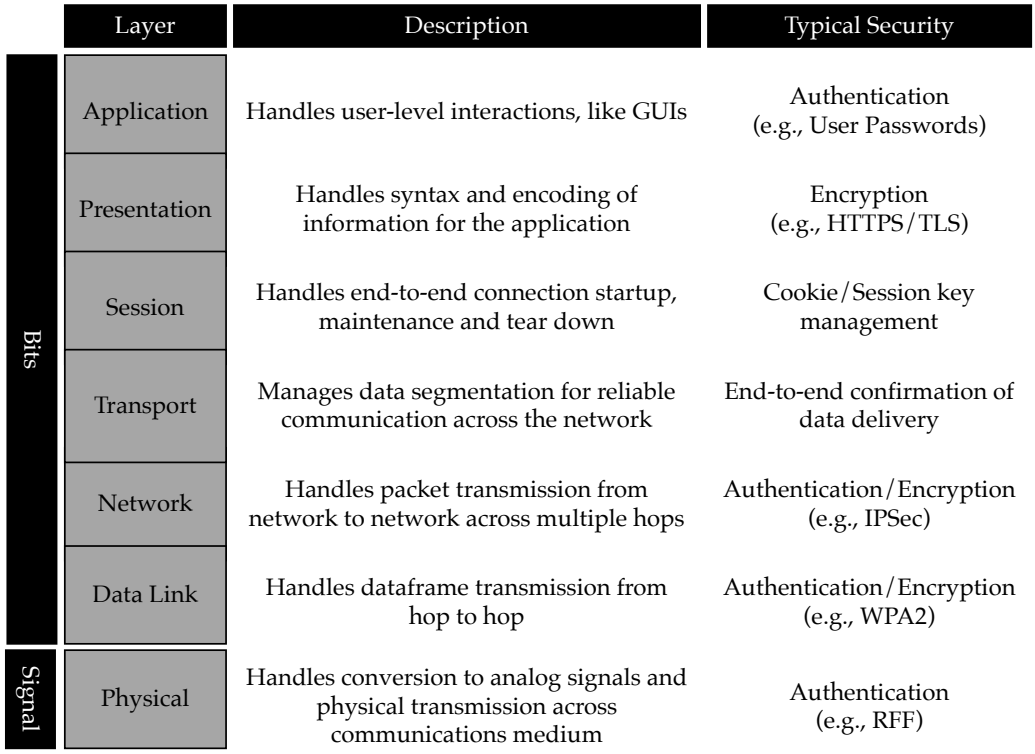


Figure A.1. OSI Model depicting types of security techniques available at each layer. Most security techniques rely on bit-level cryptographic algorithms. RFF is a statistical technique that operates directly on signal information.

Most layers operate on bit-level information, often encapsulating information from the layer below. The Physical (PHY) Layer is where that bit-level information is

converted to real signals for transmission through a communications medium (e.g., the RF spectrum in wireless communications). Layers can also provide security services, like authentication and encryption – common security protocols are also listed in Figure A.1 next to their corresponding layers. RFF is often proposed as a PHY-layer security service, using signals directly to perform authentication.

In wireless systems, the PHY layer generates signals that can be transmitted over-the-air, through RF emissions. Modulation schemes describe the process for generating those signals, and they are often engineered to achieve specific design requirements, like data rates, power constraints, and bandwidth utilization. Two PHY layer protocols and corresponding modulation schemes are used in this work to generate synthetic signals, or portions of signals. Offset-Quadrature Phase Shift Keying (O-QPSK) (from IEEE 802.15.4 [21]) and OFDM with 16-Symbol Quadrature Amplitude Modulation (QAM) (from IEEE 802.11a/g [2]) are described in detail in the following sections. Both protocols output complex signals with in-phase and quadrature components. Components are individually mixed with orthogonal carriers, summed, amplified, and finally transmitted.

A.1 Offset-Quadrature Phase Shift Keying (O-QPSK)

This section describes the half-sine O-QPSK modulation scheme from IEEE 802.15.4 [21]. Link layer bits are first divided into four-bit segments, which are mapped to 16-bit Pseudo-Noise (PN) chip sequences as a form of spreading. By effectively spreading four data bits across 16 chip bits, the system data rate is reduced for an improvement in error tolerance. Next, chips are assigned to in-phase and quadrature components in an alternating fashion, i.e., every other bit is assigned to the same component. Each chip c is then mapped to a half-sine pulse, defined from

$0 \leq t \leq 2T_c$ as:

$$p_k(t) = \begin{cases} \sin(\pi \frac{t}{2T_c}), \text{ if } c = 1 \\ -\sin(\pi \frac{t}{2T_c}), \text{ if } c = 0. \end{cases} \tag{28}$$

Chips within each component are transmitted back-to-back, but the quadrature component is delayed (or "offset") by T_c of the in-phase component. The chip duration $2T_c$ is 8 μs when operating in the 2.4 GHz Industrial, Scientific and Medical (ISM) band, but because of the delay in quadrature, a new chip is effectively transmitted every 4 μs. A diagram showing half-sine pulses and delayed quadrature component is presented in Figure A.2.

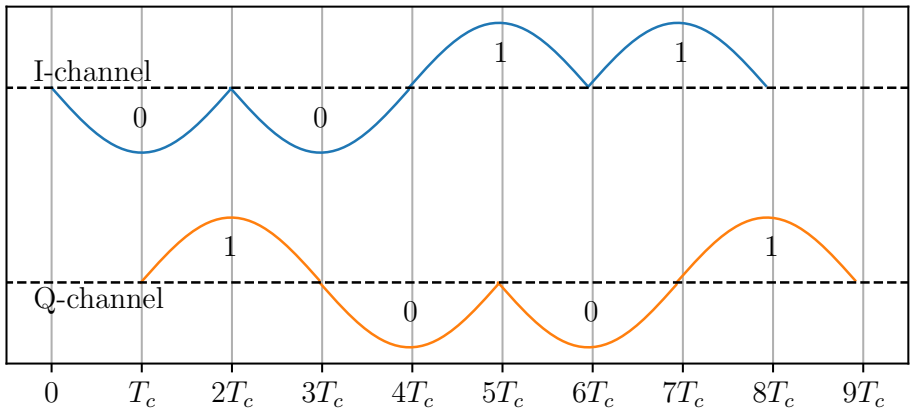


Figure A.2. Example of O-QPSK half-sine pulse shaping with I- and Q-channels for bit sequence [0,1,0,0,1,0,1,1]. Note that bits are transmitted on alternating channels, where the quadrature component is delayed by T_c.

The receiver must synchronize its internal mixer in both time and phase to recover the transmitted chips and subsequently the original data bits. One method for synchronization is through a standardized communications preamble. IEEE 802.15.4 requires a preamble of 32 consecutive zero data bits, or eight "0" symbols, which map to eight consecutive chip sequences of "11011001110000110101001000101110." Each "0" symbol is transmitted using the signals shown in Figure A.3. Preambles are used

in Chapter III for timing and phase correction, and serve as the region of interest used for RFF.

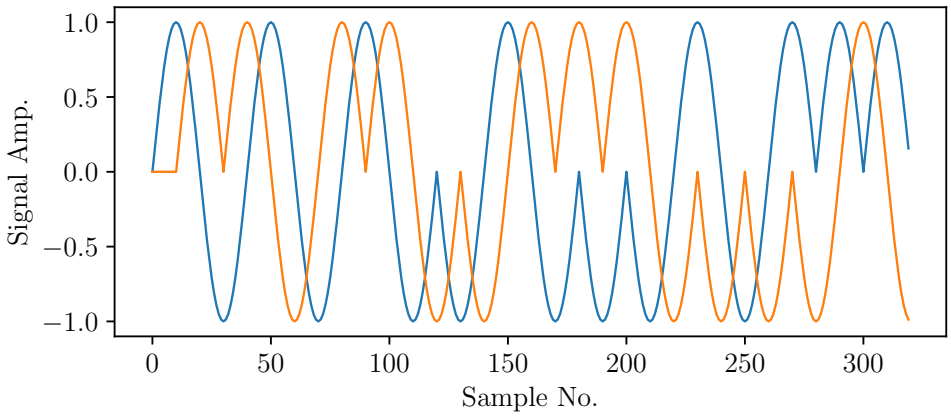


Figure A.3. First "0" symbol of an O-QPSK preamble.

A.2 Orthogonal Frequency Division Multiplexing (OFDM) with 16-Symbol Quadrature Amplitude Modulation (16-QAM)

A common technique for efficient utilization of the communications spectrum is Orthogonal Frequency Division Multiplexing (OFDM). With OFDM, multiple communications symbols are transmitted simultaneously on a set of orthogonal subcarriers, where each symbol is mapped to a single subcarrier. Orthogonality is achieved through careful spreading of the subcarriers across the available bandwidth. Because the time-domain burst representations of the subcarriers are finite in time, the frequency domain representation of each subcarrier is a sinc function, where the width of the main lobe in frequency is relative to the length of the burst in time. Sinc functions can be spread in frequency such that the peak of one sinc function aligns with the null of another, as shown in Figure A.4, effectively achieving orthogonality between them.

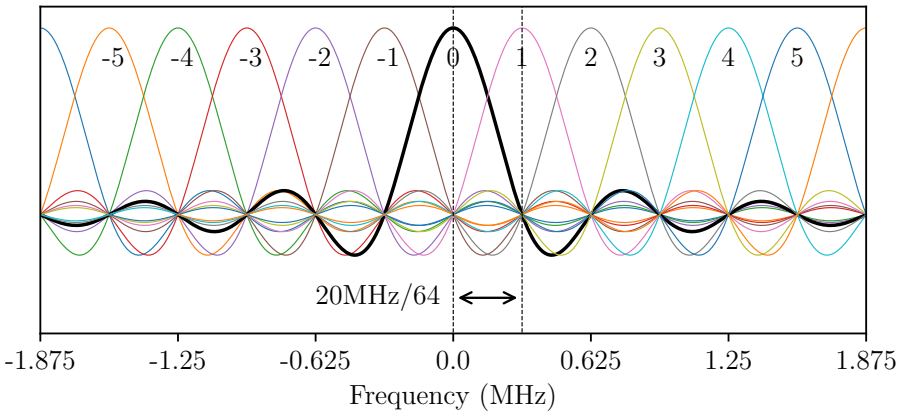


Figure A.4. OFDM subcarriers in frequency domain. Subcarriers -5 to 5 are shown, with 0.3125 MHz spacing between them, enabled by a 3.4 μs burst length.

In IEEE 802.11a/g, 64 subcarriers are spread across the available bandwidth of 20 MHz at intervals of 312.5 kHz, necessitating a time-domain burst of 3.4 μs. Of those 64 carriers, 48 are used for data, four are used for pilot tones, and the rest are left unused for interference avoidance – specific relative frequencies and purpose for each subcarrier are listed in Table A.1.

Table A.1. OFDM subcarrier frequencies (MHz) and usage ((U)nused/(D)ata/(P)ilot)

Idx.	Freq.	Use	Idx.	Freq.	Use	Idx.	Freq.	Use	Idx.	Freq.	Use
-32	-10.0000	U	-16	-5.0000	D	0	0.0000	U	16	5.0000	D
-31	-9.6875	U	-15	-4.6875	D	1	0.3125	D	17	5.3125	D
-30	-9.3750	U	-14	-4.3750	D	2	0.6250	D	18	5.6250	D
-29	-9.0625	U	-13	-4.0625	D	3	0.9375	D	19	5.9375	D
-28	-8.7500	U	-12	-3.7500	D	4	1.2500	D	20	6.2500	D
-27	-8.4375	U	-11	-3.4375	D	5	1.5625	D	21	6.5625	P
-26	-8.1250	D	-10	-3.1250	D	6	1.8750	D	22	6.8750	D
-25	-7.8125	D	-9	-2.8125	D	7	2.1875	P	23	7.1875	D
-24	-7.5000	D	-8	-2.5000	D	8	2.5000	D	24	7.5000	D
-23	-7.1875	D	-7	-2.1875	P	9	2.8125	D	25	7.8125	D
-22	-6.8750	D	-6	-1.8750	D	10	3.1250	D	26	8.1250	D
-21	-6.5625	P	-5	-1.5625	D	11	3.4375	D	27	8.4375	U
-20	-6.2500	D	-4	-1.2500	D	12	3.7500	D	28	8.7500	U
-19	-5.9375	D	-3	-0.9375	D	13	4.0625	D	29	9.0625	U
-18	-5.6250	D	-2	-0.6250	D	14	4.3750	D	30	9.3750	U
-17	-5.3125	D	-1	-0.3125	D	15	4.6875	D	31	9.6875	U

The following process was used to generate time-domain OFDM signals in this work:

1. Initialize vector of 64 complex numbers, all set to $0 + j0$
2. Assign 48 complex communications symbols (e.g., 16-QAM symbols) to data-carrying subcarriers in numerical order, starting with subcarrier -26
3. Assign pilot tones to the four pilot subcarriers, per IEEE 802.11a/g
4. Take Inverse Fast Fourier Transform (IFFT) of the 64-number vector – the resulting signal will be 3.4 μs long
5. Prepend a copy the last 0.6 μs of the time-domain signal to create a Cyclic Prefix

The output signal is 4.0 μs long and represents a single OFDM symbol. A Cyclic Prefix is added to help reduce inter-symbol interference across OFDM symbols by suppressing the effects of multi-path propagation experienced during transmission.

Mapping data bits to the 48 complex communications symbols requires a few additional steps. As with the chip spreading in O-QPSK, a technique is used to encode data bits across multiple coded bits, where coding rates include 1/2 (i.e., 2 coded bits for 1 data bit), 2/3, and 3/4. A combination of coding rate and modulation scheme dictates the communications data rate. For this work, the data rate was 24 Mbps, which was achieved through a 1/2 coding rate and 16-QAM modulation scheme. The 16-QAM modulation scheme maps 16 symbols to constellation space, as shown in Figure A.5.

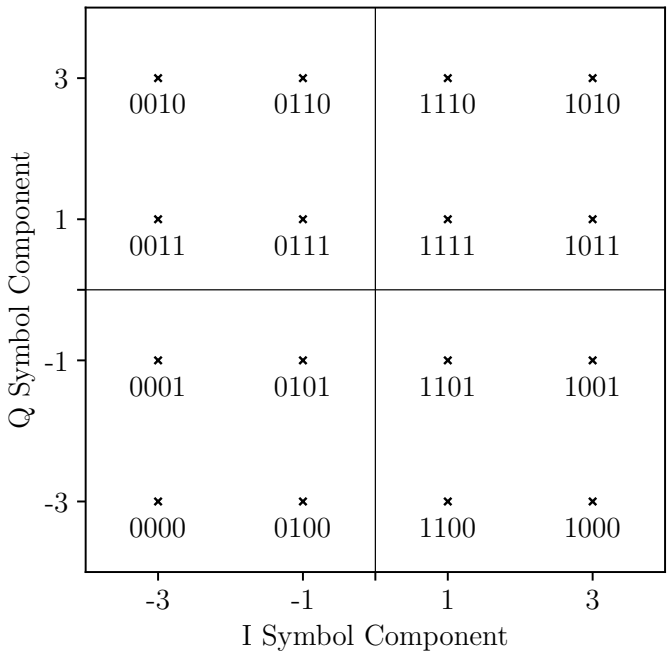
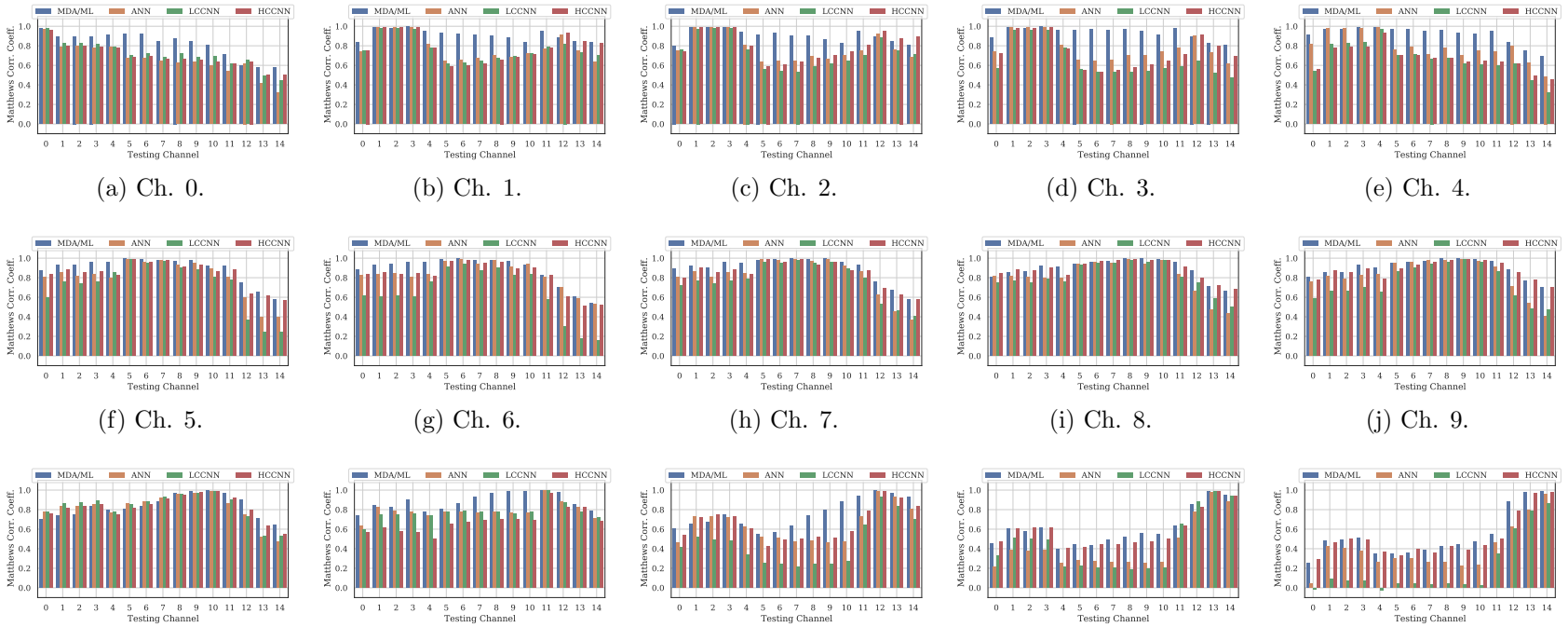


Figure A.5. 16-QAM symbols with corresponding bit sequences.

Appendix B. Additional Results from Study I

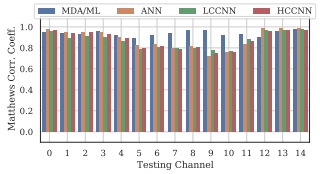


(a) Ch. 0. (b) Ch. 1. (c) Ch. 2. (d) Ch. 3. (e) Ch. 4.

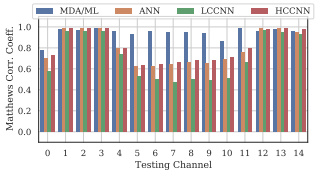
(f) Ch. 5. (g) Ch. 6. (h) Ch. 7. (i) Ch. 8. (j) Ch. 9.

(k) Ch. 10. (l) Ch. 11. (m) Ch. 12. (n) Ch. 13. (o) Ch. 14.

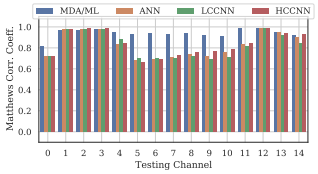
Figure B.1. Additional data from Study I, Experiment A. Single channel models were evaluated on multi-channel data. Results are shown in MCC.



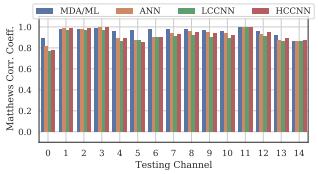
(a) Chs. 0/14.



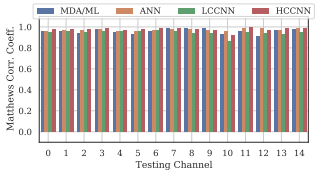
(b) Chs. 1/13.



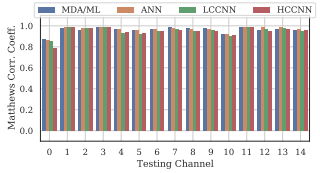
(c) Chs. 2/12.



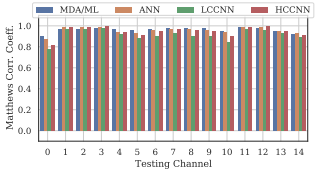
(d) Chs. 3/11.



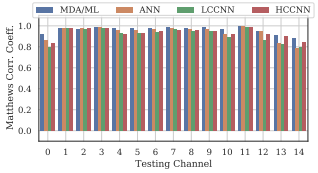
(e) Chs. 0/7/14.



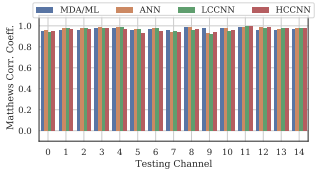
(f) Chs. 1/7/13.



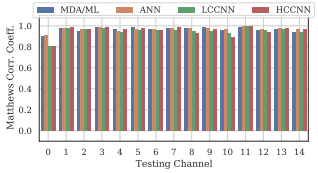
(g) Chs. 2/7/12.



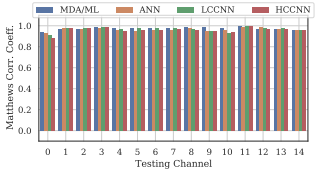
(h) Chs. 3/7/11.



(i) Chs. 0/4/10/14.



(j) Chs. 1/5/9/13.



(k) Chs. 0-14.

Figure B.2. Additional data from Study I, Experiment B. Multi-channel models were evaluated on multi-channel data. Results are shown in MCC.

Appendix C. On-the-fly Burst Detection and Collection

A typical technique for signal collection is to capture data continuously with a Software-Defined Radio (SDR), store it all in hard disk, and offload search for signal bursts to MATLAB or other signal processing tools. The key downside with this approach is that computer hard drive space fills up very quickly with non-burst data because devices communicate only a fraction of the time. One technique to help limit hard drive space usage is to reduce the bit resolution of the SDR, e.g., by reducing from 12-bit numbers to 8-bit numbers, but this potentially discards signal information useful for RFF. Another technique is to reduce the sample rate, but the consequent reduction in collection bandwidth limits the types of signals can be captured. For Chapter III, the collection goal was to capture wideband data, namely the full 2.4 GHz ISM band, which necessitated a 100 MHz collection bandwidth. To avoid bloating storage and to maximize the information available for RFF, an on-the-fly burst detection technique was implemented in GNU Radio Companion. Figure C.1 shows the finished process, which can be summarized as

1. Split the input signal into a delayed path and non-delayed path

2. On the non-delayed path, perform burst detection by measuring signal power over a moving window

3. When a burst is detected (i.e., power threshold is surpassed), tag the signal on the delayed path with "burst=True"

4. Repeat the process to detect when the power drops below a threshold, and tag the signal with "burst=False"

5. Sink the tagged signal into a Tagged File Sink, which will only record and store the samples between "burst=True" and "burst=False"

Delays can be adjusted to attach additional leading or trailing samples to the recorded burst. Naturally, burst thresholds should be configured to align with the expected signal power, and all bursts should be separately validated to discard any false positive readings. With this configuration, each burst will be individually stored on the hard drive, making it easier to process. In the end, the technique presented here enables efficient storage use, especially during wideband data collection.

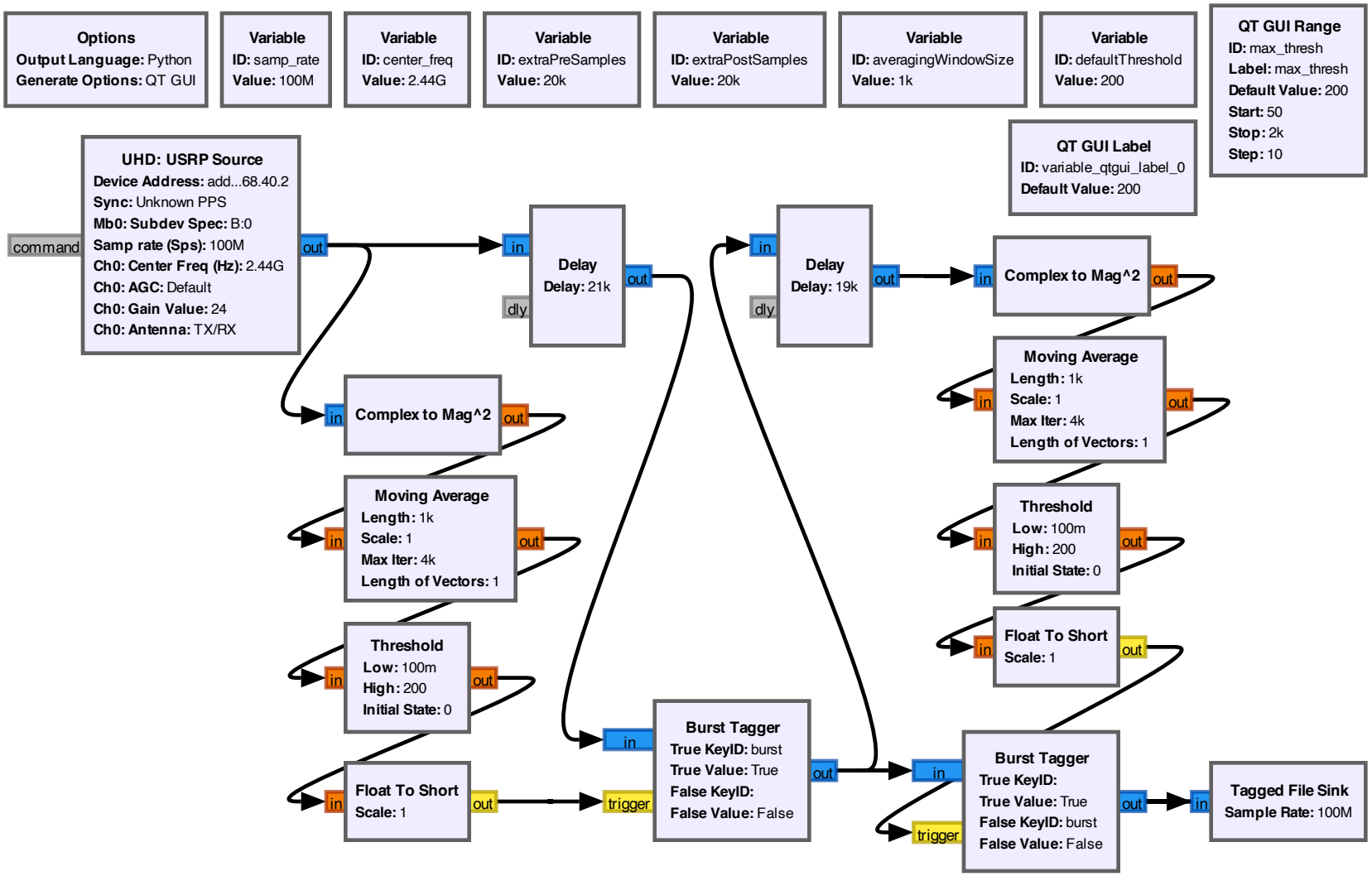


Figure C.1. GNU Radio Companion visualization of on-the-fly data collection and burst detection technique. The technique works by delaying the primary signal path to detect bursts, tagging a burst start and burst end position, and saving the information between the tags to hard drive.

Bibliography

1. United States Air Force, "U. S. Air Force Science and Technology Strategy: Strengthening USAF Science and Technology for 2030 and Beyond," 2019.

2. IEEE, "IEEE 802.11-2020: Part 11, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," *IEEE Std 802.11-2020*, pp. 1–4379, 2020.

3. C. Jacomme and S. Kremer, "An Extensive Formal Analysis of Multi-factor Authentication Protocols," *ACM Transactions on Privacy and Security*, vol. 24, no. 2, pp. 1–34, 2021.

4. C. K. Dubendorfer, B. W. Ramsey, and M. A. Temple, "An RF-DNA Verification Process for ZigBee Networks," in *IEEE Military Communications Conference (MILCOM)*. IEEE, 2012, pp. 1–6.

5. M. D. Williams, M. A. Temple, and D. R. Reising, "Augmenting Bit-Level Network Security Using Physical Layer RF-DNA Fingerprinting," in *IEEE Global Telecommunications Conference (GLOBECOM)*. IEEE, 2010, pp. 1–6.

6. T. Jian, Y. Gong, Z. Zhan, R. Shi, N. Soltani, Z. Wang, J. G. Dy, K. Chowdhury, Y. Wang, and S. Ioannidis, "Radio Frequency Fingerprinting on the Edge," *IEEE Transactions on Mobile Computing*, pp. 1–16, 2021.

7. S. D. Andrews, "Extensions to Radio Frequency Fingerprinting," Ph.D. dissertation, Virginia Polytechnic Institute and State University, 2019.

8. L. J. Wong, W. H. Clark, B. Flowers, R. M. Buehrer, A. J. Michaels, and W. C. Headley, "The RFML Ecosystem: A Look at the Unique Challenges of

Applying Deep Learning to Radio Frequency Applications," *arXiv*, pp. 1–19, 2020. [Online]. Available: https://arxiv.org/pdf/2010.00432.pdf

9. C. M. Rondeau, M. A. Temple, and C. Schubert Kabban, "TD-DNA Feature Selection for Discriminating WirelessHART IIoT Devices," in *53rd Hawaii International Conference on System Sciences (HICSS)*, 2020, pp. 6387–6396.

10. T. J. Carbino, M. A. Temple, and J. Lopez, "Conditional Constellation Based-Distinct Native Attribute (CB-DNA) Fingerprinting for Network Device Authentication," in *IEEE International Conference on Communications (ICC)*. IEEE, 2016, pp. 1–6.

11. N. Soltanieh, Y. Norouzi, Y. Yang, and N. Chandra Kamakar, "A Review of Radio Frequency Fingerprinting Techniques," *IEEE Journal of Radio Frequency Identification*, vol. 4, no. 3, pp. 222–233, 2020.

12. J. Zhang, R. Woods, M. Sandell, M. Valkama, A. Marshall, and J. Cavallaro, "Radio Frequency Fingerprint Identification for Narrowband Systems, Modelling and Classification," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 3974–3987, 2021.

13. J. A. Gutierrez del Arroyo, B. J. Borghetti, and M. A. Temple, "Considerations for Radio Frequency Fingerprinting across Multiple Frequency Channels," *Sensors*, vol. 22, no. 6, 2022. [Online]. Available: https://www.mdpi.com/1424-8220/22/6/2111

14. C. M. Rondeau, M. Temple, and J. A. Betances, "Dimensional Reduction Analysis for Constellation-Based DNA Fingerprinting to Improve Industrial IoT Wireless Security," in *52nd Hawaii International Conference on System Sciences (HICSS)*, 2019, pp. 7126–7135.

15. W. H. Mims, M. A. Temple, and R. F. Mills, "Active 2D-DNA Fingerprinting of WirelessHART Adapters to Ensure Operational Integrity in Industrial Systems," *Sensors*, vol. 22, no. 13, 2022. [Online]. Available: https://www.mdpi.com/1424-8220/22/13/4906

16. D. R. Reising and M. A. Temple, "WiMAX Mobile Subscriber Verification using Gabor-based RF-DNA Fingerprints," in *IEEE International Conference on Communications (ICC)*. IEEE, 2012, pp. 1005–1010.

17. Y. Pan, S. Yang, H. Peng, T. Li, and W. Wang, "Specific Emitter Identification Based on Deep Residual Networks," *IEEE Access*, vol. 7, pp. 54 425–54 434, 2019.

18. Y. Liu, J. Wang, J. Li, S. Niu, and H. Song, "Machine Learning for the Detection and Identification of Internet of Things Devices: A Survey," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 298–320, 2022.

19. Connectivity Standards Alliance, "ZigBee Specification," pp. 1–565, 2015.

20. FieldComm Group, "HART Protocol Technical Specification: Part I," *HART Protocol Revision 7.6*, pp. 1–99, 2016.

21. IEEE, "IEEE 802.15.4-211: IEEE Standard for Low-Rate Wireless Networks," *IEEE Std 802.15.4-2011*, pp. 1–709, 2016.

22. Bluetooth SIG, "Bluetooth Core Specification v5.3," *Bluetooth v5.3*, pp. 1–3085, 2021.

23. G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning*, ser. Springer Texts in Statistics. New York, NY: Springer Science+Business Media, 2017. [Online]. Available: http://link.springer.com/10.1007/978-1-4614-7138-7

24. Scikit Learn Developers, "Supervised Learning," https://scikit-learn.org/stable/supervised_learning.html#supervised-learning, accessed: 2022-07-09.

25. M. Kutner, C. Nachtsheim, J. Neter, and W. Li, *Applied Linear Statistical Models*, 5th ed. McGraw-Hill, 2005.

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

27. J. Brownlee, "A gentle introduction to relu," https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/, accessed: 2022-08-03.

28. M. Du, N. Liu, and X. Hu, "Techniques for Interpretable Machine Learning," in *Communications of the ACM (CACM)*, 2018, pp. 1–9. [Online]. Available: http://arxiv.org/abs/1808.00033

29. K. Koech, "Cross-Entropy Loss Function," https://towardsdatascience.com/cross-entropy-loss-function-f38c4ec8643e, accessed: 2022-08-02.

30. M. Kulin, T. Kazaz, I. Moerman, and E. De Poorter, "End-to-End Learning from Spectrum Data: A Deep Learning Approach for Wireless Signal Identification in Spectrum Monitoring Applications," *IEEE Access*, vol. 6, pp. 18 484–18 501, 2018.

31. E. Moser, M. K. Moran, E. Hillen, D. Li, and Z. Wu, "Automatic Modulation Classification via Instantaneous Features," in *IEEE National Aerospace Electronics Conference (NAECON)*. IEEE, 2015, pp. 218–223.

32. B. Tang, Y. Tu, Z. Zhang, and Y. Lin, "Digital Signal Modulation Classification With Data Augmentation Using Generative Adversarial Nets in Cognitive Radio Networks," *IEEE Access*, vol. 6, pp. 15 713–15 722, 2018.

33. T. J. O'Shea, T. Roy, N. West, and B. C. Hilburn, "Physical Layer Communications System Design Over-the-Air Using Adversarial Networks," in *26th European Signal Processing Conference (EUSIPCO)*, 2018, pp. 529–532.

34. H. Jiang and L. Dai, "Residual-Aided End-to-End Learning of Communication System Without Known Channel," *IEEE Transactions on Cognitive Communications and Networking*, vol. 8, no. 2, pp. 631–641, 2021.

35. P. Chen, Y. Guo, G. Li, and J. Wan, "Adversarial Shared-Private Networks for Specific Emitter Identification," *Electronics Letters*, vol. 56, no. 6, pp. 296–299, 2020.

36. L. Wu, Y. Zhao, M. Feng, F. Y. Abdalla, and H. Ullah, "Specific Emitter Identification Using IMF-DNA with a Joint Feature Selection Algorithm," *Electronics*, vol. 8, pp. 1–16, 2019.

37. M. Zhu, Z. Feng, X. Zhou, R. Xiao, Y. Qi, and X. Zhang, "Specific Emitter Identification Based on Synchrosqueezing Transform for Civil Radar," *Electronics*, vol. 9, no. 658, pp. 1–16, 2020.

38. X. Wang, G. Huang, C. Ma, W. Tian, and J. Gao, "Convolutional Neural Network Applied to Specific Emitter Identification based on Pulse Waveform Images," *IET Radar, Sonar and Navigation*, vol. 14, no. 5, pp. 728–735, 2020.

39. M. Kose, S. Tascioglu, and Z. Telatar, "RF Fingerprinting of IoT Devices Based on Transient Energy Spectrum," *IEEE Access*, vol. 7, pp. 18 715–18 726, 2019.

40. M. Zhu, Z. Feng, and X. Zhou, "A Novel Data-Driven Specific Emitter Identification Feature Based on Machine Cognition," *Electronics*, vol. 9, no. 1308, pp. 1–18, 2020.

41. K. Sa, D. Lang, C. Wang, and Y. Bai, "Specific Emitter Identification Techniques for the Internet of Things," *IEEE Access*, vol. 8, pp. 1644–1652, 2020.

42. H. J. Patel, M. A. Temple, and R. O. Baldwin, "Improving ZigBee Device Network Authentication Using Ensemble Decision Tree Classifiers with Radio Frequency Distinct Native Attribute Fingerprinting," *IEEE Transactions on Reliability*, vol. 64, no. 1, pp. 221–233, 2015.

43. T. J. Bihl, K. W. Bauer, and M. A. Temple, "Feature Selection for RF Fingerprinting with Multiple Discriminant Analysis and Using ZigBee Device Emissions," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 8, pp. 1862–1874, 2016.

44. C. M. Rondeau, J. A. Betances, and M. A. Temple, "Securing ZigBee Commercial Communications Using Constellation Based Distinct Native Attribute Fingerprinting," *Security and Communication Networks*, vol. 2018, pp. 1–14, 2018.

45. J. Lopez, N. C. Liefer, C. R. Busho, and M. A. Temple, "Enhancing Critical Infrastructure and Key Resources (CIKR) Level-0 Physical Process Security Using Field Device Distinct Native Attribute Features," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 5, pp. 1215–1229, 2018.

46. T. J. Bihl, T. J. Paciencia, K. W. Bauer, and M. A. Temple, "Cyber-Physical Security with RF Fingerprint Classification through Distance Measure Extensions of Generalized Relevance Learning Vector Quantization," *Security and Communication Networks*, vol. 2020, pp. 1–12, 2020.

47. G. Shen, J. Zhang, A. Marshall, and J. R. Cavallaro, "Towards Scalable and Channel-Robust Radio Frequency Fingerprint Identification for LoRa," *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 774–787, 2022.

48. S. S. Hanna and D. Cabric, "Deep Learning Based Transmitter Identification using Power Amplifier Nonlinearity," in *International Conference on Computing, Networking and Communications (ICNC)*, 2019, pp. 674–680.

49. J. Li, Y. Ying, C. Ji, and B. Zhang, "Differential Contour Stellar-Based Radio Frequency Fingerprint Identification for Internet of Things," *IEEE Access*, vol. 9, pp. 53 745–53 753, 2021.

50. L. J. Wong, W. C. Headley, and A. J. Michaels, "Specific Emitter Identification Using Convolutional Neural Network-Based IQ Imbalance Estimators," *IEEE Access*, vol. 7, pp. 34 544–34 555, 2019.

51. J. W. Dean, "Multi-Mode Wireless Device Discrimination Using Convolutional Neural Networks," Ph.D. dissertation, Air Force Institute of Technology, 2020.

52. S. Riyaz, K. Sankhe, S. Ioannidis, and K. Chowdhury, "Deep Learning Convolutional Neural Networks for Radio Identification," *IEEE Communications Magazine*, vol. 56, no. 9, pp. 146–152, 2018.

53. K. Sankhe, M. Belgiovine, F. Zhou, L. Angioloni, F. Restuccia, S. D'Oro, T. Melodia, S. Ioannidis, and K. Chowdhury, "No Radio Left Behind: Radio Fingerprinting Through Deep Learning of Physical-Layer Hardware Impairments," *IEEE Transactions on Cognitive Communications and Networking*, vol. 6, no. 1, pp. 165–178, 2020.

54. K. Sankhe, M. Belgiovine, F. Zhou, S. Riyaz, S. Ioannidis, and K. Chowdhury, "ORACLE: Optimized Radio clAssification through Convolutional neuraL nEtworks," in *IEEE International Conference on Computer Communications (INFOCOM)*. IEEE, 2019, pp. 370–378.

55. A. Al-Shawabka, F. Restuccia, S. D'Oro, T. Jian, B. Costa Rendon, N. Soltani, J. Dy, S. Ioannidis, K. Chowdhury, and T. Melodia, "Exposing the Fingerprint: Dissecting the Impact of the Wireless Channel on Radio Fingerprinting," in *IEEE Conference on Computer Communications (INFOCOM)*, 2020, pp. 646–655.

56. J. Bassey, D. Adesina, X. Li, L. Qian, A. Aved, and T. Kroecker, "Intrusion Detection for IoT Devices based on RF Fingerprinting using Deep Learning," in *4th International Conference on Fog and Mobile Edge Computing (FMEC)*. IEEE, 2019, pp. 98–104.

57. J. Yu, A. Hu, G. Li, and L. Peng, "A Robust RF Fingerprinting Approach Using Multisampling Convolutional Neural Network," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 6786–6799, 2019.

58. K. Merchant, S. Revay, G. Stantchev, and B. Nousain, "Deep Learning for RF Device Fingerprinting in Cognitive Communication Networks," *IEEE Journal on Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 160–167, 2018.

59. A. Elmaghbub and B. Hamdaoui, "LoRa Device Fingerprinting in the Wild: Disclosing RF Data-Driven Fingerprint Sensitivity to Deployment Variability," *IEEE Access*, vol. 9, pp. 142 893–142 909, 2021.

60. Q. Xu, R. Zheng, W. Saad, and Z. Han, "Device Fingerprinting in Wireless Networks: Challenges and Opportunities," *IEEE Communications Surveys and Tutorials*, vol. 18, no. 1, pp. 94–104, 2016.

61. N. Soltani, K. Sankhe, J. Dy, S. Ioannidis, and K. Chowdhury, "More Is Better: Data Augmentation for Channel-Resilient RF Fingerprinting," *IEEE Communications Magazine*, vol. 58, no. 10, pp. 66–72, 2020.

62. X. Zhou, A. Hu, G. Li, L. Peng, Y. Xing, and J. Yu, "A Robust Radio-Frequency Fingerprint Extraction Scheme for Practical Device Recognition," *IEEE Internet of Things Journal*, vol. 8, no. 14, pp. 11 276–11 289, 2021.

63. W. Wang and L. Gan, "Radio Frequency Fingerprinting Improved by Statistical Noise Reduction," *IEEE Transactions on Cognitive Communications and Networking*, vol. 7731, pp. 1–10, 2022.

64. L. Bai, L. Zhu, J. Liu, J. Choi, and W. Zhang, "Physical Layer Authentication in Wireless Communications Networks: A Survey," *Journal of Communications and Information Networks*, vol. 5, no. 3, pp. 237–264, 2020.

65. T. Jian, B. C. Rendon, E. Ojuba, N. Soltani, Z. Wang, K. Sankhe, A. Gritsenko, J. Dy, K. Chowdhury, and S. Ioannidis, "Deep Learning for RF Fingerprinting: A Massive Experimental Study," *IEEE Internet of Things Magazine*, vol. March, pp. 50–57, 2020.

66. I. Mohamed, Y. Dalveren, F. O. Catak, and A. Kara, "On the Performance of Energy Criterion Method in Wi-Fi Transient Signal Detection," *Electronics*, vol. 11, no. 269, pp. 1–15, 2022.

67. L. Peng, J. Zhang, M. Liu, and A. Hu, "Deep Learning Based RF Fingerprint Identification Using Differential Constellation Trace Figure," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 1, pp. 1091–1095, 2020.

68. Z. Langford, L. Eisenbeiser, and M. Vondal, "Robust Signal Classification Using Siamese Networks," in *ACM Workshop on Wireless Security and Machine Learning*, 2019, pp. 1–5.

69. S. U. Rehman, K. W. Sowerby, S. Alam, I. T. Ardekani, and D. Komosny, "Effect of Channel Impairments on Radiometric Fingerprinting," in *IEEE International*

Symposium on Signal Processing and Information Technology (ISSPIT), 2015, pp. 415–420.

70. F. Restuccia, S. D'Oro, A. Al-Shawabka, B. C. Rendon, S. Ioannidis, and T. Melodia, "DeepFIR: Channel-Robust Physical-Layer Deep Learning through Adaptive Waveform Filtering," *IEEE Transactions on Wireless Communications*, vol. 20, no. 12, pp. 8054–8066, 2021.

71. Thread Group, "What is Thread?" https://www.threadgroup.org/BUILT-FOR-IOT/Home, accessed: 2022-01-12.

72. Siemens Industry Online Support, "WirelessHART Adapter SITRANS AW210 - 7MP3111," https://support.industry.siemens.com/cs/document/61527553/wirelesshart-adapter-sitrans-aw210-7mp3111, accessed: 2022-01-20.

73. Pepperl and Fuchs, "BULLET - WirelessHART Adapter," https://www.pepperl-fuchs.com/global/en/WirelessHART_Adapter_BULLET.htm, accessed: 2022-01-20.

74. J. Olds, "Designing an OQPSK demodulator," https://jontio.zapto.org/hda1/oqpsk.html, accessed: 2022-01-21.

75. M. Rice, *Digital Communications: A Discrete Time Approach*. Pearson/Prentice Hall, 2009.

76. H. Patel, M. A. Temple, and B. W. Ramsey, "Comparison of High-end and Low-end Receivers for RF-DNA Fingerprinting," in *IEEE Military Communications Conference (MILCOM)*. IEEE, 2014, pp. 24–29.

77. T. J. Bihl, K. W. Bauer, M. A. Temple, and B. Ramsey, "Dimensional Reduction Analysis for Physical Layer Device Fingerprints with Application to ZigBee and

Z-Wave Devices," in *IEEE Military Communications Conference (MILCOM)*. IEEE, 2015, pp. 360–365.

78. B. W. Matthews, "Comparison of the Predicted and Observed Secondary Structure of T4 Phage Lysozym," *Biochimica et Biophysica Acta (BBA) - Protein Structure*, vol. 405, no. 2, pp. 442–451, 1975.

79. J. Gorodkin, "Comparing Two K-category Assignments by A K-category Correlation Coefficient," *Computational Biology and Chemistry*, vol. 28, pp. 367–374, 2004.

80. Sci-kit Learn Developers, "Metrics and scoring: quantifying the quality of predictions," https://scikit-learn.org/stable/modules/model_evaluation.html#matthews-corrcoef, accessed: 2022-01-20.

81. A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Advances in Neural Information Processing Systems*, vol. 27, pp. 1097–1105, 2012.

82. T. J. O'Shea, T. Roy, and T. C. Clancy, "Over-the-Air Deep Learning Based Radio Signal Classification," *IEEE Journal on Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 168–179, 2018.

83. B. Jdid, K. Hassan, I. Dayoub, W. H. Lim, and M. Mokayef, "Machine Learning Based Automatic Modulation Recognition for Wireless Communications: A Comprehensive Survey," *IEEE Access*, vol. 9, pp. 57 851–57 873, 2021.

84. G. Shen, J. Zhang, A. Marshall, L. Peng, and X. Wang, "Radio Frequency Fingerprint Identification for LoRa Using Deep Learning," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 8, pp. 2604–2616, 2021.

85. O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2015, pp. 234–241.

86. E. Akar, "How Error Vector Magnitude (EVM) Measurement Improves Your System-Level Performance," https://www.analog.com/en/technical-articles/how-evm-measurement-improves-system-level-performance.html, accessed: 2022-08-02.

87. DARPA, "Radio Frequency Machine Learning Systems (RFMLS)," https://www.darpa.mil/program/radio-frequency-machine-learning-systems, accessed: 2022-07-05.

88. T. M. Schmidl and D. C. Cox, "Robust Frequency and Timing Synchronization for OFDM," *IEEE Transactions on Communications*, vol. 45, no. 12, pp. 1613–1621, 1997.

89. T. J. Carbino, M. A. Temple, and T. J. Bihl, "Ethernet Card Discrimination Using Unintentional Cable Emissions and Constellation-Based Fingerprinting," in *International Conference on Computing, Networking and Communications (ICNC)*. IEEE, 2015, pp. 369–373.

90. M. Simonovsky and N. Komodakis, "Dynamic Edge-Conditioned Filters in Convolutional Neural Networks on Graphs," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 3693–3702.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
15–09–2022	Doctoral Dissertation	Sept 2019 — Sept 2022

4. TITLE AND SUBTITLE

Learning Robust Radio Frequency Fingerprints Using Deep Convolutional Neural Networks

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Gutiérrez del Arroyo Pérez, José A., Maj, USAF

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT-ENG-DS-22-S-019

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Force Research Laboratory, AFMC
Attn: Dr. Vasu Chakravarthy
2241 Avionics Circle, Bldg 620
Wright-Patterson AFB OH 45433-7765
Email: Vasu.Chakravarthy@us.af.mil, Comm: 937-713-4026

10. SPONSOR/MONITOR'S ACRONYM(S)

AFRL/RYW

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

DISTRIBUTION STATEMENT A:
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Radio Frequency Fingerprinting (RFF) techniques, which attribute uniquely identifiable signal distortions to emitters via Machine Learning (ML) classifiers, are limited by fingerprint variability under different operational conditions. First, this work studied the effect of frequency channel for typical RFF techniques. Performance characterization using the multi-class Matthews Correlation Coefficient (MCC) revealed that using frequency channels other than those used to train the models leads to deterioration in MCC down to 0.05 (random guess), indicating that single-channel models are inadequate for realistic operation. Second, this work introduced, developed, and demonstrated Fingerprint Extraction through Distortion Reconstruction (FEDR), a neural network-based approach for quantifying signal distortions in a relative distortion latent space. Coupled with a Dense network, FEDR fingerprints were evaluated against common RFF techniques for up to 100 unseen classes, where FEDR achieved best performance with MCC ranging from 0.945 (5 classes) to 0.746 (100 classes), using 73% fewer training parameters than the next-best technique.

15. SUBJECT TERMS

RF Fingerprinting, RFF, Deep Neural Networks, Radio Frequency Machine Learning, RFML

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dr. Brett Borghetti, AFIT/ENG
U	U	U	UU	131	19b. TELEPHONE NUMBER *(include area code)* (937) 255-3636 x4612; brett.borghetti@afit.edu