The Importance of Data in RF Machine Learning

William H. Clark IV

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Electrical Engineering

Alan J. Michaels, Co-chair
R. Michael Buerher, Co-chair
T. Charles Clancy
Paul Plassman
Mark Embree
Joseph M. Ernst

October 17, 2022
Blacksburg, Virginia

Keywords: data generation, RFML, modulation classification, machine learning, deep learning

The Importance of Data in RF Machine Learning

William H. Clark IV

(ABSTRACT)

While the toolset known as Machine Learning (ML) is not new, several of the tools available within the toolset have seen revitalization with improved hardware, and have been applied across several domains in the last two decades. Deep Neural Network (DNN) applications have contributed to significant research within Radio Frequency (RF) problems over the last decade, spurred by results in image and audio processing. Machine Learning (ML), and Deep Learning (DL) specifically, are driven by access to relevant data during the training phase of the application due to the learned feature sets that are derived from vast amounts of *similar* data. Despite this critical reliance on data, the literature provides insufficient answers on how to quantify the data training needs of an application in order to achieve a desired performance.

This dissertation first aims to create a practical definition that bounds the problem space of Radio Frequency Machine Learning (RFML), which we take to mean the application of Machine Learning (ML) as close to the sampled baseband signal directly after digitization as is possible, while allowing for preprocessing when reasonably defined and justified. After constraining the problem to the Radio Frequency Machine Learning (RFML) domain space, an understanding of what kinds of Machine Learning (ML) have been applied as well as the techniques that have shown benefits will be reviewed from the literature. With the problem space defined and the trends in the literature examined, the next goal aims at providing a better understanding for the concept of data quality through quantification. This quantification helps explain how the quality of data: affects Machine Learning (ML) systems with regard to final performance, drives required data observation quantity within that space, and impacts can be generalized and contrasted. With the understanding of how data quality and quantity can affect the performance of a system in the Radio Frequency Machine Learning (RFML) space, an examination of the data generation techniques and realizations from conceptual through real-time hardware implementations are discussed. Consequently, the results of this dissertation provide a foundation for estimating the investment required to realize a performance goal within a Deep Learning (DL) framework as well as a rough order of magnitude for common goals within the Radio Frequency Machine Learning (RFML) problem space.

The Importance of Data in RF Machine Learning

William H. Clark IV

(GENERAL AUDIENCE ABSTRACT)

Machine Learning (ML) is a powerful toolset capable of solving difficult problems across many domains. A fundamental part of this toolset is the representative data used to train a system. Unlike the domains of image or audio processing, for which datasets are constantly being developed thanks to usage agreements with entities such as Facebook, Google, and Amazon, the field of Machine Learning (ML) within the Radio Frequency (RF) domain, or Radio Frequency Machine Learning (RFML), does not have access to such crowdsourcing means of creating labeled datasets. Therefore *data* within the Radio Frequency Machine Learning (RFML) problem space must be intentionally cultivated to address the target problem.

This dissertation explains the problem space of Radio Frequency Machine Learning (RFML) and then quantifies the effect of quality on data used during the training of Radio Frequency Machine Learning (RFML) systems. Taking this one step further, the work then goes on to provide a means of estimating data quantity needs to achieve high levels of performance based on the current Deep Learning (DL) approach to solve the problem, which in turn can be used as guidance to better refine the approach when the real-world data quantity requirements exceed practical acquisition levels. Finally, the problem of data generation is examined and provides context for the difficulties associated with procuring high quality data for problems in the Radio Frequency Machine Learning (RFML) space.

Dedication

Deanna; without the loving support she provided, and the tireless weeks on end caring for our rampaging toddler, this would not have been completed.

Acknowledgments

This work is fundamentally supported by the efforts of the advisors, co-workers, and students who have supported the curation of the dataset used in this work, as well as those responsible for pushing me to develop new approaches whenever real-world conflicts caused stumbling blocks along the way.

Contents

List of Figures

List of Tables

List of Abbreviations

ADC Analog-to-Digital Converter - The device that is responible for sampling the electromagnetic spectrum to which it is attached and converting the signal into a discrete value format for digital systems to process.

AI Artificial Intelligence - The concept of a man-made system capable of updating its own behavior based on the conditions in which it is placed with capabilities at or exceeding that of a person.

AMC Automatic Modulation Classification - A problem space within RFML in which the goal is to identify the modulation of an observed signal.

ANOVA Analysis of Variance - A set of approaches to help understand the differences among statistical observations.

AWGN Additive White Gaussian Noise - A typical type of interference observed in transceivers due to thermal noise in electrical components and ambient noise collections of the spectral environment.

CLDNN Convolutional, LSTM, Deep Neural Network - An architecture found to be effective at processing time series datasets like RFML.

CNN Convolutional Neural Network - A neural network that predominately makes use of convolutional layers for processing.

COCOMO Constructive Cost Model - A tool designed to help perform cost modeling for software development by providing the expected cost of a software project by lines of code and completion timeline for a given parameter space.

COTS Commercial Off-The-Shelf - Products that can be purchased from a commercial market rather than designed specificaly for the task at hand.

CR Cognitive Radio - A concept for advance radios that are able to adapt their settings and system configuration based on the environmental conditions without the need for a human in the decision loop.

CTIA Cellular Telecommunications Industry Association - A US wireless communications organization that represents many groups within the wireless industry.

CV Computer Vision - The field that focuses on visual processing of images and videos to either augment or a replace a human operater.

DAC Digital-to-Analog Converter - The device responible for taking the discrete value data presented to it from a digital system and outputing a continuous spectrum to be used in practical systems.

DARPA Defense Advanced Research Projects Agency

DL Deep Learning - The application of deep neural networks to a problem.

DNN Deep Neural Network - This is the terminology used to refer to any neural network that consists of more than one hidden layer between it's input and output.

DoD Department of Defense - The USA's Department of Defense.

DSA Dynamic Spectrum Access - The term for utilization of the spectrum in an oppurtunistic fashion rather than preallocated frequency slices.

DSP Digital Signal Processing - The digital processing of signals.

DTV Digital Television - Broadcast waveforms that are typically considered a Primary User when performing secondary access analysis.

EM Electromagnetic - The duality of electric and magnetic waves in that one cannot exist without the other.

EME electromagnetic Environment - A generalized term for describing everything in the purview of a transceiver that effects, augments, or adds to the intended waveform.

EMI Electromagnetic Interference - The presences of other sources in the EME whose effect degrades the performance of a system when compared to quiet spectrum.

FCC Federal Communications Commission - Government agency in charge of state, local, and personal communication systems.

FFT Fast Fourier Transform - An efficient implementation of the Discrete Fourier Transform.

FIR Finite Impulse Response - A fundamental tool for filtering signals where a linear phase respone is desired.

FO Frequency Offset - The deviation of the carrier frequency from actual transmission to that of estimation or assumption. Expressed in Hz or as a ratio with observation sample rate as a percentage.

FPGA Field-Programmable Gate Arrays - A common configurable hardware component within software defined radios that supports realtime processing.

FSK Frequency Shift Keying - A modulation technique that encodes information into the frequency of a complex sinusoid and varies between discrete levels.

GAN Generative Adversarial Network - This is a machine learning approach that trains a tandem set of networks where the role of one network is to perform a task that is opposite to its counterpart's effort, such that as one network improves, it forces the counterpart network to improve as well.

GMSK Guassian Minimum Shift Keying - The initial modulation technique used in GSM.

GPS Global Positioning System - A technology based on satellites' relative positioning with that of the Earth and the observable delays in a known propagation of a waveform.

GPU Graphics Processing Unit - Graphics cards that in original design were used to display images on a screen from a computer.

GRU Gated Recurrent Unit - A type of layer within a neural network that keeps previous output to better inform current decisions through a network of gating operations as an alternative to the LSTM.

GSM Global System for Mobile Communications - The second generation of cellular phones, still widely used today.

HF High Frequency - The radio frequency range of 3MHz to 30MHz that is well suited to long range transmission requirements due to the reflective properties of the ionoshpere.

IARPA Intelligence Advanced Research Projects Agency

IIR Infinite Impulse Response - A fundamental tool for filtering signals where current input and past output are used.

IoT Internet of Things - The description of devices being networked together that don't necessarily require a human to be active.

IQ In-phase and Quadrature - This is a sampling technique that is beneficial for software processing through complex notation.

KDE Kernel Density Estimator - A tool that estimates the distribution(s) of parameters in a sample space.

LDAPM Linear Digital Amplitude and Phase Modulation - A class of modulations that are representable by a constellation on IQ phase plots.

LEEP Log Expected Empirical Prediction - A technique proposed in TL for estimating the ability of a given model to be tuned to a new data space.

LogME Logarithm of Maximum Evidence - A technique proposed in TL for estimating the ability of a given model to be tuned to a new data space.

LSTM Long-Short Term Memory - A type of layer within a neural network that keeps previous output to better inform current decisions through a network of gating operations.

LTI Linear Time-Invariant - A property of a system or technique where the input when scaled, or added to, or translated, results in the same scaling, addition, or translation in the output.

MIMO Multiple-Input-Multiple-Output - This is the terminology used when there are multiple transmitters and there are multiple receiver sensors being used to make a decision.

ML Machine Learning - A toolkit available in data processing for extracting useful information to be further applied to the problem space.

NASA National Aeronautics and Space Administration - The USA agency in charge of the civil space program.

NCE Negative Conditional Entropy - A technique proposed in TL for estimating the ability of a given model to be tuned to a new data space.

NLP Natural Language Processing - A domain in ML that focuses on applications around understand, predicting, and translating language.

NN Neural Network - An approach of machine learning that approximates the brain by allowing for operational units mimicking a neuron.

NTIA National Telecommunications and Information Administration - Government agency in charge of federal communication systems.

OOTM Out-Of-Tree Module - Terminology used to describe resources developed to enhance GNU Radio that exist outside of GNU Radio source code for others to grab as needed.

OPC Observations per Class - The number of observations available for a particular classification problem

OTA Over-the-Air - This is an indication that the sampled spectrum was captured wirelessly as opposed to through a wired sampling or simulation.

PII Personally Identifiable Information - A protected class of information in the United States pertaining to its citizens and legal alien residents.

PRNG Pseudo-Random Number Generator - A random number generator that works in the digital domain or through logicial circuitry in FPGAs to provided seeded, repeatable random number generation.

PSK Phase Shift Keying - A modulation techinque that encodes information into the phase of a complex sinusoid with discrete values.

PU Primary User - Nomenclature indicating the user of the spectrum that has been given access with rights that other users may not infringe upon.

QAM Quadrature Amplitude Modulation - A modulation technique that encodes information into the amplitude of a complex sinusoid by independently controlling the real and imaginary components.

QoS Quality of Service - The defining characteristics of how a system performs and minimum levels that must be met.

RF Radio Frequency - The oscillation of the electromagnetic field on the order of 30kHz to 300GHz, which is typically used for wireless applications.

RFML Radio Frequency Machine Learning - The application of machine learning to radio frequency problems. Most typically reliant on directly processing the sampled spectrum with machine learning techniques. Pronounced as *riff-maul* in this work.

RFMLS Radio Frequency Machine Learning Systems - A DARPA program.

SC2 Spectrum Collaboration Challenge - A DARPA program.

SCISRS Securing Compartmented Information with Smart Radio Systems - An IARPA program.

SD Signal Detection - The act of determining whether a signal is present in the observed spectrum.

SDR Software Defined Radio - The concept of providing more flexibility to a radio system by moving the ADC and DAC as close as reasonably possible to the device's antenna, allowing for waveforms to be created in software such that dynamic utilization of spectrum could be accomplished rather than as previously done solely in hardware radios.

SEI RF Fingerprinting / Specific Emitter Identification - A problem space within RFML in which the goal is to identify the transmitter of an observed signal.

SIMO Single-Input-Multiple-Output - This is the terminology used when there is a single transmitter and there are multiple receiver sensors being used to make a decision.

SNR Signal-to-Noise Ratio - The relative strength of a signal of interest to the additive noise observed with the signal, often expressed in dB.

SRM Sample Rate Mismatch - The ratio of the observed waveform relative to that of what was intended, or expected of the observation.

SS Spectrum Sensing - The terminology within the RFML problem space focused on signal detection from sampled spectrum.

STRS Space Telecommunications Radio System - A project at NASA to help facilitate communications between open architecture systems and ground based SDRs.

SU Secondary User - Nomenclature indicating the user of the spectrum that has been given access, but may not infringe upon a Primary Users' access.

SVD Singular Value Decomposition - A factorization approach for matrices that decomposes a matrix into rotational and scalar factors.

SWaP Size, Weight, and Power - The typical contraints needed to be minimized in order to make a device practical for portable application and usage.

TL Transfer Learning - The practice of training a system using an initial dataset, then fine tuning or modifications to the approach using a *target* dataset.

TPU Tensor Processing Unit - Processing units that are designed for performing tensor operations.

TRL Technical Readiness Level - A scale to describe the state of a new technology or system in terms of where it is between proof-of-concept design and a fieldable technology with years of viability to show.

UAV Unmanned Aerial Vehicle - A autonomous craft that is able to fly such as a drone.

YOLO You Only Look Once - A popular computer vision / image processing machine learning techinque that is able to produce meaningful results while balancing realtime performance.

Chapter 1

Introduction

Radio Frequency Machine Learning (RFML) is a relatively new field, with its true definition still in flux with regard to the level of refinement of features compared to the raw source data used [3], while being defined by all of the contributors adding their work to its literature body. Consistent with [1], this dissertation defines the field of RFML as the application of Machine Learning (ML) within the Radio Frequency (RF) domain, where the ML application is applied as close to the digitization of the sampled RF spectrum as a baseband signal. Allowances are given for the usage of preprocessing techniques that can help simplify the problem space without deteriorating the information context beyond an unusable threshold. In the most basic application of this constraint on preprocessing, the performance of an ML system should not decrease when preprocessing is applied.

Fundamental to the body of RFML is the radio and its interaction with the electromagnetic environment (EME). The radio is a device that is capable of carrying information that its user intends to relay to another via modulated electromagnetic (EM) waves allowing for the propagation path of the message to not require physical matter to act as a conduit. This device allows for information to travel great distances at the speed of light minus some processing delay of the devices, along with the ability to not be location-bound in the sense that traditional physical mail is bound; e.g., the post office and mailbox system. While Marconi is the credited creator of the first radio device [4], his work rides on the shoulders of the EM giants of Faraday, Maxwell, and Hertz who each in turn harnessed the idea of EM waves as something tangible that could be manipulated through human ingenuity [5]. It is on the shoulders of these giants of thought and engineering, as well as those that have advanced the radio over the last century to the point where there are more radios in the country than people [6], that the body of RFML owes its foundations and ongoing developments. A modern day information flow for a radio link is provided in Figure 1.1; however, getting to this flow took significant experimentation and experience as the first radios directly connected the *information* blocks to the transmitter and receiver *frontends*. One key distinction between the usage of EM waves for radio transmissions from that of RFML is that the primary function and goal of the radio is overcoming the distortions that are perceived by the receiver's view of the EME that deform the information to convey as much information between users as possible, whereas for RFML the actual information content of the message is usually not the focus. Rather RFML is more typically focused on secondary characteristics of the waveform, or the EME as a whole, instead of the information being transmitted as in communications-focused ML.

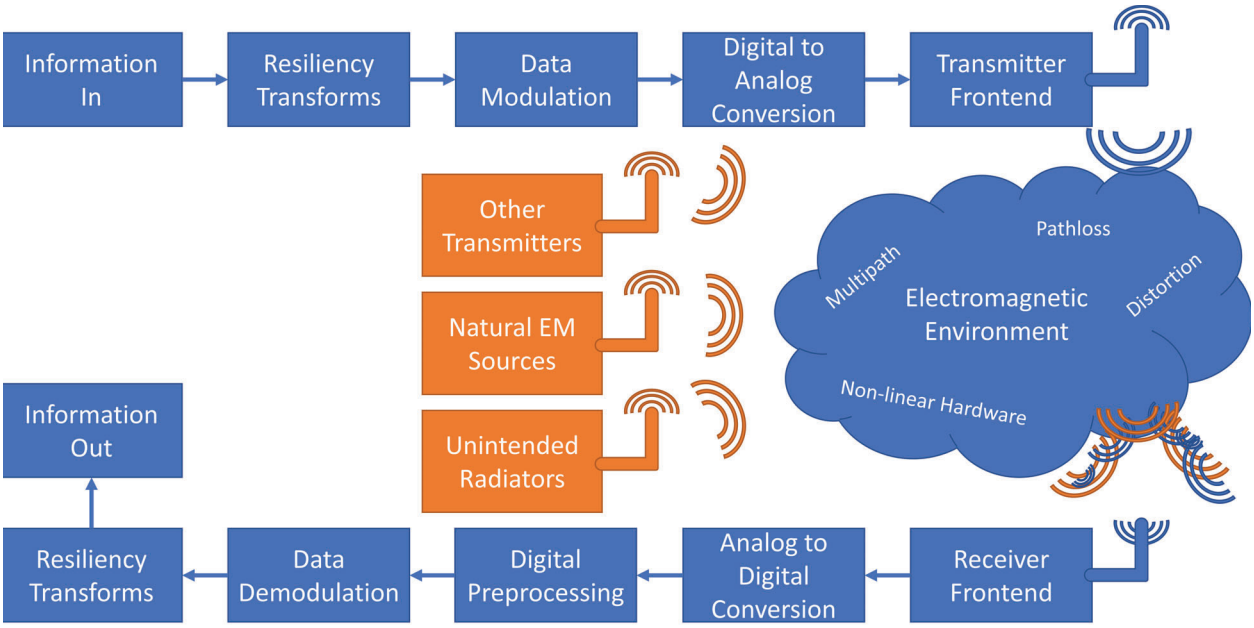

Figure 1.1: Radio Information Flow: The flow of information from the originator or transmitter through the EME to a receiver that then inverts the processes to recover the information. While this is a general example of flow, the original radios only consisted of the information and frontends, and only through experimentation, experience, and dedicated effort from the research community did this current flow come to fruition. The receiver ends up receiving the intended message, delayed and differently distorted copies of the intended message, along with all of the other observable perturbations in the EME.

Recent works have taken the loose definition provided earlier for RFML and focused the definition to look more at the wireless systems that are put in place, while having the ML applied as closely as possible to raw sampled RF spectrum [1]. The main effect of RFML systems is reducing the amount of highly skilled preprocessing needed and getting the raw samples themselves as close as possible to the input into the system. Figure 1.2 highlights where ML can be considered as part of the *RFML Ecosystem*. For the most part, RFML applications are dominantly focused on better understanding the EME around the receiver, but that does not and should not neglect the matching role it can have within the transmitter as the research continues progressing. In fact some work has already begun looking at replacing portions of the Digital Signal Processing (DSP) chains in both the transmitter and receiver to allow for an end-to-end ML system that can more effectively match or adapt to the EME present [7]. It is important to remember that the *RFML Ecosystem* [1] goes beyond just the concept of placing an ML routine in the radio path, as the inclusion of ML in the system must also consider:

- The data space that is necessary to train such a system and whether the data is available and feasible to create such a system.

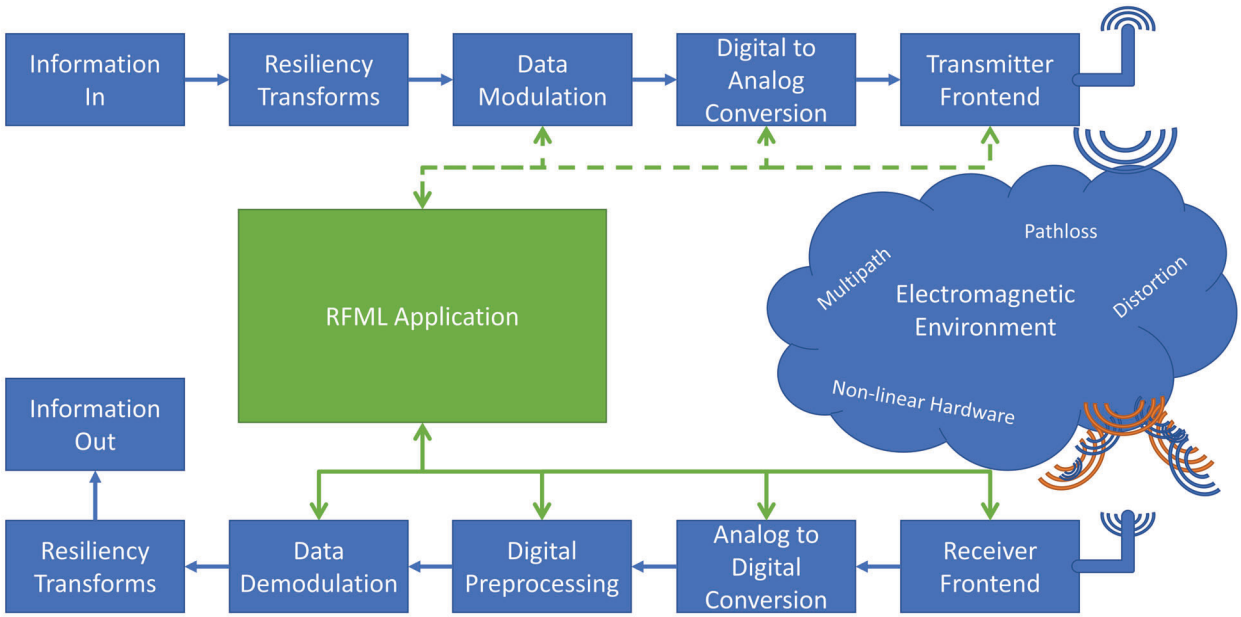

Figure 1.2: Role of ML in the field of RFML. Focus is typically heavier on the receiver side of the radio system, with ML having control of a finite state space in terms of the configuration settings within the analog components. The ML is used more heavily on the feature and signal space on the digital side of the system. There is nothing that stops it from being incorporated into the transmitter as well in terms of a matched replacement, but stops short of the resiliency transforms applied to the information space such as the role of coding theory.

- Understanding that new techniques and approaches present new surfaces for adversarial attacks and new challenges for security of any system that makes use of it.
- Development of trust in the tool must also be considered due to the difficulties associated with describing tactical radio decisions to an end user where the user's life or others' lives are on the line for such decisions.
- The operational requirements for developing and deploying such a system and its effect on the larger system to which it will be intertwined.

This focus on minimizing the expert-in-the-loop preprocessing of RF spectra leads to two general questions:

- First, why is Machine Learning needed within the RF problem space?
- Second, what foundational training, in terms of data quality and quantity, is required in order to deploy a system that abides by this definition?

The former question is touched on briefly next, while the latter is described throughout this dissertation.

1.1 Motivation for ML in RF Applications

In one fashion, the marriage of RF and ML can be argued to be almost as old as either application, and by another, the argument can be made that the advent of Software Defined Radio (SDR) truly is what allowed the two to coexist [8, 9]. The main driving force behind SDR was to bring flexibility to an individual radio by removing as much pre-defined rigid functionality as possible between the device's antenna and the components responsible for converting between analog and digital domains, namely the Analog-to-Digital Converter (ADC) and the Digital-to-Analog Converter (DAC), thereby allowing the programmable software/firmware to craft the interaction with the wireless spectrum [10]. This in turn leads to the concept of Cognitive Radio (CR), though through independent means, which had the goal of improving the utility of the radio to the user by giving a reasoning engine contextual control over the device's resources [9]. An important point to keep in mind, is that union of CR and SDR is what allows for this to be possible. The CR is the predominant source of reasoning that allows decisions to be made and actions to be scheduled, but it is the SDR that sets the capabilities and action space available to the CR. In much the same way, someone might think about being an Olympic class athlete, but unless they've also got the stamina, reflexes, and focus that is needed to perform at that level, it would remain as only thoughts, without the accompanying action.

While not the first use case of ML within an RF system, since there are examples from the 1970's and earlier [11], CR is probably the easiest to conceive the overlapping domain spaces of an RF device and Artificial Intelligence (AI), the envelope in which ML resides as shown in Figure 1.3. At this point, the question of what ML can do for the RF domain has been the focus of research for at least three decades under the umbrella of SDR and CR, and now extending into the RFML domain space. The question of *why use ML in the RF domain problems* comes down to the common reasons to do anything as seen in the literature: ML has the potential to enable RF systems to be better, faster, and cheaper. *Better* due to their potential for handling the complex problems of spectrum shortage and holes, along with the ability to adapt their waveforms to fit the spectrum [12]. *Faster* in the sense that the algorithm can make observations and adjustments at rates faster than a human observing the same situation. *Cheaper* in the sense that knowledge contained within the ML system can be replicated through software and distributed in a way that no training of human operators can achieve.

While there is great potential for the application of ML in the RF domain, an important starting point is the application spaces that exist in the RF domain to which ML can be applied. Looking at the statistics collected by the Cellular Telecommunications Industry Association (CTIA) about consumer growth in wireless usage, the usage of the RF spectrum

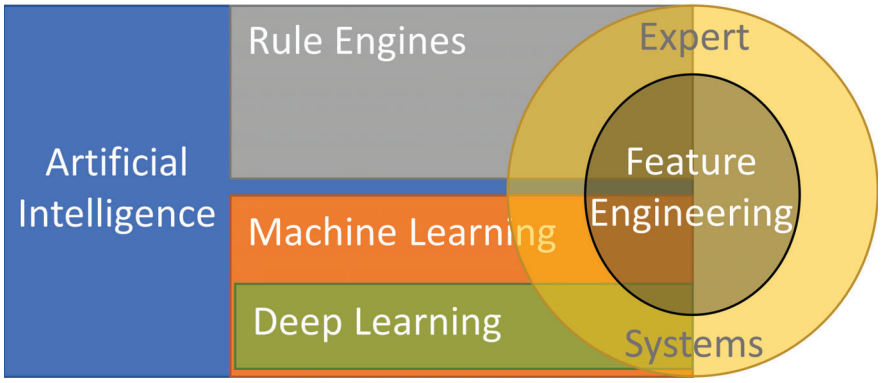

Figure 1.3: Venn Diagram for the relationship between ML and AI along with expert systems, which can make use of AI/ML components, but are typically driven by a knowledge base.

continues to grow at exorbitant rates [6]:

- From 2000 to 2017, the number of wireless subscriptions in the United States saw roughly a factor of four growth, from 107 million to 400 million subscribers, roughly 1.2 subscriptions per citizen.
- In 2017, roughly 77% of the population owned at least one smartphone.
- Rough estimates project a current number of unique global user to be 5.1 billion.
- Sales of spectrum usage rights have raised an estimated $200 billion dollars in the US.
- Wireless spectrum usage enables more complex technology such as self-driving cars when 5G enabled, predicted to save 21.7 thousand lives and $447 billion dollars per year.
- Improvement of 5G networks in general is estimated to bring $500 billion in economic growth while creating 3 million jobs.
- Future predictions are expecting 31 billion devices being connected globally using wireless technology next year, 2023.
- Hotspot usage increased 60% since the start of the Covid-19 pandemic.

The general summary of the statistics from CTIA go on to show that the commercial market for RF spectrum usage is rapidly growing, empowers and enriches nation-level actors through lease and access to the resource, and shows no signs of stopping in the near future, thereby enforcing the concept that wireless usage is a fundamental part of the developed world, and even critical for maintaining economic functionality while handling global crises.

An important component in understanding many challenges in the RF domain revolve around the lease and usage rights of RF spectrum. The static allotment of the resource leads to the problem known as "spectrum shortage," where the demand for spectrum vastly exceeds the amount of unallocated spectrum available for use. Now the spectrum shortage can be argued to be of an artificial nature, but that is largely due to a mixture of legacy static allocations put in place over the last century and not having the technical knowledge on how to implement techniques that would be efficient [13]. The legacy allotment of spectrum can otherwise be described as a historical record of how regulatory and procedural understanding viewed the *optimal* usage of spectrum. Going forward the challenges are generally be understood as: how can the consortium of spectrum users agree on how to share spectrum that is fair, able to be regulated, and still maintain an expected level of quality for providing service. The technology on the other hand is the underpinning for why interest in CR exists, by providing a fundamental mechanism by which Dynamic Spectrum Access (DSA) can be implemented without a strict set of regulatory rules that are feared to lead back to the problem of spectrum shortage [13].

Commercial usage of the spectrum isn't the only source of contention and problems; helping to fuel the spectrum shortage are the allocations made for military systems. A large portion of the static assignment is allocated to the military and sits idle domestically as it is only engaged in times of war, or periodically used in low duty cycle sensing operations, thereby utilizing the spectrum for only small fractions of the time. The military standpoint on spectrum usage then must be adaptable to share spectrum, but the vast amount of adaptability, and dynamics that are introduced by the concept of CR, DSA, and SDR systems that can change at superhuman rates, means that the domains of Spectrum Awareness (SA) and strategic signals intelligence will also need to improve and be capable of handling these changes to continue to provide utility. Many such RF applications are defined and leveraged in operational strategies highlighted in the Joint Chiefs' Electromagnetic Spectrum Operations (JEMSO) strategy, with RFML techniques offering user-selectable accelerations of real-time battlefield SA [14]. Fundamentally, control of information, and by extension the means of relaying information, determine the outcome of battles and war, as they have done since ancient times. In present times, the use of the RF spectrum cannot be understated; having access to it, and control of it, will undoubtedly provide a tactical edge in combat. During wartime, commercial allocations are national at best, yet Electromagnetic Interference (EMI) caused by commercial devices potentially degrade battle effectiveness (e.g., 5G vs. Radar altimeters).

From commercial to military, there is a clear demand for access and use of the RF spectrum. Figure 1.4 presents an example slice of spectrum cut out for a finite time (vertical) and finite frequency (horizontal) stretch. In the center bottom, the spectrum is observable and distinguishable in terms of occupied or not in orange and black, respectively. While this might be enough to help answer the question of what spectrum is empty, and what is in use, it doesn't provide enough information for either commercial or military applications. For example, in the top left corner, we can see two red signals overlapping where a wide

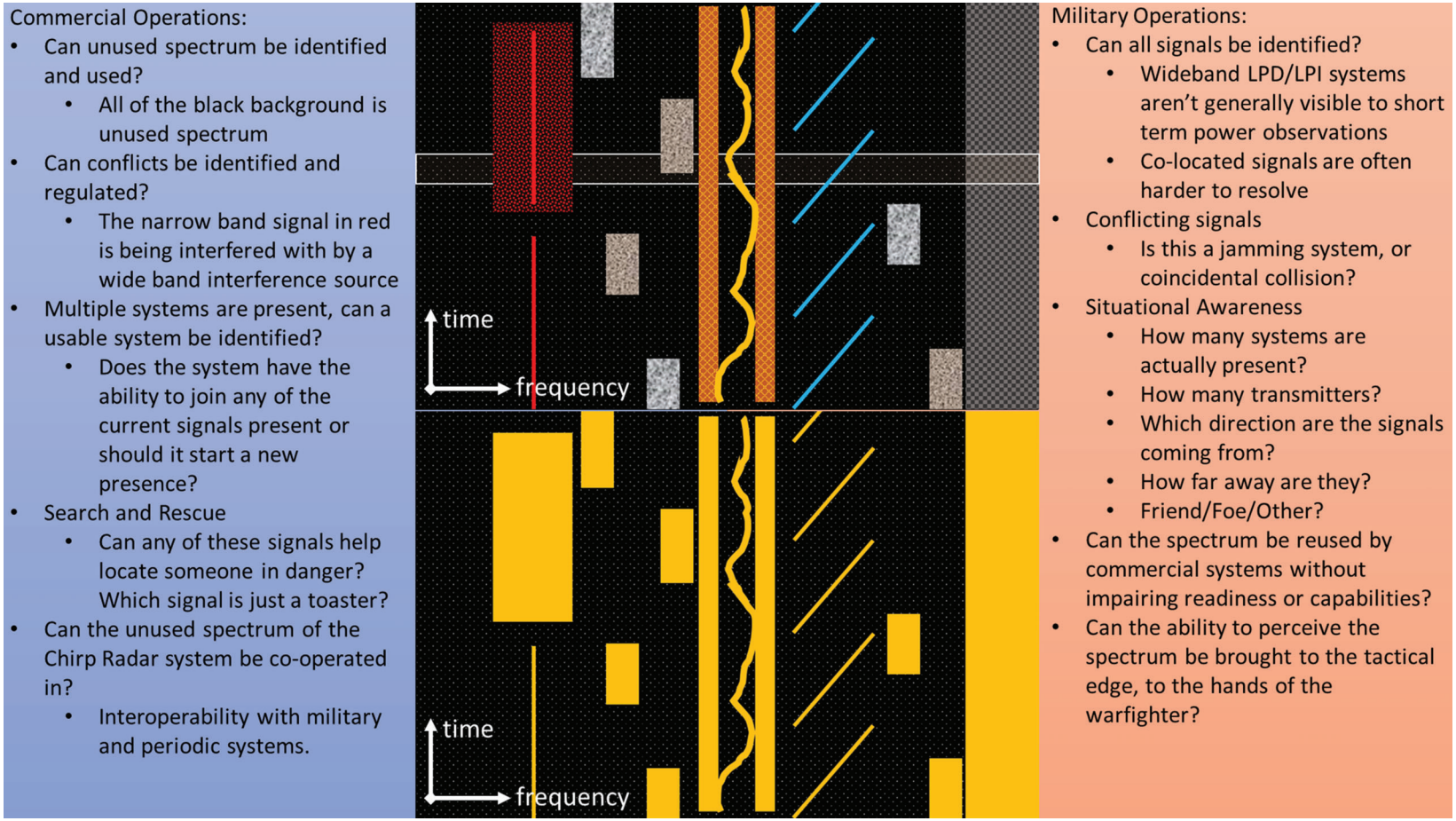

Figure 1.4: A conceptual drawing of a slice of spectrum in time and frequency. This provides a view of what situational awareness brings to both commercial and military applications by expanding the detail and context of the spectrum to allow for more useful questions to be answered in the hopes of increasing the overall utility to the end user. The figure at the bottom in black and orange represents what might be seen by a short term energy detection system, while the top figure, with additional colors and texture, provides greater context and is able to help make more practical use of the spectrum.

rectangular signal is overlapping with a narrowband signal. In the commercial realm, this could be interference that degrades system performance and needs to be regulated to prevent future collisions; however, an observer looking at this spectrum would not be able to tell which of the two colliding signals had right-of-way and who was interfering. In order to better understand this collision useful context is needed to resolve these conflicts over the presence of energy alone; the knowledge of who was transmitting, and from where they were transmitting would be needed. As for military operations, this could be jamming of the narrowband system by creating a wideband interference signal compromising the quality of the narrowband signal, thereby preventing information transfer. If this is unexpected, *could there be hostile actors nearby*, or if intentional, *then was the jamming act successful* are questions that would be hard to judge from a power observation. While there are bullet points provided for both commercial and military operations in the spectrum, there is nothing that makes any one operation explicitly in one application space or another. For example, understanding the location of a transmitter might help in a military operation to neutralize a threat; it's just as likely that it could be used to find a friendly unit. Flipping that to commercial search and rescue operations, finding where a signal is originating could indicate and differentiate a life needing to be saved over a toaster messaging the current temperature. Another way to view the commercial application of such technology is in disaster relief where Unmanned Aerial Vehicle (UAV) beacons indicate where critical issues need to be addressed, such as flash fires, and require immediate response to prevent worse damage from occurring. While the operations touched on in Figure 1.4 are some of the more researched areas and applications of RFML in the literature, this is by no means an exhaustive list.

Here are a few questions conceived while thinking through the issues discussed above:

- How can the RF spectrum be made available to an order of magnitude more radios than people in the near term?
- How can the RF spectrum support multiple orders of magnitude more radios than people in the long term?
- What is the maximum density of transceivers, based on available bandwidth and data rate requirements, given at least one end of a communications link is within a defined area?
- How can quality of service be guaranteed without dedicated resources?
- Should a wireless service be centralized, or distributed?
- How can such a vast quantity of radios and systems be well regulated?

Efficient and robust usage that will require intelligent machine capabilities to change and adapt to make the wireless medium highly congested and contested will be required. The question is then, "*How can machine learning alleviate the problems that are currently affecting the usage of RF spectrum?*"

1.2 Importance of Data in RFML Applications

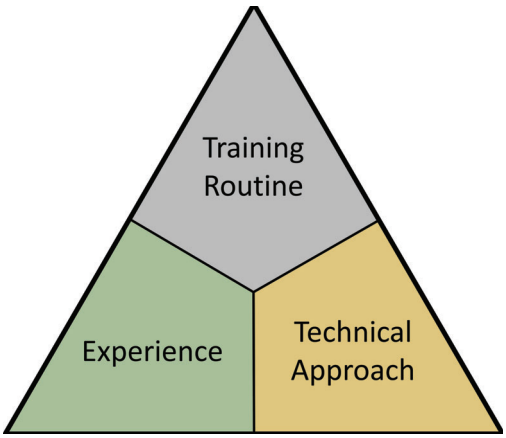

Figure 1.5: Three components for creating a successful ML system. The *technical approach* used to resolve the problem, or Deep Learning (DL) as used frequently here with a specific layer architecture. The *experience* that is observed and learned, the observations of data, and the focus of this work. Lastly, the *training routine* that utilizes the technical approach and experience, then modifies and adjusts the technical approach's weights in the goal of perfecting the routine. All together, these three components must be continuously adjusted in the pursuit of a better spectrum access solution.

Given the hopes for the utility of ML applied in the RF domain, understanding the strengths and trappings of ML are essential. To that end, there are three fundamental components to any ML system as shown in Figure 1.5: the technical approach (Deep Learning (DL) models are the focus in this work), the training routine or learning mechanism (the iterations between the experience and technical approach that result in a final solution), and finally the data or experience that defines the behavior of the model, the latter being the fundamental focus of this dissertation. While the details of ML will be discussed in greater length in Chapter 2, at the fundamental core, ML is still growing and should not be assumed to have found the peak of its capabilities. In terms of model development, breakthroughs should continue to be expected as was the case when the Convolutional Layer was introduced by LeCun and Bengio [15]. Looking at improvements in the training routine, the introduction of back-propagation to be incorporated into ML training by Rumelhart, Hinton, and Williams [16], as well as the inclusion of Transfer Learning (TL) as surveyed by Pan and Yang [17] saw great improvement in ML applications. Further, the combination of model architecture and the unique qualities of sampled RF data could lead to further improvements and innovation. The unique qualities of sampled RF data refer to the complex baseband representation that allows for critically sampled observation represented in complex notation, $\mathbb{C}$, for which models are still being devised on how to best properly utilize this representation [18, 19].

The three cornerstones of ML as shown in Figure 1.5 are in the most general sense vital to any ML application. Without all three, there is no system.

- Without a training routine, data moves through the technical approach with no way to improve or refine based on imperfections observed.
- Without a technical approach, there's no means of extracting common features from the data and therefore nothing on which the training routine can alter.
- Without available data, the technical approach cannot provide clues that the training routine can leverage for refining the approach in a useful manner.

While the three intertwined components of ML all interact to create a final system that can be used for the application, it is the experience, or data, that the developer has the least control over, if any at all. Often, the data for most applications is collected by another entity, likely for a different purpose, as the process of curating a sufficiently diverse dataset exceeds the resources of time, storage, and/or money, needed to assemble the dataset. This dissertation serves as a guide for curating data for RFML applications such that the required diversity of observation and environment can be planned for in some minimal capacity, without succumbing to the need for an endless source of data to train a system.

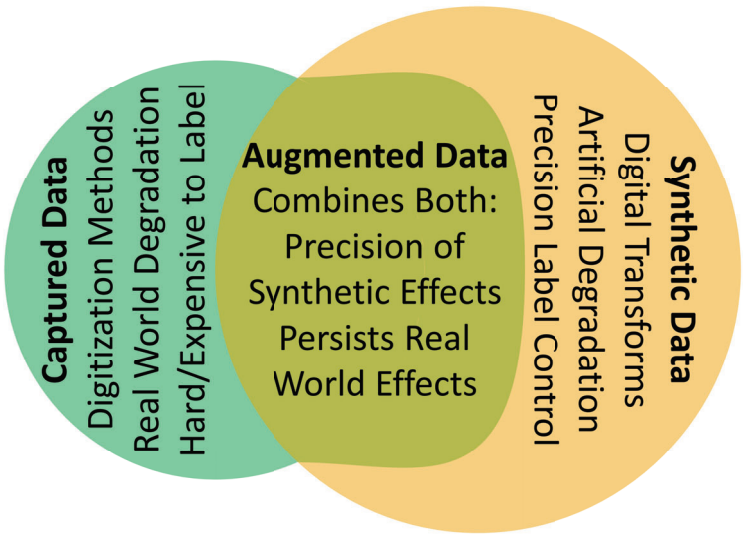

Figure 1.6: The makeup of data available to ML systems. Captured data consists of direct observations of the real world that has been made available to an ML system during training or inference and contains the most accurate representation of data at the point of capture. By contrast, Synthetic data is easier/cheaper to acquire assuming a model exists for representation in a digital form, but comes at the cost of simulation/developer assumptions that might not perfectly reflect the real world use case. Augmented data is then the combination of real world captures and synthetic permutations in the attempt to expand the dataset size of captured data alone, while maintaining true degradations, where synthetic techniques effectively *stretch* the available size of the captured data.

Figure 1.6 shows a Venn Diagram of the data sources available to typical ML systems. The three typical sources of data are *Captured*, *Synthetic*, and *Augmented*. The process known within this work as data collection procures captured data, which is arguably the best data if collected native to the application space, and is seen in computer vision as the process of recording with a camera, or similar device, the real world in a digitized form for use in an ML system. By direct contrast with captured data, synthetic or simulated data is instead the featurization of real world objects in a mathematical representation such that a computer simulation can be made to create more data than can reasonably be collected through a data collection process. By making use of both captured and synthetic data and techniques, the realm of augmented data becomes available. Augmentation can be seen as the application of synthetic techniques to distort or otherwise alter a captured observation to provide a new observation with different characteristics (albeit with redundant information), but can also be the case of using a captured observation as a placement of the real world with synthetic observations being added to such a placement to create synthetic drop-in augmentation.

Unlike other domains, image and audio for example, where the application of the ML system is to design a machine counterpart to a human operating and interpreting the analog world through digitization, RFML applications typically stem from human manipulation of the RF domain through a direct mathematical formula, meaning that RFML problems can precisely simulate the waveform using this mathematical representation [20]. This in turn means that it is comparatively simple to create an academic data generation routine that is able to create vast amounts of synthetic data under precise academic assumptions in the RFML domain. This condition allows for the explosion of research as has been seen over the last decade in terms of RFML problem spaces, it is a double-edged sword in that there are severe limitations of going directly from training in the synthetic space and applying the trained system to the real world [21]. Looking back at Figures 1.1 and 1.2, the simulated data creation process effectively cuts out the DAC through the ADC steps and replaces it with mathematical models that can over-idealize the problem space for the sake of simplified computation. However, these simplifications allowed for developing traditional algorithms and analysis techniques that enabled the current radio systems that are free of any ML routines. One major simplification often used in the field is the simplification of the EME to an Additive White Gaussian Noise (AWGN) transmission channel, where the only difference between the original transmitted waveform and the received observation is the addition of circularly symmetric Gaussian distributed random variables with a controllable variance to affect the Signal-to-Noise Ratio (SNR) of the observation. AWGN channel assumptions allow for an idealized receiver to use a matched filter as the optimal approach to maximize the SNR of the received signal and thereby provide the best recovery of information contained in the transmitted waveform. Other simplifications that have been applied in this domain have assumed a Nyquist critically sampled waveform where the complex symbols, a direct mapping of information data bits to a point on the complex plane, are received with the AWGN added in, thereby bypassing the difficult process of time, frequency, and phase synchronization. While works using these simplifications are justified based on the field and literature for an investigative proof-of-concept, the direct work of transitioning such a trained model to

real world practical use requires significantly more work and will fail without any further revision [22]. The work presented in this dissertation shows that systems trained on the more realistic scenario where only an AWGN channel is observed between the ideal transmit waveform and receiver, along with a few mismatched parameters that can be expected in a dynamic system [22], fail to have any practical value on real world observations [21].

To summarize why data and the understanding of the impact made by using said data is important:

- Without data, ML is non-existent. There are no experiences on which a machine may learn.
- Without *representative* data, ML systems at face value are likely worthless as they are untested.
- Without understanding what *representative* data is for the problem, trust in a system cannot be established.
- Without understanding how to procure *representative* data, time, money, and other commodities are wasted.
- Data is a fundamental component to ML systems; moreover, data is the hardest to change in a meaningful manner.
- In order to develop an ML system quickly, and produce meaningful results, sufficient high-quality data must be available or procurable in a reasonable timeline.

With the available software packages that can allow for diverse ML techniques to be applied to any data available in a handful of lines of code, such as Pytorch [23], Tensorflow [24], and WEKA [25] to name a few, both the technical approach and training routine can be modified and allow for blind, brute force approaches to determine the hyperparameter space that best suits the problem space. By direct comparison, the data available is not flexible and serves as the fundamental limiter for most ML applications, and therefore demands greater care in terms of curation to allow for achieving the highest level of performance for a system.

1.3 Contributions of this Research

Solidifying the concept of RFML in a manner that brings more value to the problem space rather than just applying ML to RF problems is a fundamental contribution within this dissertation and the underlying articles and authors from which this dissertation is compiled. The work does more than just distinguish between the two concepts, it allows for others to have a predictive estimator for what is needed in terms of time and complexity to take the many proof-of-concept articles, which have ballooned within the last six years, and see them

transitioned to applications applied in practical problems. An additional contribution that this work provides is a comparison point for the amount of data that is needed at this point in time, which can be used to show improvement with new approaches in the RFML problem spaces as research progresses, such as improvements in training routines and how changes in architecture can better fit the application. A summary of each of these contributions follows in this section.

Radio Frequency Machine Learning The application space of Radio Frequency Machine Learning was refined and described through the contributions made in works [1, 2, 21]. While the full *RFML Ecosystem* was developed as a joint work with several authors, the contributions provided in pursuit of this degree were predominantly in understanding of the application space that RFML currently resides, the underlying data space that encompasses each application, the concerns that need to be overcome with deploying a real system, and, to a lesser extent, the trust and assurance that is to be expected of such a system. In a sense, the work of RFML follows the logical progression laid out by the concept of SDR, under which the premise is to move as much of the processing as possible out from the hardware behind the ADC and DAC, and in the pursuit of CR, moving the ML as close as possible to the ADC and DAC, and thereby having ML as close as possible to directly interacting with RF spectra. From this progression, RFML is the understanding of interactions in RF spectra through ML systems, while minimizing and incorporating the effects of devices' imperfections representing said spectra. Note that this includes the degradation from emitters, the propagation effects observed within the wireless channels, as well as additive effects while using the channels, the degradation of the receivers, and any preprocessing that separates the ML from the digitization, while minimizing the preprocessing as is within reason.

L. J. Wong, **W. H. Clark IV**, B. Flowers, R. M. Buehrer, W. C. Headley, and A. J. Michaels, "The RFML Ecosystem: Considerations for the Application of Deep Learning to Spectrum Situational Awareness," in *IEEE Open Journal of the Communications Society*, vol. 2, pp. 2243-2264, 2021, doi: 10.1109/OJCOMS.2021.3112939. [1]

Data Quality and Quantity Relationship With the presumption that the problem space is observable and an approach has been devised that can extract relevant information from observations of that space, this contribution provides a means for comparing the quality of different datasets as well as a practical approach for quantifying how much of a given dataset is needed for a desired performance. One of the most fundamental components of any ML system is the available datasets that are used to train the approach. Through a case study on the RFML problem space of Automatic Modulation Classification (AMC), a large scale, diverse dataset was collected over a four month period in 2019 at Virginia Tech's Kentland Farms. This dataset was then used to understand the effect of several aspects about the propagation channel, what happens between the transmitter's DAC and the receiver's ADC. This case study and the developed dataset allow for an understanding of the effects

that quantity and quality have on a specific RFML problem space and generalizations that can be made to the full RFML data space.

W. H. Clark IV, S. Hauser, W. C. Headley, and A. J. Michaels, "Training data augmentation for deep learning radio frequency systems," *The Journal of Defense Modeling and Simulation*, 2021. [Online]. Available: https://doi.org/10.1177/1548512921991245 [21]

Data Quantity Prediction From the understanding provided in the AMC case study and the lessons and insights derived from the generation of the datasets used, the relationship between system performance and available quantity is expanded, providing deeper insight into quantity projections needed for training a model from random initialization into a high performance model on a known test dataset. By making use of metrics originating in the study of transfer learning, which have been developed to take a trained system and project its potential on a new dataset or data space, those same metrics can be applied within the same data space to evaluate a trained model's performance on a known test dataset while varying a parameter of interest to understand how that parameter affects a generalized performance. By then regressing the relationship between the transfer learning metrics and available data quantity during training, an improved estimate of total data quantity needed for the current training configuration can then be made that improves over the performance metric of accuracy given the asymptotic nature of the relationship between accuracy and data quantity, which in turn allows for estimations to be made from smaller quantities of available data with linear fits.

W. H. Clark IV, and A. J. Michaels, "Quantifying and Extrapolating Data Needs in Radio Frequency Machine Learning," *(submitted) Transactions on Machine Learning in Communications and Networking*, 2022. [26]

Quantifying Dataset Quality in Radio Frequency Machine Learning This work was an initial attempt to quantify the relationship between data quantity and the accuracy of the networks that were trained on said data. In this work, we showed that the data quantity and network performance can be regressed and used to quantify different datasets with respect to achieved performance for a consistent Deep Neural Network (DNN) architecture and training procedure.

W. H. Clark IV, and A. J. Michaels, "Quantifying Dataset Quality in Radio Frequency Machine Learning," *MILCOM 2021 - 2021 IEEE Military Communications Conference (MILCOM)*, 2021, pp. 384-389, doi: 10.1109/MILCOM52596.2021.9652987. [2]

Multi-Antenna Pre-processing for Improved RFML in Congested Spectral Environments By making better use of the resources available in large and dynamic radio systems, the underlying EMI present in the EME can be suppressed in a less restrictive manner than would be typically be allowed within a system reliant on expert features. This

relaxation to the difficult problem of EMI suppression is well suited to the RFML problem space as a preprocessing stage to simultaneously reduce the miscellaneous energy that can be inferred as a distraction while reducing the overall data throughput that would be required of an ML without it.

M. R. Williamson, W. C. Headley, **W. H. Clark IV**, *et al*, "Multi-Antenna Pre-processing for Improved RFML in Congested Spectral Environments," *2021 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, 2021, pp. 288-295, doi: 10.1109/DySPAN53946.2021.9677275. [27]

Developing RFML Intuition: An Automatic Modulation Classification Architecture Case Study The primary focus in this work was examining the work of expert systems that performed AMC and replicating that general decision structure with DNN to see if there was any benefit in a DL system. Effort was spent to take the initial architectures proposed by the coauthors to rebalance the networks to have similar number of weights and constrain the problem space such that the decision structures could more directly be compared rather than other aspects that vary within a DNN.

W. H. Clark IV *et al*, "Developing RFML Intuition: An Automatic Modulation Classification Architecture Case Study," *MILCOM 2019 - 2019 IEEE Military Communications Conference (MILCOM)*, 2019, pp. 292-298, doi: 10.1109/MILCOM47813.2019.9020949. [28]

When is Enough Enough? "Just Enough" Decision Making with Recurrent Neural Networks for Radio Frequency Machine Learning Looking into ways to further utilize a real-world system by reducing the volume of data that must be ingested before action can be taken. Given the standards of today's portable systems, any reduction in data volume through a processing system that produces some kind of analysis equates to more characterization provided over any device that must process the full observation before returning a result. By examining ways that RFML systems can provide reliable decisions on less data allows for more real-world capable devices.

M. Moore, **W. H. Clark IV**, R. M. Buehrer and W. C. Headley, "When is Enough Enough? "Just Enough" Decision Making with Recurrent Neural Networks for Radio Frequency Machine Learning," *2020 IEEE 39th International Performance Computing and Communications Conference (IPCCC)*, 2020, pp. 1-7, doi: 10.1109/IPCCC50635.2020.9391569. [29]

Overall, the cumulative contributions provided from my research into the RFML problem space have been focused on accelerating the pipeline from training a RFML model through actual operation in real-world environments. The work has helped define the RFML problem space, identify the limitations of transitioning the research into real-world applications, and examining different aspects of the problem to understand and reduce where applicable the experience and data volume constraints that exist in such applications.

Transforming RFML Data Generation Techniques for Real-World Applications Over the last decade, a significant research and development effort has been focused in understanding, improving, and creating new approaches for data generation to support the development of RFML systems [30, 31]. After developing a MATLAB implementation of dozens of modulators for research used in [32], there was a clear hole in the available tools for diverse modulation generation in order to support the RFML research community. The Out-Of-Tree Module (OOTM) for GNU Radio 'gr-signal_exciter' [33] was then created in the hopes of providing consistency, and enabling easier access such that RFML research could be done. This tool has been specialized on multiple projects [30, 31] to go beyond the initial modulation generation focus, though the core modulation structure hasn't changed, as well as supporting multiple masters and PhD efforts since its creation. The specializations focused on wideband spectrum aggregation of bursty signals [30], realtime streaming of narrowband signals for data collection campaigns [31], and expanding the available modulation formats and updating the codebase to keep up with new releases of GNU Radio. Unfortunately, not everything has made it back into the open source code base due to contract conflicts and lack of personal time available to handle the required effort to deconflict the source code. From this effort, the foundations were created, and continually utilized to create the foundations of what became SignalEye [34]. The capabilities from this effort also led to supporting the PHOTON framework developed by MITRE, though it is not understood if it has been directly or indirectly incorporated. At a fundamental level, the described effort helped develop understanding of how to create an experimental baseline for developing real-world tactical radio training datasets as well as tackling challenging Department of Defense (DoD) use cases.

W. Headley, **W. Clark IV**, and L. Wong, "General Dynamics (TREX) CogRF Phase III Final Review," Virginia Tech, Tech. Rep., 2018. [30]

W. Clark IV, and Z. Leffke, "Cyborg Phase II Final Report," Virginia Tech, Tech. Rep., 2019. [31]

1.4 Dissertation Outline

The remainder of this dissertation is organized as follows:

In Chapter 2, the background information for establishing the focus of RFML is presented, along with considerations that should be taken for bringing a problem from the lab to the field. The chapter starts with the fundamentals of ML, and DL in particular, then goes on to explain the general desire of ML applied to commercial and military applications. From there, the scope of using ML in the RF space is narrowed and an overview of RFML systems is presented.

In Chapter 3 the focus switches to the problem of generating the data. Understanding the differences between the origins of a dataset in terms of the process of how the waveform be-

comes translated from EME into bits is discussed. The challenges of creating a dataset from the observations of real transmissions are then discussed through experience in a small scale collection campaign over an extended observation window, and through observations about datasets curated through governmental programs. Additionally the inability to understand and forecast the costs of doing such a collection is addressed. Some open source toolsets are discussed, while focusing on the tools developed for this work. Discussion then examines the shifting from software based collection systems to hardware, and finishes with lessons learned through the course of this work.

In Chapter 4, the focus is put on understanding what role the dataset used while training the RFML system plays in the overall performance. In order to better understand the effect the data plays in a trained network, the other aspects of an ML training regimen, namely the approach and architecture, are held constant while the data is parametrically varied while training many networks. This approach for analyzing the data allows for quantitative observations that objectively provide a quality to different types of data sources, while allowing for an understanding of how the quantity of available data plays a crucial role in the system's final performance. Finally, the available ways of maximizing performance while subject to limited data availability are discussed within the context of the RFML problem space.

In Chapter 5 the considerations to bring a system to the field and have positive results are discussed. Focus is given to commercial whitespace, signals intelligence, signal detection, RF Fingerprinting / Specific Emitter Identification (SEI), and hardware realization of data generation tools. The focus on commercial whitespace and SEI are heavily based on observations of published works and public projects, while signals intelligence, signal detection, and hardware realizations are backed by first hand experience.

In Chapter 6 the works discussed herein are tied together to better give an overview of the RFML problem space. Based on the work within this dissertation, and the experience gain while conducting the experiments and subsequent analysis, new possibilities and suggestions for future research are provided.

Chapter 2

Background

The RFML application space is the fusion of ML and problems within the RF spectrum that attempt to characterize, understand, and predict the usage for developing improved utility. To that end, this chapter provides background into the fundamental components of such systems. First, the primary ML approach used within this dissertation known as DL is discussed and imparts why this technique is well suited to problems in the RF domain. Next, the problems within the RF domain being examined with ML are discussed in terms of the commercial and military application spaces. The overall RFML problem space is then examined in terms of the concerns and considerations that need to be present with the fusion of ML approaches to RF problems. Finally, the considerations that go into the data for improving performance in RFML applications are discussed.

The concepts described here in are provided to serve as a foundation on which future chapters are built and whose understanding will enrich the concept and highlight need for RFML applications. By no means will this serve as complete volume of the topics discussed within, but rather shed enough light on them to provide insight for the experiment's utility.

2.1 Deep Learning

While the utilization of DL technology is fundamental to this dissertation, attempting to piece together the fundamental history of the field is beyond the scope of this work; however, for those interested in understanding how deep learning came to the current iteration, Schmidhuber [35] provides an in-depth look into the history of DL, which has technical roots in the 1800s, with its present form more easily identified in the 1960s, with special attention provided to the inspirations that allowed the field to make leaps in technical capabilities. That being said, a greater focus here is foundational understanding of DL along with the core concepts that are utilized for better performance within RFML.

DL is fundamentally built on top of Linear Algebra, Probability and Information Theory, and 'Mu'ltivariable Calculus that is within the category of Neural Networks (NNs) within ML. Goodfellow *et al.* provide a detailed dive into the topic of DL and DNNs [36]. ML consists of the input-output relationship $\hat{y} = f(x)$, where the examples, $x \in \{X\}_1^N \in \mathcal{X}$, are processed through the technical approach, $f(\cdot)$, to produce an output, $\hat{y}$, that provides more utility to the desired task. The outputs, $\hat{y} \in \{\hat{Y}\}_1^N \in \hat{\mathcal{Y}}$, are then processed through a metric of

performance, which when *supervised* makes use of an expected *target*, $y \in \{Y\}_1^N \in \mathcal{Y}$, or alone in the *unsupervised* case, though the lines between *supervised*, and *unsupervised* approaches are often blurred [36]. In order to be qualified as a learning system the net performance on the examples as a whole should improve with every additional example or batch of examples passed through the technical approach. This means that adjustments should be made within the technical approach (the weights) based on the metric observed in order to improve performance without a human-in-the-loop tuning the system. In DNN systems, the technical approach can be described in terms of layers, which in a strict *Sequential* approach can be described as $\hat{y} = (\circ_{i=1}^{N} f_i)(x)$, or for two layers $\hat{y} = (f_2 \circ f_1)(x) = f_2(f_1(x))$, where $\circ$ notation represents composition of functions. However, in the general sense, nothing stops the layers from operating on the same input value in parallel branches, or from a layer operating on multiple inputs or previous layers as long as causality is followed, e.g. $\hat{y} = f_3(x, f_1(x), f_2(f_1(x)))$. Traditional definitions of layers describe the layer as an operation on the input with some stored weights and/or biases followed by an activation function to express non-linear operations such as $\boldsymbol{y} = \sigma(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b})$, the *sigmoid* $(\sigma(\cdot))$ of a matrix multiplication and vector addition; here, for simplicity, the approach described in PyTorch's implementation [23] as a module where a layer may be constructed in terms of a functional input and output of any kind, the previous example can be described as two layers, where only the first layer has stored weights and biases, $\boldsymbol{y_1} = f_1(\boldsymbol{x}) = \boldsymbol{W}\boldsymbol{x}+\boldsymbol{b}$; $\boldsymbol{y} = f_2(\boldsymbol{y_1}) = \sigma(\boldsymbol{y_1})$. This separation of activation and operation with weights allows for a more explicit definition of the overall architecture and leaves less room for misinterpretation of model descriptions, as well as allows for deterministic transforms such as the Fast Fourier Transform (FFT) to be encompassed in the layer notation. An additional benefit of this structure decouples the activation from the weights in the layer such that systems that change the activation as part of the training of the network can just swap layers cleanly rather than changing a component of a layer.

2.1.1 Understanding the Functional 'Layer'

A practitioner of DSP will recognize much of the technical structure involved in the layers that make up a DNN, only that instead of focusing on preserving the linear nature of a signal as is typical in DSP, DNNs abandon linearity in favor of non-linear functions. The choice of how the layers connect and intertwine within a DNN architecture are what make up the Technical Approach, which serves as one of the three pillars upon which successful ML systems are built. In order to understand the concept of a layer better, here some of the more common layers and activations are described for convenience:

Linear / Dense / Fully Connected Layer The Linear layer is the most functionally basic layer that utilizes weights and can be thought of as a matrix transform layer. That is, given some input multi-dimensional matrix, X, of shape $(\ldots, F)$ that ends with F values in

the last dimension, the Linear layer then is a matrix multiplication of the form

$$\begin{aligned} \boldsymbol{y} &= \boldsymbol{W} \cdot \boldsymbol{x} + \boldsymbol{b} \\ \boldsymbol{Y} &= \boldsymbol{X} \cdot \boldsymbol{W}^T + [\boldsymbol{b}^T, \ldots, \boldsymbol{b}^T], \end{aligned} \tag{2.1}$$

where $\boldsymbol{x}$ is the vector of length F that is transformed by the weight matrix $\boldsymbol{W}$ of shape (F', F) and added to the bias vector $\boldsymbol{b}$ of length F'. The output of the layer maintains all of the same dimensions except for the final one to take on the shape of $(\ldots, F')$. In general, the bias is optional in this layer, and in the case it is omitted, then $\boldsymbol{b} = \boldsymbol{0}$ and is not allowed to change, whereas the weights matrix must be present and update when the training approach increments. A visual representation of this layer is given as the last layer in Figure 2.1, where the input vector is of length 3060 and the output is reduced to 10.

Convolutional Layer The convolutional layer operates on an input multi-dimensional tensor of shape $(\ldots, C, D_1, D_2, \ldots, D_N)$ and utilizes a weight tensor of shape $(C', C, K_1, \ldots, K_N)$ and bias vector of length C'. This layer performs the discrete convolution operation between the weights and input tensor with C channels and produces an output with C' channels, where the N output dimension's lengths change as a function of the input dimension size $(D_1, \ldots, D_N)$, the kernel size $(K_1, \ldots, K_N)$, and the hyperparameters of stride and dilation along each data dimension. Most common implementations limit the value of data dimensions $N \in \{1, 2, 3\}$, though in the most general sense, there is no limit to the number of dimensions that can be present.

$$\boldsymbol{y}[:, n_1, \ldots, n_N] = \sum_{d_1} \ldots \sum_{d_N} \boldsymbol{W}[:, :, d_1, \ldots, d_N] \cdot \boldsymbol{x}[:, n_1 - \delta_1 \cdot d_1, \ldots, n_N - \delta_N \cdot d_N] + \boldsymbol{b} \tag{2.2}$$

The resulting multiplication is effectively a linear layer operating on a subset of the input data for each summation step. The benefit of this approach is a learned location invariance for the output of the kernel such that each of the resulting (C', C) matrices extracted from the weights kernel W can be fine tuned to find a specific characteristic of the data without concern of exact position in the data structure. While this seems to have significantly more computation than the linear layer, the structured formula is well suited to a Graphics Processing Unit (GPU) and allows for the size of the kernel to be significantly smaller than a linear layer that operates on the product of dimensions $(\prod(C, D_1, \ldots, D_N), \prod(C', D_1, \ldots, D_N))$. If the number of data dimensions is set to $N = 1$ and the input and output number of channels are bound such that $C = C' = 1$, then the convolutional layer functions identical to that of a DSP Finite Impulse Response (FIR) filter on a time series signal. In a typical implementation, if the above kernel tries to access data in the input matrix outside of the valid dimensions, a zero is assumed in the place of the data. A visual representation of this layer is given as the first layer in Figure 2.1 where the input In-phase and Quadrature (IQ) observation is first changed from complex baseband to a two channel time series tensor, one for I and the other for Q. The two channels then get increased into a 10 channel output tensor where the length of the time series is reduced by a length of 4, due to the kernel size of the layer.

Recurrent Layer There are several types of recurrent layers, with two of the most popular layers in RFML applications being the Long-Short Term Memory (LSTM) and Gated Recurrent Unit (GRU) layers. These layers fundamentally operate on the concept of an ordered sequence, most commonly with a temporal assumption. In these cases, the layer operates on the sequence in an elemental order rather than in mass parallel approaches such as the way a convolutional layer can operate, which in turn typically makes these layers significantly slower than other layers. The typical assumption is that there is a temporal feature space that changes over the ordered sequence, such that the input multi-dimensional matrix takes on the shape $(\ldots, L, F)$ and produces a sequential output of $(\ldots, L', F')$; however, a common approach to these sequence layers is to "unroll" along the ordered sequence, allowing for the L' value to vary based on the unrolling from 1 to L based on the implementation. At a fundamental level, these operate on a similar principle to DSP Infinite Impulse Response (IIR) filters with a single pole, though this is more easily observed in the GRU layer rather than the LSTM layer, with the caveat that the weight of said pole is driven by a mixture of both past output and current input rather than statically assigned. The computation then falls back to a linear layer in terms of a simplified recurrent stage (2.3),

$$\boldsymbol{h}_l = \boldsymbol{W} \cdot [\boldsymbol{h}_{l-1}; \boldsymbol{x}_l] + \boldsymbol{b}, \tag{2.3}$$

where the previous step output, $\boldsymbol{h}_{l-1}$, is concatenated to the features of the current step, $\boldsymbol{x}_l$, before going through the linear layer process. Most recurrent layers have multiple linear layers within them rather than a single one as shown here for a single sequence step. A visual representation of this layer is given as the second layer in Figure 2.1 where the 10 channel time series tensor is compressed into a 3 channel output of the same length.

Batch Normalization The layers above utilize their weights in a manner to transform and separate features in the data in order to better perform the task of the network as a whole, while Batch Normalization instead focuses on minimizing the covariate shift that the other layers introduce through their processing [37]. Essentially, as the training procedure goes on, it's possible for a layer's output to produce values excessively large or small, making the deeper networks have problems where a small differential applied to the weights becomes insignificant and harder to tune relative to the overall result. This problem in essence is caused in part by the distributional changes associated with different looks at data within each training batch, and in order to minimize the effect the first two moments are normalized by a single pole IIR filter applied to the batch estimates of mean and biased standard deviation measurements. These learned normalization values are then applied during inference and have the best effect when the distributions between training and inference data spaces are similar. Additionally, this layer has a scale and center shift value per channel to allow for changing the first two moments to better utilize the following activation function whose defaults are 1 and 0, respectively, to start from a state where only the normalization step takes place and any changes should be a learned benefit. A visual representation of what a Batch Normalization Layer should do in a well-trained system per feature is provided in Figure 2.2 top left corner.

Dropout One major difficulty in DNN, especially when the size of the dataset is on the smaller end, is the problem of overfitting, which is where the network learns the exact pattern of data being trained on and fails to generalize to the problem space at large. In these cases, the application of Dropout helps combat that learned memory by dropping out values randomly during training such that, between two layers, the full value is never seen during training. Dropout is then bypassed, or disabled, during inference and the full network is able to be utilized to come to a decision. A visual representation of what a Dropout Layer should do during system training is provided in Figure 2.2 bottom left corner for a single observation, where with each batch, the indices that become zero would change.

Interestingly, enabling dropout during inference is one method of determining the *uncertainty* of the network in the current observation [38].

Sigmoid (σ) The sigmoid activation is primarily used for constraining the values of a system to the range of $[0, 1]$ and heavily utilized in layers with gating functions, such as the LSTM and GRU layers.

$$\sigma(x) = \frac{1}{1 + \exp(-x)} \tag{2.4}$$

A visualization in the domain range of $[-2, 2]$ is provided in Figure 2.2 bottom right, with the upper and lower limit bounds shown.

Hyperbolic Tangent (tanh) The hyperbolic tangent function is another function that is heavily used in bounding output values into a constrained range, in this case the range $[-1, 1]$. Similarly to the sigmoid activation, it is most frequently used within recurrent layers in order to influence the gating effects.

$$\tanh(x) = \frac{\exp(2x) - 1}{\exp(2x) + 1} \tag{2.5}$$

A visualization in the domain range of $[-2, 2]$ is provided in Figure 2.2 bottom right, with the upper and lower limit bounds shown.

Rectified Linear Unit (ReLU) The most common activation utilized when training a model from a random initialization (or some extended version of this) has been found empirically to cause networks to converge more quickly than the sigmoid or hyperbolic tangent activations.

$$\text{ReLU}(x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{o.w.} \end{cases} \tag{2.6}$$

Another common tendency is to swap out the ReLU activation with a leaky version, by which it means that instead of negative values being clamped to zero, some reduced amount

is allowed through, be it the Leaky ReLU activation where negative values are attenuated by a scalar value, or Sigmoid Linear Unit (SiLU), $\text{SiLU}(x) = x \cdot \sigma(x)$, where small negative values are passed through at a diminished value while larger negative values quickly clamp to zero. A visualization of ReLU in the domain range of $[-2, 2]$ is provided in Figure 2.2 bottom right, with the lower limit bounds shown.

Softmax Softmax activation is often used in classification problems where there is a finite choice between classes to provide a probability estimate of which class is actually present. An important note is that in the general case, this probability estimate is not very good in terms of the true probability for each class's presence, but because values are bound in $[0, 1]$ and sum to 1, interpretation as a probability density function is often abused for simplicity.

$$\text{Softmax}(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \tag{2.7}$$

A visualization of a single input realization with 10 output values is shown Figure 2.2 top right, in blue the input values are shown, while in red the output values are shown.

Element Operations Here, again for simplicity, operations that take in inputs and produce an output are abstracted in the concept of a layer, and those operations include the addition and multiplication of matrices when combining the outputs of other layers and inputs. Additional operations that fall into this category combine matrices through concatenation, as well as those whose operation alters the shape of a matrix such as the transpose operation and reshaping or flattening of a matrix.

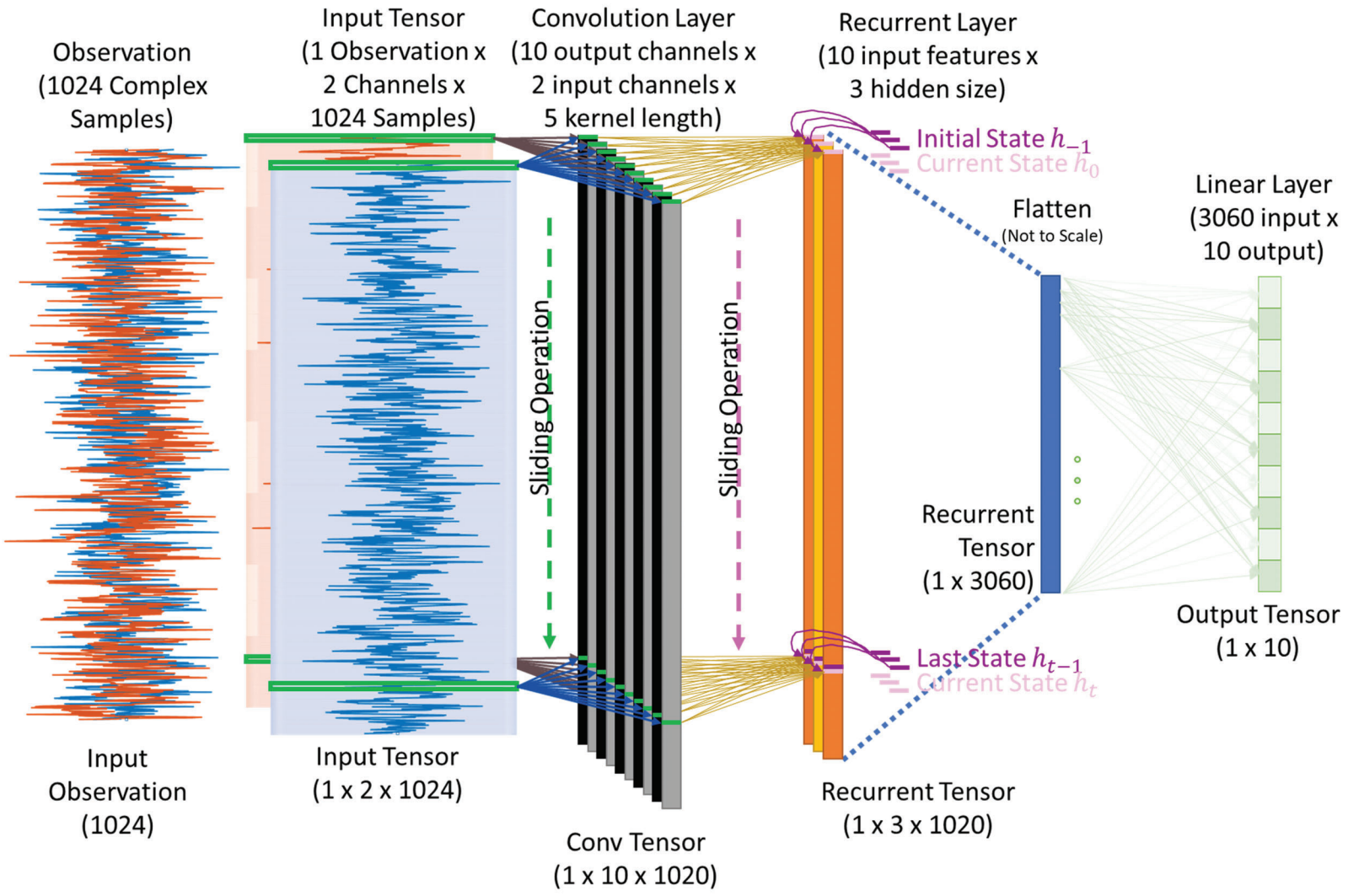

Figure 2.1: This simple network consists of a single layer of Convolutional, Recurrent, and Linear. The input is a synthetic signal and shows how the input observation can be reshaped into a tensor for interfacing with the network. Convolutional Layers can perform all steps of the sliding operation at the same time, while Recurrent Layers must iterate sequentially in order to make and use the 'Last State' value.

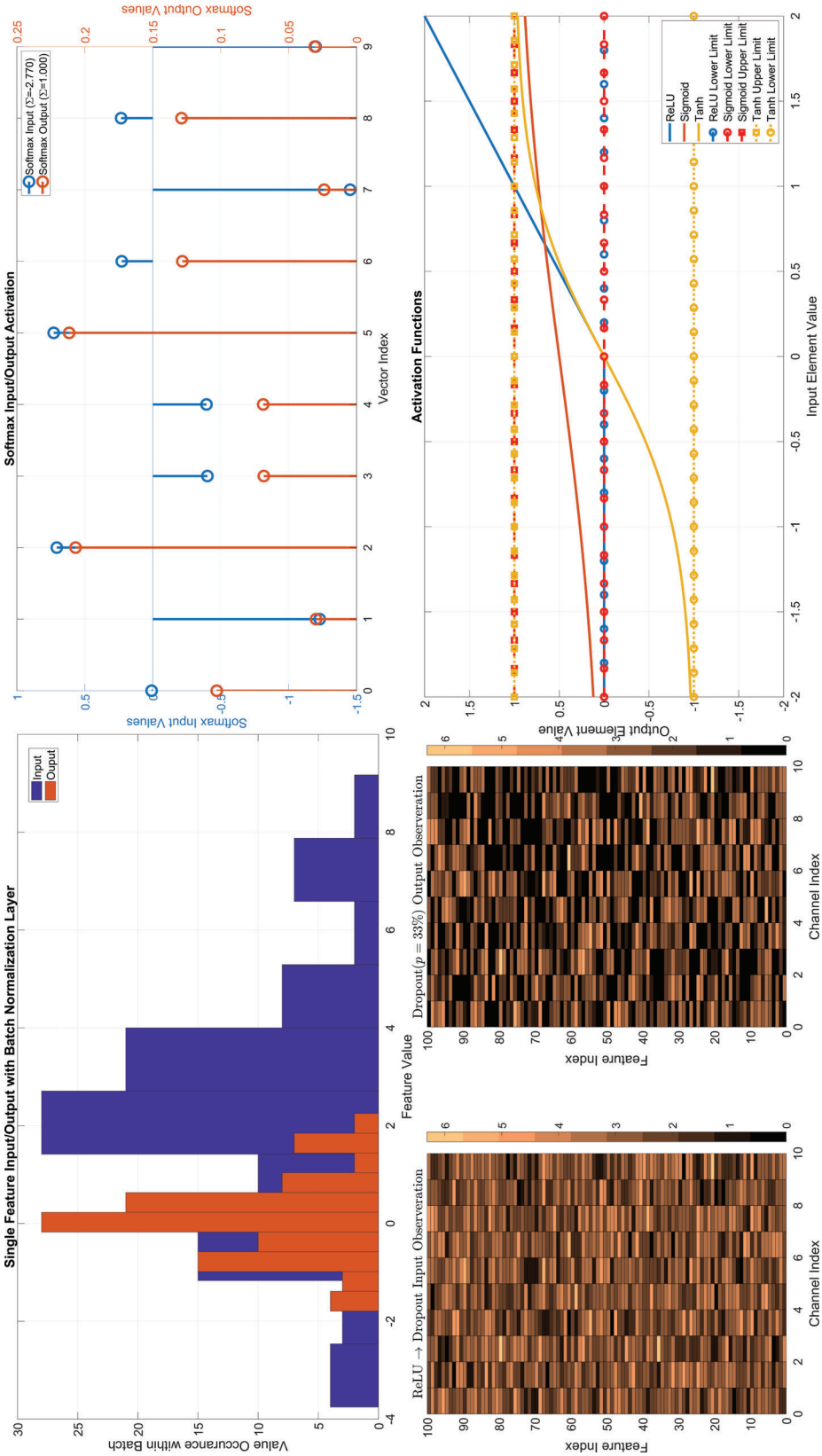

Figure 2.2: The left side of the figure shows the intended role of regularization within a network where the top left corner shows how a Batch Normalization Layer is intended to offset the covariate shift that occurs in network layers, while the bottom left corner shows the effect of Dropout (shown after a ReLU activation for clarity) where roughly 33% of the image area becomes black (zero valued) following the 33% dropout layer. The top right shows Softmax Activation that acts on the full input vector as a whole, while the bottom right shows the Activations that act per element, and provide the functional bounds were possible.

2.1.2 Training the DL Architecture

The *Training Routine* for ML systems typically used with DNN architectures relies on back-propagation of loss, or the performance metric on which the system is measured, to provide the gradient for how the system performance is expected to change for each weight and bias in the full network through the application of partial derivatives by using the chain rule in calculus [36]. The optimization algorithms that leverage that gradient are then responsible for translating the gradient into updates for each tunable parameter in the model during training. The choice of optimization algorithm, all of the choices that go into the structure of the architecture, as well as what data is made available to the training routine and when that data is available, are all parameters that the training routine has control over. The collection of all design choices with the model are called the *hyperparameters* of the architecture and part of the training routine can optimize over these often discrete options. Two fundamental approaches for selecting high performance *hyperparameter* sets are *grid-search* and *random-search*.

Grid-search optimization built into the training routine is effective at establishing trends of how individual parameters in the hyperparameter space affect the output performance; however, the use of a grid implies a direct understanding that there are characteristics that will not be observed between the search points. Fig. 2.3 presents a hyperparameter search when looking at just one parameter's effect on the overall performance of the system. The solid blue line represents the true effect this parameter has on the performance of the system, while the scatter points represent different trials and the resulting performance on some validation dataset. Now since a trained model's performance is dependent on more than one parameter, the observed performance is a noisy observation of the true performance of the chosen parameter. By performing a grid-search, the general quadratic trend could be observed, but if too few points are observed as in the Wide Grid observation in Fig. 2.3, establishing a trend in the wrong region is possible as a feasible region here would be in the range $[2, 4]$, while an ideal region would be centered on 5. By direct contrast, using a Fine Grid will not be so easily misled, allowing for a more ideal range centered on 5. The tradeoff comes to training significantly more models and spending more time and resources to examine this problem, while still not being able to observe periodic differences in the parameter that has a more significant impact than the precise value of this parameter. Ideally, choosing a medium size grid that can help narrow the parameter region, which this example would allow for a narrowing to the range $[3, 6]$, would be more practical. After narrowing the range with the medium grid, a random search for a final selection can be done while also optimizing other parameters; however, knowing what size grid is appropriate comes from experience and/or detailed knowledge of the data being observed. In the case of a truly unknown problem space, utilizing an initial random search often leads to faster convergence to a near optimal result, or can minimally establish a range over which a grid search might be appropriate.

The general assumption made in this dissertation is that a model is being developed from

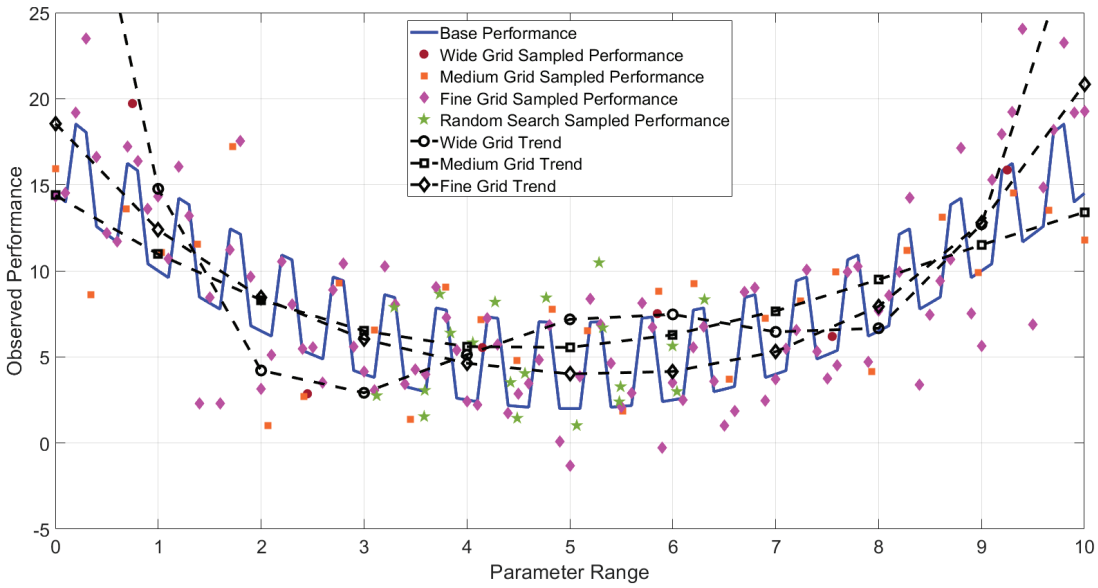

Figure 2.3: An example of how Grid-search and Random-Search can be used to optimize on the hyperparameter space by finding trends (dashed lines) for a parameter. The true effect of changing this parameter (an unknown) is modeled with the solid blue line, but due to other parameters, there is some unknown noise on the final performance, but the overall goal is to choose the parameter value at the minimum, or a value near to the minimum. With too wide of a grid both large and fine scale trends can be missed, while too fine of grids result in a large search space, which still doesn't guarantee that fine scale characteristics are observed. Choosing a medium size grid typically comes from experience or expert knowledge about the data space. However, predicting the fine scale properties will likely become an impractical search problem, so after narrowing to a likely operation region, switching to a random-search is more practical than choosing a specific value while optimizing over the full hyperparameter space.

scratch, which means that the model's initial weights and bias values are randomly initialized. When models are being developed for a classification task, the random initialization results in a system whose performance is on par with random guessing and often requires a large number of training steps to move away from random guessing in any significant manner. By curating the data to a simplistic set, or by drastically limiting the available data during training, thereby forcing the model to overfit, the challenges of overcoming a random initialization can more quickly be addressed over initially providing all available data. Choosing what data to provide to initially *prime the network* is a research area in and of itself and is not covered in any detail here, but some observations on ways this might be achieved are discussed in Chapter 4 based on observations taken in the AMC application study, though it's important to note that recent trends have networks being pre-trained with unsupervised techniques [39]. Techniques seen within the image and larger ML community rely on TL [17, 40], where a much more complex dataset along with long training routines are used to

create a primed network that can then be used to overcome the initial randomization. While TL is not the focus of this work, concepts developed within the research area will be utilized to help understand data quality later on. TL is viewed here as a staged approach taken by the training routine; after all, the initial model that is used as the *source* must be trained in some fashion to overcome the random initialization, so all discussion of TL within this work is taken as means to overcome the random initialization and not with regard some highly trained model that can be refined to the problems at hand.

Transfer Learning The most general definition of TL is taking a trained model on some *source* task, then allowing this model to go through a new training procedure to perform on the *target* task. This approach can allow for the full network to go through *fine-tuning*, where all weights and biases are free to change during the training on the *target* dataset [41]. In contrast to fine-tuning, the second most common approach freezes the weights and biases in the majority of the network, where these parameters are considered part of the feature extraction routine, and replaces the networks *head(s)*, thus only training a new *head* for the *target* dataset. The most common usage of a *head* is a linear layer or series of linear layers that take the features extracted from the frozen layers and fuse them to produce the result of the task at hand.

2.1.3 Role of Data

DL with a DNN architecture relies on gradients commonly produced with backpropagation and an optimization algorithm to tune the layers in a manner to make them useful for the desired task. The role of data is explicitly to provide a means for producing *meaningful* gradients. For the RFML task of AMC there's nothing that stops a developer from training a network using pictures of cats and dogs to train a network for that task, but there are more efficient approaches to take.

One of the most basic ways to train an efficient network to do a task is to provide the exact task that it should accomplish and then iterate until the task is successfully complete. However, a fundamental assumption of the above is that the problem space is sufficiently small and accessible that practice can occur until perfection, but it does not take into account the generalizations to achieve real world operation. For example, here are two descriptions that if properly executed should perform the same task:

- "Take the spoon from the edge of the counter at a height of 1 meter and place it on the table 3 meters away with the handle pointing to the nearest edge of the table."
- "Take *that* from the counter edge to the table."

Both commands above should be able to take a spoon from a counter and place it on the nearby table, but the first description carries significantly more precise instructions for the

situation. If a system becomes highly tuned to this first command, then the generalization of the system to move forks, cups, bowls, or any other object along with counters and tables of varying heights and distances will be lost.

The fixation on specific details in the data rather than the task at hand can be interpreted as a failed generalization in two ways. The first way is commonly called overfitting and can be treated as the problem with the network and training routine effectively memorizing the data, creating a matched filter that knows how to exactly extract the desired information it has seen before. Overfitting can be addressed through regularization layers, as discussed with **Batch Normalization** and **Dropout** layers, being added to the network to offset memorization. By enlarging the dataset to exceed the memory capacity of the network, the training routine can potentially force the network to make the desired generalizations in order to improve the performance metric. The second way is often more subtle and only really becomes obvious when a system is put into active use. This problem is called *machine bias* and occurs where a trained system used in practice mimics unseen/known biases in the data itself [39, 42]. The most concerning types of machine bias with regard to sociopolitical concerns come into play with focus around humans and the information surrounding them [43]. Humans are imperfect by nature and any bias or ignorance they posses can be reflected in the data and analysis produced by them, but it is important to remember that systems produced by humans can inherit those biases, for example search engine results [44].

In order to provide meaningful gradients for networks to train on, the dataset used to train the network must in turn be of high quality, where quality can be described with three principles [45]: *Comprehensiveness*, *Correctness*, and *Variety*. Chen *et al.* [45] go into detail on what each concept encompasses. Summarizing their work, *Comprehensiveness* is the meaning that the dataset has the information relevant to the task at hand. *Correctness* entails that any underlying metadata that is incorporated during training is accurate to the task, thereby free of preventable biases and not subject to adversarial manipulation. The third principle of *Variety* is the primary focus within this dissertation and consists of making sure the dataset used for training has appropriate distributions within it to the intended task. For the sake of simplicity, *Comprehensiveness* will be assumed throughout the work for all datasets used, as will *Correctness*; however, discussions on how to preserve *Correctness* will be addressed as appropriate within RFML application spaces. The concept of *Variety* will be explored in more detail in the RFML domain to better understand and learn how this principle hurts the final performance of networks when not appropriately addressed.

2.2 Commercial RF

From the statistics collected by CTIA [6] that were discussed in Chapter 1, the use of wireless technology is growing exponentially. With the current release of 5G and the ever growing list of devices that will be present in the Internet of Things (IoT), ML is becoming a fundamental technology to meet the heavy demands and constraints [46].

Table 2.1: The three principles of dataset quality: *Comprehensiveness, Correctness*, and *Variety*. Provided within this table there is one case where the principle is met, and another where it is lost for the case of training an AMC network.

Principle	Higher Quality	Lower Quality
Comprehensiveness	All modulations present and balanced	Only a single modulation scheme present
Correctness	Every observation represents the desired modulation and is labeled properly	Co-channel interference present during recordings, potentially adversarial
Variety	Observations represent the expected degradations for usage in the field	Collections are purely synthetic and contain no distributions of nuisance parameters

2.2.1 Spectrum Shortage

A fundamental application space for ML in the commercial domain is getting a better handle on the problem of Spectrum Shortage. The principal understanding of the shortage of spectrum comes from the limited resource known as 'radio spectrum,' which resides in the finite electromagnetic frequencies between 3kHz and 300GHz, combined with continual exponential growing demand [13]. Since the establishment of the Federal Communications Commission (FCC) in 1934 and the National Telecommunications and Information Administration (NTIA) in 1978, the RF spectrum has been carved up into dedicated slices of spectrum for specific system or application uses in the USA and has often influenced worldwide allocations. Berlemann and Mangold define wasted and/or unused spectrum as spectrum that is allocated to failed systems, occasional or sparse utilization of allocated spectrum, or spectrum that has been allocated in a manner that fails to match technological improvements [13]. Therefore, by their estimates in 2009, 90-95% of the spectrum at any given location is unused and thus the 'scarcity' of spectrum is "an artificial result of the way spectrum is regulated" [13].

The fundamental problem is then: how one both utilizes and regulates the unused spectrum that is beneficial to a Secondary User (SU) without infringing on the Primary User (PU), all while maintaining a desired level of Quality of Service (QoS) to support the business infrastructure surrounding the commercial RF domain. What makes this problem even more difficult is the understanding that the burden of utility and onus of impairment compensation fall to the SU as oftentimes the PU will continue to act as if it has sole access to the resource. Therefore the SU has to operate typically in an uncooperative manner, meaning it must both adapt and be regulated at speeds that will exceed the capabilities of human operators such

that PU QoS is not impacted.

2.2.2 Spectrum Sensing

In simplest terms, the solution to the problem of spectrum shortage is being able to understand when and where spectrum is and isn't being used. For the cases where spectrum is allocated for a failed system, or where technical advancements have left a known deterministic hole, the spectrum can be thought of as idle and can be used by a SU. However, even in these cases where the PU can be thought of as absent, while still ignoring the cases where the PU is an infrequent user whose access is at best predictable if not outright non-deterministic, the SUs must still contend among themselves in order for any true effort at QoS to be made among SUs.

Frequency usage is influenced by a diverse set of deterministic and non-deterministic parameters, such that for any one device to have complete knowledge should be considered impossible. For instance, even if all of the deterministic parameters can be accounted for in terms of where a two-way radio can make use of the spectrum, whether or not the spectrum is ever in use is determined by random presence of a user at the time and location along with having such a device and then using it. Taking it one step further, the problems inherent to finite devices give rise to the complications of the hidden and exposed node problems [4] where even a perfect observer prone to make no error in the realm under its purview cannot make the ideal decision because there exist things outside the perceivable domain. Figure 2.4 presents visualizations of the hidden and exposed node problems, where the circles represent the transmission footprint or interference range of the color matched transceiver. At the top of the figure, the hidden node problem exists where SU-A senses the desired spectrum is free at its location, and even with perfect knowledge has no way of knowing on its own that PU/SU-C is also using that same slice of resources. For the sake of argument, assume that spectrum is used in a first-come/first-served basis and PU/SU-C is already using the resource for their system. When SU-A transmits with perfect knowledge of the local EME stating that the spectrum is free, SU-A produces interference to system C's receiver in a way that, without external feedback, SU-A would not be aware of. The hidden node problem is therefore a difficult problem to overcome without *a priori* knowledge provided in a cooperative system, the likes of which might be infeasible or impractical for every scenario that ultimately serves as a limit to QoS in a shared medium. On the opposite side of the problem space shown as the bottom configuration in Figure 2.4, the exposed node problem is shown. In this case, SU-B wishes to transmit to SU-A while PU/SU-C is actively transmitting. SU-B senses the use of the spectrum and chooses to concede the use of the spectrum to be fair; however, if SU-B transmits, no interference would befall system C, meaning that reuse was possible and potentially ideal in terms of reaching the ultimate capacity of the spectrum. In this scenario, there is still the problem of Spectral Shortage as by regulation of being fair; usable spectrum sits idle.

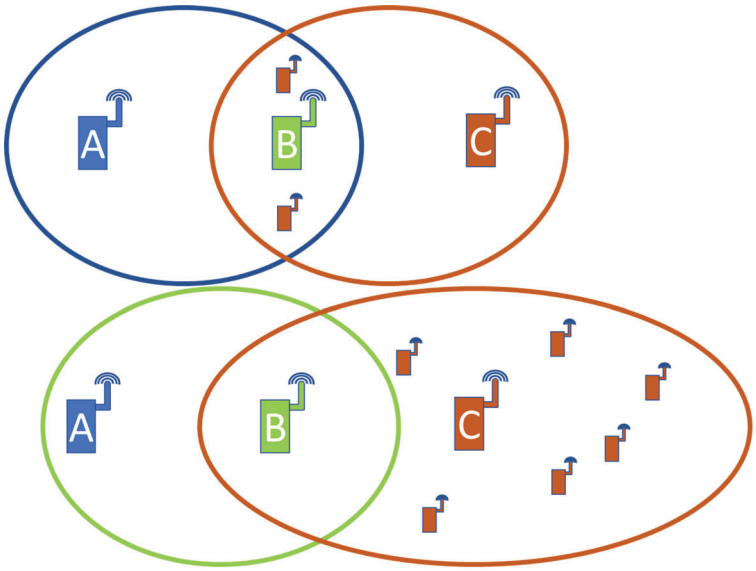

Figure 2.4: Examples of the hidden node problem on top, and the exposed node problem on bottom. In the *hidden node problem*, SU-A wishes to communicate with SU-B in the frequencies occupied by PU-C, however in doing so will interfere with receivers in the PU system, yet has no means on its own to detect PU-C's presence. In the *exposed node problem*, SU-B wishes to communicate with SU-A, but because it detects that PU-C is actively using those frequencies, it will choose not to use this frequency band even though SU-B's usage will cause no interference with the PU system.

While both conditions present limitations on what is feasible, they both examine the situation where each device is limited in the resources available and must fend for itself in the decision making process. The body of work in the realm of Spectrum Sensing (SS) is ever evolving and has a fair amount of ML work well established within it, while focusing on how to broaden the knowledge available to the radios to make better decisions about when and where to use the spectrum [47, 48, 49, 50, 51, 52, 53, 54].

2.3 Military RF

The exchange of information through communication systems or the acquisition of more physical situational awareness through the use of signal intelligence (SIGINT) and radar systems are reasons that the military has a vested interest in RF spectrum allocations. A fundamental truth for the military is that if a threat cannot be observed, it cannot be handled or mitigated. Therefore, it should come as no surprise that Spectrum Sensing is a critical component of military systems and has the additional hurdles of not just caring about finding when the spectrum is empty, but caring about the signals that are present as

well. Monitoring the of whole of the RF spectrum is desired in order to provide any possible tactical advantage, especially in the field of battle, but the techniques and tools available to monitor the spectrum are highly advanced, tuned, and require significant training to even begin understanding. The ideal simplification for the military would be to automate this monitoring, minimize the possibility of missing a critical observation, and distill down the many gigahertz of spectrum into actionable intelligence that any soldier can understand. This includes improving the hardware capabilities, while reducing the Size, Weight, and Power (SWaP) constraints of the radio enough to have them be mass produced, at the same time condensing years of training and experience at spectral surveillance into an application that not only can analyze the spectrum, but also produce bite size information such as *allies are to the east, and threats are approaching from the south*, all working in real time.

Over the last few years, numerous competitions and programs have been created to encourage the government, industry, and research communities to apply their joint understanding of ML into the RF problem space. Among the competitions were the Army's Signal Classification Challenge [55, 56], focusing on AMC with poisoned datasets, and the Defense Advanced Research Projects Agency (DARPA) Spectrum Collaboration Challenge (SC2) [12], focusing on spectrum sensing, planning and usage. Programs outside of the challenge space like DARPA's Radio Frequency Machine Learning Systems (RFMLS) focused on problems that had yet to see as much public research as AMC, like SEI [57]. Active programs like Intelligence Advanced Research Projects Agency's (IARPA's) Securing Compartmented Information with Smart Radio Systems (SCISRS) program take that a step further to characterize unintended, anomalous, and/or covert waveforms, for which training data may be difficult to collect as well as blind detection and reaction in an RF space to a waveform that may not conform to regulatory confines [58].

2.3.1 Spectrum Reallocation

The continued commercial growth of spectrum usage and the value seen in selling off the spectrum to the highest bid has caused a reevaluation of spectral use by the federal government in general. One of the more recent concessions relinquished 100MHz of military radar spectrum for commercial use, intended to supplement the ongoing rollout of 5G in the cellular market, from the DoD at 3.45-3.55GHz [59]. However, the reallocation of spectrum from military is nothing new and often has significant costs for the federal government to accomplish as the hardware in use for those bands is typically highly specific and requires acquisition of new hardware and training for all those affected [60]. One hope for the usage of SDR systems going forward is that the reallocation costs can be minimized to software changes rather than replacing the physical hardware and retraining the users.

Fundamentally, the challenge is to investigate and understand the overall impact these reallocations have on military readiness. For instance, if there's interference generated nearby, physically and spectrally, from commercial sources, that could hamper the functionality of

critical military systems and should be managed. As wars typically are not nicely planned out, having the ability to regulate, and potentially remove conflicting systems from military contention are desireable to have prior to the release of such a strategic resource.

2.3.2 Competitive Use of the RF Spectrum

In times of peace, use of the RF spectrum typically is commercial in nature and, outside of preallocated bands, competition among different entities falls in the domain of trying to acquire and use as much as possible according to some rules of conduct. In times of conflict those rules of conduct no longer apply, and while still in theory limited by the rules of war, the goals of such systems can shift between desire to use, and desire to deny use to others. The observed contention shifts from one of accidental and/or greedy interference to one of malicious denial and jamming resulting in a much more difficult and perilous environment to operate in in terms of resource access and physical defense. The ongoing Russo-Ukrainian War offers first-hand understanding of this kind of competition and how with proper planning and staging it can have a devastating effect, but with that same kind of planning and staging the effect can be mitigated as well [61].

2.3.3 Looking to the Future

Older static systems have shown fundamental flaws in both times of peace and during competitive conflicts. During peace, they add unused allocations to the military and cause spectrum shortage that might otherwise find a desirable use in the commercial markets, while at times of war they serve as a potential risk if their purpose is targeted by an adversary who creates a countermeasure that targets the band of operation and renders the system inoperable. The primary drivers of SA for regulation and SIGINT result in a fundamental need for RFML systems with the abilities to find and identify malcontents in the spectrum as well as the ability to attribute these emissions back to a source with SEI.

While all sources point toward and are pushing for RFML systems, there's the added complication for federal entities created by the Privacy Act (5 U.S. Code 522a) and for all entities that have similar limitations placed on them for the purposes of protecting the privacy of its citizens. The Privacy Act limits federal government agencies from abusing a citizen's Personally Identifiable Information (PII) in broad terms [62]. This means that caution must be exerted when collecting RF data at the federal level when intending to create any form of curated data on which a system may be trained, which might contain such information through both well-defined protocols and otherwise unknown protocols. Therefore, a means to understand and procure data in a cost effective manner when needed is important to the U.S. RFML portfolio.

2.4 RFML Ecosystem

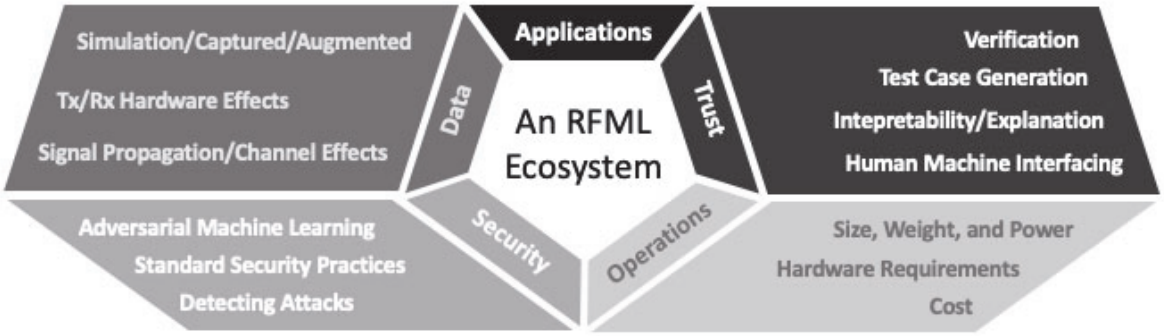

Figure 2.5: The composition of the *RFML Ecosystem* providing a greater focus on the aspects of Data, Trust, Security, and Operations as important, closely intertwined facets that support the Application space [1].

A joint effort between several researchers at Virginia Tech sought to draw feasible bounds around the scope of RFML in order to better highlight the aspects that felt absent, or lacked more detailed study in the literature. At the start of work toward the *RFML Ecosystem* the literature was heavily focused on the application space and showing the gains that ML provided to each unique application, but in general consistently glossed over the remaining areas, even if acknowledging them as areas that needed more focus and study before the results could move from a theoretical solution to one that would survive contact with the real world. The *RFML Ecosystem* [1] emphasizes the full scope of intertwined issues that should be considered, while trying to go from solving a single problem in the RF domain to fully implemented application or solution live in the field. Figure 2.5 presents the five general areas to consider within the ecosystem: the applications, user trust, security, operations, and underlying data. The work in this dissertation is predominately focused within the areas of data, operations, and user trust while showing their inner workings through application space of AMC, with observations made from relevant spectrum sensing programs as well.

2.4.1 Applications

A fundamental part of RFML research is the understanding of context, or secondary information, observable from sampled RF spectrum. The primary information of a waveform can be described as the *what* in regards to a transmission. Two fundamental uses of RF spectrum are: carrying information from one point to another in communications systems, and sounding pulses in attempts to search, locate, and/or track physical objects within radar systems. Looking to understand the secondary information, or context, surrounding a transmission examines the other fundamental questions: *when, where, why, who*, and *how*. By instead asking questions about the *when* and *where* of a transmission in regards to the

spectrum causes the question to fall into the realm of spectrum sensing and signal detection, the process of putting bounds on where the transmission exists in time and frequency. An additional aspect of *where* comes into play in the physical domain, to better understand relative positioning of the emitter from the reception point (e.g., forward, backward, left, right), or more specific localization coordinates to produce a mappable emission. The questions of *how* and *who* can then be thought of as the problem spaces of AMC and SEI, respectively, when enough understanding of the transmission exists; otherwise, the problem can fall into the realm of anomaly detection for characterizing a transmission that falls into an unknown category. The final fundamental question, *why*, in the RFML problem space is significant for providing a deeper understanding of the EME itself. In terms of channel modeling and emulation, this *why* comes along the lines of understanding how the channel takes an ideal representation of the transmission and the results of the degraded form observed at the receiver.

Next the application spaces discussed above are explored in greater context below, except for SS, which was discussed before in Section 2.2.2.

2.4.1.1 Automatic Modulation Classification

AMC is the body of work within the RFML literature that aims to ask *how* information is being transmitted between a transmitter and any number of receivers. In effect the aim is to down-select from the set of possible modulation techniques to either the exact technique in use, or a much smaller subset of modulation techniques such that more computationally expensive, or otherwise costly routines such as trial and error approaches, can be used to determine the correct classification.

Typical approaches rely on a combination of feature extraction stages followed by feature fusion or pattern recognition to make the decision on a given observation. Prior to raw IQ being used in the RFML field, likelihood-based and feature extraction routines designed by domain experts were prevalent, often with rule-based engines or other ML backends handling the final decision from the feature space [32, 63].

Often the simplest version of an AMC problem is presented by the classification of two nearly identical waveforms, Binary Phase Shift Keying (BPSK) and Quadrature Phase Shift Keying (QPSK). When distilling the waveforms down to their symbol spaces, their distinct differences can be shown in an IQ scatter plot as seen in Figure 2.6 on the left, where the symbols are perfectly clear given time, frequency, and phase synchronization. The problem becomes more complex with the introduction of an AWGN channel to the symbol space as seen in the middle column of Figure 2.6, even with just the loss phase synchronization, with few symbols to observe, the classification becomes more difficult. Finally, on the right of Figure 2.6 is the synthetic raw IQ data more commonly seen within the RFML problem space where there still exists time, frequency, and phase synchronization. As the observer views the images from left to right, the ability for a human to distinguish the two modulations

from each other becomes increasingly difficult, and yet this still is neglecting the sources of complexity in the problem space without opening up the option space to a larger pool of modulations.

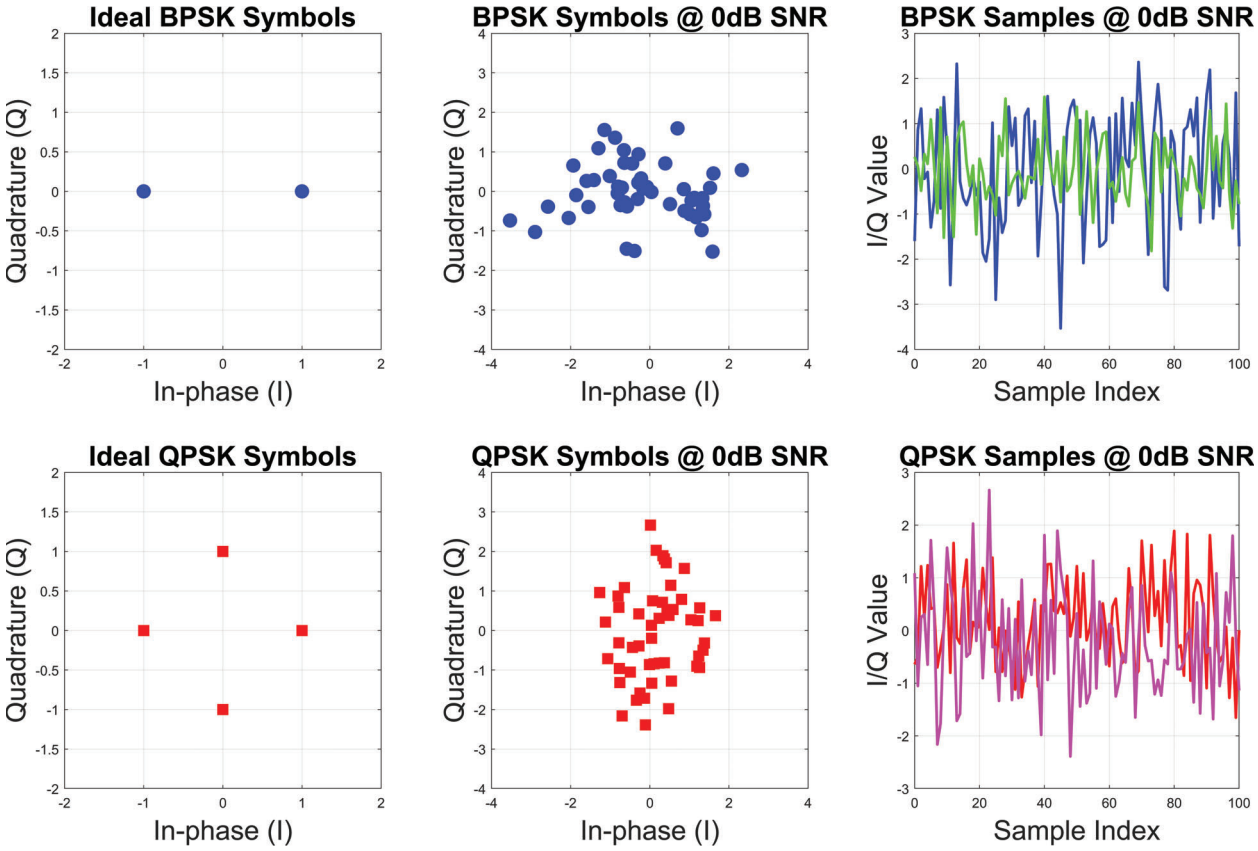

Figure 2.6: One of the simplest problems in AMC, the classification between BPSK and QPSK waveforms based on symbol space, and the more complex problem of sample space. This still assumes time, frequency, and phase synchronization for both symbol and sample images.

2.4.1.2 RF Fingerprinting / Specific Emitter Identification

The goal of Specific Emitter Identification (SEI), also known as RF Fingerprinting, is to identify the transmitter responsible for sending a signal of interest. Slight, but consistent differences between emitters, such as IQ imbalances, amplifier non-idealities, and other imperfections caused during the manufacturing process [64] make SEI possible. These differences not only exist between transmitter brands and models, but amongst transmitters of the same production lot. Further, work presented in [65] showed geographical differences including propagation channels and angle of arrival to have a dramatic effect on SEI performance as well. Compounding the difficulty of examining and contrasting these transceiver

imperfections that oftentimes will have a calibration stage to further minimize their observation, the ability to model the RF component chains in these devices is something done with approximate modeling rather than high fidelity simulation due to the sheer difficulty in perfectly representing and understanding all of the effects that have been seen.

Given the vast number of existing devices, each exhibiting nearly imperceptible differences from one another, SEI in particular has benefited greatly from the advent of RFML, which doesn't need to rely on expert features derived from the approximate models [64, 66, 67]. While traditional SEI techniques have focused on the difficult and laborious task of defining expert features to distinguish between emitters [68], recent RFML-based solutions have used CNNs to learn the discriminating features for identifying transmitters more reliably than the hand-crafted features, and have shown the ability to identify unknown emitters [64, 66].

2.4.1.3 Positioning/Localization

Positioning and localization play a crucial role in both military and commercial communications. For example, as the quantity of consumer-focused wireless devices continue to grow, positioning and localization become increasingly useful in emergency and safety applications, such as search and rescue operations [69, 70].

Traditionally, localization techniques have relied on expert-defined features, such as received signal strength [71, 72]. However, in recent years, a more rich set of RF measurements including channel transfer functions, frequency coherence functions, and channel state information have been used [70, 73]. While channel state information has been used to reach state-of-the-art and cm-level accuracy on indoor positioning tasks [70], little-to-no work has made progress towards performing localization using raw RF data.

Unique considerations take place while curating datasets for this application space. Particular care should be taken as more well kept metadata will be required for this application space as one needs to understand and document the propagation environment, and transceiver locations with as much precision as possible (often beyond commercial Global Positioning System (GPS) precision). The receiving system will require some means for direction of arrival estimation with the means to coordinate multiple observations with knowledge of the configuration of the receiving system. These differences will likely make datasets created for AMC and SEI with a single observer be less than ideal for investigation and require explicit dataset curation by scenario until a better understanding is established.

2.4.1.4 Anomaly Detection

An emerging RFML application area is that of anomalous event detection, where DL models are used to learn a baseline environment and subsequently detect/classify deviations from this baseline (so-called anomalies). An example of this budding area of research can be found in

[74], where RF spectrum activities are monitored and analyzed using deep predictive coding NNs to identify anomalous wireless emissions within spectrograms. Similarly, in [75], the authors utilized recurrent neural predictive models to identify anomalies in raw IQ data.

This application space is harder to quantify in general than most in terms of data fidelity and quantity, as the approach and representation drastically affect the performance of such systems. In the most naive sense, an anomaly could be inferred as anything new, so starting with a system that only knows the qualities of background noise, one could design an SS system that is able to detect everything and qualifies those as anomalies and reduces the reporting of such events based on a backend rules engine. In more well researched and engineered systems, there could already be a knowledge base that constitutes the *normal* and then the system is simply designing a similarity system that quantifies whether or not a given observation is close enough to the knowledge base to be ignored, otherwise flagging it as an anomaly. In any approach, there needs to be enough data to parameterize the concept of *normal*, and then enough data to establish the decision space that can then separate *normal* and *anomalous* behavior. At a fundamental level, this application is trying to identify the unknown and is actively being researched across all ML domains, and this dissertation does not attempt to describe it.

2.4.1.5 Channel Modeling/Emulation

The channel plays a defining role in the performance of RFML systems. As a consequence, including realistic channel effects, captured or simulated, into the training of RFML systems, is critical to achieving top performance. In the case that sufficient data can not be captured, channel modeling is a critical component of creating realistic simulations of RF systems.

Traditionally, channel modeling requires understanding the multi-path propagation effects of a wireless channel and stochastically recreating those characteristics using mathematical approximations during simulation. However, such approaches are often computationally expensive [76]. The area of RFML-based channel modeling and/or emulation is currently limited, but continues to grow as the need for data grows. For example, in [77, 78], an ML-based channel "stand-in" is used, which allows for channel emulation within an end-to-end RFML training routine. Alternatively, in [79], the goal is channel translation, where signal captures collected in one channel environment are augmented to resemble a different channel environment.

2.4.1.6 Application Dependencies

As alluded to while discussing the application spaces, the scale and scope of different applications can lead to vastly different hardware and SWaP requirements. For example, a Raspberry Pi 0 has been shown to be suitable for performing event-triggered packet-based SEI for IoT networks [67], but much larger systems are needed to realize real-time 5 GHz instanta-

neous continuous spectrum monitoring systems [57]. For most RFML applications, decisions must be made locally due to bandwidth and time constraints [80]. As a result, environmental effects on the hardware must also be considered, in addition to the SWaP requirements necessary to execute RFML algorithms on varying devices. For example, when deploying RFML algorithms aboard small spacecraft which are impacted by radiation-induced single event upsets [81, 82], without the addition of radiation shielding and/or extensive mitigation strategies, the performance of the ML structures fail to achieve the necessary performance to be practically useful [83, 84, 85, 86, 87, 88].

Broader dependencies include harnessing the more rapid decision making available through RFML. More specifically, many of the applications discussed above cite rapid decision making as a benefit of using a DL-based approach over traditional approaches. However, additional work is required to make the outputs of such RFML systems fully actionable.

2.4.2 Deployment Considerations

Though early adoption of RFML systems has already taken place in a variety of military systems [34, 57, 89, 90], a broader interest is expected in the roll-out of commercial cellular [91, 92, 93], IoT [67, 94, 95, 96], and satellite communications systems [97, 98, 99]. This section evaluates the practical SWaP constraints encountered in the transition to real systems, highlighting that the requisite hardware and algorithmic technologies for RFML deployment are well under way. More specifically, given the low processing and storage requirements for RFML algorithms, compared to Computer Vision (CV) algorithms, and current availability of RF sensors on board low-SWaP mobile devices such as cell phones, the barriers to entry for deployed RFML algorithms are primarily the cost of training data, decision-making infrastructure, and trust/assurance.

2.4.2.1 Real-world Data Collection

The primary difference between laboratory-measured or synthetic data and observed data is typically that the laboratory or synthetic environment is pristine in comparison to an observed environment. This is largely due to the multiple overlapping phenomena, not typically encountered in simulation or a laboratory, that degrade signals which have propagated in the physical world [100]. These real-world effects can generally be categorized as hardware variations and channel effects, and can significantly impact RFML performance, if not considered when developing the training, validation, and test datasets, as discussed in this subsection.

Hardware variations refer to the variances between transmitter and receiver hardware platforms and the resulting impact on the received waveform. More specifically, different transmitter and receiver pairs distort waveforms from the ideal in varying degrees as a result of manufacturing variations, environmental operating conditions (e.g., temperature), and

access to supporting devices like reference oscillators. These distortions take the form of non-linearities, additive noise, timing offsets, frequency offsets, phase offsets, sample rate mismatches, and/or amplitude offsets, all of which may be time varying. Depending on the application, distortions to the waveform caused by the transmitter may be a parameter of interest, or may be considered a nuisance parameter. In the latter case, an ensemble of transmitters is required to model an average transmitter, and as a result, adding varying transmitter imperfections to the training data is critical for model generalization. For example, applications such as SEI depend upon transmitter imperfections to distinguish between transmitters. Meanwhile, for applications such as AMC, transmitter imperfections are considered nuisance parameters, as the goal of AMC is to identify the modulation class, regardless of the emitter. Similarly, in the case of receiver distortions [22, 72], natural reception variations such as sampling rate differentials, frequency offsets, and varying SNR, must also be varied in the training data to encourage generalized learning [101].

Lack of synchronization between devices will also exacerbate the distortion caused by the transmitter and receiver, as well as the channel itself. To improve synchronization, detection and isolation routines are used to select spectrum of interest. However, these algorithms introduce measurement errors in the form of time, frequency, and phase offsets between the devices which must also be modeled in order create a realistic simulated dataset. It should also be noted that higher quality hardware, such as military transmitters, tend to cause less severe distortions than lower quality hardware, such as IoT transmitters, and the non-linearities that contribute to these variations are often dependent upon technology and hardware configurations.

The second category of real-world consideration for RFML system performance, signal propagation and/or channel effects, add noise and further degrade the signal of interest. While the baseline simulated or laboratory training environment used in most RFML works is an AWGN channel, real-world channels have time-varying, often colored spectra, and uncontrolled RF interference sources such as other signals, impulsive noise (e.g., lightning), and non-linear effects associated with bursty packet transmissions. While many of these effects may be approximately modeled [102, 103], preliminary work in [104] has shown superposition of a live captured effect onto synthetic datasets through augmentation to yield better performance. However, additional work is needed to confirm this hypothesis.

The physical medium (channel) through which the signal propagates can also change over time, if the transmitters/receivers or environment is mobile, causing delayed imperfect reflections of the signal to overlap with the direct path resulting in time and frequency varying interference. Therefore, relative motion between platforms, co-channel/adjacent channel interference, and multi-path propagation must also be considered in the development of RFML datasets. Many of these channel variations can be modeled stochastically. However, it is important that the training dataset not be biased so heavily towards learning the channel that it fails to learn the desired behavior [57].

2.4.2.2 Size, Weight, and Power

Many DL techniques employ significant computing infrastructures during their training phases, which makes training in the field infeasible [105]. When considering deployment, we are most concerned with a DL algorithm's computational requirements post-training, when attempting to process incoming data inputs. Current state-of-the-art RFML techniques often utilize NNs significantly smaller than CV techniques, with 2-3 orders of magnitude fewer trainable parameters, boding well for deployment on low-SWaP devices. Further, RF sample frames occur on the order of 1 kHz compared to image inputs that might occur on the order of 1 Hz in inexpensive commercial devices. Therefore, the evaluation time of a NN processing raw IQ data must meet more stringent real-time requirements than a NN processing images.

In an effort to further reduce processing requirements, some RFML implementations have also embedded traditional signal processing techniques such as Fourier and wavelet transforms, cyclostationary feature estimators, and other expert features directly into the NN [97, 106, 107]. Meanwhile, other research has focused on reduced precision implementations of NNs, enabling a path towards realtime implementation [108, 109, 110]. However, reducing realtime computational resources to mobile systems remains a challenge that must be overcome, especially if online learning techniques are to be developed for future RFML systems [111, 112].

Given the highly effective miniaturization of digital electronics, a deployed system's weight is primarily driven by its power consumption and the associated batteries or heatsinks [113]. In a spectrum situational awareness system, the instantaneous bandwidth of the spectrum analyzed, the density of signals within the environment (affecting the number of calls to an RFML algorithm), implementation in hardware vs. software, and the environment where the device is used will all contribute to the system power usage. Realtime signal detection [114], signal characterization [34], and SEI [67] systems have already been achieved, either through the assumption of vehicle power or a tightly regulated and small duty cycle, showing the feasibility of using these algorithms in current mobile systems. Further, the use of wake-up circuits for periodic/event-triggered execution of an RFML function can be used to further reduce average power draw, permitting the use of such techniques in extreme low-power applications such as the IoT [67]. Finally, the integration of RFML processing with energy harvesting techniques are of particular interest for battery-powered IoT and solar-powered satellites, but have yet to be investigated.

2.4.2.3 Cost

Beyond SWaP, cost is typically considered the next most important operational consideration. Because the quality of the training data drives the overall functionality of an RFML system and often requires human-intensive labeling and/or pre-processing [57], as discussed in more detail in Section 2.5, the primary cost drivers of current RFML systems are the

curation of datasets used for training/evaluation/testing, the training hardware, and the RF hardware to be deployed.

The cost of the training process itself is in part driven by power consumption [115], and in part driven by the purchase of parallelized processors such as GPUs, Tensor Processing Units (TPUs), or other special purpose hardware. While the purchase of specialized hardware is typically a one-time expense, current training approaches for most RFML algorithms require complete retraining of the underlying model when new training data is added, as online, unsupervised, semi-supervised, and transfer learning techniques have yet to successfully be employed. As a result, power requirements for maintaining RFML models can be high. Improvements in online, incremental, and transfer learning approaches are necessary, not only to learn behaviors associated with new signals or environmental changes, but to minimize re-training to when performance degrades.

The cost of the RF hardware is dependent upon the quality, and will impact the performance of the RFML algorithm and resultant learned behaviors, as discussed in Section 2.4.2.1. For example, SEI algorithms are better at differentiating between low-cost sensors, such as those used in the IoT, than between high-cost sensors, due to more significant variations during manufacturing. Conversely, the peak accuracy of an AMC algorithm will decrease, as the model is forced to generalize its learned behavior across the imperfections present in low-cost hardware [22]. Such phenomena are not confined to the transmit side of the RF chain, and can be induced by low-quality receiver hardware as well. Therefore, RFML system design must consider the impact of the cost of the RF hardware on each side of the communications link in the creation and/or expansion of the training datasets and in the expected performance of the RFML system.

2.4.3 Discussion

While RFML algorithms have already begun to make their way onto deployed military systems, it is expected that RFML will become a vital component of future commercial cellular, IoT, and satellite communications systems in the near future. The scale and scope of the different RFML applications to be deployed will lead to different hardware, SWaP, and bandwidth requirements which will need to be considered. However, the discussion above highlights the feasibility of using current state-of-the-art RFML techniques on even low SWaP and Cost (SWaP-C) devices, given the use of significantly smaller models and the speed with which raw RF data can be collected and batched for processing.

The quality of these potential deployed RFML systems is largely dictated by the quality of the training data, i.e., how closely it resembles the data observed during deployment. Therefore, the collection and labeling of training data is also one of the largest cost drivers, in addition to the training hardware (i.e., GPUs and TPUs), power consumed during training processes, and RF hardware. However, the development of online, unsupervised, and/or semi-supervised learning techniques will mitigate these costs to some extent by limiting the

amount of model re-training that must occur when hardware or environments change.

This dissertation focuses on the critical training processes that enable the *RFML Ecosystem*, with significant focus given to understanding the intertwined aspects of *Application*, *Data*, and *Operations*, while dropping hints along the way of how *Trust* comes into play. Chapter 4 goes through an application study of AMC, where considerations are given toward the data used during training in order to better understand and qualify the statements made here. Going through the process of training thousands of models with parametrically varying constraints on the dataset allows for a deeper understanding of how data plays a role in *Trust* as well. More explicitly, *Trust* in the realm of data comes down to the concepts of "Test Case Generation" with understanding of *in-set, near-set*, and *out-of-set*, which help demonstrate how an end user can expect the system to function, and more importantly help understand where and how it will break.

- *in-set* - Data observations whose distributions and choices exist completely within the training dataset.
- *near-set* - Data observations whose fundamental choices exist within the training dataset, yet whose distributions are not found in the training dataset.
- *out-of-set* - Data observations whose choices and potentially their distributions are not found in the training dataset.

Under the most ideal situations, the *in-set* category of data includes the deployment environment and therefore the best understanding and trust can be applied to the whole development cycle in terms of performance on the chosen training, validation, and test datasets used. In most practical real-world situations, while some of the deployment environment will exist within the *in-set* training data, a more common outcome will have that the real-world data will instead exist in the *near-set* category where all training, validation, and test datasets might indicate high performance, while when deployed in the real-world performance can degrade to on par with random guessing, if not worse.

2.5 Data Requirements

At this point, two different ways to think about data have been discussed. The first way to consider data is in terms of *Quality*, as seen in Section 2.1.3, and is useful in talking about what is desired in a training dataset in terms of *Comprehensiveness, Correctness*, and *Variety*. While the second way to consider data blends the concepts of *Comprehensiveness* and *Variety* by presenting the concepts of *in-set, near-set*, and *out-of-set* while thinking about the test dataset and on a larger scale the deployment environment in Section 2.4.3.

Starting with the end, understanding the desired deployed environment is in effect the bounding placed on the system in the hope of describing what the *in-set* dataset must be privy to in terms of observables, rather than the intimate knowledge and parameterization of such a data space. For example, the observables could be defined in performance goals such as:

- Must work in the forests of Virginia
- Expected operational time is 8AM - 5PM
- Needs to incorporate the use of radio systems X, Y, and Z.
- Operational distance is within 5km.

Then, everything that falls within the above is part of the *in-set* data that must be portrayed. From that, more parameters may be inferred, like the modulations that each radio system can use, and the frequency bands in which they operate. Depending on the frequency bands and operational time of day, that in turn might help qualify the assumptions about the radio channels that can be expected if operating at a carrier frequency of 3MHz, or 3GHz. Information about the radio systems, channel environments, and operational distance enable inference the relative SNR range to expect. What is harder to identify and constrain are unknowns like:

- How are the forests of Virginia unique in contrast to forests in general?
- How does the weather affect the systems in question?
- How well are the radio systems maintained?
- How much variation can be expected over the user base and does the individual user affect their radio's base performance?
- Does the speed of the transceiver matter?
- Should considerations be taken for vehicle travel on land, water, or sky?

While these questions can be thought of in this context, there may be many more that cannot as easily be known. These unknowns in turn expand and contrast with the definable *in-set* and establish a blurred understanding of the *near-set*. Helping understand the last concept of *out-of-set* would be taking the same system with no modifications and expecting it to work in the same way if put in New York City. There, the channel environment is likely different enough that even if the same radio systems were deployed, which is unlikely, the channel environment would differ in significance.

A subject matter expert might be able to help better align the *in-set* definition such that the *near-set* differential can be minimized in terms of dataset representation. Caution must

be taken however, because if the assumptions made do not align well enough, the operational space that the system will operate in could fully reside in the *near-set*. Conversely, the validation and test sets would be in the *in-set*, providing false confidence in how well the system will perform. The application study in Chapter 4 gives a strong example of this where the assumptions put the entire operational point in the *near-set* for some training/-validation/test datasets providing tests and metrics with strong performance, but fails once operational observations are used. One point of caution to mention: this application study contains a controlled and intentionally curated dataset where the *Correctness* property of *Quality* is largely assumed to have zero error, or at least significantly close to zero error within the data. Due to this careful curation, and the use of early stopping mechanisms, the phenomena known as the "double-descent" issue within machine learning can largely be avoided [116].

The concept of dataset quality then comes into play with moving from defining the *in-set* data space and curating the training dataset in terms of *Comprehensiveness*. The training dataset must not only contain the information space needed for operation (ex: modulation classes for AMC), but also the nuisance space of significance (ex: fading, and absence of perfect synchronization) as well. Where *Comprehensiveness* and *Variety* blend in this work is when distributions that make up the degradation of signals when thought of as impulses at a single point, or constant value rather than a distribution with variance (in simulation this is often the case if the degradation is ignored). The attribution of this random constant to either of the two areas is correct; however, based on the significance of such distributions on the overall performance, their absence might be better attributed to a lack of *Comprehensiveness* rather than to the mismatch of *Variety*.

These conditions lead to the desired requirements for dataset curation:

- Training datasets should be *Comprehensive* in that they incorporate as much of the operational space as possible in the *in-set* category, while minimizing the occurrence of the *near-set* and removing the possibility for *out-of-set* observations.
- Training datasets should be *Correct* in that labels have minimal noise and caution should be taken to avoid the occurrence of adversarial poisoning.
- Training datasets should be tuned such that the *Variety* matches as closely as possible to the expected distributions as long as that doesn't conflict with performance goals.

Fulfilling these requirements will lead to datasets with greater *Quality* than otherwise would occur if not observed. One example to be aware of in terms of *Variety* where distributions and performance goals can conflict is when a performance goal is defined per class, while a dataset might sample the classes true to observed distributions. The problem of imbalanced classes in a dataset, while good in terms of *Variety*, could lead to undesired performance based on the goals and is not considered within this work.

Chapter 3

Challenges of Dataset Generation

At this point the role of data in an ML system, and more specifically RFML systems, should be understood as a commodity. This is due to the nature of data, and the resulting quality associated with a dataset. Higher quality datasets that originate from within the actual application space should always be the goal when money and time are of no consequence; however, money and time are never of no consequence. So here a discussion of how to acquire data, and the challenges associated with the collection, are discussed.

3.1 Data Observation Origins

At the end of the day in the digital domain, if two files or observations are bitwise identical, the origin is a moot discussion. To the best of my knowledge, no such process can create a bitwise exact replicate of waveforms in a specific real-world situation, let alone for the general case. For the sake of argument, let's assume there exists some such system that is capable of performing such a general task. This means that there exists some combination of software/hardware system that is able to bitwise recreate the exact observation that two radios in the field produce and observe.

The question of what it would take to run such a simulation is an important discussion. The first part of this simulation system is that there must be enough understanding of the physical space and the electromagnetic properties of all materials in the location to a point of quantization precision. Next, every propagation path, of significance, that exists between the transmitter and receiver would need to be accounted for. Moving into the realm of imagination, the exact state of the EME would be needed such that all interference that is present in the observation is also accounted for, which would include the quantum state of the transceiver devices to better understand the distortion and noise introduced. Finally, the complete understanding of all biological life that might be interacting, moving, or obfuscating the devices would need to be accounted for. At this point in discussing what is needed for this system, the performance requirements would be unquantifiable and likely be of the order of performance of all supercomputers on Earth, if not orders of magnitude larger, all to produce a single observation in a dataset. Based on the system discussed in Chapter 4, this would need to be repeated for a *quadrillion* observations in order to reach the target performance goal.

Backing away from the idealized perfect simulation, stochastic approximations are well within the realm of possibility. Instead of the absolute knowledge, all that is needed are general distributions on parameters of interest, like transceiver distortions, channel coherence time and bandwidth, and interferences. Now here, the assumption should be that the stochastic observation of any single simulated observation will never be bitwise perfect, but that over the full dataset the measurable distributions are alike enough to real-world to still be usable.

The problem is therefore one of how well the distributions of interest can be characterized and validated prior to setting down the simulation path. This problem also reduces to one of data and characteristic extraction, be it historical or contemporary. Therefore, the observations created by simulation and by extension, augmentation, are dependent on the distributions present in the real-world environment and thus require an understanding of the real-world as described through captured data.

3.2 Challenges to Collected Data

The problem for all forms of dataset generation fall back on needing some form and amount of collected data, so why not just say that only collected datasets are of value. Pound for pound, collected observations carry the greatest return when they are germane to the problem space; however, the associated costs of procuring a single observation of collected data often are orders of magnitude more expansive.

3.2.1 The Cyborg Dataset Collection Campaign

The captured dataset used in Chapter 4 for the AMC application study consists of over 2 million observations per class with an SNR value greater than -10dB and a length 1024 samples, with a stride from the start of one observation to the next being 2048 samples, implying that on the order of 4 million non-overlapping observations were captured per modulation. This collection was performed with a point-to-point setup using directional antennas to maximize the distance and power that could be observed without adding additional amplification outside of the Ettus B210 SDRs [117]. Despite having the waveform generation software available with GNU Radio [118] and the OOTM 'gr-signal_exciter' [33] and the SDRs on-hand, the development occurred over a period of eight months prior as part of an Undergraduate Research effort by pre-entry level software developers. Prior to performing the collection that became the research dataset, two additional attempts were made, each revealing bugs that made the metadata wrong in comparison to the observations collected. With each attempt, new structure and controls had to be implemented in order to properly handshake between the transmitter and receiver such that the metadata updated with each new capture, and every detail was recorded in a way that was correct. In all, the collection took place over two fewer months than originally allocated to the collection, in

order to catch and address the bugs within the collection routine. Even after all of the work preparing this dataset, and establishing an automated collection routine on a point-to-point setup, lack of understanding in stochastic modeling resulted in observations deviating from the desired uniform spread, as both the transmitter and receiver used uniform spreads to cause convolutions of the distribution in terms of SNR, Sample Rate Mismatch (SRM), and Frequency Offset (FO).

3.2.2 Government Furnished Dataset Curation

There are several projects run by government entities that have collected, or seek to collect, data that is suited for solving RF related problems. The DARPA RFMLS program was focused on the problem of SEI and as a result required a significant dataset in order to be able to apply ML to the problem. This process was labor intensive and required significant planning and precise execution in order to guarantee the protection of citizen PII, as opposed to taking an easier approach of recording similar devices such as WiFi at a neighborhood coffee shop. This level of care required the acquisition and control of many IoT based devices, and produced terabytes of raw IQ data restricted to government use.

A newer program being run by IARPA, SCISRS, in many ways is a spiritual successor to the RFMLS program and has come across the same issue with the need of datasets to properly address the problem of SEI. However, as SEI is not a primary focus of this program, which is more interested in understanding the presence of anomalous signals instead, the program structure has placed severe restrictions on access to raw IQ data in order to avoid potential PII complications. Understanding the difficulty of performing SEI under these constraints, the focus of SEI was adjusted in later phases to give the performers and program runners more time to understand if there's a better path to accomplish this aspect of the program.

Another program run as a competition by MITRE on behalf of the Army Rapid Capabilities Office chose to instead rely on simulated data to avoid the difficulties associated with PII for dataset curation. Even within the simulated data, there existed irregularities within the datasets that prevented a seamless competition as reported by the performers. However, this was discussed by the program runners and described as a combination of intentional and accidental dataset poisoning within the program [56].

In all, the government led programs show the significant difficulties associated with curation of datasets in both the collected and simulated domains. The difficulties lie in the domains between a software developer and a domain analyst that can understand what the data represents. By and large, there are too few people with adequate experience to wear both hats, and this often causes greater expense in order to properly bound the problem, or risk non-critical bugs in the software execution domain to slip through, causing label and metadata noise to reside in the datasets.

3.2.3 Cost Modeling

Performing a collection has largely been conducted by the operator for the express purpose of generating data for their usage alone. These generation, collection, and storage routines have mostly been single use developments that are either discarded after use, or simply shelved in the case more data is needed, but remain inaccessible to the general public. The OOTM 'gr-signal_exciter' [33] was initially developed to help alleviate the need for reinventing waveform generation tools every time a new project was needed; however, it too has been shelved in large part due to lack of time available to develop the code base while pursuing the degree for which this dissertation is being created. By making use of the shelved version of 'gr-signal_exciter' as a physical layer simulator and using the National Aeronautics and Space Administration's (NASA's) Space Telecommunications Radio System (STRS) Constructive Cost Model (COCOMO) Calculation tool, a rough estimation for just developing the tool would be $1.3 million for simulation purposes in a semi-detached project. The cost estimate increases to $2.9 million when developing the system for embedded system deployment that places increased demand in the categories of Execution Time Constraint, Main Storage Constraint, and Computer Turnaround Time. These estimates only cover the costs associated with developing the codebase as an extension to GNU Radio and require greater attention and research to substantiate. In general, a limitation in RFML is that there doesn't exist a means of projecting the cost associated with dataset curation, often resulting in them being under-budgeted, leading to mistakes and delays due to under-staffing of the effort. Developing a toolset to perform collections in a reliable way as well as to help do a better job of cost estimation for curating the datasets that are needed are a critical hole in the RFML body of work.

3.3 Open Source Data Generation Tools

There are several toolsets that exist to enable simulation of RF waveforms and have been used in the literature to establish the RFML body of work. Among those that exist, three predominant toolsets are GNU Radio [118], Liquid DSP [119], and RedHawk [120], which all have roots extending from Virginia Tech.

While any of these toolsets would perform an adequate job at keeping track of the labels under the assumption that proper bookkeeping was written into the source code, an OOTM was created as 'gr-signal_exciter' in order to better keep track of the information. To accomplish this, the concept of a signal was collapsed into a concept of 'symbols' in the case of digital modulations, where a symbol was a compact representation of amplitude and phase. For analog modulations, the concept of 'symbol' is overloaded and instead captures a 'message' that represents an analog waveform, typically an audio waveform conceptually, as a series of floats. The 'symbols' can then be requested by the routine responsible for generating the 'signal' and within this function, per modulation, the knowledge of how to

convert ‘symbol’ into ‘signal’ at the desired sample rate is preserved. To further simplify the procedure, all waveforms can be instantiated with a signal parameter structure that contained all relevant parameters needed to instantiate any modulation, and was able to be serialized for independent process creation of the desired waveform, and another process capable of implementing such a request. This procedure ends up being modular in that each waveform can be created in a polymorphic manner with the same API in place. Figure 3.1 shows a few examples of what is possible with the ‘gr-signal_exciter’ OOTM. This routine also allows for the creation of co-channel, adjacent channel, and vast arrays of signals to all interact with the truth known, with just a small example of this being visible in the waterfall plot within Figure 3.1, which is simulating a congested WiFi band.

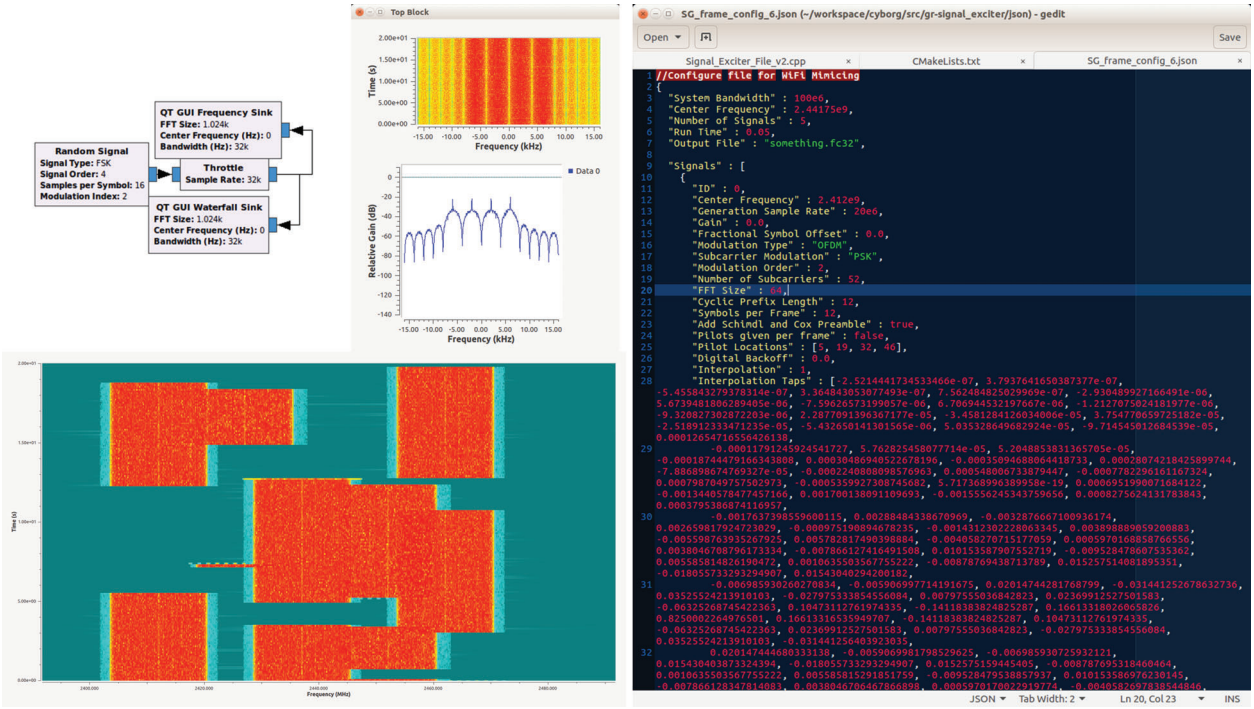

Figure 3.1: Here are examples of the waveforms that were created using the ‘gr-signal_exciter’ OOTM with GNU Radio. In the top left is a FSK signal whose pulses are unfiltered, leading to main-lobes related to twice that of the symbol rate. In the bottom left is a combination spectrum of multiple WiFi-like signals that occupy and overlap each other while simulating an active and congested 2.4GHz WiFi band. The configuration file on the right is the application’s generation information which is read in, allowing for multiple signals to be generated in aggregate.

For the dataset used in Chapter 4, high quality timing was not available as the information link between the two transceivers was supported by an LTE cellular internet backbone. As a result, the collection was not designed for detecting the start and end of bursts, but rather focuses on being able to classify waveforms from a steady state under the assumption that an observation is definitely active for the duration. Due to the nature of the point-to-point

setup, the receiver responsible for recording the waveform would form a request stating the specific settings before sending it to the transmitter. The transmitter unit would ingest the request, confirm a valid setting for the waveform of interest, and in the case of incompatible parameters or undefined parameters, a default would be fallen back to. In the case that the transmitter had to make any changes to the request, the transmitter updated the parameter structure and replied back to the receiver once transmission was started. The receiver would then log the metadata into a SigMF [121] file and start recording.

This setup for logging of metadata was convenient for logging metadata in a JSON file format alongside a binary file consisting of the samples. In this way the dataset was created consisting of as much detail as possible about each and every capture, and additionally was able to keep track of any discontinuities in the samples from the B210 driver. One key understanding needs to be preserved: the metadata is still subject to the imperfections of the transceivers, such as time/frequency/phase drift and offset that are not compensated for, therefore the metadata labels of SNR, FO, and SRM are still subject to some irreducible noise. An example of the metadata collected is provided below for a Phase Shift Keying (PSK) observation.


```
{
    "annotations": [],
    "captures": [
        {
            "core:datetime": "2019-07-10T13:10:57.761204749999999Z",
            "core:frequency": 914749999.9999986,
            "core:length": 11264000,
            "core:sample_start": 0,
            "cyborg:noise_floor_dB": -104.48491096496582,
            "cyborg:snr_estimate_dB": 40.56185722351074,
            "cyborg:snr_technique": "{\"parameters\": {\"overlap\": 768, \"
               nwin\": 1024, \"window\": \"blackmanharris\", \"
               single_side_frequency_region\": 0.1875, \"
               above_noise_floor_filter\": 3.0, \"nfft\": 1024, \"
               psd_range_noise_floor_region\": [0.4, 0.5]}, \"script\": \"
               snr_reference.py\", \"method\": \"psd estimation\", \"function
               \": \"snr_estimation_energy\"}"
        }
    ],
    "global": {
        "core:author": "",
        "core:datatype": "cf32_le",
        "core:description": "",
        "core:hw": "b200_30FD80E_30FD80E_na_na_b200_30FD82C_30FD82C_na_na",
        "core:license": "",
        "core:sample_rate": 1000000.0,
        "core:version": "0.0.2",
        "cyborg:modulation_parameters": "{\"Pulse Shape\": \"RRC\", \"Phase
           Offset\": 0.0, \"Symbol Overlap\": 11, \"Samples per Symbol\": 4,
           \"Rolloff Factor\": 0.35}",
        "cyborg:nyq_est": 0.006750000000000001,
```

```
24          "cyborg:rx_ant_ang": 310,
25          "cyborg:rx_decim": 125,
26          "cyborg:rx_freq_offset": 0.0,
27          "cyborg:rx_interp": 6,
28          "cyborg:rx_location": "[37.197326, -80.579223]",
29          "cyborg:seed": 294668621,
30          "cyborg:seedling": 334006877,
31          "cyborg:sr_mismatch": 1.0,
32          "cyborg:symbol_rate": 5000.0,
33          "cyborg:tx_ant_ang": 128,
34          "cyborg:tx_decim": 1,
35          "cyborg:tx_freq_offset": 0.0,
36          "cyborg:tx_interp": 50,
37          "cyborg:tx_location": "[37.201727, -80.588871]",
38          "cyborg:tx_samp_rate_mismatch": 1.0,
39          "cyborg:version": "v0.2.0",
40          "modulation:order": 2,
41          "modulation:scheme": "BPSK",
42          "usrp:rx_ant_model": "11 Element 900 MHz Yagi",
43          "usrp:rx_ant_uhd": "TX/RX",
44          "usrp:rx_bb_gain": 1,
45          "usrp:rx_center_freq": 915000000.0,
46          "usrp:rx_clk_src": "gpsdo",
47          "usrp:rx_db_ser": "na",
48          "usrp:rx_db_type": "na",
49          "usrp:rx_dsp_freq": 250000.0,
50          "usrp:rx_rf_gain": 50,
51          "usrp:rx_ser_tag": "30FD82C",
52          "usrp:rx_ser_uhd": "30FD82C",
53          "usrp:rx_time_src": "gpsdo",
54          "usrp:rx_type": "b200",
55          "usrp:rx_uhd_clk_src": "gpsdo",
56          "usrp:rx_uhd_time_src": "gpsdo",
57          "usrp:tx_ant_model": "11 Element 900 MHz Yagi",
58          "usrp:tx_ant_uhd": "TX/RX",
59          "usrp:tx_bb_gain": 0.25,
60          "usrp:tx_center_freq": 915000000.0,
61          "usrp:tx_clk_src": "gpsdo",
62          "usrp:tx_db_ser": "na",
63          "usrp:tx_db_type": "na",
64          "usrp:tx_dsp_freq": 250000.0,
65          "usrp:tx_offset": 250000.0,
66          "usrp:tx_rf_gain": 70,
67          "usrp:tx_ser_tag": "30FD80E",
68          "usrp:tx_ser_uhd": "30FD80E",
69          "usrp:tx_time_src": "gpsdo",
70          "usrp:tx_type": "b200",
71          "usrp:tx_uhd_clk_src": "gpsdo",
72          "usrp:tx_uhd_ip_addr": "na",
73          "usrp:tx_uhd_time_src": "gpsdo",
74          "usrp:version": "v0.1.0"
```

```
75        }
76    }
```

3.4 Combating Overfitting and Improving Data Generalization

One major challenge of DL systems is overfitting, or when the network learns characteristics of the observations themselves rather than a more general fit. This was touched on in Sections 2.1.2 and 2.1.3 with the introduction of two regularization layers **Batch Normalization** and **Dropout**, but adding these layers is not the only way to reduce the overfitting of a network. Additional approaches allow for steps being added to the training routine such as early stopping: a technique that uses the hold out validation set to better understand loss on unseen data, rather than training loss that is seen in the training batches. A significant portion of techniques revolve around the training routine and adjusting the hyperparameters associated with the learning algorithm during training , while other focus on increasing the batch size such that each iteration of the learning routine has to take a more generalized step given more observational data being available [122]. Other approaches aim at adjusting the labels to encourage less confidence with the approach while learning with label smoothing, input mixup, and manifold mixup [36, 123, 124, 125]. Early stopping is used within the training routine along with **Batch Normalization** within the model's architecture as well in the application study in Chapter 4. Additionally, the batch size used during training in the application study is set to a constant of 1500, while other ML domains will see typically smaller batches on the order of tens to hundreds depending on model complexity and observation size [126].

At a fundamental level, the generalization approaches above make a general assumption that everything is already contained within the dataset, therefore all that needs to be shown is an increased accuracy, or a decreased loss, which prevents any understanding for analyzing generalization over unobserved observation spaces. This implies the underlying assumption of what is needed is the improvement (smoothing or sharpening for instance) of the decision regions, though it leaves the question of whether the generalization can be applied to parts of a problem space that are not contained within the dataset. The remainder of this section's focus is what can be done with the data within the available datasets to improve the model's generalization as a result.

In the general sense, there are two ways to improve generalization through the dataset: augmentation and extension. The process of augmentation is domain specific alteration(s) of any given observation. Domain specific because the exact transform in the way that sequence based systems like Natural Language Processing (NLP), which focus on the flow of words, won't benefit from image based augmentations like adjusting the saturation levels, but can benefit from transforms that occlude portions of the observation (objects can be blocked,

and words can be lost). In effect, augmentation replicates an observation by reusing it with some synthetic modifications applied altering the data while preserving the labels, which can be a useful approach to grow the dataset size [36, 127]. When it comes to the types and kinds of augmentation available in each data domain, the image domain has a natural advantage as humans are well versed and highly skilled at image perturbations, as we are constantly immersed in objects changing perspective as we move about our lives. There is an innate understanding of the kinds of augmentations that can be performed, objects in broad daylight also can be seen at night. In a study of image processing that was looking into cross-dataset performance, making use of both extension (discussed below) and augmentation through Generative Adversarial Networks (GANs) there were several cases where increased performance can be seen over real data alone [128]. One benefit of GAN approaches is that they provide a means for taking observations under one set of conditions and transferring them to another set of conditions in terms of real-world degradation [79, 128, 129]. The key understanding is that as much of the real world effects that can be considered in the domain of the image problem were modeled and implemented in the augmentations and extensions. Augmentation through synthetic drop-in is performed in [130] in the image domain working with version 4 of the You Only Look Once (YOLO) architecture [131]. By making use of real-world backgrounds and placing synthetic objects over top of the backgrounds, more complex and varying images are able to be generated to push the limits of the YOLO algorithm, but again relies on expert understanding of the data space in order to apply many effects of lighting to the generated images. These examples make up the broad categories of augmentation: Expert Manipulation, ML Manipulation.

Table 3.1: The two generalized types of augmentation for dataset generalization in ML. The process of taking existing data and applying transforms to create another data point.

Name	Concept Description
Expert Manipulation	Using expert knowledge approaches to apply intentional transforms/degradations to available data. Need to understand the parameter space for generalization.
ML Manipulation	Having observations of a target observation space, use ML (often GANs) to transform an observation into the target observation space. Commonly described as Domain Adaptation. Need to have observations to train for and produce generalizations.

While augmentation can be viewed as a type of extension, the adding of more observations to a dataset, the main point of extension is the generation of new observations rather that a manipulation of the available data space. Extension can be seen as a synthetic extension, the creating of more data through simulating the data space and creating unique observations [129], or through physical means by replaying original data through new environments [127]

in an effort to include more real-world permutations. The separation of physical duplication from augmentation is purely a conceptual one, based on the idea that augmentation are synthetic manipulations of the observation, while physical duplication is truly creating a new observation through the real world interaction. For a better conceptual understanding, in NLP this duplication can come about by having another speaker say the same phrase that has already been recorded, or even the same speaker say the same phrase in a different environment. Synthetic extension is often performed through GAN networks, expert simulations, or through latent space manipulations from some kind of autoencoder network. One of the most commonly known forms of synthetic extension through a GAN can be found online at thispersondoesnotexist.com.

Table 3.2: The three generalized types of extension for dataset generalization in ML. The process of creating new observations data.

Name	Concept Description
Expert Synthetic Generation	Using expert knowledge approaches to apply intentional transforms/degradations to an available model of interest. Need to understand the parameter space for generalization.
ML Synthetic Generation	The training of a ML system to generate observations with transforms/degradations observed with the intent that the observation is one possible from the data space. Need to have observations to train for and produce generalizations.
Regeneration	Taking available data and reproducing it in the real world. Causes a new observation of an underlying subject of interest. Need to have observations to be able to reproduce from.

In general the best way to better generalize with datasets are to observe and utilize the information within those observations to broaden the distributional data space available while training. Outside of being able to affect the dataset's observations, relying on regularization within the model architecture or taking steps in the training routine to better help generalize must be done in order to combat overfitting in the trained model.

3.5 Moving from Software to Hardware

There are three general ways to move from a software infrastructure to hardware. The first, and arguably the least practical, is dropping whatever tools and code that are currently in use and switching to a hardware setup that already does everything that is desired and

needed. The reason for this being the least practical is that in the realm of research it is rare to find such a system that gives everything that is needed, and at a cost that is within reasonable means. For instance, recreating a full protocol stack for simulation can be a challenging hurdle to overcome and the temptation is to just replace that with hardware; however, hardware systems are rarely enabled to provide metadata relevant to ML. As an example, such a task could be to replicate a home-based system using the ZigBee protocol. Now depending on the goal, this could be a reasonable approach, but like all engineering it will depend on the desired application and use case. If the goal is to show that ZigBee packets can be observed and demodulated, then this is likely a suitable solution. The goal of being able to perform SEI on a per emitter situation will likely require this acquisition of hardware at a minimum to establish distributions related to the hardware in order to properly simulate and likely cannot be avoided. On the opposite side, though, if the goal is to provide data to a regulatory tool under which the precise time, frequency, and power bounds can be quantified, there will likely be significantly more work in store over creating a simulation for which all of those values can be precisely recorded. In this last example, the cost and difficulty from creating the protocol is being traded to being able to make precise bounding boxes (the point of the project being worked on) by hand.

The second approach is often more practical and relies on the purchase of hardware that is already designed to interface with software such as an SDR. In this case, there's a nice union of software and hardware and the approach doesn't require having high level skills across the board, but will end up costing in terms of quality and/or precision based on the task. The work of Virginia Tech researchers helped bring an application of AMC from a Technical Readiness Level (TRL) in the range of 3-4 to that of a fielded technology at TRL 6 with further private development bringing the technology to a TRL of 7-8 [34], likely with movement of of the system to the third approach discussed below. It's worth pointing out here that in terms of datasets, the TRL of simulated datasets at best register at level 3, while datasets in this category fall most reliably in the realm of TRL 5-6. These devices still ultimately rely on the computer to handle most of the processing and as a result the instantaneous observable resources are constrained in contrast to the third option.

The third approach is the move from relying on the computer to perform the processing of a routine to that of using hardware devices like Field-Programmable Gate Arrays (FPGA). Here in this third approach is where dataset collection routines predominantly in use can be seen in the TRL 7-9, but often are tied behind proprietary company restrictions. One way the original 'gr-signal_exciter' [33] was enhanced was converting from a software generation structure to a fully FPGA environment. By leveraging larger FPGAs and with the improvements in Pseudo-Random Number Generator (PRNG) cores designed for FPGAs [132], custom cores were created that allowed for the generation of Linear Digital Amplitude and Phase Modulation (LDAPM) waveforms at rates far exceeding the constraints observed with computers, while maintaining the label creation needed for precision RFML dataset generation. This particular system resides at TRL 4 and is currently paused while the search is ongoing to find the time and resources to dedicate to continuing the development

to higher TRLs.

3.6 Lessons Learned

Throughout the course of this body of work, the need for reliable and repeatable data generation has been a constant driving force. At the heart of that generation is a PRNG, and making sure to use a complex and well designed generator is imperative in producing useful results that do not end up biasing any observations. Generation within these applications rely on a Mersenne Twister while on a computer and running in software due to the desire to never repeat within a generation execution, with seeding the PRNG from a non-deterministic source. An additional benefit to this use case for PRNGs is that as the problem goes from single stream generation, where one waveform is generated at a time, to parallel and diverse stream generation, the tying of a PRNG to each generator allows for each generator to be well seeded and minimize conflict between the random streams. Initial designs defaulted to the older style of relying on the system clock to act as the seeding source for the waveforms and results in dozens of repeated waveforms reducing the application's ability for rapid generation on more robust servers.

Another difficulty found in this process is being consistent in property definitions across the diverse set of waveforms. With arbitrary bandwidths being available, the properties associated with time and frequency bounds become harder to precisely define. For example in digital systems where there is a distinct understanding of a 'symbol' the start time can just be thought of as the group delay of the symbol through the Linear Time-Invariant (LTI) FIR system. Contrasting that with analog signals with no such finite entity can lead to differences in what is considered the 'start time' and 'duration' of a signal. Throw into the mix the ability to change the SNR of the signal, and the power profile will change as well. Determining and constraining the problem in such a way that signal definitions are consistent takes some forethought while building up the code base. This problem only becomes more complex as real devices are brought into the mix and the lack of a precision time synchronization between them can cause offsets at the receiver from what's reported and observed in addition to the jitter present in such devices as well.

One additional point of note is constraining the competing desires to create minimal data storage with the need for meticulous labeling for the system. For the interest of data science and trying to understand all of the possible differences that can occur in the real world, metadata labels can be created, if one chooses, to precisely label information down to the number of leaves in the trees nearby, but often result in unmanageable data volumes. However, on the opposite side of things, having the ability to log with some form of redundancy is beneficial, as these collections are generated and managed with humans in the loop that are prone to simple errors that might otherwise go undetected. The initial collection window started earlier than the four month window, as described in Section 4.1, but because there was enough redundancy within the metadata collected we were able to identify failed log-

ging due to human operator error. The data observations that were identified were removed and deleted from the dataset due to improper metadata being passed in, for example all waveforms reporting they were PSK rather than their true waveform. Other mistakes in the metadata labels were able to be corrected for with detailed note-taking, at one point when the transceivers were moved, the GPS coordinates were not updated within the scripts, yet the timestamps allowed for patches to be applied to the labels in error after the fact.

Moving the focus out from isolated signals to more complex wideband observations, the nice assumption of an AWGN environment starts to fall apart. Taking into consideration the hardware configurations that are being used, and the effect of the position of the observation in the frequency domain causes the noise floor to become colored rather than following a white noise assumption. When pushing the boundary on RFML applications to their limits, having the ability to analyze, characterize, model, and combine data in an appropriate way to maximize the EME of the observation data to the task at hand is a vital skill to create useful datasets.

A summary of these points can be boiled down to:

- Make sure the PRNG in use is well suited to the task at hand.
- Identify parameters of importance and understand what they describe; where the values cannot be consistent to what was observed, rectify them as appropriate.
- Be verbose in the metadata and keep external logs for system configurations - human error is extremely common.
- Don't disregard the background, but rather make use of it to improve augmentation and synthesis.

Chapter 4

Training Data Quantity and Quality

For a given problem within the RFML space, once a reliable training routine and a network of sufficient capability have been identified, how well a trained network is able to solve the problem often comes down to the quantity and quality of the data available [36]. Effectively, there are three sources of data that can be used to train networks within the RFML space: simulated/synthetic [20, 22, 28, 29, 48, 64, 74, 79, 104, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182, 183, 184], captured/collected [47, 48, 66, 71, 73, 100, 104, 136, 147, 159, 168, 173, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194, 195, 196, 197, 198, 199, 200], and augmented [79, 104, 168, 183, 189, 195], which is a combination of the first two using domain knowledge (focus of this work), or using Generative Adversarial Networks (GAN) as performed in Davaslioglu *et al.* [79]. Due to the nature of the RFML data space, simulated data is inexpensive thanks to open source toolkits like GNU Radio [118], where observations can be generated uniquely in parallel, with the only bottleneck being the available compute resources [20]. Comparatively, performing an Over-the-Air (OTA) collection costs many orders of magnitude more in terms of time and money due to procurement and configuration of the hardware transceivers and having to generate data in real time rather than in parallel as is done in simulation, and unless directly examining Commercial Off-The-Shelf (COTS) equipment, all the work done in order to simulate the data is still needed. That cost only increases once collection is moved onsite to environments of interest for acquisition of the highest quality data, because robust mobile systems need to be assembled in order to perform the collection while recording and storing the vast quantities of IQ data needed for training. This is a familiar problem in other domains such as image processing, where labeled training datasets are augmented to expand the size of the datasets and improve neural network generalization performance [201, 202, 203], such that data augmentation becomes a viable alternative and builds off a comparably smaller collected dataset. In Wang *et al.*, using synthetic permutations of an AWGN class and combining that with other classes was enough to increase the performance of their network in the Army Rapid Capabilities Office's Blind Signal Classification Competition with an augmentation factor of seven; i.e., adding seven augmentations per observation to the original dataset [183].

This dissertation provides an application study that takes a look at the problem space of AMC to better understand how the source of data affects the system, as well as how the distributions of parameters affect both synthetic and augmented dataset quality and

performance. A common rule-of-thumb in ML is more data is better, but caution must be advised because without high quality data, more data can wind up as wasted cycles, and when performance is asymptotically limited more data can't overcome that limitation. This is especially true of synthetic data that doesn't do enough to address real-world challenges. The results in this work rely heavily on the ability to assume *Correctness* in the dataset. The labeling of a dataset is critical in being able to draw out the best of it's performance, while having even noisy data can lead to harder to understand effects [116, 204].

4.1 An Application Study - AMC

In order to understand the impact augmentation brings to RFML, an application space is needed. Due to the widely studied signal classification problem of AMC within RFML, the AMC problem space provides an opportunity to test the promise of augmentation in RFML data without having to perform a full exploratory study determining the network and training routines needed in order to perform well. Work performed by Sankhe *et al.* showed, in the RFML category of SEI, that when the channel, or propagation path, changes and the new channel is not in the training data, the performance of the RFML system rapidly degrades [100], thus implying that typical approaches are not channel invariant. From that, it can be concluded that the quality of a dataset will be higher if the propagation path of the intended environment for field usage is within the dataset. In simpler terms, the training data is known to be better when collected from the same physical location and conditions as what the deployed system will encounter. Additionally, not only does the environment matter, but the role of the algorithms in detecting and isolating a signal also play an important role since these imperfections in the algorithms have an impact on a network's performance in AMC when not considered [22]. One final factor known for AMC is that for diverse waveform spaces, a significant amount, i.e., $> 1M$ observations, of data is required [173]. In order to better gauge augmentation, focus will be on augmenting the detection imperfection space, FO and SRM, along with varying the effective SNR, as these are computationally inexpensive augmentations rather than trying to quantify the imperfections observed in the propagation path. Here, propagation path refers to any effects that deviate the signal from ideal digital representation, which include everything from the transmitter's DAC up through the receiver's ADC. The detection imperfections are taken as a post-processing effect imposed by the detection algorithm after the receiver's ADC. While the work in O'Shea *et al.* [173] does perform AMC with OTA data collection, it does so in a relatively benign environment, therefore the work is closer to what Sankhe *et al.* [100] called a *static* channel, rather than a more realistic *dynamic* channel, which used significantly more preprocessing to overcome.

This chapter seeks to investigate and answer several questions open in the field. First, with no first hand knowledge of the degradation of the signals to be seen, how well does a synthetically trained, validated, and tested network actually perform under real-world conditions? The major investigation here is the contrast between a synthetic dataset where, through

simulation, distortion is applied to the signals and compared to the field collection of data where all of the distortion is taken from the signals' propagation through the environment.

Second, what value does augmentation bring in contrast to just performing an extended capture? This question addresses the initial data collection concerns when starting a new problem or a repeat application within a new environment, which is because deep learning typically has a nonlinear relationship between performance and the number of observations in a dataset. For narrowband signals, getting another order of magnitude of examples may change the length of time running a collection from days into months of field time. In the absence of enough data, augmentation is relied upon to provide a greater observational data space, and by comparison to a traditional campaign is relatively cheap in terms of time and effort; however, the value of one augmented observation contrasted with one collected observation in terms of performance is not well understood. Further, many military spectrum access applications require modeling of channel effects that cannot be practically tested live (e.g., electronic warfare or atmospheric scintillation).

The third question investigated here is whether understanding the distributions of degradation sources impacts the ability of a network to achieve peak performance. That is, can an approximation made by an expert over a narrow range be sufficient to allow the network to generalize over the full degradation space, or will taking measurements of the degradation space and utilizing those measured values prove to be more beneficial to the network's performance in that measured range? There are two cases to examine under this question: synthetic generation and augmentation. For the synthetic portion of this inquiry, the focus is on whether drawing parameters from the assumed degradation region, or drawing parameters from estimations extracted from the observation space, allows for any change in performance of the trained network when tested against collected data. With the augmentation portion, the question is whether it matters how the degradation space is resampled in the augmented observations; for example, can augmentation be performed using an assumed parameter degradation region like with the synthetic example, or should the resampling come from the collected observation space instead?

4.1.1 The Approach

As the focus of this work is to examine the effects of the quantity and quality of the available data to the RF problem space of AMC, the architecture for the model to be trained and a DL training routine were identified and remain consistent for all aspects of the results shown within this study unless otherwise stated. The experiment, shown in Figure 4.1, consists of training a Convolutional, LSTM, Deep Neural Network (CLDNN) for a maximum of 50 epochs through all available training data after a 90%/10% split for training and validation datasets respectively. For the case where there are 101 observations for a waveform, 91 Observations per Class (OPC) are in the training dataset while 10 are in the validation dataset. A second condition for stopping is allowed for, in the form of early exiting when the

validation loss does not decrease for four epochs, and is responsible for all results presented here as the maximum number of epochs reached is 38. A similar training routine and process were used in our prior work, with multiple such examples achieving operational deployment [22, 34]. A discussion for the selection of the architecture, training routine, and datasets follow.

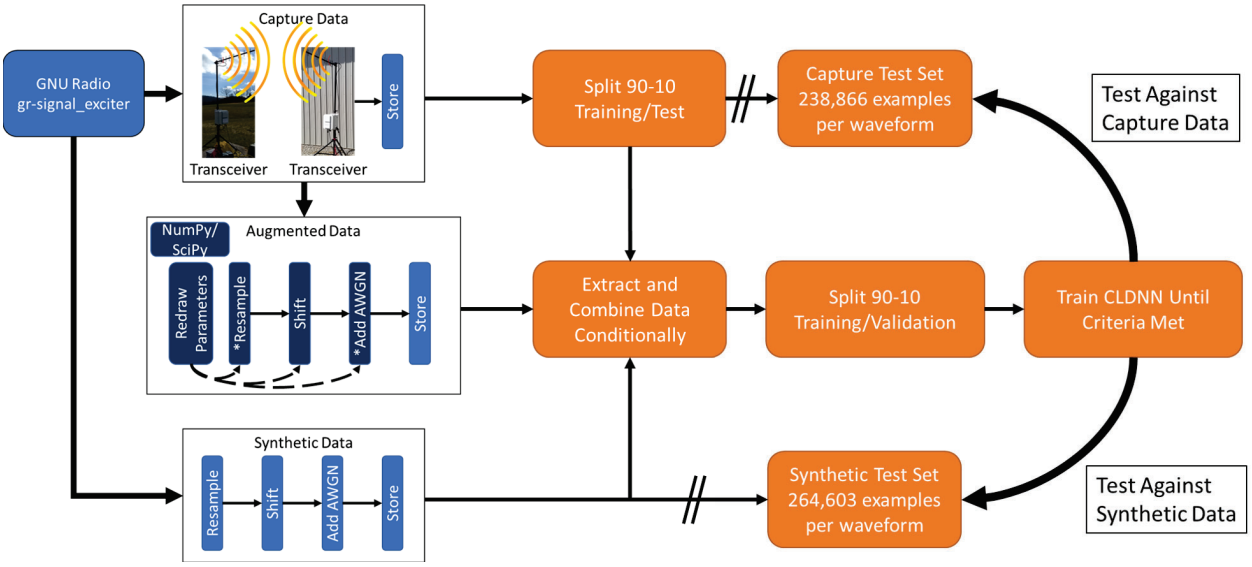

Figure 4.1: Experiment Configuration. At the start of each experiment, data is extracted from either a synthetic database, a capture database, and/or conditionally from an augmented database based on the extraction from the capture database. All data is then combined and split into 90%/10% training/validation datasets used to train a CLDNN network until an exit condition is met. The '//' indicate a separate file has been created to better isolate the testing data from the training data.

4.1.1.1 Network Architecture

An extensive investigation of RFML architecture was under taken by West *et al.*, where different networks from the literature were compared and contrasted [156]. The dominant network for performing AMC was found as the CLDNN, and a practical description for implementation was given by Flowers *et al.* with the addition of batch normalization throughout the network [205]. From these works, the network used in this experiment, as seen in Figure 4.2, is then a CLDNN, which accepts an input of two channels (In-phase and Quadrature floats) with 1024 samples passing through three 1D convolution layers with 50 output channels, using a 1×8 kernel size, whose input is padded with zeros such that the output sequence is of the same length as the input, followed by a ReLU activation, and a 1D batch normalization layer. The output of the first and third such layers are then concatenated along the channel dimension and passed through a single LSTM layer such that the channels are taken

as the features while the time sequence is fed through the LSTM's memory structure. For simplicity, the LSTM has a hidden size equal to the number of classes being used in the problem. The output of the LSTM is then flattened and passed through a linear layer with 256 output nodes with a ReLU Activation and 1D Batch Normalization. The final batch normalization layer is then connected to the final linear layer with the number of outputs equal to the number of classes with a Softmax Activation. This network structure was chosen as the baseline for the experiment for two main reasons, the first being the high performance seen in West *et al.* [156], and the second being that the convergence time with the training routine given next was shown to be quick in terms of epochs by Flowers *et al.* [205].

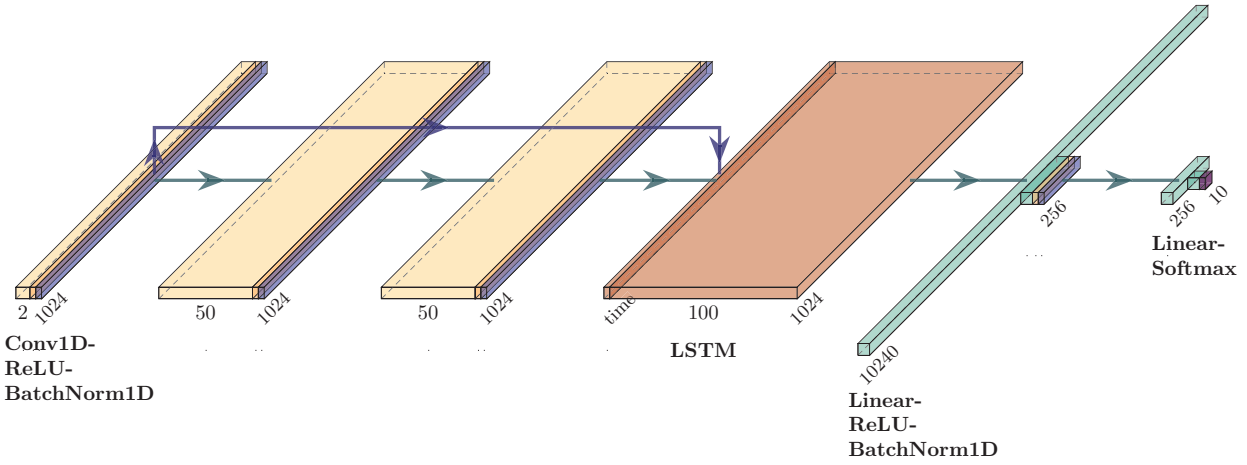

Figure 4.2: CLDNN Architecture for the Φ_{10} (Discussed in Section 4.1.2.4) waveform dataset.

4.1.1.2 Training Routine

The CLDNN network is trained consistently for all dataset quantities and sources. As discussed next, the dataset for each experiment is selected, and then split 90%/10% into training and validation datasets respectively. These two datasets are then distinct for that experiment and model trained. The data is then processed in batches consisting of 1500 total observations. Using the default Adam optimizer [206] within PyTorch [23] with Cross Entropy Loss, the network is allowed to continually train until 4 epochs have passed without any improvement in the validation loss metric or until a total of 50 epochs have been processed (in the experiment, 38 maximum epochs seen in a single training pass). The experiment keeps the weights whose validation was the least across all trained epochs for the model because, as the validation rises after such a minimum is reached, the training routine assumes the weights have begun overfitting.

While this training routine doesn't allow for significant deviation in the event a local minimum is found, it is chosen for the ideal properties of having a quick consistent goal and a limited processing window within which to achieve said goal. In order to best understand

the relationship between performance and quantity of available data, hundreds to thousands of networks need to be trained in order to regress the relationship while accounting for unknown outlier potentials; allowing a more lenient training routine would only extend the total time taken to train a single model. The approach above was selected while taking into consideration the total processing time needed to perform the investigation.

4.1.2 The Dataset

The database being used in this investigation was made from live collection of numerous waveforms at Virginia Tech's Kentland Farms over the course of 4 months in 2019 using two weather enclosed software-defined radios (shown in Figure 4.1); photos from the longer collection path are shown in Figure 4.3. The database consists of multiple waveforms of varying duration and quality. Based on the metadata available, the data was filtered to only select observations whose SNR was estimated to be above −10dB. Here the SNR is the direct measure of estimated total signal power over total noise power as

$$\text{SNR} = 10\log_{10}(\sum |s[i]|^2) - 10\log_{10}(\sum |\nu[i]|^2), \tag{4.1}$$

but, given the range of signal bandwidth observed, a normalization for bandwidth of the observations can be performed with

$$\text{SNR}_{BW} = \text{SNR} + 10\log_{10}(8 \cdot \text{SRM}). \tag{4.2}$$

Additionally, due to the nature of hardware collections and an active environment, observations that were found to have irregular sample values were filtered out from the dataset. The data was then segmented such that observations of 1024 samples could be extracted in a continuous fashion with no two observations being contiguous outside of 1024 samples. In other terms, given one observation starting at sample 0, the next observation could not start until sample 2048, assuming there are at least 3072 samples available in that record. The final filter placed on the data left all waveforms evenly balanced in terms of available observation counts, resulting in 2,388,667 total observations for each waveform class considered. Of the data that remained, the detector imperfections that were estimated showed that the SNR was between −10 and 80 dB, with the majority of the data being below 20dB. Figure 4.4 shows the distribution of SNR observed prior to any filtering of the dataset for the 10 classes examined in the study. The estimated FO were found to be bounded by ±20% of the receiver's sampling rate though heavily concentrated between ±5%. Figure 4.5 shows the distribution of FO observed prior to any filtering of the dataset for the 10 classes examined in the study. The SRM found signals in the range of 2-32 times that of the Nyquist rate, though roughly twice as likely to be between 2-8 as between 8-32. Figure 4.6 shows the distribution of SRM observed prior to any filtering of the dataset for the 10 classes examined in the study. In order to better make use of the estimated distributions of the captured data, a joint Kernel Density Estimator (KDE) was performed per modulation on the imperfections of SNR, FO, and SRM to be used while augmenting and synthetically generating datasets. Table 4.1 provides an overview for the datasets used in this work.

Figure 4.3: From left to right, the collection node sitting outside the Kentland Farms Experimental Aviation Systems Laboratory, the collection node sitting roughly a kilometer away uphill, the view looking back at the first collection node from the second along the antenna. Photos taken by Zach Leffke.

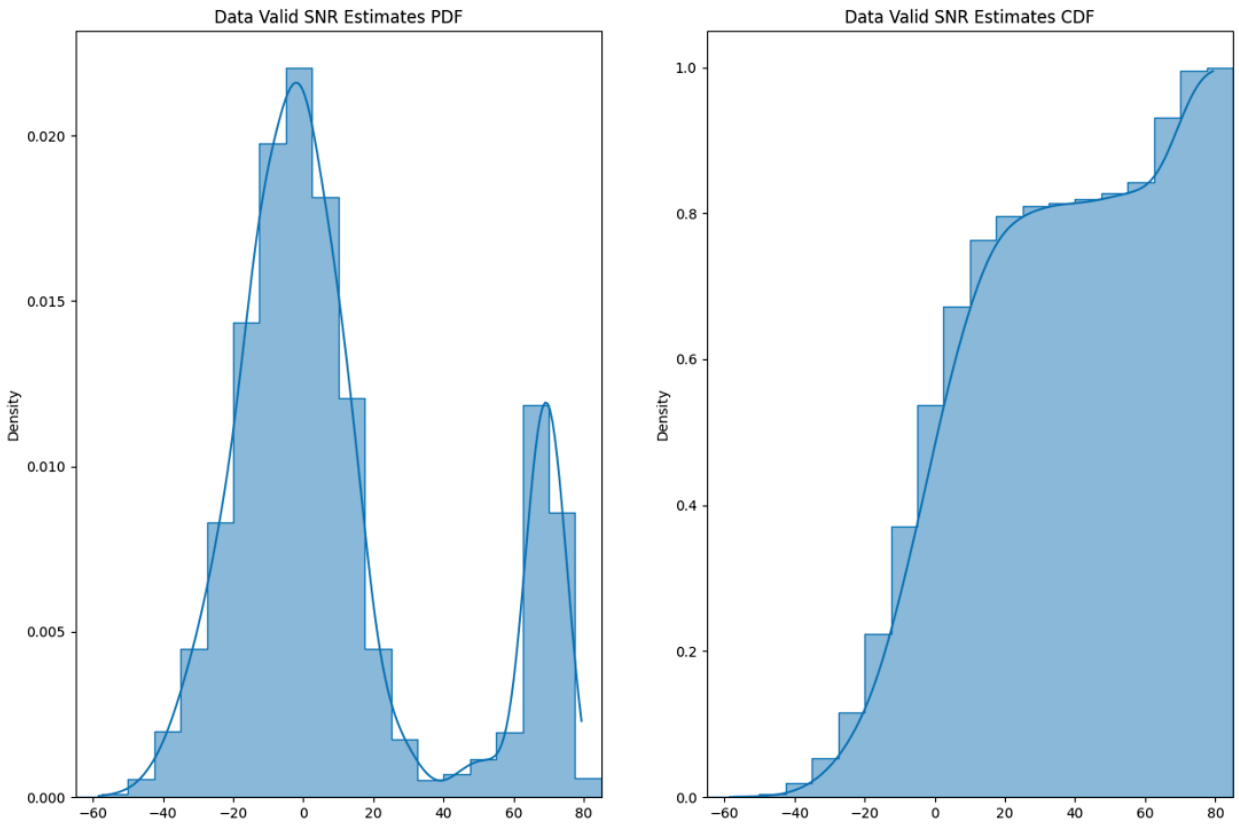

Figure 4.4: The observable distribution of SNR in the dataset for the 10 waveforms prior to any filtering, based solely on the estimated metadata

4.1.2.1 The Capture Dataset

All the observations in the dataset described above are then split into a general training set (Ω_C) and a test set (Ω_{TC}) against which all results will be compared. The training/testing split is 90%/10% consisting of 2,149,801 and 238,866 OPC respectively. As part of the investigation, the number of observations drawn from the training set varies across iterations, but every trained model is tested against the full Ω_{TC}.

4.1.2.2 The Augmented Dataset

Additionally, there are other datasets that can be used while training with captured data. The first is an augmented dataset that is linked to the capture training set and, for every observation, 10 augmentations are made. The data drawn from the augmented dataset is conditionally linked in such a way that only augmentations of data selected from the capture training dataset will be available to be drawn, and then when the augmentation factor is less than 10, which exact augmentation is drawn is left to random uniform sampling. There

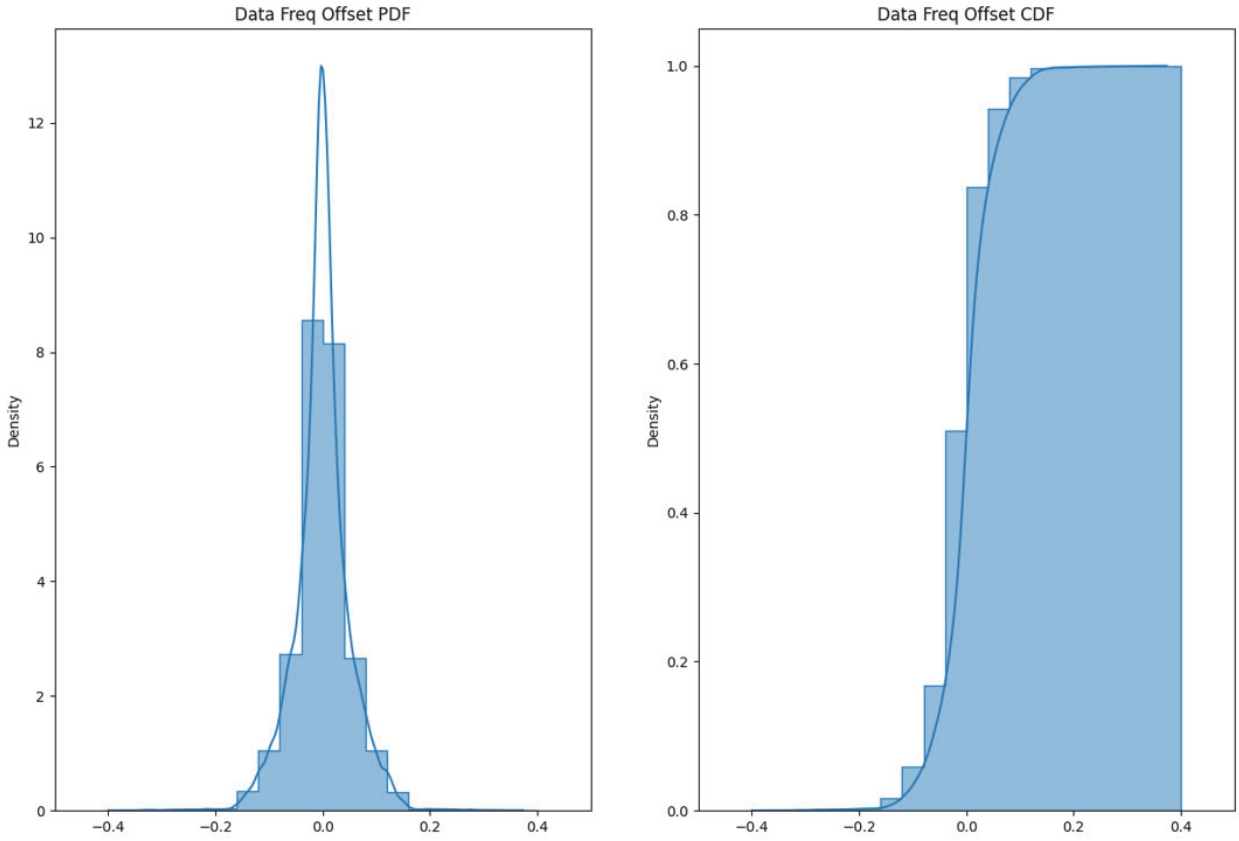

Figure 4.5: The observable distribution of FO in the dataset for the 10 waveforms prior to any filtering, based solely on the estimated metadata

are two distinct augmented datasets to understand what effect, if any, the distribution of parameters has on the performance during training.

The first augmented dataset takes on a range of parameters given as an expected performance range of the capture data prior to performing any capture. The signals are augmented in such a way that the SNR is uniformly drawn from the range of 0-20dB, the FO is taken uniformly in the range of $\pm 10\%$ of the sampling rate, and the SRM is taken to be uniform in the range of 2-8 times that of the Nyquist rate for the captured signals. Given the available metadata and the observations of 1024 samples, any time a random value is drawn that cannot be achieved, for example a signal with an estimated SNR value of 5dB attempting to augment the signal to an SNR of 10dB, the augmentation is nulled and whatever the current estimate is holds. Likewise, SRM augmentation that requires decimation reducing the number of samples below the desired 1024 observation length is nulled. This dataset is the Ω_{AS} dataset, and the parameter space is drawn from three independent distributions using NumPy [207].

The second augmentation dataset makes use of a Gaussian joint kernel density estimate,

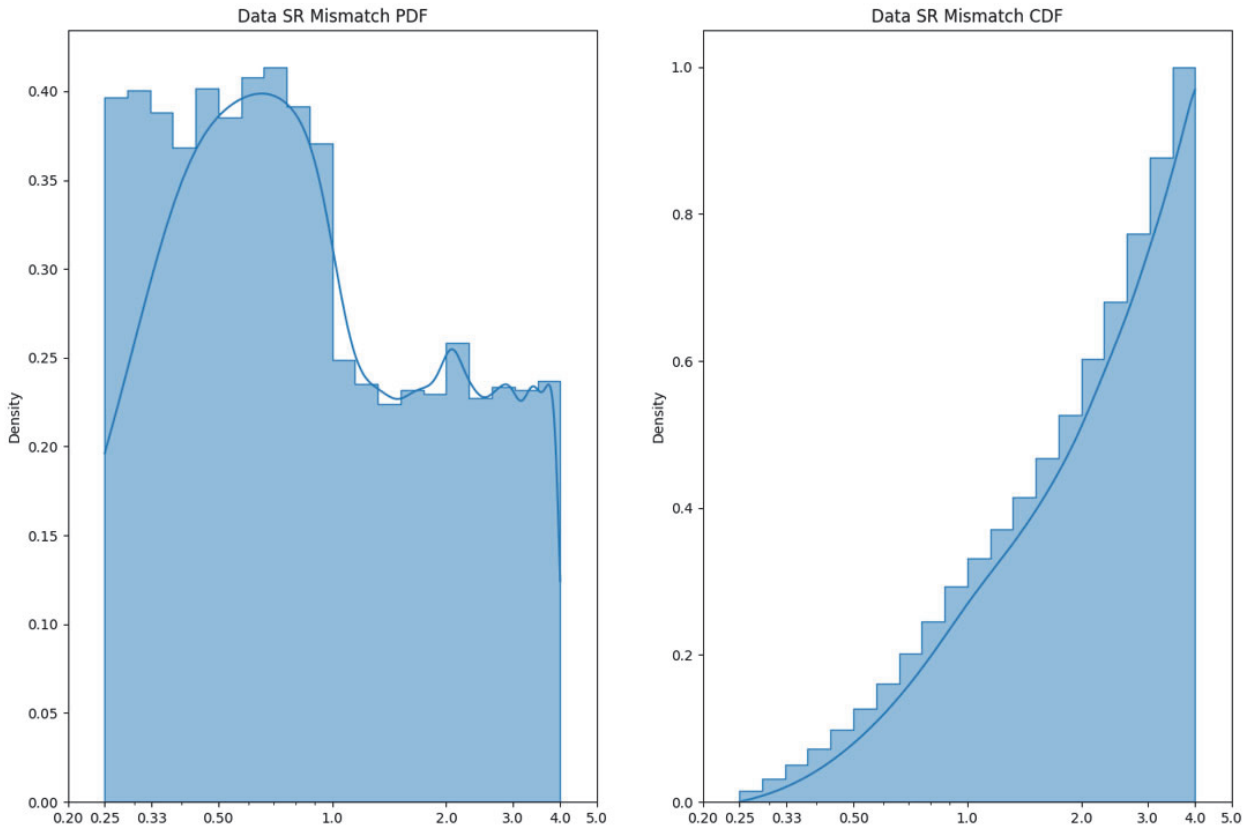

Figure 4.6: The observable distribution of SRM in the dataset for the 10 waveforms prior to any filtering, based solely on the estimated metadata

using SciPy, of the available data per captured waveform and uses a random draw from that joint estimate to perform the augmentation [208]. Again, any augmentation that is not feasible is nulled. This dataset is the Ω_{AK} dataset. In this way, the investigation can contrast the value of data analysis on the captured data with regard to augmentation or whether a general blind practical range will suffice.

Additionally, each of these two augmented datasets has a counterpart that makes use of both the augmented and captured data. In order to augment, the captured data must be available, so making use of both makes sense. These datasets are the $\Omega_{AS\cup C}$ and $\Omega_{AK\cup C}$, respectively.

4.1.2.3 The Synthetic Dataset

The final two datasets consist of simulated synthetic data for the waveforms under test. Using the same distribution assumptions for the SNR, FO, and SRM as the Ω_{AS} dataset, the synthetic dataset randomly generates observations for each waveform to be used during

Table 4.1: Description of fundamental datasets used within this study.

Training		
Symbol	Source	Description
Ω_C	Capture	Consists of only capture examples
Ω_{SS}	Synthetic	Consists of simulated examples using an assumed synthetic distribution
Ω_{SK}	Synthetic using KDE	Consists of simulated examples using the KDE of the capture dataset
Ω_{AS}	Synth Augmentation	Consists of only augmentations using the synthetic distributions
Ω_{AK}	KDE Augmentation	Consists of only augmentations using the KDE of the capture dataset
Testing		
Ω_{TC}	Capture	Consists of only capture examples
Ω_{TS}	Synthetic	Consists of simulated examples using an assumed synthetic distribution

training, Ω_{SS}. A second synthetic training set is used under the assumption that better metrics are known for SNR, FO, and SRM based on the targeted detection routine in place on the observer device and draws the parameters from the KDE discussed with the Ω_{AK} dataset; the Ω_{SK} dataset. This approach can help quantify the value of real world data that undergoes true transceiver and channel degradation that is not as easily replicated through simulation. Additionally, a single testing dataset, Ω_{TS}, is created using the blind distributions as a means for comparing what a purely synthetic test set would say about the performance of a trained model with the results of testing captured data from the field.

4.1.2.4 Waveform Space

The waveforms selected from those available in the data consist of 3, 5, and 10 classes denoted by Φ_3, Φ_5, and Φ_{10}, respectively. The waveforms associated with each space are provided in Table 4.2. By having three different dimensions for the class size, the work is able to examine any differences that data quantity might have with regard to the difficulty of the problem. The two smaller subsets are chosen due to the frequent usage in traditional feature based AMC approaches [209].

Table 4.2: Waveform Spaces. Different waveforms used to examine the problem space as the complexity changes.

Group	Waveforms
Φ_3	BPSK, QPSK, Noise
Φ_5	BPSK, QPSK, QAM16, QAM64, Noise
Φ_{10}	BPSK, QPSK, QAM16, QAM64, BFSK, GMSK, AM-DSB, FM-NB, GBFSK, Noise

4.1.3 Clipping of Outliers

In general, there are several networks that fail to converge in either the synthetic or captured data space. This failure to converge is primarily identified by the network not being able to escape a random guessing state for the problem space. Initial removal of outliers in network results relied on this notion and treated any network that didn't achieve a statistical separation in terms of performance from that of the convergent network. As more data points were observed, another strong correlation was observed. Networks that failed to converge by this definition predominantly shared a secondary trait in that there appeared to be a transition point in terms of total number of batches seen by the network during the training stage on the order of one hundred batches. Figure 4.7 plots all data points observed while training networks and shows the number of batches observed and the best performance of the network on either the captured test data, Ω_{TC}, or the synthetic test data, Ω_{TS}. To find a reasonable clipping point for the number of batches a network must have seen to separate between a 'good' network data point and outliers that failed to converge, Analysis of Variance (ANOVA) was conducted to look at the four independent variables relative to network performance.

1. The training data source(s) available
2. The number of waveforms in the training data
3. The source of randomization; whether using the KDE or Expert Approximation
4. The number of batches seen during training acting as a good/bad binary split.

Initial 1-way analysis reveals that #1, #2, and #4 must reject the null hypothesis that these parameters have no correlation with the resulting network performance, while the result is that #3 consistently must fail to reject the null hypothesis across the range while using a p-value of 0.03. Excluding the KDE condition, 2-way analysis can be conducted with the remaining three variables and reveal that the interaction between any two requires the rejection of the null hypothesis. However, when trying to perform a 3-way analysis there

are holes where controlling for all three results in zero available observations. Controlling for the independent variables #1 and #2, independent sweeps are taken per subset and a 1-way ANOVA is performed over the sweep region for #4. As the resulting p-values in this sweep exceed the range of double precision to be combined in a meaningful manner, the median p-value across all combinations is used to contrast the value selected as the clipping point. By this measure, p-values with a clipping point in the range of $[107, 147]$ batches seen during training are equivalent to machine precision, therefore the value of 122 is chosen to be the cutoff point. This cutoff point cuts out all networks that failed to converge under the original outlier search in terms of network performance, and only cuts two observations out from those that can be considered as converged. An untested expectation is that this could be used for other model architectures, but would require further study, and most likely would result in a model architecture dependency to select the cutoff value. For simplicity, outliers are removed from most plots and are not used in the regression of the relations. Figures 4.8-4.16 show outlier regions that were originally created based solely on inability to escape the random guessing region by a factor unique to each problem's observation space.

4.1.4 Performance Regression

The analysis starts with the first major question in this work; for a given synthetically trained network, how well does the network perform when applied to real-world data from the field? To answer this question, three different waveform spaces, $\{\Phi_3, \Phi_5, \Phi_{10}\}$, are examined with regard to the dataset source used while undergoing training. Each trained model is then evaluated on both the synthetic and the capture test sets, $\{\Omega_{TS}, \Omega_{TC}\}$. Plotting the accuracy of the Ω_{TC} against the accuracy of the Ω_{TS} in Figures 4.8-4.10, the overall performance can be seen for the waveform spaces $\{\Phi_3, \Phi_5, \Phi_{10}\}$ respectively. For each plot, a vertical and horizontal black solid line, without markers, are used to indicate random guessing along each axis, with a third diagonal line indicating where performance would be equivalent across the two test sets. This diagonal line is clipped where it meets the first two lines because falling closer toward zero with any significance would require intentional or adversarial manipulation of the training routine or data, which is not considered in this study [210]. The trained networks are then represented with different color and marker combinations representing different dataset sources used during training.

In terms of an ideal performance, non-filled markers should be concentrated in the top right corner of each plot, indicating high accuracy on both the Ω_{TC} and Ω_{TS} datasets, which can be used as an indication that the nuisance parameter space in the capture data that is not being modeled in the synthetic data has been well generalized over. Instead of this ideal performance, two distinct cases are seen: higher performance on Ω_{TS}, or higher performance on Ω_{TC}. The data types that performed best on the synthetic test set were the models trained from the synthetic training datasets Ω_{SS} and Ω_{SK}, using triangles as the marker shapes, whereas the data types that performed best on the capture test set were the models trained from the capture and augmented datasets Ω_C, Ω_{AK}, and Ω_{AS}, using marker color/shapes

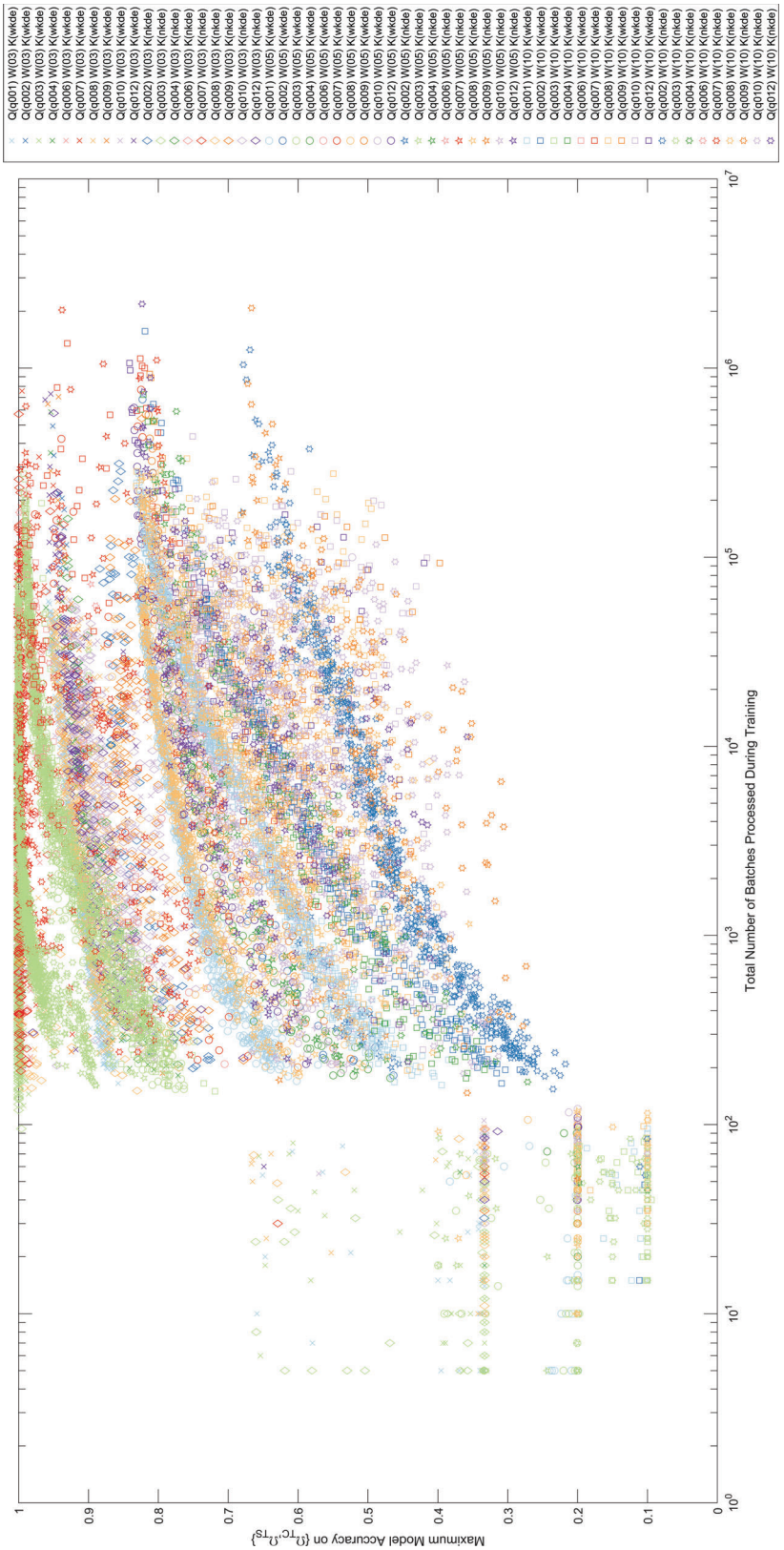

Figure 4.7: Here the number of batches each network processed during training are examined; while the figure contains too many different types of results to properly display, the key point here is there is a clear junction point around ∼120 batches seen, below which the majority of failed convergence networks can be split from convergent networks.

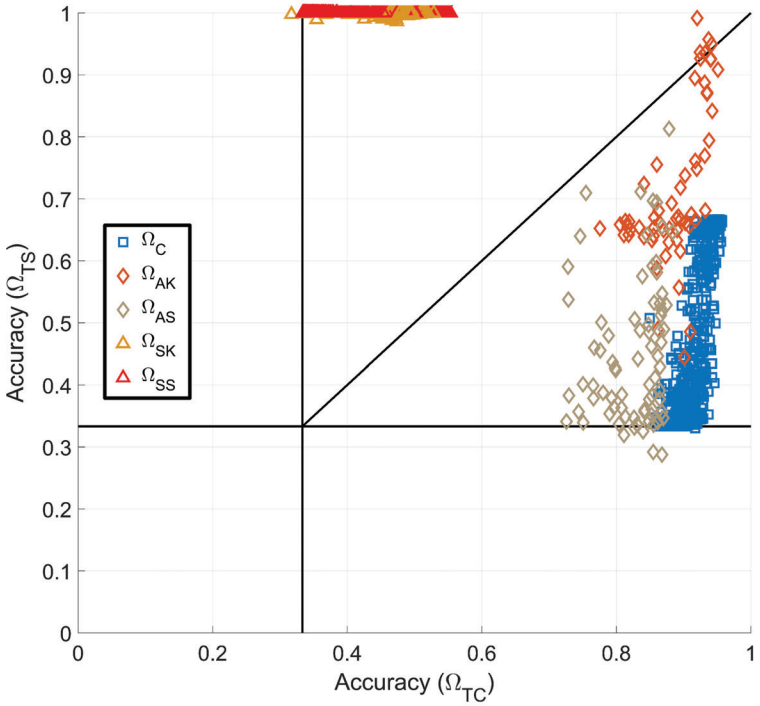

Figure 4.8: Performance of individually trained networks using the five datasets on the Φ_3 waveform space contrasting the performance observed on the synthetic (Ω_{TS}) and capture (Ω_{TC}) test sets. While ideal performance is in the top right corner, an acceptable performance is to the right in general for real-world performance. While high performance is found on data similar to their respective training datasets, neither synthetic nor captured data performs well on the other. Through augmentation of captured data, a greater performance is observed on the synthetic data, as a result indicating better generalization over the full parameter space.

blue/square, red/diamond and tan/diamond respectively.

While this bias for alike datasets intuitively makes sense, there are some unique outcomes that are not obvious. The outliers, which are results that failed to converge to a performance and were designated as such with the approach discussed in Section 4.1.3, have been removed from the plots, but consist of points clustered around the intersection of the three black lines.

Observing the figures for the synthetic trained datasets (Ω_{SS}: red triangle and Ω_{SK}: yellow triangle markers) on the performance comparison show that the synthetic data is very easily learned when tested on Ω_{TS}, but generally fails to do better than twice that of random guessing on Ω_{TC}. By looking closely at how the distribution affects the performance of synthetic data we can start to answer the third question of how the assumed distribution affects the performance of a system, but this will be revisited later. Table 4.3 shows the relational change in performance on both Ω_{TS} and Ω_{TC} when contrasting the two synthetic datasets. Contrasting the average accuracy as a ratio, $\overline{\alpha_{\Omega_{SK}}}/\overline{\alpha_{\Omega_{SS}}}$, there is a marginal loss

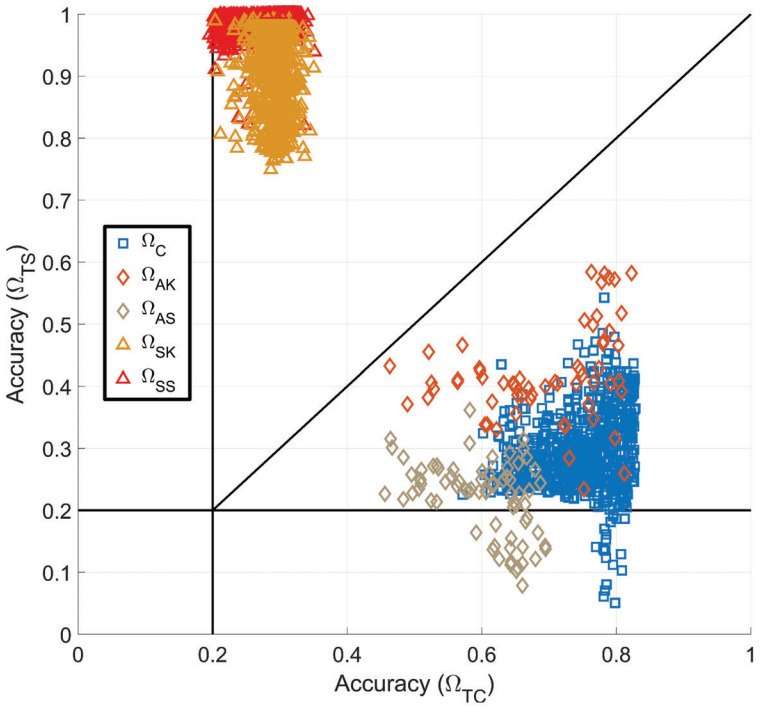

Figure 4.9: Performance of individually trained networks using the five datasets on the Φ_5 waveform space contrasting the performance observed on the synthetic (Ω_{TS}) and capture (Ω_{TC}) test sets. While ideal performance is in the top right corner, an acceptable performance is to the right in general for real-world performance. In contrast to Figure 4.8 the generalization over the parameter space is less pronounced when using augmented data in the same quantity range per waveform; however, the augmented dataset with knowledge of the parameter space (Ω_{AK}) still provides improved generalization, while augmentation without such knowledge ends up reducing generalization instead.

in performance on the smallest waveform space, Φ_3, when using the synthetic data created with the KDE, while the KDE drawn synthetic data performs a little better on average with Ω_{TC} than the non-KDE drawn synthetic data; using Welch's two-sample t-test, the significance of such improvement rejects the null hypothesis that both datasets should have the same average performance [211]. By contrast, the decrease in average accuracy on Ω_{TS} and the increase in average accuracy on Ω_{TC} for Φ_5 show a much greater significance indicating that the use of the imperfections from the intended environment can improve the fidelity of synthetic datasets. However, the improvement on the capture test set is lost as the waveform space continues to grow with Φ_{10}. In general, there is a slight possibility that creating synthetic data that only considers the detector imperfections can be of high enough fidelity to train as the number of synthetic examples increases by many orders of magnitude, but overall these results show that modeling only the detector imperfections while ignoring the propagation path is not significant enough to properly train a system heading to the field. This result answers the first question in that when given a network

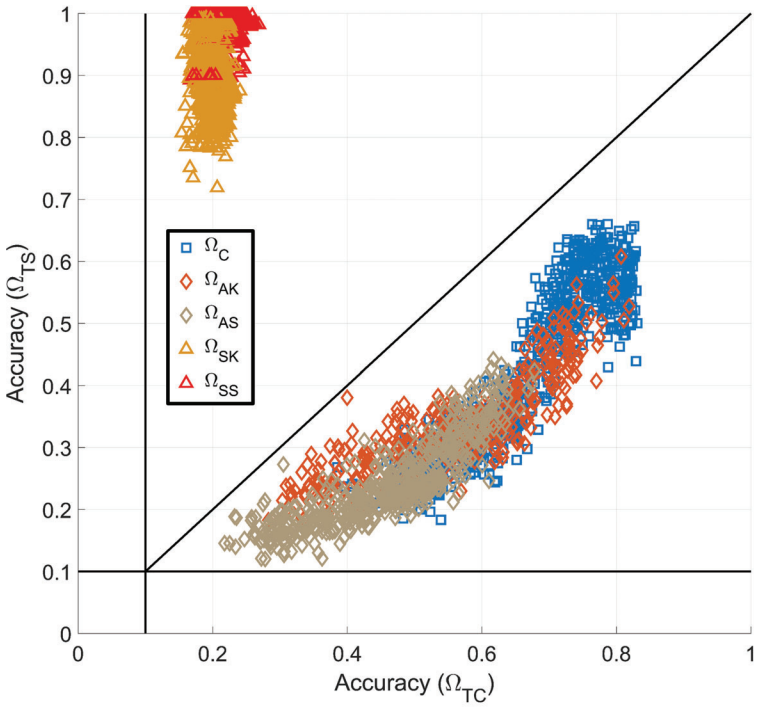

Figure 4.10: Performance of individually trained networks using the five datasets on the Φ_{10} waveform space contrasting the performance observed on the synthetic (Ω_{TS}) and capture (Ω_{TC}) test sets. While ideal performance is in the top right corner, an acceptable performance is to the right in general for real-world performance. In contrast to Figures 4.8 and 4.9 the generalization observed through augmentation is only observable at low accuracies on Ω_{TC} and becomes negligible as accuracy increases. As the difficulty increases, the improved generalization that can be observed with augmented data within lower difficulty problems disappears in higher difficulty problems when constrained to the same quantity/quality availability during training. It is unclear whether quantity or quality directly play a role of significance in this observation.

trained and tested in the synthetic space, that network will not perform well in a real system without a much higher fidelity simulated dataset. Additional work is still needed to determine at what threshold simulated data can be considered high enough fidelity when designing and developing a deployable system. In all likelihood, finding that threshold is going to be very dependent on the target operating environment. This includes how much is known about the transceiver-to-transceiver propagation path, which includes everything from the transmitter's DAC through the receiver's ADC and any effects of the detection and isolation stages inherent to that receiver.

Table 4.3: Test Accuracy's Observed Response to Synthetic Datasets Ω_{SS} and Ω_{SK}. Examines the significance of where the detection imperfections are drawn from while contrasting to what is tested against. Smaller p-values (< 0.05) indicate larger significance of where the simulation degradation is drawn from. p-values found using Welch's two sample t-test. Results indicate there is a small, but significant change in network performance on both test sets when choosing synthetic distributions for simulated datasets.

Waveform Space	Average Accuracy Ratio $\overline{\Omega_{SK}}/\overline{\Omega_{SS}}$		p-values	
	Ω_{TS}	Ω_{TC}	Ω_{TS}	Ω_{TC}
Φ_3	0.9992	1.079	1.204e−44	2.015e−31
Φ_5	0.9277	1.053	5.188e−154	1.237e−21
Φ_{10}	0.9421	0.9512	2.892e−103	2.514e−31

4.2 Data Augmentation for RFML

To start answering the second question of what value augmentation brings to the problem, the attention shifts focus to the proximity of the models trained with Ω_C, Ω_{AK}, and Ω_{AS} datasets (blue/square, red/diamond, and tan/diamond markers, respectively) to the diagonal line where there are performance generalizations that become less pronounced as the waveform space grows. For the capture dataset models, the clusters show that Ω_{AK} typically achieves better performance on Ω_{TS} than both Ω_C and Ω_{AS}, indicating that the degradation encountered from imperfect detector estimation, when accounted for in augmentation, does help the network better generalize over the nuisance parameters present in the capture data. Conversely, and more surprisingly, augmenting the dataset with the assumed synthetic range actually made the performance on the Ω_{TS} worse than without the augmentation. One conclusion that can be drawn from this is that the degradation encountered between one transceiver's DAC to another transceiver's ADC has a greater effect on performance than the degradation caused by the detection algorithm's imperfections, assuming detection and isolation are achieved, and that simply redrawing the parameters observed by one detection routine for another detection routine will not be sufficient without taking into account the propagation degradation on the path between the DAC and ADC in this new environment. Such refinements will become more important as RFML systems begin to incorporate learned behaviors that have been trained with and transferred from another node.

Figures 4.11-4.13 show the relationship between the achieved performance on Ω_{TC} of each individually trained network on the y-axis, with the x-axis corresponding to the total number of uniquely stored OPC available during the training of the network. By looking at the relation between accuracy achieved and total data per class used during the training, two

Table 4.4: Log-Linear fits, $\text{qty} = 10^{\left(\frac{\alpha - p_2}{p_1}\right)}$, for data presented in Figures 4.11-4.13.

Dataset	Waveform Space (p_1, p_2)		
	Φ_3	Φ_5	Φ_{10}
Ω_C	$0.03351, 0.7424$	$0.07231, 0.3893$	$0.1286, 0.04641$
Ω_{AK}	$0.04676, 0.6323$	$0.1007, 0.1684$	$0.1613, -0.2051$
Ω_{AS}	$0.03951, 0.6108$	$0.05988, 0.2917$	$0.1271, -0.1460$
Ω_{SS}	$0.01207, 0.4199$	$1.3866\text{e}{-11}, 0.2918$	$0.0001972, 0.1988$
Ω_{SK}	$0.01223, 0.3831$	$0.02400, 0.1590$	$0.004765, 0.1874$

important pieces of information can be extracted. First, the trend lines further to the top for a given total quantity exhibit a higher quality within the data, because a better performance is achievable. Using this, the quality of data decreases in the following order of the capture datasets across all examined waveform spaces: Ω_C, Ω_{AK}, Ω_{AS}. Second, assuming the log-linear trend holds, as shown in Table 4.4, without an asymptotic bound on accuracy (an asymptotic bound should be expected, and is given with the dotted lines), a forecast can be made on just how much data of each type is required in order to achieve ideal performance, and these quantity values are shown in Table 4.5, with the total continuous capture time required to perform such a capture, as has been done for this dataset for each waveform space, is given in Table 4.6 in terms of days. As the trends are not consistent across all waveform spaces, also plotted is the 95% confidence region around those linear trends in shaded regions bounded by dashed lines with the same marker. However, assuming that the trends are consistent given enough observations, the general results align well with intuition in that data captured directly from the test environment is of highest quality and needs the least number of observations to achieve a target performance for the given model architecture and training routine. Second to the captured data, augmentation of data to match the nuisance parameter distributions from the test environment provides the next highest quality data for training, followed by naive augmentation that doesn't consider the full nuisance parameter distributions. Coming in last, by many orders of magnitude, is synthetic data that only considers detection imperfections while simulating the waveform spaces.

Table 4.5: Quantity of examples per class needed to achieve 100% accuracy for each dataset source and waveform space. Extrapolated from linear trends in Figures 4.11-4.13. Assuming no asymptotic limit.

Dataset	Waveform Space		
	Φ_3	Φ_5	Φ_{10}
Ω_C	48.3e6	277e6	25.7e6
Ω_{AK}	72.8e6	180e6	29.3e6
Ω_{AS}	7.05e9	672e9	1.03e9

Table 4.6: Quantifying the duration of a continuous capture, with no down time needed, in order to capture all data required to fulfill the Ω_C requirement for each waveform space assuming a 40kHz sampling rate of a 5kHz baud rate signal in Days. Assuming no asymptotic limit.

Dataset	Waveform Space		
	Φ_3	Φ_5	Φ_{10}
Ω_C	85.9	823	152

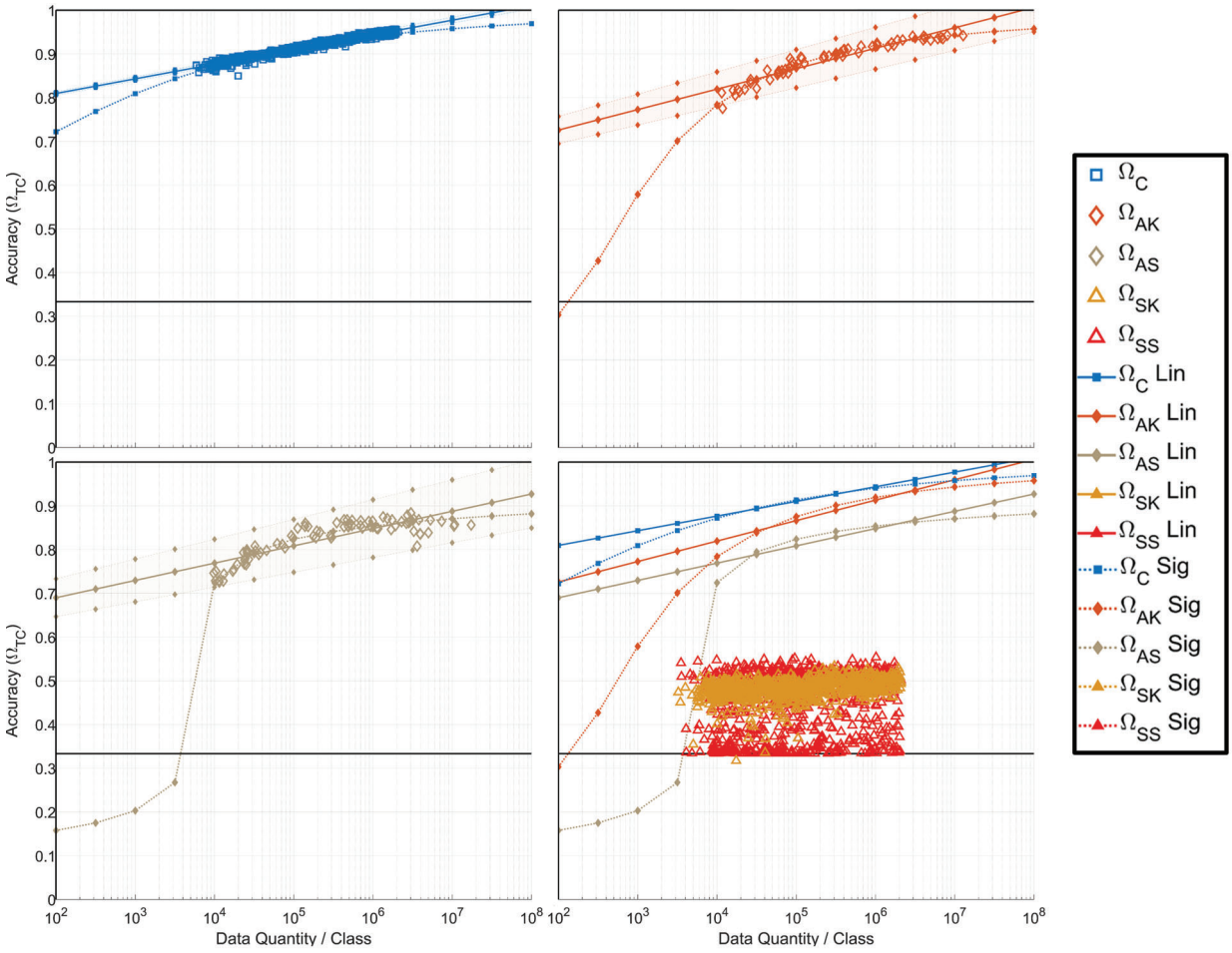

Figure 4.11: Performance of models trained using the five elemental datasets on the Φ_3 waveform space for a given data quantity. The solid horizontal line represents a network that is performing as well as a random guess. Solid lines with markers represent the trend in terms of examples per class that are needed to achieve a given accuracy on the capture test data, while the shaded regions between dashed lines with the matching markers indicate a 95% confidence region for that trend. Synthetic datasets are omitted from the trend analysis as no significant trend was observed from these datasets. Trends are derived with outliers removed. The further to the top the trend line, the higher the quality of data. Additionally, a dotted line is fit to the data using a log-sigmoid regression with the assumption that 100% accuracy is possible given the asymptotic curve perceived in the data and the impossibility of performance greater than 100% accuracy.

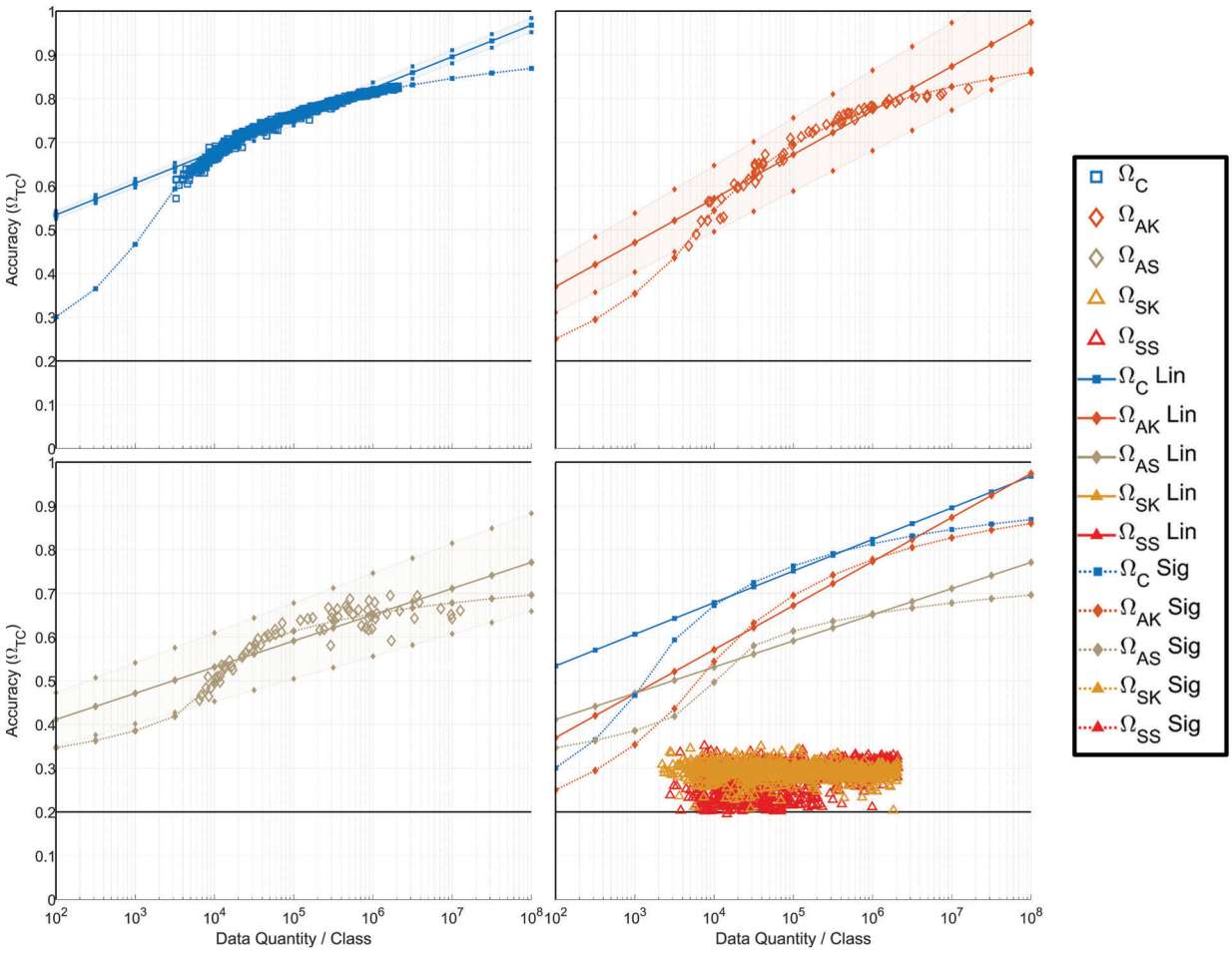

Figure 4.12: Performance of models trained using the five elemental datasets on the Φ_5 waveform space for a given data quantity. The solid horizontal line represents a network that is performing as well as a random guess. Solid lines with markers represent the trend in terms of examples per class that are needed to achieve a given accuracy on the capture test data, while the shaded regions between dashed lines with the matching markers indicate a 95% confidence region for that trend. Synthetic datasets are omitted from the trend analysis as no significant trend was observed from these datasets. Trends are derived with outliers removed. The further to the top the trend line, the higher the quality of data. Additionally, a dotted line is fit to the data using a log-sigmoid regression with the assumption that 100% accuracy is possible given the asymptotic curve perceived in the data and the impossibility of performance greater than 100% accuracy.

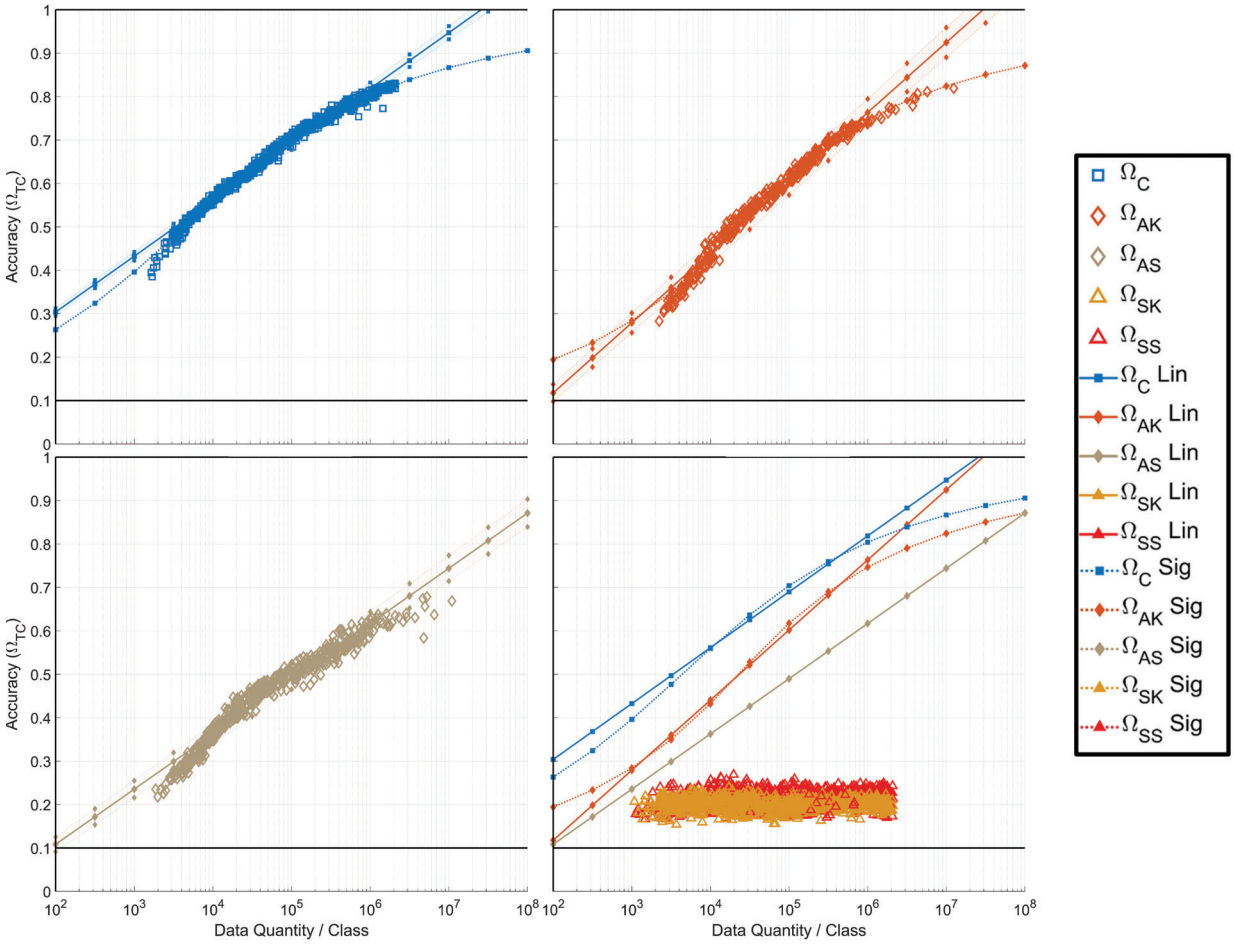

Figure 4.13: Performance of models trained using the five elemental datasets on the Φ_{10} waveform space for a given data quantity. The solid horizontal line represents a network that is performing as well as a random guess. Solid lines with markers represent the trend in terms of examples per class that are needed to achieve a given accuracy on the capture test data, while the shaded regions between dashed lines with the matching markers indicate a 95% confidence region for that trend. Synthetic datasets are omitted from the trend analysis as no significant trend was observed from these datasets. Trends are derived with outliers removed. The further to the top the trend line, the higher the quality of data. Additionally, a dotted line is fit to the data using a log-sigmoid regression with the assumption that 100% accuracy is possible given the asymptotic curve perceived in the data and the impossibility of performance greater than 100% accuracy.

Table 4.7: Log-sigmoid fits, $\alpha = 0.5\frac{b\cdot(\text{qty}-c)}{(1+|b\cdot(x-c)|^k)^{(1/k)}} + 0.5$, for data presented in Figures 4.11-4.13.

Dataset	Waveform Space (b, c, k)		
	Φ_3	Φ_5	Φ_{10}
Ω_C	$0.231, 5.96\text{e}{-7}, 2.70$	$0.610, 3.11, 0.974$	$0.333, 3.64, 1.90$
Ω_{AK}	$0.626, 2.74, 1.58$	$0.464, 3.79, 1.15$	$0.393, 4.35, 1.53$
Ω_{AS}	$39.5, 3.76, 0.416$	$1.42, 4.00, 0.420$	$0.254, 5.08, 16.7$

Table 4.8: Quantity of examples per class needed to achieve 95% accuracy for each dataset source and waveform space. Extrapolated from logistic fit in Figures 4.11-4.13.

Dataset	Waveform Space		
	Φ_3	Φ_5	Φ_{10}
Ω_C	3.2e6	2.66e19	17.1e9
Ω_{AK}	26.2e6	1.51e16	1.92e12
Ω_{AS}	7.24e47	∞	456e6

Instead of using log-linear trends, a log-sigmoid parametric fit's parameters are shown in Table 4.7 and results in the observation quantities in Table 4.8 in order to achieve a 95% accuracy in the problem space, given that 100% accuracy would require an infinite quantity of data. This results in the more likely capture durations shown in Table 4.9. These results show that for the smallest waveform space an order of magnitude less capture duration can be performed for giving up the 5% accuracy of the linear prediction, whereas for the larger waveform spaces several orders of magnitude longer capture duration will be needed even while giving up that 5% performance. One more note is that the sigmoid regression assumes that 100% accuracy is an asymptotically achievable feat, while it is more likely that given the model and training style that the peak performance achievable would still be less than 100%.

Table 4.9: Quantifying the duration of a continuous capture, with no down time needed, in order to capture all data required to fulfill the 95% accuracy requirement using dataset Ω_C for each waveform space assuming a 40kHz sampling rate of a 5kHz baud rate signal in Days.

Dataset	Waveform Space		
	Φ_3	Φ_5	Φ_{10}
Ω_C	5.72	44.7e12	101e3

One question that naturally follows this quality comparison of the datasets is that if augmented data is of lower quality, why not just focus on getting more captured data? The primary reason for relying on augmentation is cost, both in terms of time and money. In terms of time, the capture dataset was collected over a 4-month window in 2019, while the augmented datasets were generated over the course of 2-4 days each and contain an order of magnitude more observations per dataset. For full comparison, the synthetic datasets were generated over the course of 7 days for each dataset and are of the same order of magnitude as the captured data. One contributing factor for the increased generation time of the synthetic data was the design decision of extracting only one observation per execution of GNU Radio flowgraph, rather than extracting many observations from a single execution, which was done to decrease any dependence between observations within the dataset. The second cost is the monetary expenditures for procuring the transceivers, and making them robust enough to last 4 months of continuous use, paying for the power and space needed to make the transmissions, and the personnel for setting up and maintaining the capture. Determining the value of data is beyond the scope of this work.

So far the results have been shown in total number of observations used, but there is one more important way to look at the augmentation performance, in that there must be some foundation of capture data from which to augment. Figures 4.14-4.16 shuffle the results of the capture and augmented datasets to show the accuracy achieved on Ω_{TC} as a function of the capture data quantity that went into each model's training. This means that for a given value on the x-axis, all data points required the same number of capture OPC in order to achieve the performance shown. What these figures do not show is the augmentation factor used by each augmented network result. In this work, the augmentation factor is upper bounded by 10 due to the choice of having the augmentations performed prior to training the network and the storage constraints of the servers used, rather than augmentation performed online during training that would be one-offs unique to each augmented network as is done in image-based ML such as YOLO version 5 networks [126]. From Figures 4.14-4.16, two more beneficial aspects of augmented data can be observed. The first benefit of augmentation is that it allows for network convergence when the number of capture observations is not substantial enough to converge on their own. This is tremendously beneficial when planning for a capture event and determining how long the event must be in order to achieve a desired performance level by establishing the trends, like what was done in Table 4.5 but performed in an order of magnitude smaller time window as an exploratory capture event. The second benefit is seen when there are only a set number of observations available within the capture dataset, and characterization about the degradation due to the detection algorithm, which can be measured, as under these conditions the accuracy of the networks trained with augmentation exceed those of the networks with only capture data alone, which becomes more pronounced as the problem complexity increases. From these results, while remembering that the augmentation used in this work is a static augmented dataset with a bounded number of augmentations set to 10, the full effect of augmentation and how performance changes with dynamic, large augmentation factors (>10) and as to whether there are any diminishing returns as the augmentation factor increases is outside the scope of this

work and is an area for future work.

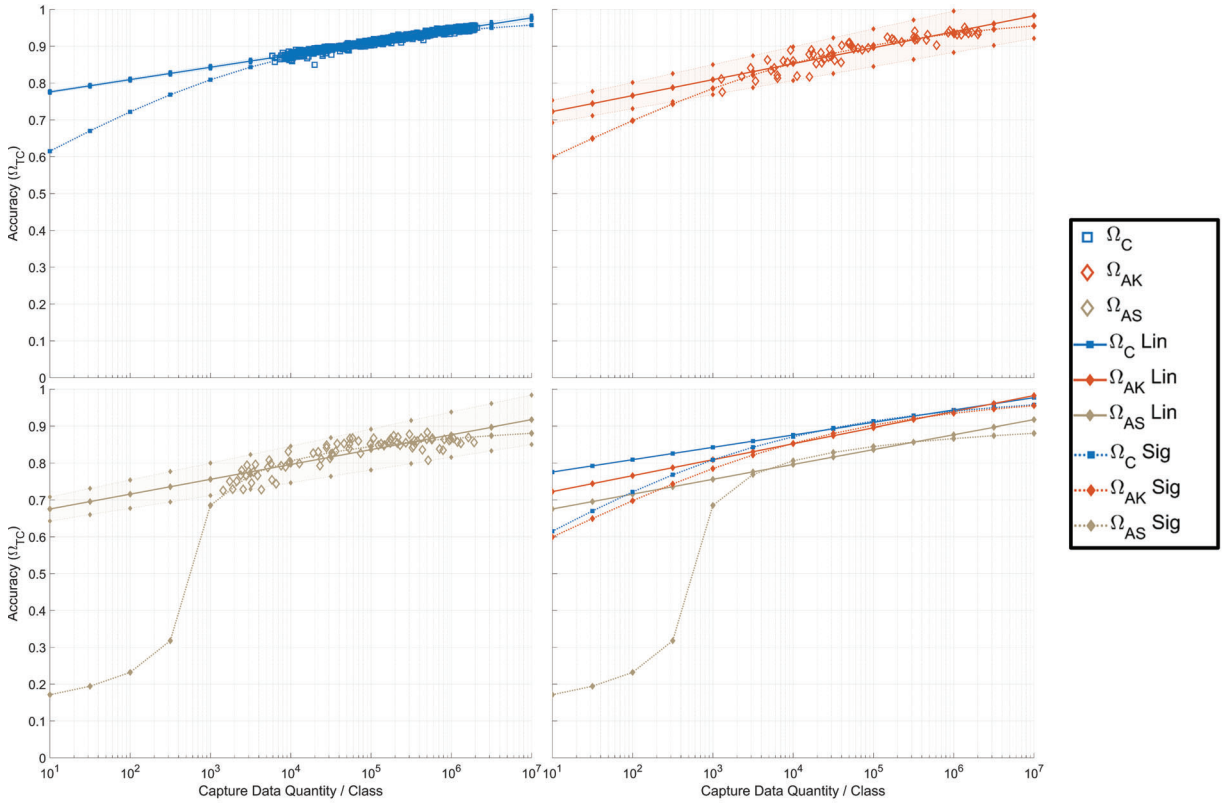

Figure 4.14: Performance of models trained using the three datasets with reliance on capture data on the Φ_3 waveform space. Solid lines represent the trend in terms of examples per class that are needed from Ω_C to achieve a given accuracy on the capture test data with or without any augmentation, while the shaded regions between dashed lines with the matching markers indicate a 95% confidence region for that trend. Trends are derived with outliers removed. The higher the trend line, the higher the quality of the dataset. The dotted lines represent a log-sigmoid regression to account for the asymptotic curvature observed in the results. Under both sets of regression, the models using dataset Ω_C exhibit a higher quality data with increasing data quantity; however, Ω_{AK} appears to catch up just beyond the observed range.

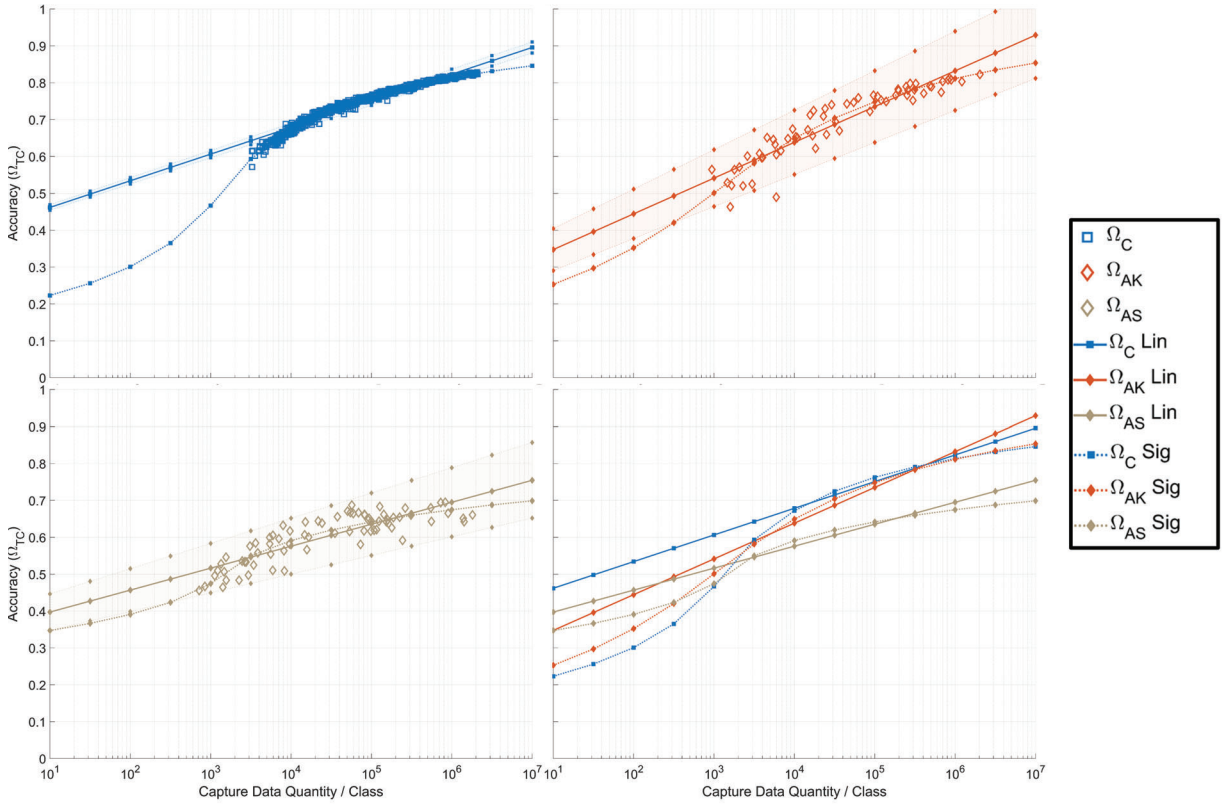

Figure 4.15: Performance of models trained using the three datasets with reliance on capture data on the Φ_5 waveform space. Solid lines represent the trend in terms of examples per class that are needed from Ω_C to achieve a given accuracy on the capture test data with or without any augmentation, while the shaded regions between dashed lines with the matching markers indicate a 95% confidence region for that trend. Trends are derived with outliers removed. The higher the trend line, the higher the quality of the dataset. The dotted lines represent a log-logistic regression to account for the asymptotic curvature observed in the results. Under both sets of regression, the models using dataset Ω_C exhibit a higher quality data with increasing data quantity; however, Ω_{AK} catches up just in the range of 10^6.

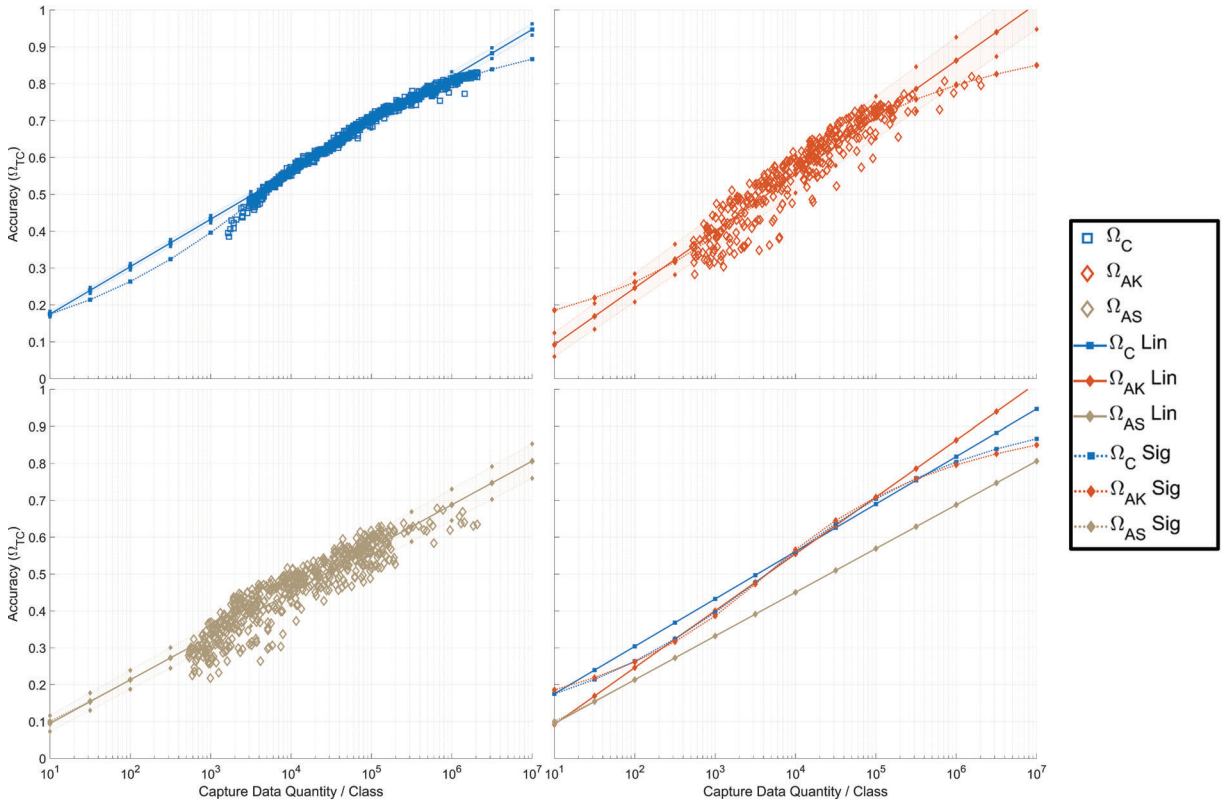

Figure 4.16: Performance of models trained using the three datasets with reliance on capture data on the Φ_{10} waveform space. Solid lines represent the trend in terms of examples per class that are needed from Ω_C to achieve a given accuracy on the capture test data with or without any augmentation, while the shaded regions between dashed lines with the matching markers indicate a 95% confidence region for that trend. Trends are derived with outliers removed. The higher the trend line, the higher the quality of the dataset. The dotted lines represent a log-logistic regression to account for the asymptotic curvature observed in the results. In contrast to Figures 4.14 and 4.15 the full observed trends from networks Ω_{AK} match the trends of when Ω_C is used, with individual models from Ω_{AK} exceeding the Ω_C model cluster below 2×10^5, indicating the benefit of augmentation as the complexity of the problem space increases.

4.2.1 Degradation Distribution Effect on Augmentation

The last insight offered by Figures 4.11-4.16 addresses the third question of whether knowing the distribution of the degradation is beneficial for augmentation, as it has already showed a significant, albeit marginal, effect on the synthetic datasets. Through contrasting the performance of Ω_{AK}, where augmentation draws from the nuisance parameter KDE, with that of Ω_{AS}, where augmentation is performed on an assumed subset of the application space, the quality of the augmented data can be examined.

For each of these figures, in the bottom right plot the log-linear and log-sigmoid trends for datasets Ω_C, Ω_{AK}, and Ω_{AS} are overlaid. As the difficulty of the problem space increases, with regard to the plots for both the total available data (Figures 4.11-4.13) and looking at the performance for a fixed quantity of captured data (Figures 4.14-4.16), the trends derived from models trained with data using Ω_{AK} consistently outperform the trends derived from models using Ω_{AS}. Examining the trends within Figure 4.16 for the most complex waveform space considered in this work, Φ_{10}, the trends for performance using the Ω_{AK} dataset are projected to be greater than the trends of the captured data for most of the considered quantity span. Comparatively, the projections made while using the Ω_{AS} dataset do not exceed those of the captured data. Across all waveform spaces when considering the amount of capture data needed as shown in Figures 4.14-4.16, using Ω_{AS} results in worse performance compared to using the capture data on its own as long as there is enough data available to result in a convergent network while using the capture data. When the only goal is peak performance, the expertise to evaluate the available data is lacking, and insufficient capture data is available, then the naive augmentation will help achieve a better result than the capture data alone.

One further consideration not addressed in this work is the difference in the transmitter and modulation specific correlations. To understand this point of contrast, all observations were between two distinct software radios capable of transmitting every waveform of interest, and therefore the data presented here does not take into consideration any correlations between the transmitting device and imperfections observed. One such unique consideration is that a Global System for Mobile Communications (GSM) waveform from a base station should have a distinct set of degradations that would help improve the identification of the Gaussian Minimum Shift Keying (GMSK) waveform over a hand held two-way radio's Frequency Shift Keying (FSK) for a given environment. Applying degradation from one type of transmitter (two-way hand held) to another modulation (GMSK) could end up hurting the performance of the overall system. Investigating the effect of COTS systems and their distinct characteristics in contrast to more versatile software-defined radios should be another avenue investigated for enhancing the performance of a system that must perform with a wider range of transceivers present.

These observations suggest that care should be taken when creating augmentation routines such that the distributions on the nuisance parameters are considered during the augmentation in order to achieve peak performance for a given number of capture observations.

Additionally, naive methods of augmentation utilizing GANs should be cautious of developing one network capable of augmenting *any* waveform, though developing a network *per* waveform as was done in Davaslioglu *et al.* still might be a viable alternative to the domain knowledge approach in this work [79].

4.3 Maximizing Performance through Data Selection

Up to this point, the datasets have been kept as individual buckets of observations where the only choice is to select and use one bucket. Specifically for the use of captured data and augmentations from said data, this might not be the most beneficial approach, as the original and augmentations are frequently used together. This section looks further into how data from the buckets can be combined, either directly by combining the different buckets, or through minimal alteration of the training routine by performing transfer learning from one or more buckets, to one or more buckets. For convenience, the consideration of understanding how the distributions affect the performance is set aside and only the datasets that make use of the KDE are employed, and focus is given to the most complex waveform group, Φ_{10}, rather than across all three groupings. Table 4.10 explores the shortened nomenclature for the training datasets and how that combines the different datasets into a single bucket, or across multiple training iterations. Figure 4.17 presents the performance of each dataset as a function of the total number of OPC to reorient the understanding of what the datasets produce on their own.

4.3.1 Dataset Mixing Results

The performance of the base datasets, $\{\Omega_C, \Omega_A, \Omega_S\}$, used in a single pass are shown in Figure 4.17 for a given quantity. The first and most important conclusion is that if the goal is to extract the best performance with the smallest dataset size, then there's no need to look any further than the capture dataset when only these three datasets are available; other datasets could exist from other capture events or different assumptions when creating augmented/simulated data. While the plot shows that augmented data does catch up under a log-linear assumption at larger quantities, the more likely outcome is that it will catch up only asymptotically. Another important point to remember is that augmented data is derived from a base set of captured data, and thus can not be fully separated from the Ω_C size. As shown in Figure 4.17, the synthetic only training routines fail to produce high performing results, so this particular style of result is omitted going forward. From here on, the question is shifting from 'what is the best performance that can be achieved per total quantity of data' to instead the problem of 'what is the best performance that can be achieved for a given amount of captured data.' This shift highlights the importance of collected data to achieving high performance within the RFML problem space, while also acknowledging that collected data will often be a limited quantity for training real systems.

Table 4.10: Experimental Dataset Combinations - Ω_A implies Ω_{AK} and Ω_S implies Ω_{SK} from Table 4.1 unless otherwise specified.

Experiment Symbol	Training Iterations	Dataset Pass 1	Dataset Pass 2	Dataset Pass 3
Ω_C	1	Ω_C		
Ω_A	1	Ω_A		
Ω_S	1	Ω_S		
$\Omega_{A\cup C}$	1	$\Omega_C \cup \Omega_A$		
$\Omega_{S\cup A}$	1	$\Omega_S \cup \Omega_A$		
$\Omega_{S\cup A\cup C}$	1	$\Omega_C \cup \Omega_A \cup \Omega_S$		
$\Omega_{S\to C}$	2	Ω_S	Ω_C	
$\Omega_{S\to A}$	2	Ω_S	Ω_A	
$\Omega_{S\to A\cup C}$	2	Ω_S	$\Omega_C \cup \Omega_A$	
$\Omega_{S\to A\to C}$	3	Ω_S	Ω_A	Ω_C

The reframing of the performance of the base datasets in terms of the capture data quantity that was used either to train directly on, or that provided the base for augmentation, is shown in Figure 4.18 along with the other dataset combinations. The simulated dataset is omitted due to having zero captured data in use during training and having an observed stagnant performance around 20%. Within the figure, there are experimental results from networks trained purely on augmented data that exceed captured data performance. This shows the fundamental strength of augmentation that allows the training routine to see much more data that contains information relevant to the problem than just looking at the captured data alone.

Since the target search is to find a combination of datasets and staged training to outperform that of captured data alone, a performance normalization is applied to the remaining plots in this section. The remaining performance is observed in terms of accuracy deviation or

$$\Delta_\alpha = \alpha_X - f_{\text{sigmoid}|C}(\text{capture\_qty}_{\alpha_X}),$$

where the accuracy of a trained network α_X first has the expected sigmoid performance of captured data for that amount of captured data used, $f_{\text{sigmoid}|C}(\cdot)$, subtracted from it, thus showing the performance difference, Δ_α, that is achieved with the training method and dataset combination.

Shown in Figure 4.19, further examination of the different combinations of the datasets, defined in Table 4.10, have the individual results plotted on the left, while showing the logistic regressions on the right. There are several key points that can be taken away from these

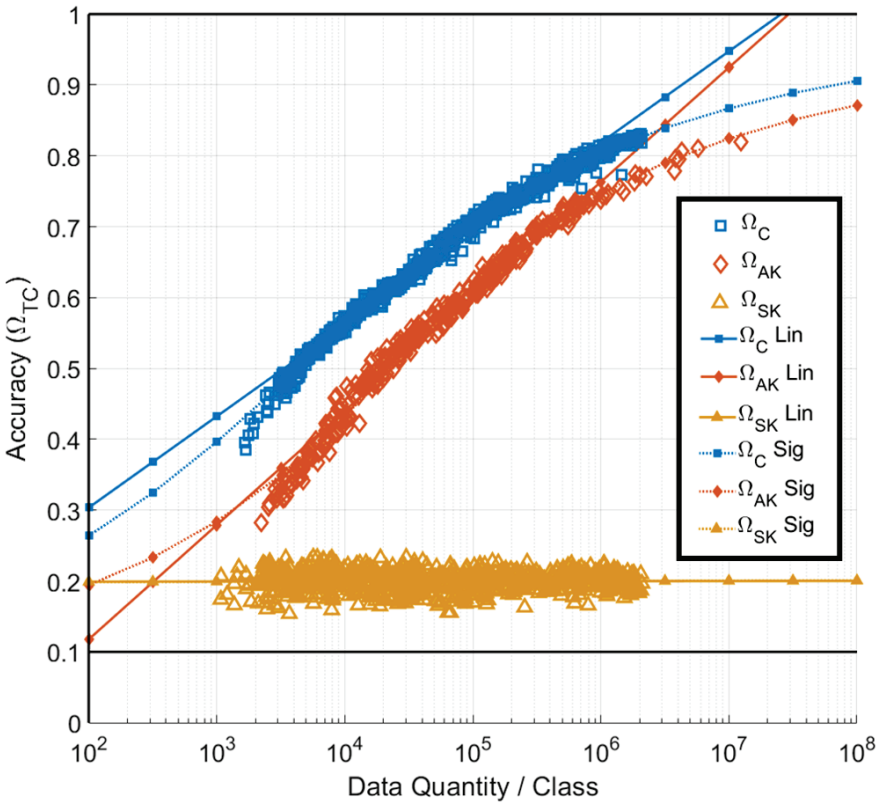

Figure 4.17: Plotting the performance of a trained network on the Φ_{10} waveform set on Ω_{TC}, constrained on the total number of observations used while training the network. Horizontal thick black line represents random guessing by the network. Solid lines represent a linear regression, while dotted lines represent a sigmoid regression of the data. Captured data provides the greatest return in performance per observation used during training.

results. First, the use of transfer learning between datasets has definite benefits. By priming the network with the synthetic set, Ω_S, the random seed of the network can be overcome and a network that understands the problem space, albeit in a unrefined manner, can then be given the datasets that contain more true to form data. This allows the $\Omega_{S\to C}$ experiment to converge with capture data at a lower available capture data quantity (514 OPC in this context) than seen with capture data alone (1645 OPC for a 3.2× less available data quantity). Additionally, the networks trained in experiment $\Omega_{S\to A\cup C}$ consistently outperform capture data alone and perform better than experiment $\Omega_{A\cup C}$ when the amount of capture data is low, indicating that a synthetic priming of the network allows for better results than training from scratch. The last point of benefit is that the final transfer learning experiment $\Omega_{S\to A\to C}$ results in consistent increased performance on Ω_{TC} across the full observed span. This approach seems to suggest that the networks are capable of being trained much like the human brain, in stages where the problem space becomes increasingly more complex, similar to the way math is taught, arithmetic → algebra → calculus, for example. However, the

network's generalization performance in Figure 4.20 across both Ω_{TC} and Ω_{TS} shows that the way the data is presented to the network has a significant effect on the ability of the network to generalize over all observations.

While using the synthetic dataset on its own provides little value to the test captured dataset in terms of observable performance, value in its use along with higher quality datasets can be observed. The second main take away is then that using all three kinds of data can have a net benefit to the final performance of the system over using the high quality dataset alone. By examining the three different experiments that make use of all base datasets, $\{\Omega_{S\cup A\cup C}, \Omega_{S\rightarrow A\cup C}, \Omega_{S\rightarrow A\rightarrow C}\}$ as shown in Figure 4.21 more understanding of how to use the dataset types can be found. All three of these approaches typically outperform the use of Ω_C alone, with an exception for $\Omega_{S\cup A\cup C}$ as the available capture quantity decreases. Looking at the generalization space in Figure 4.22 shows that even though the performance of these networks is low on the Ω_{TC} test set, strong performance is still observed on the Ω_{TS} test set. The explanation for the poor performance of these networks is not immediately clear, so the concentration of the different datasets within the full training data space is shown in Figure 4.23, which breaks down the deviating performance of each network as a function of the percentage of total data coming from Ω_C (Top Left), Ω_A (Top Right), and Ω_S (Bottom Left). Common to all of the worst performing networks is the fact that as the concentration of the dataset becomes heavily drawn from Ω_S, the performance decreases comparatively. The experiment $\Omega_{S\rightarrow A\rightarrow C}$ shows resiliency to this, which is attributed to having multiple transfer learning stages further minimizing the worse effects of the Ω_S dataset on the final performance of the network.

Expanding these observations back to all datasets observed with capture data in play in Figure 4.24 shows that this concentration of synthetic data has the most significant effect when it is in use during the last training stage. Overall, the best networks were seen when the captured data was present with augmented data and used in the range of 5-15% and 80-90% respectively. This shows that augmentation factor, the ratio of augmented data to capture data, in the range of 6-10 has beneficial returns over capture data alone when augmentation is accounting for receiver detection errors, but there are no conclusive results on what the best value for augmentation factor is, or whether an upper bound for performance was found. This leads to an understanding that unknown and low quality data can be used to prime a network, but it shouldn't be used in the final stages of a training routine as a dominant source of the training data. At the same time, completely disregarding the data used to prime the network might lead to undesirable losses in generalization, so there needs to be some combination of data used for priming, in order to produce a quickly convergent network, followed by a significant reduction and rebalancing of data from the unknown/lower quality dataset with those of the higher quality dataset in order to preserve more generalization.

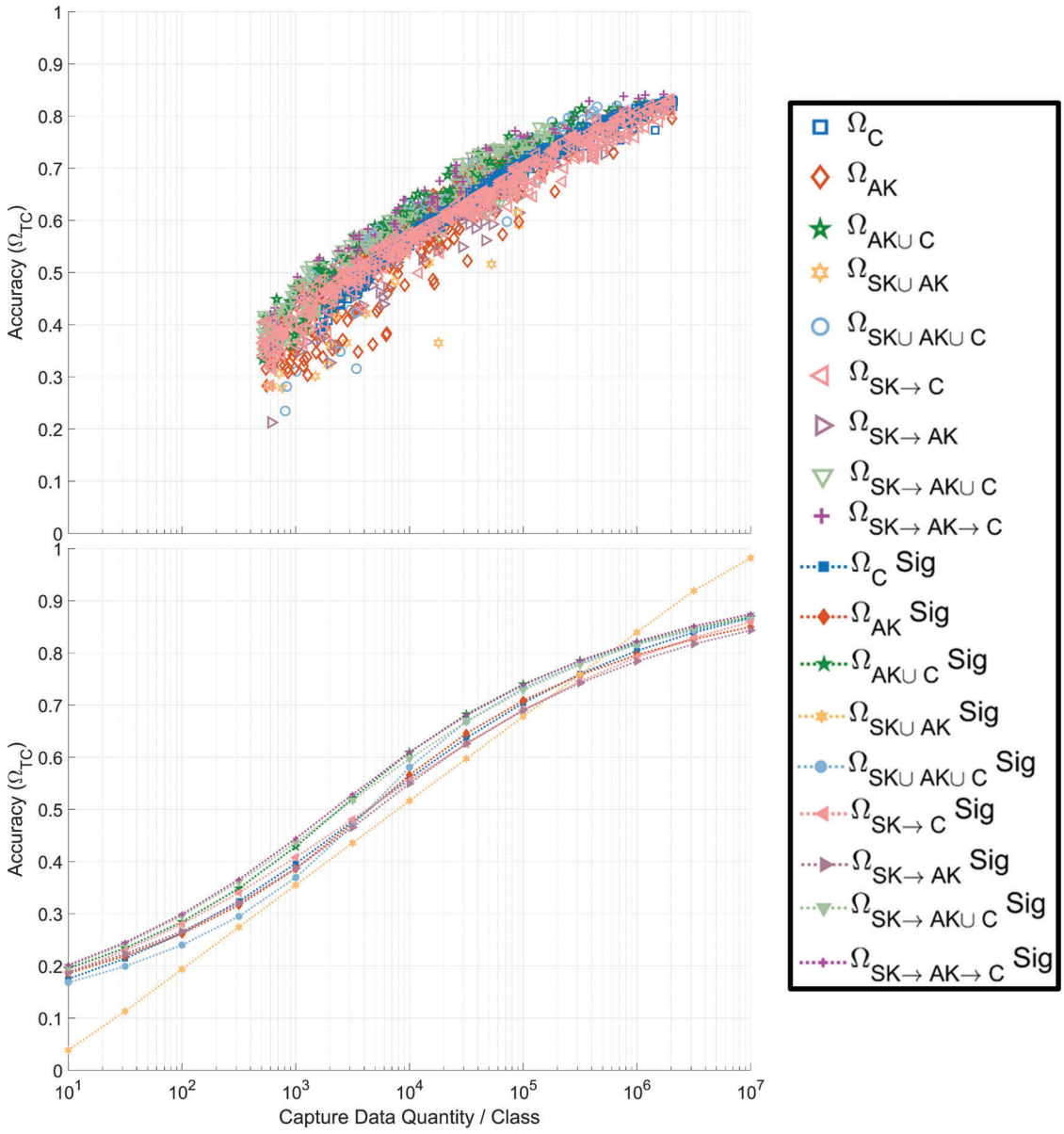

Figure 4.18: Plotting the performance of each trained network on the Φ_{10} waveform set on Ω_{TC}, constrained on the number of capture observations used while training the network (top). Dotted lines represent a sigmoid regression of the data in that experiment category (bottom).

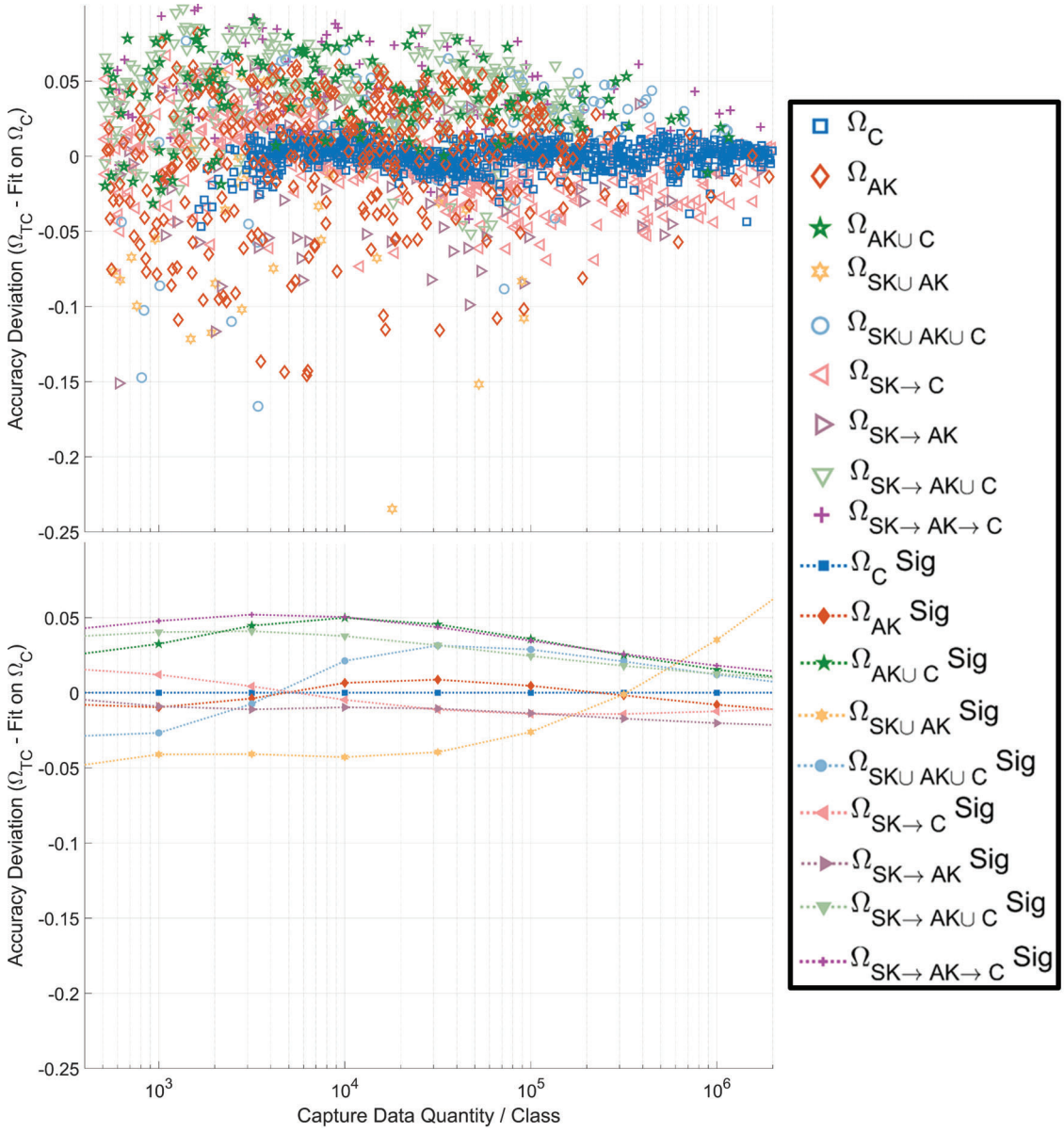

Figure 4.19: Plotting the performance deviation of each trained network on the Φ_{10} waveform set when evaluated on the Ω_{TC} test set. Each result is compared on the number of capture observations used while training the network subtracted by the sigmoid regression of performance when using only that much captured data (top). Dotted lines represent a sigmoid regression of the performance by experiment category subtracted by the sigmoid regression of Ω_C experiment category (bottom). Results show that even when the available data comes from a dataset of high quality, there are mixing techniques with lower quality datasets that can result in a better performance.

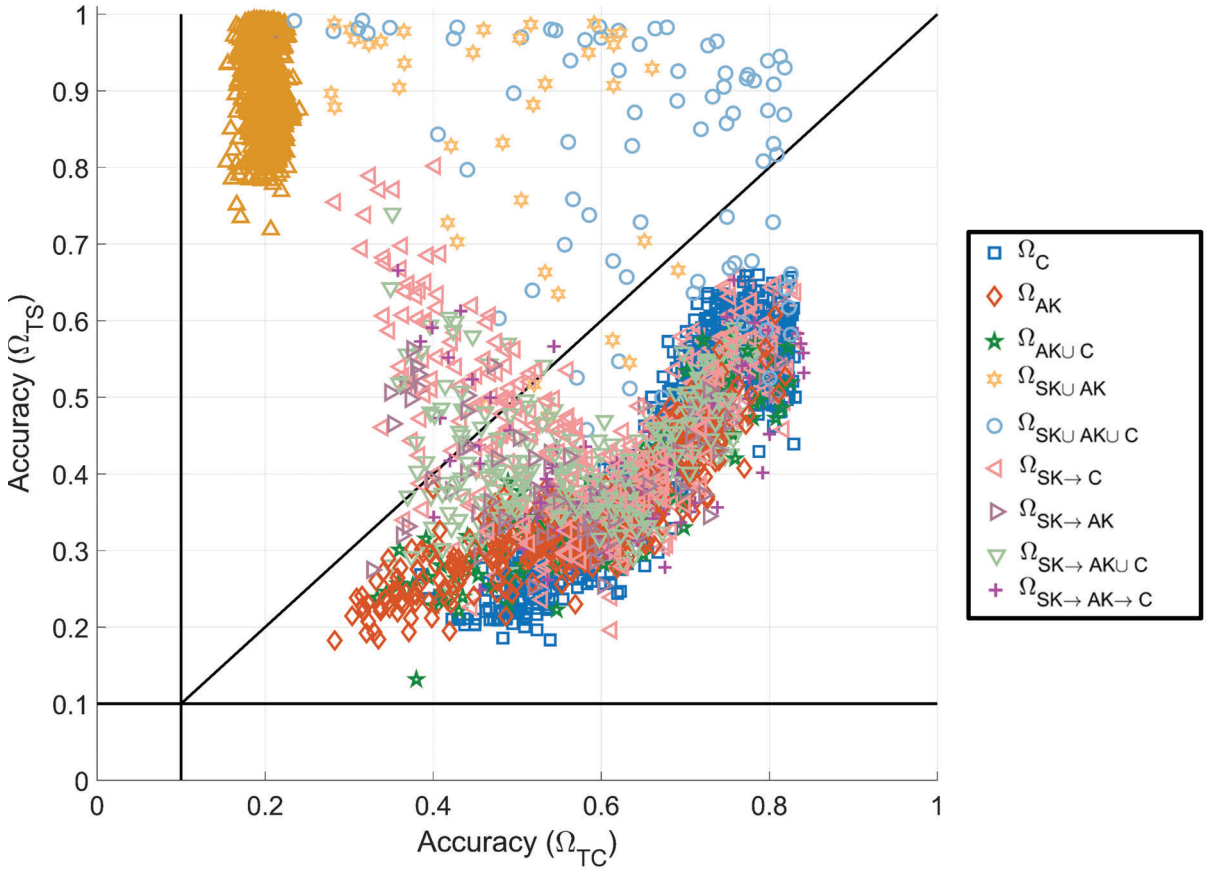

Figure 4.20: Examining performance contrasts between Ω_{TC} and Ω_{TS} for considered approaches on the Φ_{10} waveform set.

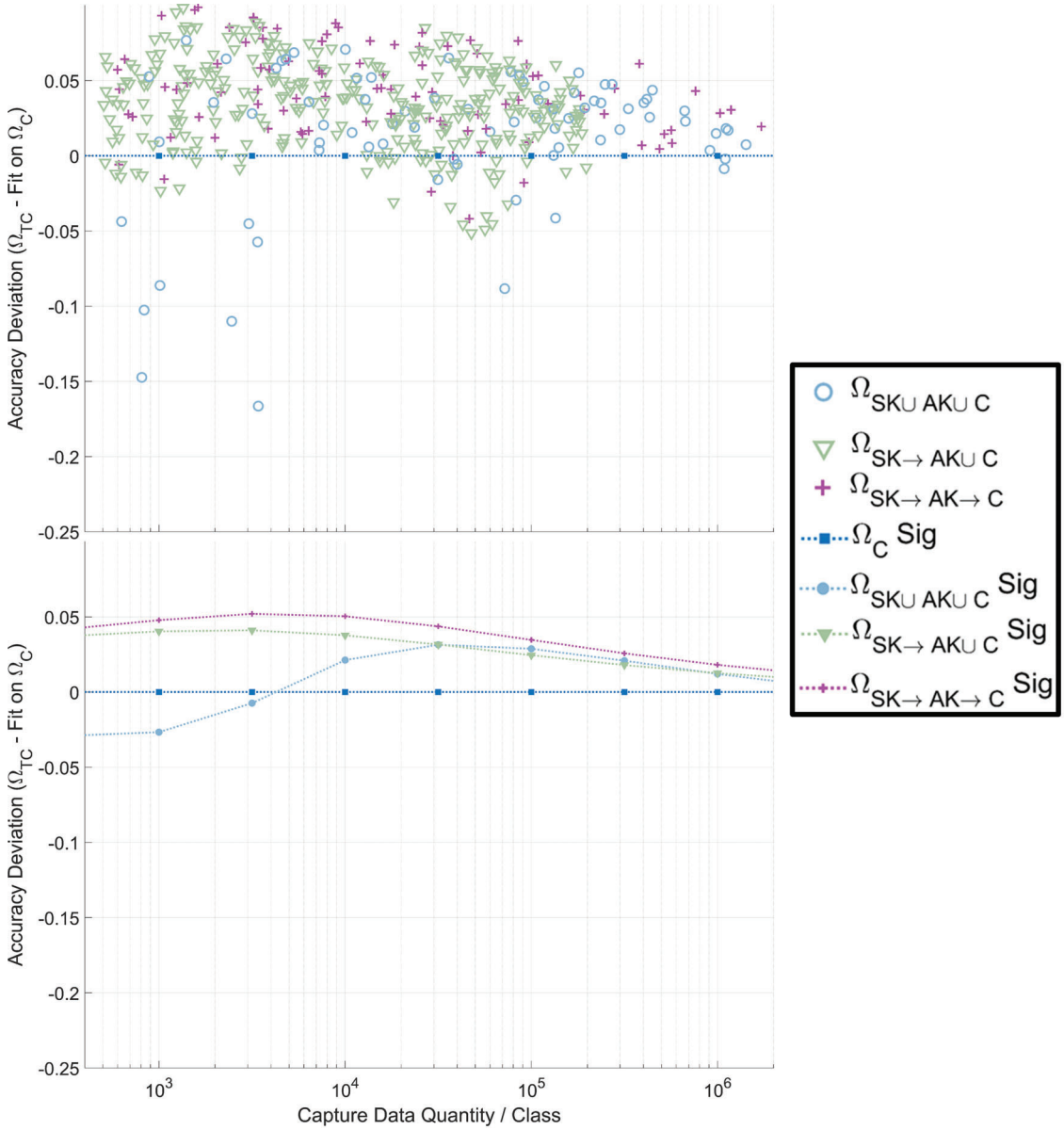

Figure 4.21: Plotting the performance deviation of trained networks on the Φ_{10} waveform set when evaluated on the Ω_{TC} test set that make use of all three base datasets. Each result is compared on the number of capture observations used while training the network subtracted by the sigmoid regression of performance when using only that much captured data (top). Dotted lines represent a sigmoid regression of the performance by experiment category subtracted by the sigmoid regression of Ω_C experiment category (bottom). Results show that even when the available data comes from a dataset of high quality, there are mixing techniques with lower quality datasets that can result in a better performance.

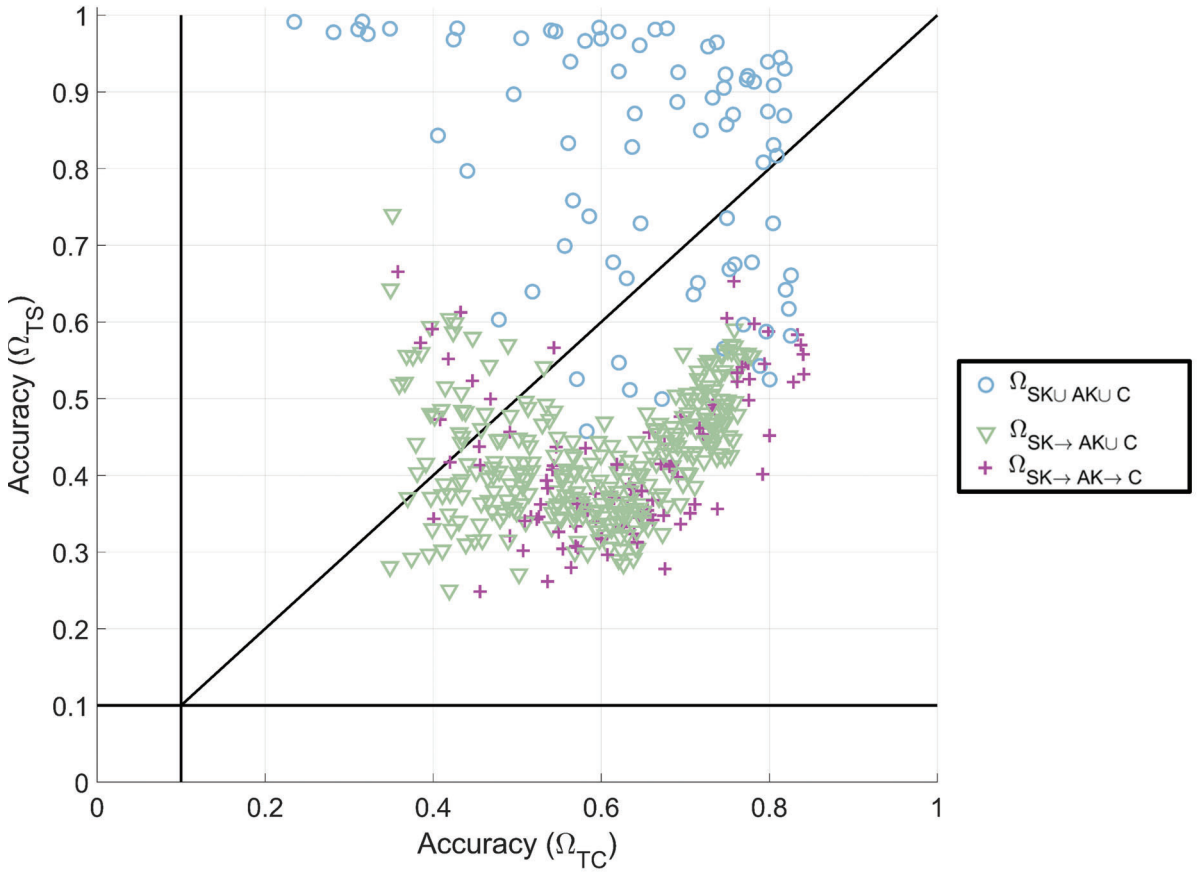

Figure 4.22: Examining performance contrasts between Ω_{TC} and Ω_{TS} for considered approaches on the Φ_{10} waveform set that make use of all three training datasets. Inclusion of the Ω_{SK} dataset in the last stage of training helps with better generalization, but can result in over saturation toward the synthetic dataset if no caution is taken.

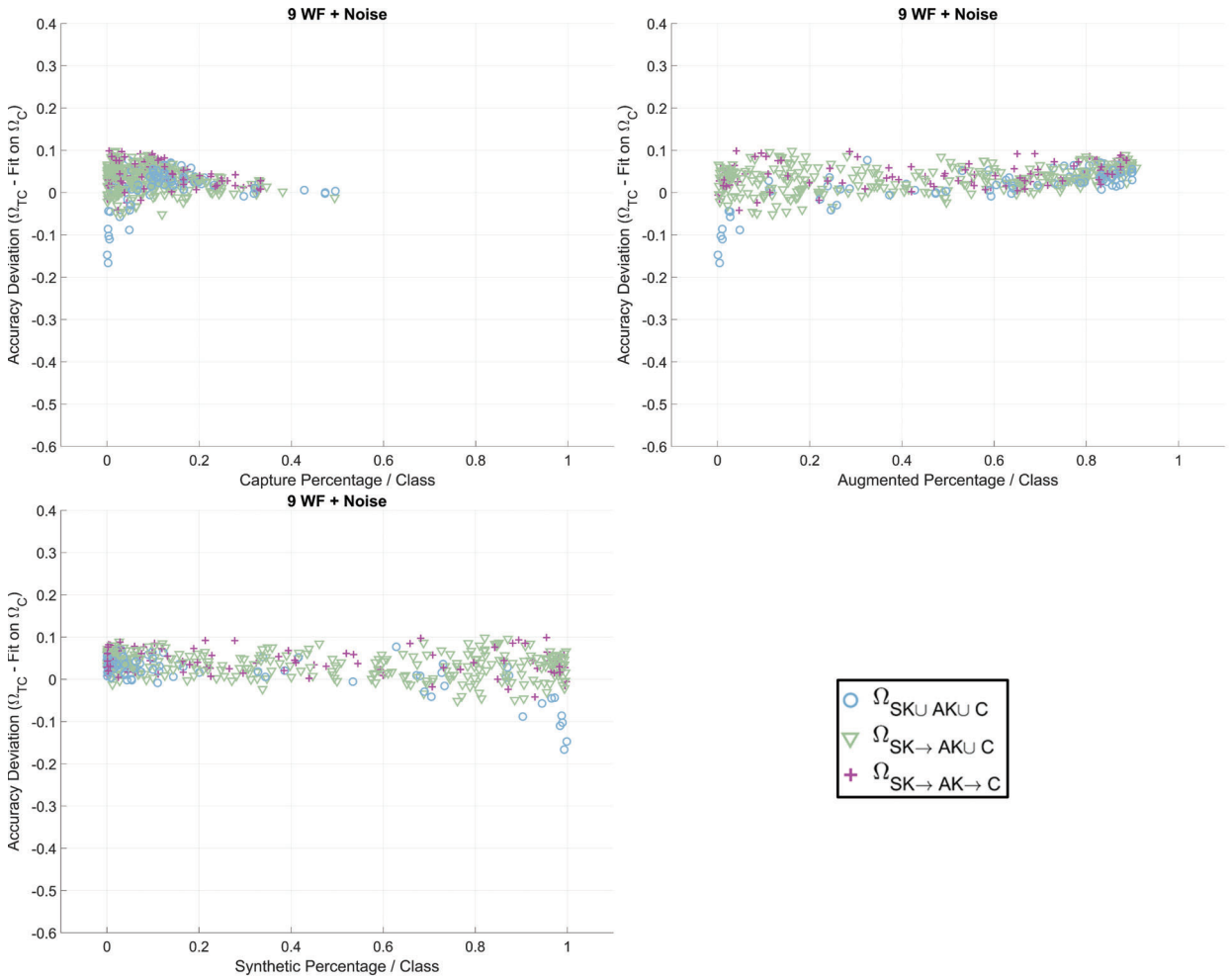

Figure 4.23: Plotting the performance deviation of a trained network on the Φ_{10} waveform set on Ω_{TC}, that is, performance adjusted for the expected performance (logistic fit) of captured data alone, compared to percentage makeup of data used while training the network. (Top Left) Shows the total percentage of capture data. (Top Right) Shows the total percentage of simulated data. (Bottom Left) Shows the total percentage of augmented data.

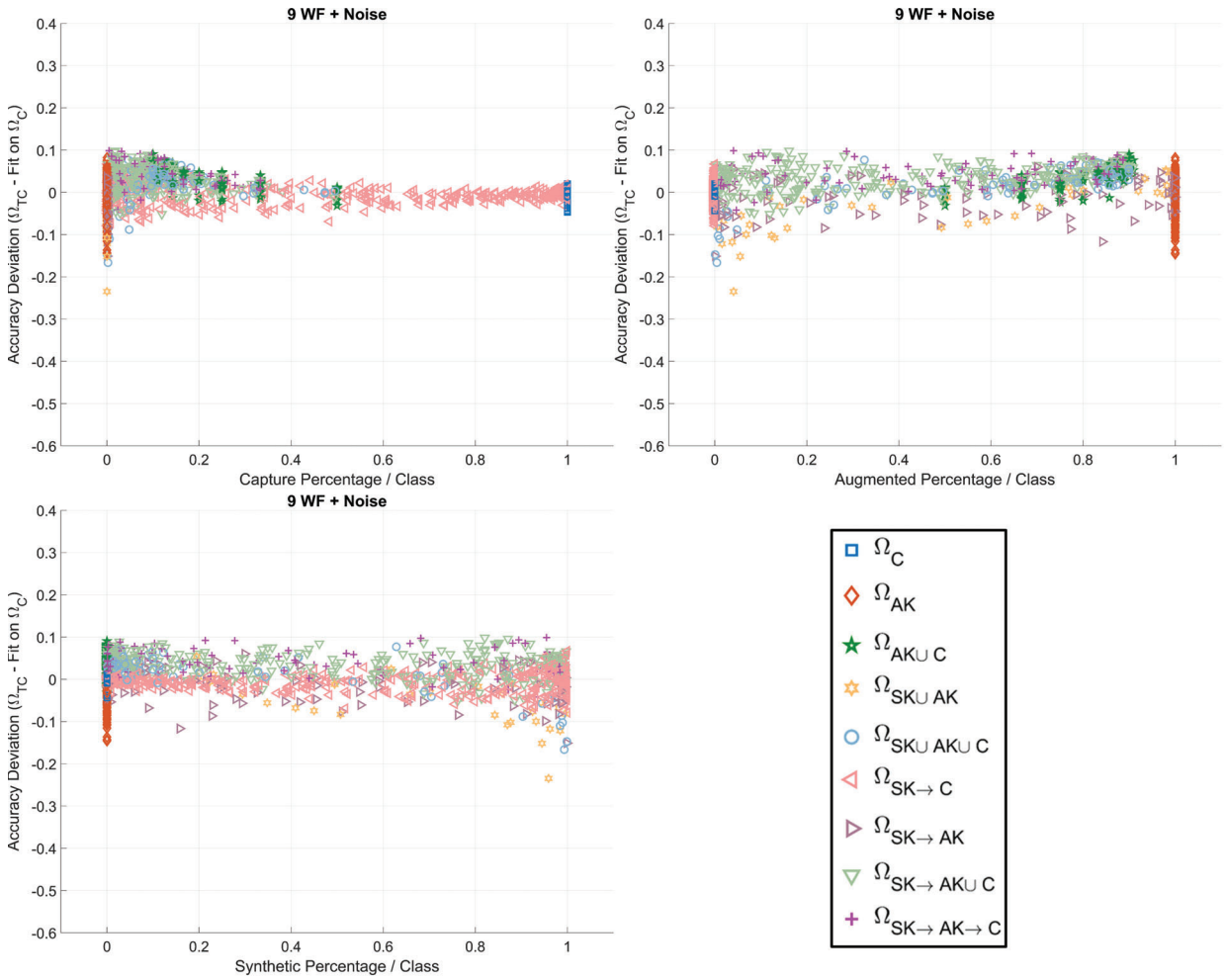

Figure 4.24: Plotting the performance deviation of a trained network on the Φ_{10} waveform set on Ω_{TC}, that is, performance adjusted for the expected performance (logistic fit) of captured data alone, compared to percentage makeup of data used while training the network. (Top Left) Shows the total percentage of capture data. (Top Right) Shows the total percentage of simulated data. (Bottom Left) Shows the total percentage of augmented data.

4.4 Data Quantity Forecasting

In Section 4.1, the datasets and approach were defined and established a relationship between the quantity of data and the performance of a network. The comparative quality of the augmented datasets are examined within Section 4.2, and the effect of combining different datasets and understanding larger variance in performance are discussed in Section 4.3. Now, focus turns to quantifying the concept of quality, and using the relationship between data and performance to predict what is needed in terms of data quantity to achieve high performance models under a given approach.

4.4.1 Data Quantity and Quality

The work performed in Clark *et al.* [21], and restated here in Sections 4-4.2, showed that, within the realm of AMC, the quantity of data has a functional relationship to the performance of a trained system given all other variables are constant. Additionally, the work showed that the performance could be found as log-linear relation to the quantity of data for lower performance regions, but a log-sigmoid relationship is more appropriate as performance reaches a maximum. In [2] this conceptual relationship between dataset quality and the regression between data quantity and performance is examined more closely. The process of regressing the relationship between quantity and performance was then suggested as a quantification measure of dataset quality in [2], where different datasets could then be compared across different quantities with the expected accuracy (e.g. dataset A needs X observations, while dataset B needs $2X$ observations to achieve an accuracy of 90%), or other metrics of performance, taken as the quality ($X|90\%$ or $2X|90\%$ in the previous example) of the dataset. The inherent quality of any dataset can be described in three generalized terms of *Comprehensiveness*, *Correctness*, and *Variety* [45]. In this work the datasets are already examined and confirmed to be *Comprehensive* in that all the information being sought is included within the dataset, and *Correct* in that the observations for each modulation are correctly identified and labeled. The main concept of quality being examined is then that of *Variety* or rather that the distributions on the observations within the datasets match, approximate, or deviate from the distributions of the test set, and therefore only the effect of quality in terms of *Variety* can be examined in this work.

While these works gave an initial understanding of the data quantity and quality that fundamentally drive the process of an ML system, they provide minimal utility when trying to understand how much data is needed in order to achieve ideal performance and therefore reliably plan a data collection campaign. For example, in Figure 4.25 of [2] looking at the 10-class classification performance, the log-linear fit predicts a performance of 90% accuracy at roughly an order of magnitude less data than the corresponding log-sigmoid fit, while both fits use the full range of trials available to regress the fit. The results discussed above all depend on some initial *good dataset* to contrast with, and while this work does not alleviate

that requirement, here the question is answered of how to best use a limited *good dataset* to forecast how much total data would be needed during training if neither the model, nor training approach, is modified.

An ideal approach would be to use a metric that is strongly correlated with the desired performance of the system, such as accuracy, in terms of Kendall's τ and has a relationship with data quantity that can be linearly derived from minimal data; however, a metric that reduces the error with a linear estimator over that of performance directly regressed with quantity will be sufficient. For this reason, the metrics that have been developed to predict the transferability of a pre-trained model onto a new *target* dataset, discussed in Section 4.4.1.1, are repurposed to predict data quantity requirements and provide a new metric of quality for a model's training dataset with regard to the *target* dataset, which is the evaluation dataset in this work Ω_{TC}.

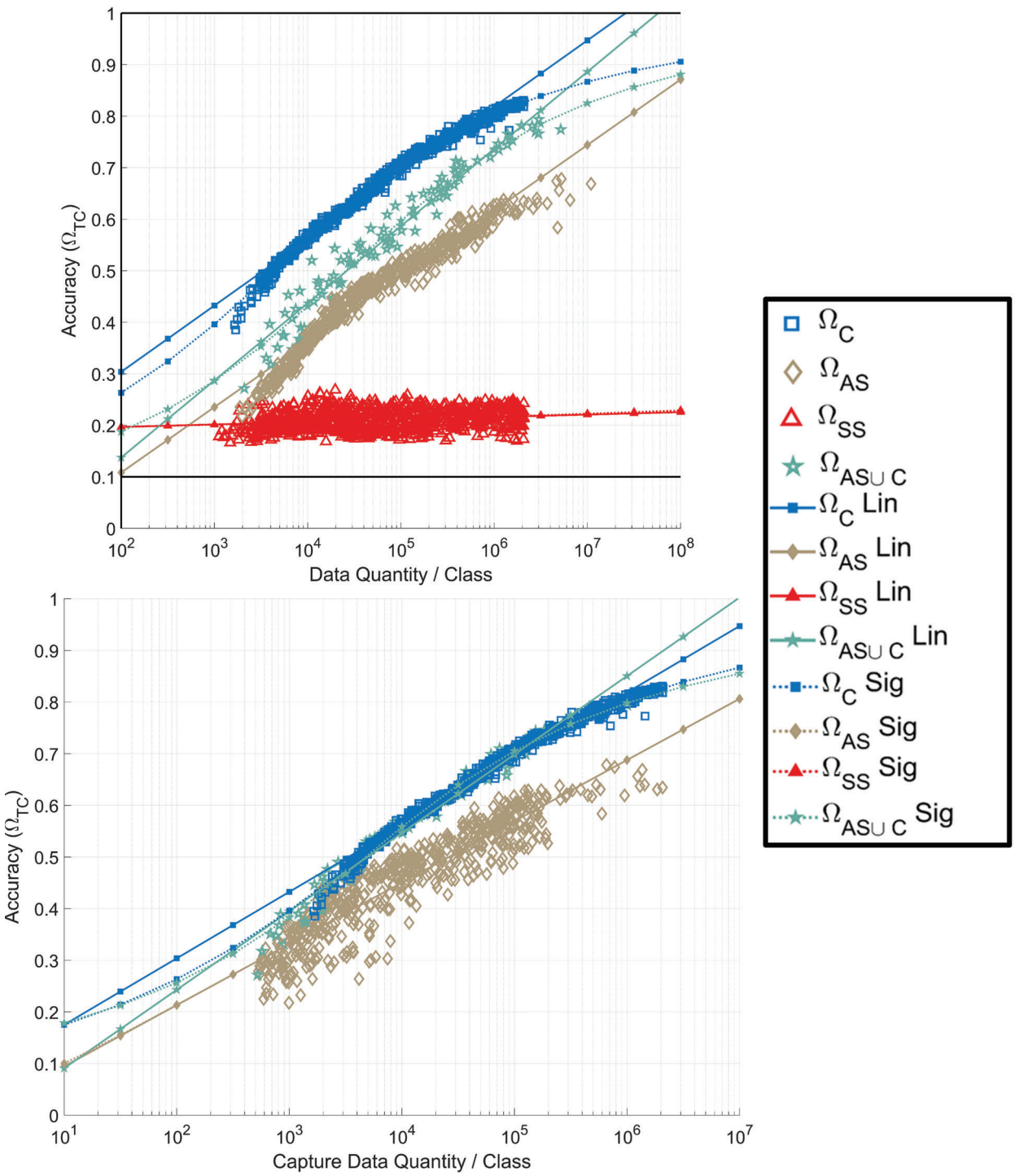

Figure 4.25: Plotting the performance of a trained network on the Φ_{10} waveform set on Ω_{TC} compared to the total number of observations (left) and the number of observations of capture data (right) used while training the network. (All) Solid lines represent a linear regression, while dotted lines represent a sigmoid regression of the data. [2]

4.4.1.1 Metrics for Transferability

TL is the practice of training a model on one dataset/ domain (*source*), or otherwise taking a pre-trained model, and training with a new dataset/ domain (*target*) instead of starting from a random initialization [17]. Depending on assumptions between the *source* and *target*, the TL application can be categorized as *homogeneous*, where differences exist in the distributions between *source* and *target*, or *heterogeneous*, where the differences are in the feature space of the problem [17, 40]. A valuable discussion for understanding the concepts of *homogeneous* and *heterogeneous* is provided by Wong and Michaels [41] by explaining the change in distributions as a change of the dataset's collected/generated domain, while the feature space of the problem can be associated with the intended task the *source* model is trained on and can be contrasted to the task of the *target* problem. The two most common types of TL include retraining the classification head, where early layers are frozen during training preserving feature extraction, and fine-tuning of the whole model [212]. In this work, TL is applied in a *homogeneous* problem space where the underlying distributions on the data vary, but the generalized problem space is the same between datasets, otherwise called a *Domain Adaptation* from [41] and more specifically an *Environment Platform Co-Adaptation*. Additionally, wherever the retraining is done in this work, the fine-tuning approach is utilized, allowing for adjustments to the feature space, which might not be observable in the *source* dataset. An important note here: this work does not evaluate any aspect of TL on the problem space, rather it makes use of metrics developed for the purpose of TL. Understanding how TL is best used within RFML is beyond the scope of this work.

The study of TL is complex and well explored [17, 40, 41, 212, 213, 214], and from the effort to understand how to choose an optimal pre-trained model for a desired application, the metrics Negative Conditional Entropy (NCE) [213], Log Expected Empirical Prediction (LEEP) [212], and Logarithm of Maximum Evidence (LogME) [214] are repurposed to analyze the relationship between available data quantity during training and system performance for a given evaluation set.

In this paper, the evaluation set has the same labels as the training set, but the distributions are not assumed to be equivalent. Therefore the evaluation set, $\{X\}_1^N \sim \mathcal{X}$, is drawn from an observation space, $\mathcal{V}$, which is inherent to the generalized problem space, $\mathcal{V} \subset \mathcal{S}$. For clarity going forward, due to the shared labels between *source* and *target* in this work, the *source* labels are found through a forward pass of the evaluation set through the network, therefore the i^{th} observation's *source* label is given by $\breve{l}_{y|xi} = \phi(x_i) \in \mathbb{R}^C$, with the inference given as $\breve{c}_{y|xi} = \text{argmax}(\breve{l}_{y|xi}) \in \mathbb{Z}^1$. By contrast, the *target* label directly gives $c_{x,i} \in \mathbb{Z}^1$ by the truth of the i^{th} observation and can be one-hot encoded to provide $l_{xi} = \text{OH}(c_{xi}, C) \in \mathbb{R}^C$. Given

the above notation, NCE is given as

$$\begin{aligned}\mathrm{NCE}(\breve{c}_{y|x}, c_x) = \sum_{j=1}^{C} \hat{P}(\breve{c}_{y|x} = j) \\ \cdot \sum_{k=1}^{C} \hat{P}(c_x = k|\breve{c}_{y|x} = j) \log(\hat{P}(c_x = k|\breve{c}_{y|x} = j)),\end{aligned} \tag{4.3}$$

where $\hat{P}(\cdot)$ are the empirical distributions found as

$$\hat{P}(\breve{c}_{y|x} = j) = \frac{1}{N} \sum_{i=1}^{N} \breve{c}_{y|xi} == j, \tag{4.4}$$

$$\hat{P}(c_x = k|\breve{c}_{y|x} = j) = \frac{1}{N} \sum_{i=1}^{N} (\breve{c}_{y|xi} == j) \cdot (c_{xi} == k). \tag{4.5}$$

The *source* labels are iterated over with j, while k iterates over the *target* labels. LEEP is given as

$$\mathrm{LEEP}(\breve{l}_{y|x}, c_x) = \frac{1}{N} \sum_{i=1}^{N} \log\left(\breve{l}_{y|xi} \cdot \hat{P}(c_x = k|\breve{l}_{y|x})\right), \tag{4.6}$$

where the empirical conditional probability, $\hat{P}(c_x = k|\breve{l}_{y|x})$ is given as

$$\begin{aligned}\hat{P}(c_x = k|\breve{l}_{y|x}) = \left[\hat{P}(k|j = 1), \ldots, \hat{P}(k|j = C)\right]^T \\ \hat{P}(k|j) = \hat{P}(k, j) / \sum_{k'=1}^{C} \hat{P}(k', j) \\ \hat{P}(k, j) = \frac{1}{N} \sum_{i=1}^{N} \breve{l}_{y|xi}[j] \cdot (c_{xi} == k).\end{aligned} \tag{4.7}$$

The LEEP score, for the combination of the model and evaluation set, is given as the average log of all probabilities of getting the correct label in the evaluation set given the empirical probability of the labels provided by the model under test [212]. Whereas LogME is given as

$$\mathrm{LogME}(\breve{l}_{y|x}, c_x) = \frac{1}{NC} \sum_{k=1}^{C} \log(p(c_x = k|\breve{l}_{y|x}, \alpha, \beta)), \tag{4.8}$$

where α and β are iteratively solved to maximize the evidence, $p(c_x = k|\breve{l}_{y|x})$, for a linear transform applied to $\breve{l}_{y|x}$, which is then averaged over the number of classes, C, and normalized by the number of observations, N, in the evaluation set [214].

In the most general sense, the importance of these metrics is how well correlated, either positively or negatively, the metric is with the desired performance of the network after

Table 4.11: Kendall's τ weighted correlation across datasets (Ω) and waveform sets for Accuracy and (NCE, LEEP, LogME). Strong correlations will have an absolute value near 1, while no discernable correlation will be around 0. **Bold** values represent the combination of problem set and metric with the highest correlation with accuracy on the evaluation set.

Set	Ω	NCE	LEEP	LogME
Φ_3	Ω_C	**0.9774**	0.9533	0.8033
	Ω_S	**0.8249**	0.8144	0.7382
	Ω_A	**0.9666**	0.9639	0.9377
Φ_5	Ω_C	0.9438	**0.9443**	0.8788
	Ω_S	**0.6554**	0.6553	0.6334
	Ω_A	**0.9794**	0.9791	0.9582
Φ_{10}	Ω_C	**0.9794**	0.9688	0.9609
	Ω_S	0.5165	0.4262	**0.5298**
	Ω_A	**0.9836**	0.9808	0.9764

being retrained on the *target* dataset. Within this work the explanation provided by You *et al.* [214] for using Kendall's τ coefficient [215] is utilized as the most significant relationship between performance and the metric of choice is a shared general monotonicity that allows for a trend in the metric to indicate a trend in performance as well.

4.4.1.2 Examining the Correlation between Performance and Metrics

The first step is confirmation that the chosen metrics correlate in a beneficial manner with the performance value of interest, classification accuracy in this case. In order to understand whether a metric is well correlated with classification accuracy, the weighted Kendall's τ is calculated using the SciPy implementation [208] and found for three datasets (Table 4.1: $\Omega_C, \Omega_A, \Omega_S$) and compared against three sets of modulation classification sets (Table 4.2: $\Phi_3, \Phi_5, \Phi_{10}$). The Kendall's τ weighted correlations are presented in Table 4.11 and show high values of correlation for all three metrics in the case of Ω_C and Ω_A datasets; however, the Ω_S dataset shows a worse correlation between accuracy and all three metrics. Looking at the relationships between performance and the proposed metrics in Figure 4.26 shows that the performance and metrics are tightly clustered, while for Ω_C and Ω_A, definite trends are observable. Looking at the performance of the different datasets as a function of quantity used during training in Figure 4.27 helps to further explain this decrease in correlation in that the performance results of networks trained on Ω_S are comparably independent from the quantity of data used for the synthetic observations. Therefore, the classification accuracy and metrics extracted from the networks trained on Ω_S are more akin to noisy point measurements than a discernable trend to examine.

The main observation is that when there is a discernable trend between performance and

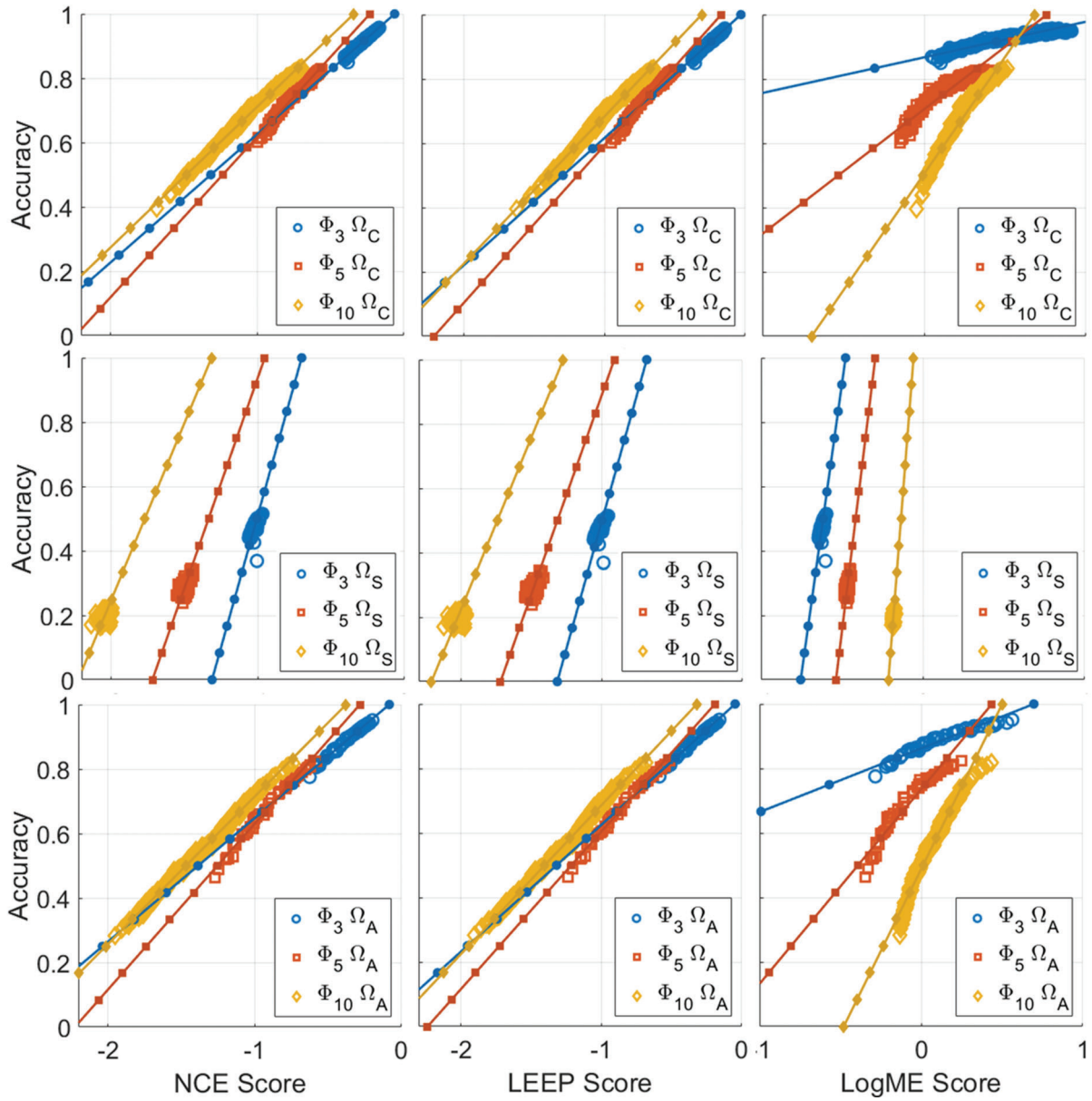

Figure 4.26: Visualization of the relationships between the three metrics (Left column: NCE, Middle column: LEEP, Right column: LogME) and the performance metric (Accuracy) of each network when measured on the results of the evaluation set Ω_{TC}, or the *target* dataset in TL vernacular. Each dataset used for training are positioned along the rows (Top row: Ω_C, Middle row: Ω_S, Bottom row: Ω_A). Linear trends shown between the metrics and accuracy for better clarity in the relationships.

data quantity, the correlation of all three metrics are considerably high, and therefore are potential metrics with which to regress the relationship with data quantity in search of a quantity estimator for the total data needed to achieve a desired performance.

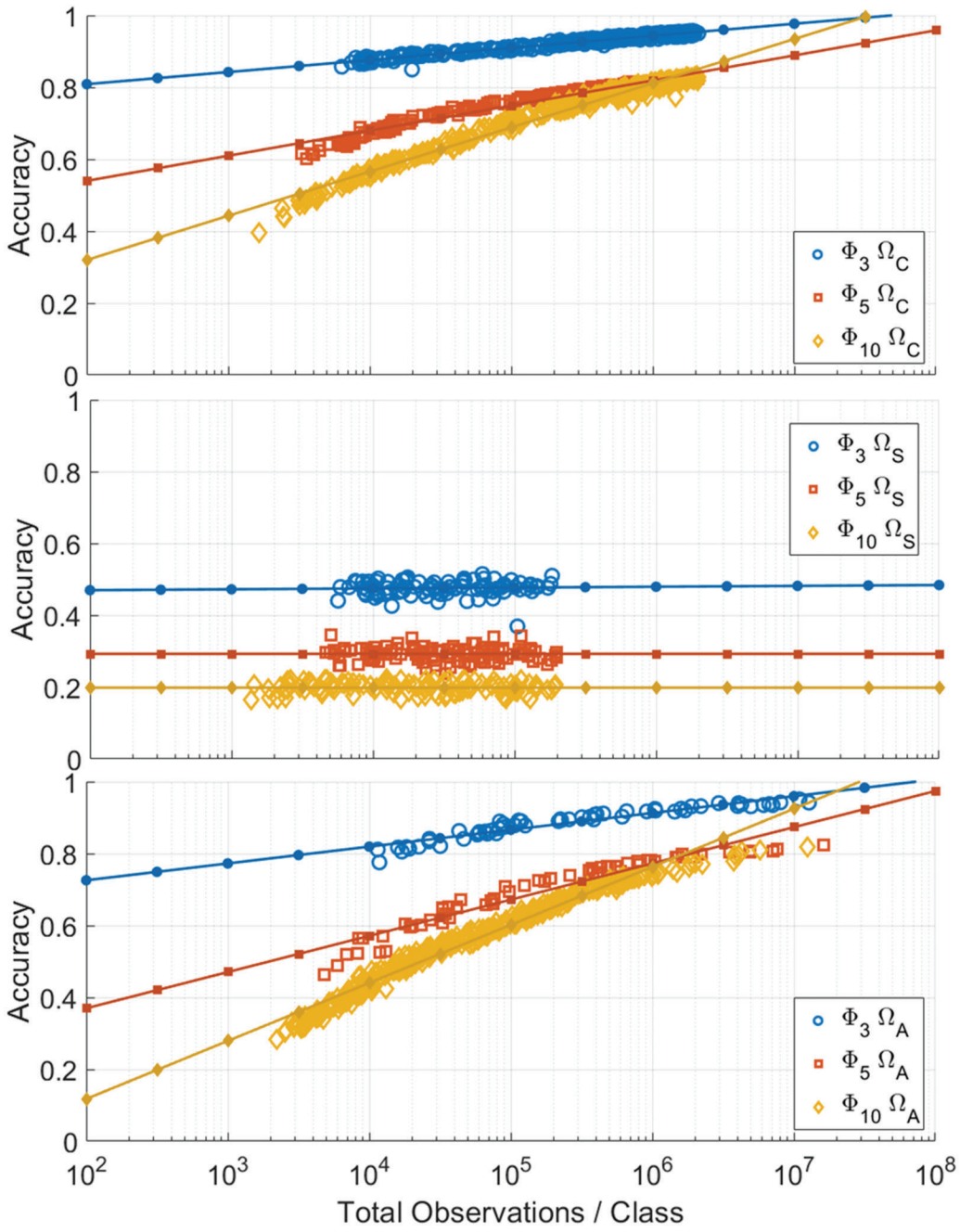

Figure 4.27: Plots showing the relationship between quantity of data used from each dataset (Top: Ω_C, Middle: Ω_S, Bottom: Ω_A) and the Accuracy achieved by networks trained on that amount of data. In general, the networks trained from datasets Ω_C and Ω_A have an increasing relation, but the network trained using Ω_S have a stagnant relation to performance in regards to quantity of data used to train.

4.4.1.3 Regression of Quantity and Metrics

With the confidence that the TL metrics discussed above have a positive and significant correlation with the performance of the system when performance increases with regard to the quantity of data used during training, the goal is to now derive the relationship between those metrics and data quantity, with preference being given to the metric that has a better goodness-of-fit (GoF) with a form of linear regression. In this case, a log-linear regression is used between the metrics and the data quantity. Starting with the accuracy of each network as shown in Figure 4.27, the log-linear fit is able to provide a quality value in terms of the accuracy achievable for a given OPC for the three datasets. Looking at the Φ_{10} problem set shows the quality quantification as

- $\Omega_C \rightarrow 81\%$ accuracy | 1M OPC
- $\Omega_S \rightarrow 20\%$ accuracy | 1M OPC
- $\Omega_A \rightarrow 76\%$ accuracy | 1M OPC,

but the quality can just as easily be defined as the OPC needed in order to achieve a given accuracy, given the linear fit can be inverted as

- $\Omega_C \rightarrow 5.25$M OPC | 90% accuracy
- $\Omega_S \rightarrow \infty$ OPC | 90% accuracy
- $\Omega_A \rightarrow 7.04$M OPC | 90% accuracy.

However, the log-linear regression between data quantity and accuracy has an undesired effect between the data points and the linear fit, which is that at the ends of the available data there is increased error relative to the center of the data points. Additionally, because the sign of error is the same at both ends, this suggests that the linear fit between data quantity and accuracy when there is minimal data will severely underestimate the data quantity needed to achieve high performance systems. For a better look at this issue, Figure 4.28 examines the residuals for the Φ_{10} waveform set across the three dataset types.

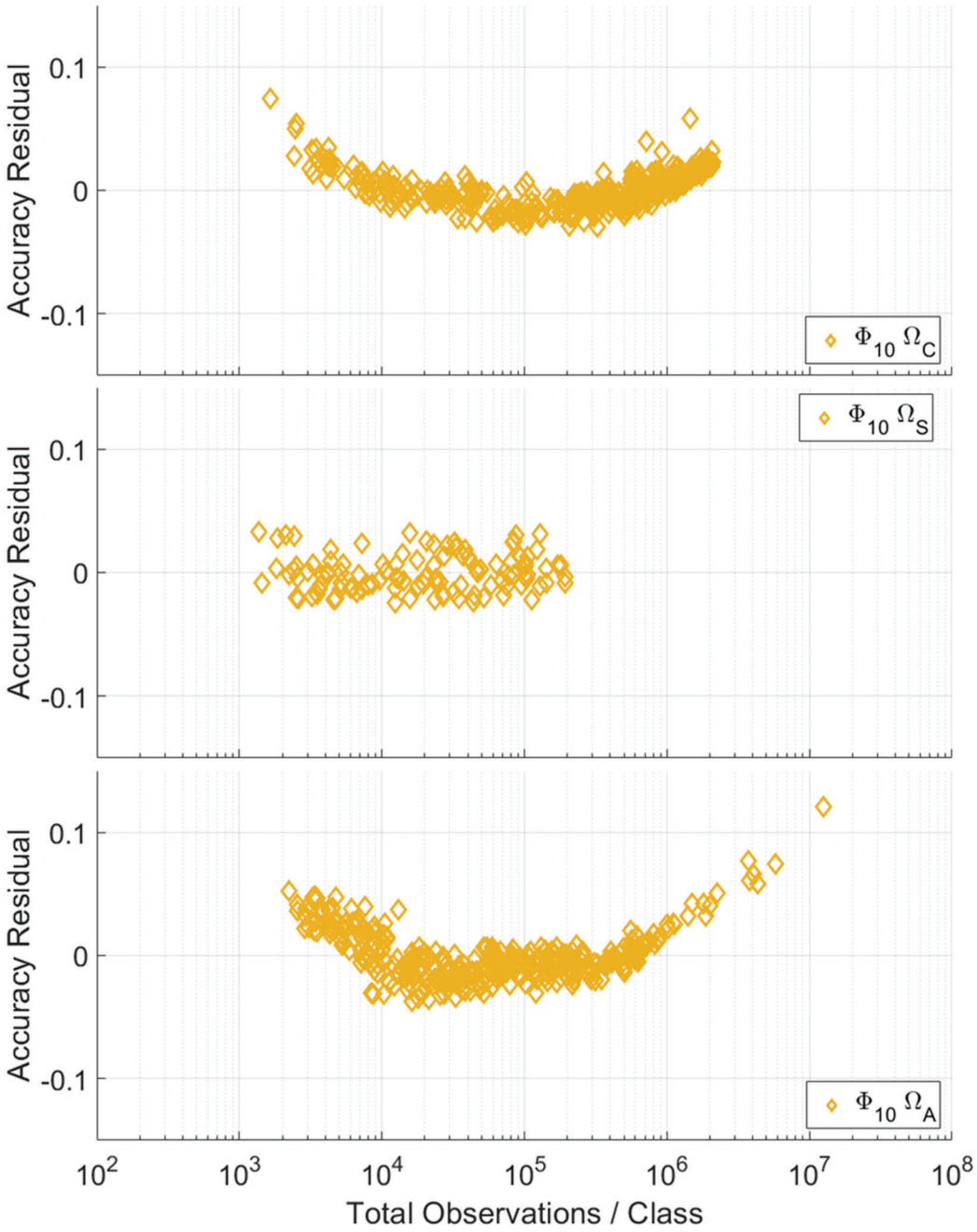

Figure 4.28: Plots showing the residuals between the regressed log-linear fits of quantity of data available during training and the accuracy of each trained network and the observed accuracy of each network. Plots show similar trends across the used datasets (Top: Ω_C, Middle: Ω_S, Bottom: Ω_A), with the edges of the available data deviating in the same direction, indicating a log-linear fit is not the ideal relationship between data quantity and accuracy.

This same sign of error at the ends of the available data suggest that a non-log-linear fit would be more appropriate for regressing the relationship between accuracy and data quantity, which is poorly suited to understanding the full relationship as available data becomes more limited to a narrow subset of the full data range. For example, just looking at a narrow portion of either end, high or low data quantity, does not provide enough context to predict a good non-linear fit. Therefore, a GoF measure that weights the outer errors more significantly than the errors toward the center of the data range is desired. Additionally, since both edges of the residual are of equal significance and the results are non-uniformly sampled across the observation space, a weighting that balances the weights into histogram bins will be used to normalize equal significance in the edges of the GoF measure. For simplicity, three bins will be used indicating lower, mid, and high data quantity observations relative to the log-linear fit. The weights are suggested as

$$
\begin{aligned}
w'(q_x[i]) &= \begin{cases} |q_x| \,/\, |b_{low}| & q_x[i] \in b_{low} \\ |q_x| \,/ 3\, |b_{mid}| & q_x[i] \in b_{mid} \\ |q_x| \,/\, |b_{hi}| & q_x[i] \in b_{hi} \end{cases} \\
w(q_x[i]) &= \frac{w'(q_x[i])}{\sum_{j=1}^{|q_x|} w'(q_x[j])}
\end{aligned}
\tag{4.9}
$$

where the middle bin has one third the weight of the edges, without which all three regions would be equally weighted. The division edges between bins is taken as evenly spaced on log scale between the minimum and maximum data quantities in the set, with $|q_x|$ being the number of elements in the set, while $\{|b_{low}|, |b_{mid}|, |b_{hiw}|\}$ are the number of observations within that bin. Those weights are then normalized such that their sum is unity. The GoF is then taken as the Normalized Root Weighted Mean Squared Error (NRWMSE)

$$
\text{gof}(\alpha_x, q_x, \hat{f}_{ll}) = \sqrt{\frac{\sum_{i=1}^{|q_x[i]|} w(q_x[i]) \cdot (\hat{f}_{ll}(q_x[i]) - \alpha_x[i])^2}{\text{Var}(\alpha_x)}} \tag{4.10}
$$

where the quantities (q_x) and accuracies (α_x) are use to derive the log-linear fit ($\hat{f}_{ll}$); however, the accuracies and fit can be swapped out for any other metric and matching fit.

The GoF for accuracy, NCE, LEEP, and LogME metrics are given in Table 4.12. A general conclusion is that all three metrics have potential to provide a better prediction of data quantity needed to achieve high performance; however, the correlation presented in Table 4.11 in addition to these results suggests that NCE will be the most consistent estimate, with LEEP being a close second. LogME, by comparison, offers the most promise with regard to the augmented dataset, but has the highest variability among the three metrics examined here. One more unique attribute about the linear regressions of the metrics is that accuracy, NCE, and LEEP all have residuals typically indicating that the true quantity of data that is needed will be underestimated, while LogME's residuals are inverted suggesting that that LogME's regression will overestimate the amount of data, giving soft bounds of the required quantity of data being between the estimates of NCE and LogME predictions.

Table 4.12: Goodness-of-fit (GoF) for a log-linear regression between dataset quantity available for training across datasets (Ω) and waveform sets (Φ) for Accuracy (α) and (NCE, LEEP, LogME). Perfect fit would have a value of 0. **Bold** values represent the best GoF value for the log-linear regression between the metric and data quantity available during training.

Φ	Ω	α	NCE	LEEP	LogME
Φ_3	Ω_C	0.2478	0.2054	**0.1885**	0.2662
	Ω_S	1.0196	**0.9515**	0.9521	0.9531
	Ω_A	0.3120	0.2674	0.2636	**0.1672**
Φ_5	Ω_C	0.2499	0.1552	**0.1458**	0.1987
	Ω_S	0.9491	**0.9367**	0.9451	0.9652
	Ω_A	0.3016	0.2163	0.2208	**0.1433**
Φ_{10}	Ω_C	0.1514	**0.1138**	0.1173	0.1179
	Ω_S	0.9853	0.9783	0.9731	**0.9697**
	Ω_A	0.2706	0.2652	0.2797	**0.1102**

4.4.1.4 Predicting The Data Quantity Needed

Now that the metrics have been compared in terms of a regressed log-linear fit with the quantity of data used to train the model, the question is how to determine what value of the metrics will provide a desired performance. Looking back at Figure 4.26 shows that the metrics and accuracy don't have an easily fit relationship that would map a metric back to accuracy, and in fact would only be trading one non-linear regression for another. To overcome this problem, label whitening to acquire near perfect performance is proposed to act as a quasar that can help map the performance of the metrics with accuracy.

The procedure starts with label smoothing [36] (4.11) of the truth labels for the evaluation set, followed by a logit transform (4.12), which without the label smoothing would not be a useful approach as infinite values would be returned for the correct class and negative infinity for all other classes.

$$\tilde{l}_x = l_x - \gamma \cdot \left(l_x - C^{-1}\right) \tag{4.11}$$

$$m_x = \log\left(\frac{\tilde{l}_x}{1 - \tilde{l}_x}\right) \tag{4.12}$$

Label smoothing applied on its own does not affect the value of accuracy, NCE, nor LogME, but it does affect the LEEP score and is dependent on the smoothing factor, γ, and number of classes in the classification problem, C. The effect of γ on the LEEP metric can significantly affect the metric, so γ is chosen to be the minimum value that the approach of $|l_x - \tilde{l}_x| > 0$ within the chosen machine precision. The effect of label smoothing and the logit transform allows for the values to now sit at a finite coordinate to which noise can be

added to stochastically decrease the accuracy of the system in a controlled manner. The normal distribution is used to whiten the logits in this case where the standard deviation of the noise, σ, can be chosen for a degradation of accuracy, ϵ, of the true labels given the number of classes in the problem space and the label smoothing γ in use.

$$\tilde{m}_x = m_x + \mathcal{N}(0, \sigma^2) \tag{4.13}$$

$$\sigma(\epsilon) = \frac{\log\left(C^2(1-\gamma) + \gamma^2(C-1)\right) - \log\left(\gamma^2(C-1)\right)}{2 \cdot \text{erf}^{-1}\left(2 \cdot \sqrt[C-1]{1-\epsilon} - 1\right)} \tag{4.14}$$

With the whitened logits the inverse logit, or logisitic, transform is applied and balanced such that the sum of any result is unity, $\sum \hat{l}_{xi} = 1 \; \forall \; i$.

$$\hat{l}_x = \frac{\exp(\tilde{m}_x)/(1+\exp(\tilde{m}_x))}{\sum_{k \in C} \exp(\tilde{m}_x[k])/(1+\exp(\tilde{m}_x[k]))} \tag{4.15}$$

Figure 4.29 shows the effects of this procedure on the error and metrics for a given ϵ averaged over 1000 iterations, and shows a trend that can be maintained with increasing ϵ; however, an important note is that this type of error does not properly reflect the distributions of error that can be expected, so smaller values ($\leq 1e-5$) of ϵ will likely be more appropriate than larger values (0.1). Looking at the residual error, (4.16), in terms of the dependent variable, ϵ, relative to the measured value, $\hat{\epsilon}$, as seen in the top left plot of Figure 4.29, the minimum average error across the three classes is achieved at $\Delta(1e\text{-}5, \bar{\hat{\epsilon}}) = 0.0169$, with the average normalized residuals being nearly equal at the extremes ($\Delta(1e\text{-}8, \bar{\hat{\epsilon}}) = 0.178$; $\Delta(1e\text{-}1, \bar{\hat{\epsilon}}) = 0.182$).

$$\Delta(\epsilon, \hat{\epsilon}) = \frac{\epsilon - \hat{\epsilon}}{\epsilon} \tag{4.16}$$

At this point a means for determining the value for each metric has been proposed that won't suffer from the need to have a perfect response that can be used, and will help with metrics such as LogME where the maximum is not immediately known given the iterative solution that is employed to produce the score. These values for a given small ϵ can then be used to regress the corresponding metric's data estimate for achieving such performance. The log-linear regressions for each metric, dataset, and waveform space combinations are shown in Figure 4.30, while the log-linear regressions for accuracy are shown in Figure 4.26, and together help to better visualize the GoF results given in Table 4.12.

Making use of the whitening procedure above and the log-linear regressions between data quantity and the metric's score a prediction for data quantity needed to achieve arbitrarily high performance can then be found. For example applying an error of $\epsilon = 1e\text{-}5$ to each examined problem space for the metrics and selecting a value averaged over 1000 iterations, the data quantity predictions can be made for each metric as shown in Table 4.13. Where the predictions are found by inverting the linear fit to estimate the quantity from the predicted metric as

$$\tilde{q}_\chi = \log_{10}(\hat{q}_\chi) = \frac{\hat{M}_\chi - b_x}{s_\chi}, \tag{4.17}$$

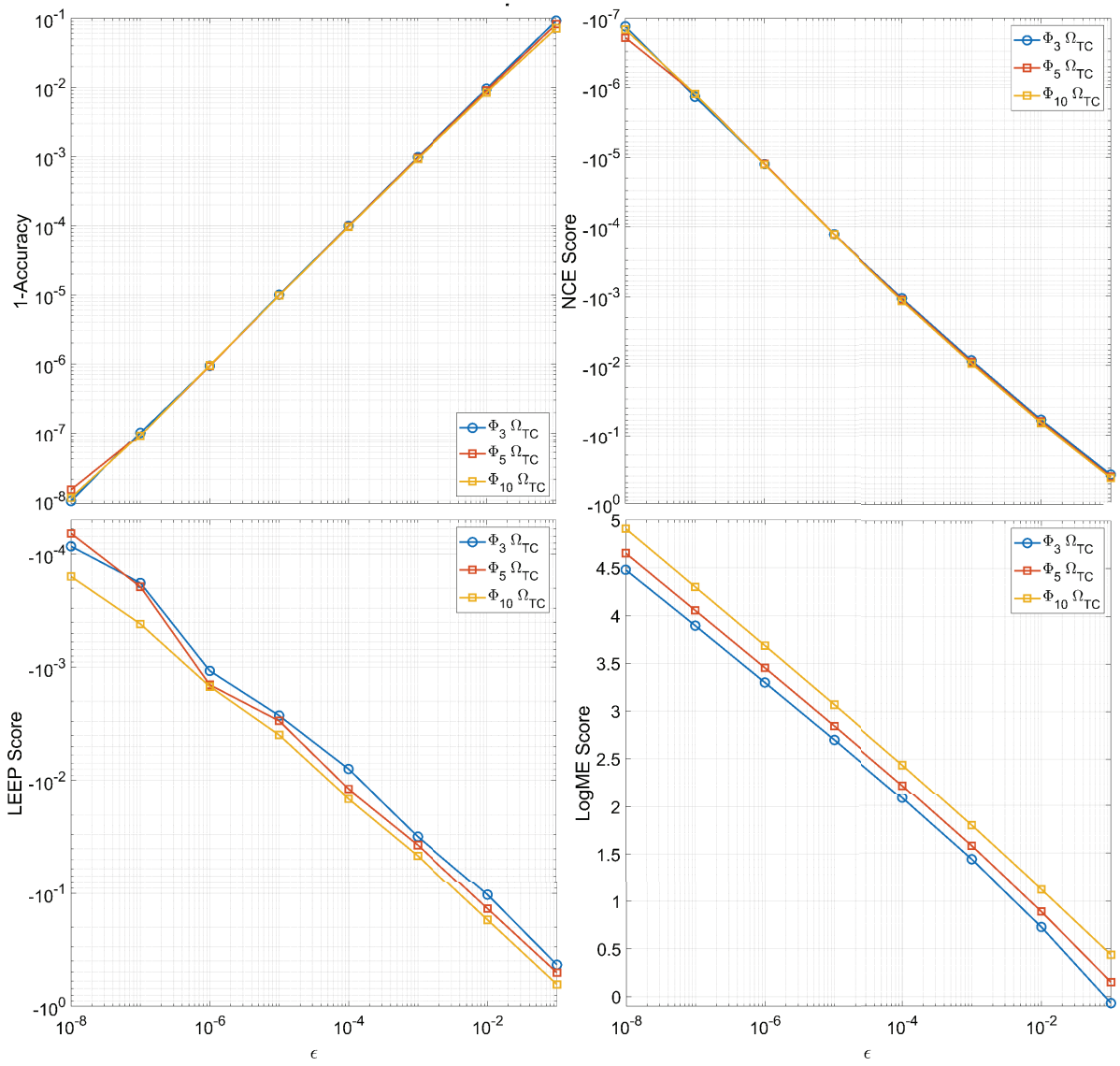

Figure 4.29: The change in the value of {Top Left: Accuracy (1-Accuracy, logscale); Top Right: NCE (logscale); Bottom Left: LEEP (logscale); Bottom Right: LogME (linear)} as a function of the induced error, ϵ, expected to accuracy from whitening the truth labels of the evaluation set Ω_{TC}. Results plotted are the average values over 1000 iterations per data point.

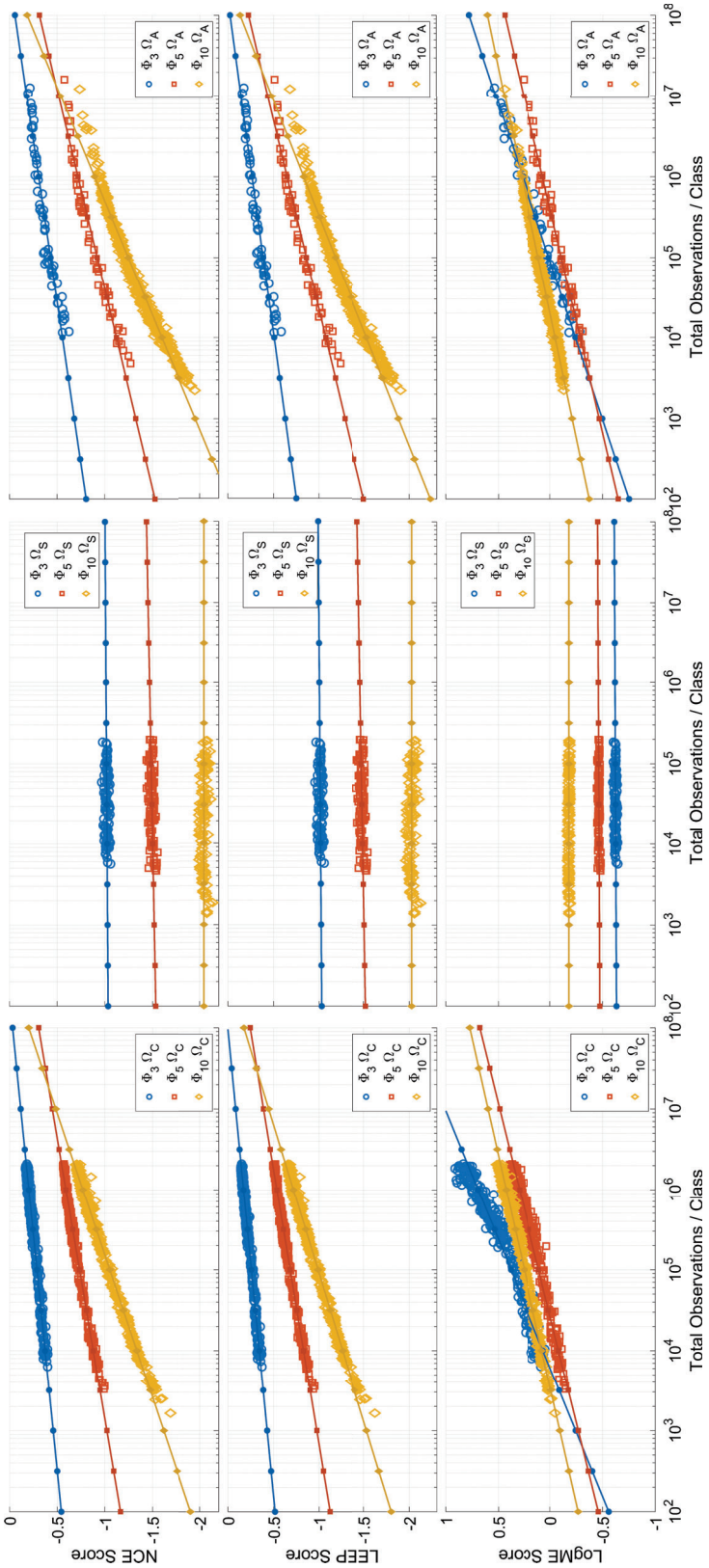

Figure 4.30: Plots showing the relationship between quantity of data used from each dataset (Left: Ω_C, Center: Ω_S, Right: Ω_A) and the metric (Top: NCE, Middle: LEEP, Bottom: LogME) achieved by networks trained on that amount of data. In general, the networks trained from datasets Ω_C and Ω_A have an increasing relation, but networks trained using Ω_S have a stagnant relation to performance in regards to quantity of data used to train. With a solution for a system that can perform arbitrarily close to perfect on the evaluation set, these linear regressions can then predict how much data would be required to achieve such a system.

Table 4.13: The performance of Accuracy (α), NCE, LEEP, and LogME for the label whitening procedure proposed in Section 4.4.1 for a desired error $\epsilon = 1e-5$. Additionally the data quantity needed per metric for the log-linear regression of the metric with the data quantity used during training is provided.

Φ	α Metric	NCE Metric	LEEP Metric	LogME Metric	Ω	α Quantity	NCE Quantity	LEEP Quantity	LogME Quantity
Φ_3	0.999990	-1.275e-4	-2.668e-3	2.696	Ω_C	48.7e6	237e6	88.5e6	2.46e12
					Ω_S	6.42e224	1.64e190	1.38e158	∞
					Ω_A	72.8e6	291e6	144e6	3.47e15
Φ_5	0.999990	-1.272e-4	-2.972e-3	2.842	Ω_C	394e6	13.5e9	4.45e9	2.57e19
					Ω_S	∞	9.15e95	6.57e100	∞
					Ω_A	181e6	3.60e9	1.11e9	2.41e21
Φ_{10}	0.999990	-1.289e-4	-3.963e-3	3.066	Ω_C	34.2e6	514e6	430e6	1.61e21
					Ω_S	∞	∞	∞	∞
					Ω_A	29.3e6	337e6	233e6	1.57e23

where the $\tilde{q}_\chi$ value is the logarithm base ten of the quantity estimate for the selected metric $\chi \in$ [Accuracy, NCE, LEEP, LogME], $\hat{M}_\chi$ is the metric value found through the whitening procedure, and s_χ, b_χ are the slope and y-intercept, respectively, of the log-linear regression given a logarithm base ten applied.

4.4.2 Prediction Performance with Minimal Data

The prior sections made use of all data points taken in order to establish the best predictions for data quantity with their given metric. As these are estimates that are intended to predict the data quantity needed to achieve high performance systems through the increase of available data alone, certifying any result in particular is beyond the scope of this work, as the expected predicted quantities will far exceed the available data acquired. Instead, the focus shifts to how less available data during training relatively affects the prediction capability for each metric in comparison to greater quantities of available data.

Due to the performance of the synthetic dataset stagnating, further analysis will ignore this case going forward. For the purpose of finding how well the log-linear regression with each metric is able to predict the data quantity needed, the data quantities provided in Table 4.13 with preference for a quantity estimate given by the GoF in Table 4.12 will be used such that the metric that achieved the best GoF will be used as the truth for the problem space. Therefore predictions of the models making use of Ω_C will use the LEEP metric's quantity prediction for Φ_3 and Φ_5, but will make use of the NCE prediction for Φ_{10}, while the predictions for Ω_A will all make use of the LogME metric and these quantity predictions

Table 4.14: Quantity estimates being taken as truth for the combinations of waveform groups, Φ, and training datasets, Ω. The augmented quantities are estimated using the LogME metric regression, while the captured quantities are estimated from either the NCE or LEEP metric based on the GoF in Table 4.12.

Φ	Ω_C	Ω_A
Φ_3	88.5e6	3.47e15
Φ_5	4.45e9	2.41e21
Φ_{10}	514e6	1.57e23

are summarized in Table 4.14. The predictions for each metric can be seen in Figure 4.31 where the top row shows the predictions when using the Ω_C dataset, while the bottom shows the predictions for the Ω_A dataset. The columns consist of waveform spaces $\{\Phi_3,\Phi_5,\Phi_{10}\}$ from left to right respectively.

The general understanding given in Figure 4.31 is that both NCE and LEEP will give a more realistic prediction for data quantity than Accuracy alone, while the prediction given by LogME can serve as an upper bound. Due to the log-linear regression any deviation can result in orders of magnitude error in either underestimation or overestimation, and, without having enough data to acquire the metric that produces the best GoF regression, a midpoint estimate is recommended. The midpoint estimate seeks to balance the two extremes and the estimate becomes $\tilde{q}_{\text{MidPoint}} = 0.5 \cdot (\tilde{q}_{\text{NCE}} + \tilde{q}_{\text{LogME}})$, such that the midpoint estimate averages the quantity estimates on the log scale rather than the linear.

Returning to Table 4.14, it is important to understand how long a sequential collection of data of this kind would take, that is, for a collection that records at 10kHz, these waveforms from three waveform groups $\{\Phi_3,\Phi_5,\Phi_{10}\}$, collection of the number of observations implied would require [1.72, 144.5, 33.4] years to acquire for the target observations needed for the Ω_C predictions and require [0.989, 82.9, 19.1] terabytes of storage to store in an uncompressed state. While this could be feasible if the collection was performed in parallel rather than a sequential collection, the suggestion that should be taken rather than immediately starting a long term collection is to instead improve the training routine and model architecture to allow for this procedure to produce a regression with a more significant slope than the approach used to produce these results.

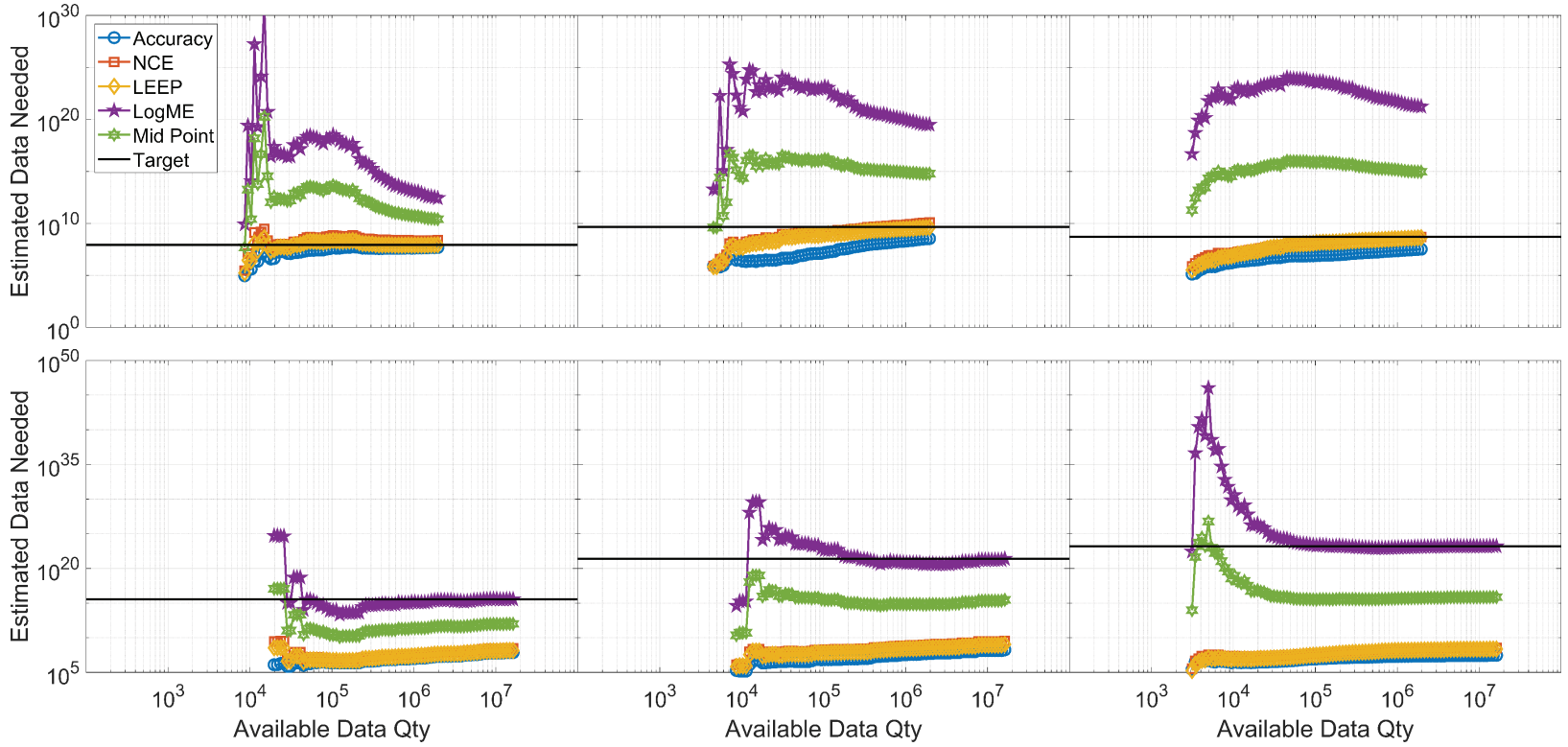

Figure 4.31: Plots showing the quantity predictions based on a limited amount of available data used to regress the estimate on the {Top Row: Capture Ω_C; Bottom Row: Augmented Ω_A} datasets when being used to estimate across the {Left: Φ_3; Center: Φ_5; Right: Φ_{10}} waveform space. The lines represent using {Accuracy: circle, NCE: square, LEEP: diamond, LogME: pentagram, Log Scale Midpoint of NCE and LogME: hexagram, Target: none} to predict data quantity needed, while Target is determined in Table 4.14. The midpoint serves as a balance between the underestimates produced by Accuracy/NCE/LEEP and overestimates from LogME for when a Target is not known *a priori*.

4.5 Summary of Training Data Needs for AMC

There are three main aspects for using the available data and maximizing the performance of an ML system: (1) high quality data that adequately reflects the application space, with as precise and accurate labeling on the metadata as is possible; (2) understanding the degradation space such that augmentation can be used to provide more visibility into the distributions within the data without warping the distributions outside the target application space; (3) being able to overcome the initial random state a model must first occupy before any training can be performed.

The first and second aspects refer to the quality of the data available and, while crucial, aren't always under the control of the developer creating the model, yet without the first aspect even the best ML practitioner would be unable to provide results. Tapping into the skills of an analyst, the dataset can be expanded to provide: more observations, more batches, more learning steps as a whole that can make up for, or minimally roughly patch, areas that the available data might not adequately reflect through augmentation. The third aspect, overcoming random initialization, is where all properties of data can best be utilized. By incorporating stages of training, the model can more easily overcome the random initialization and learn how to do *something*, even if its not useful directly to the end goal of the system. This concept of being able to prime the network, as was done by stepping up from $\Omega_S \rightarrow \Omega_A \rightarrow \Omega_C$, allows for larger structures to be learned more quickly with an easier data space, and then become more fine-tuned to the exact problem, rather than a significant portion of the learning procedure trying to identify all the fine-detail nuance of the data space from the very first step.

However, when and *if* there's a chance where the ML developer is able to plan out the data acquisition, being able to forecast data needs can be critical. Having the ability to create a small dataset to act as a compass for the ML approach can lead to significant gains if the quality is high and well thought through. Understanding the performance gains relative to the available data quantity can determine whether the training data is adequately paired to the real environment. By better quantifying distributions in the nuisance space, the sample space that should be augmented over can be better explored. These performance gains can then provide a starting point for quantity predictions required to achieve a target goal and help avoid risks in what is mostly a trial and error system.

Chapter 5

Real-World Applications

The value of this dissertation is in the understanding and skills needed to bring RFML from the lab to the real world. To that end, key information about what should be considered in the data and examples of the consequences when it's not present have been presented. The following sections speak not only to this data consideration, but how incorporating these principles for data quality into sponsored programs provides significant value to the goal of the program.

5.1 Tactical SIGINT

The program that supported this development eventually went on to become the SignalEye [34] product. This multiyear effort led to a lot of discovery and understanding about the problem space and agrees with lessons learned and discussed in this dissertation. The fundamental purpose of this routine was to provide an AMC application that could classify signals once they had been found and isolated. The preliminary focus was on positive SNR observations with stretch goals to push the performance range as low as it could go. Initial development was started on synthetic data and grew to wideband observations with multiple (non-overlapping) signals present. The detection routines were left to expert approaches predominantly relying on some form of energy detection in order to minimize the reliance on interaction between ML approaches that had not been studied in detail.

After initial work in the synthetic data space showed viable results, pushes for real-world applications came and the team of researchers made the assumption that an AWGN simulation would not be sufficient based on the experience and skepticism in ML of the team. This led to synthetic data being generated with frequency-independent IQ imbalances being applied to individual signals as well as imperfections related to signal detection routines in terms of SRM and FO being applied, as well as varying SNR observations to be synthesized. With each iteration more real-world degradation was added into the synthesized datasets with the ultimate goal of being able to perform at a collection event planned for spring 2018. As the collection event approached, field studies were performed to characterize the effect of the channel in terms of Rayleigh/Rician fading channel environments that could additionally be used in the synthetic signal generation system. Of all of the degradations applied in the synthetic system, the only degradation that did not come from field measurements

and characterization was the value around the frequency-independent IQ imbalance, as at the time there was no way to know, let alone characterize, the transmitters that would be present at the event.

The threshold for success of the system was to be able to characterize on the order of 20 waveforms in real-time, or as near to real time as possible, with an accuracy of 80%. Some observations made during this last push saw that one large network trying to classify everything ultimately sacrificed accuracy and had a longer than desired compute time on the target platform while trying to generalize over the full range of what was expected. By switching to a hierarchical design, many smaller models could be used more effectively while minimizing the generalizations any one model needed to learn. By understanding the limitations of the platform, the characteristics of the EME, and focusing models to perform more specific tasks, the goal of 80% accuracy was met, and after multiple assumed revisions by the company, the work came to fruition with the release of the SignalEye product.

5.2 Signal Detection

Another project's focus was on the problem of detection in congested environments, and when outside of congested environments technical approaches can actually fall back to a lot of the image processing domain when considering stronger signals [48]. From the start, the assumption is taken that strong signals can be identified even in this congested environment, therefore the focus of this work is trying to figure out how weak of a signal can be still be detected. One additional assumption is that such a weak signal will typically be of a more wideband nature than the strong narrowband signals in order to minimize interference and still function as a communication system [27]. One additional consideration is that the receiver system has multiple receive chains, which may or may not be aligned, that are available for observation.

In order to focus on the problem of detection, it is assumed that the waveform is QPSK, to help minimize the generalization for the ML problem, and will pass through a unique time-varying multipath channel before arriving at the sensor array while being independent of other nuisance signals (with their own time-varying multipath channel) present in the observed spectrum. Assuming the array is minimally time and frequency aligned, the other characteristics like phase are assumed to not necessarily be aligned. The final application seeks to know if the weak signal is present, rather than which antenna can see the signal if it's present, and therefore at some point the information space needs to collapse from M sensors down to one decision. An examination of an aggressive interference rejection technique for an unknown array configuration was considered and contrasted against a fully ML network that could ingest the M sensors directly. The interference rejection technique relies on the Singular Value Decomposition (SVD) to eliminate stronger singular values that are above a specified threshold given knowledge about the noise floor in the observations and emphasizing weaker ones believed to be the subspace where the weak signal will occupy.

This approach has the potential to degrade the weak signal's full time/frequency coverage and would likely prevent typical techniques from then identifying the presence of the signal with traditional means. The ML approach would then be trained to understand these kinds of degradations and learn to become invariant to them. Exploratory studies found that providing both the time and frequency representation of the signal enhanced the network's ability to learn in this problem space.

What was observed is that intelligent preprocessing applied to better present the problem space to the ML algorithm resulted in smaller networks, smaller data transfer pipelines, and as a result an overall faster system could be created in this approach, with the underlying assumption the the preprocessing could occur in the FPGA or similar system prior to being piped into the ML model. Not only did this preprocessing result in a faster system overall, but the trained network became more consistent and had higher performance in terms of detection.

In terms of raw performance, the raw sensor network was able to perform better at lower SNRs for a set decision threshold of 0.5 after a sigmoid activation, but it came at the cost of the false-positive rate, and when both approaches were constrained to a false-positive rate of 1%, the improvement in performance of the raw-sensor network became marginal at best.

5.3 Wideband Aggregate Spectrum Generator

The Wideband Aggregate Spectrum Generator or WASP Generator, shown in Figure 5.1 as the hardware setup and in Figure 5.2 hopping a 20MHz QPSK signal, is the porting of signal generation from software, which even in high end test equipment is still done in software and loaded for playback [216], to hardware in an FPGA system. As the fundamental backbone of 'gr-signal_exciter' relies on a reliable, well conditioned Random Number Generator (RNG) with a repeat cycle longer than the generation of the signals themselves, this work is heavily dependent on the implementation of such a PRNG, as was presented in [132].

This process calls for the redesigning of block based function processing into logical circuits that rely on control signals for intricate timing between components and cores. Initial development saw the fruition of LDAPM modulations being loaded into the FPGA fabric with controls to allow for the modulators to be controlled precisely in time, frequency, phase, and power, allowing for wideband and high data rate data generation. By making use of Xilinx's VIVADO and Matlab's Simulink interaction, the logic and design could be iterated through initial 'is it possible' prototypes through near ideal cores that could be deployed within Xilinx FPGA fabrics. Unlike the software source, additional focus was given to allowing for burst transmissions to better emulate and understand the transients that are often produced in hardware systems.

By the end of the initial phase of the project, the WASP Generator was capable of generating any LDAPM waveform whose symbols could be drawn from a lookup table. Additional

Figure 5.1: Here the WASP Generator is shown in terms of the Mercury Systems hardware in use along with its connection to a Keysight Signal Analyzer for visualizing the generated spectrum.

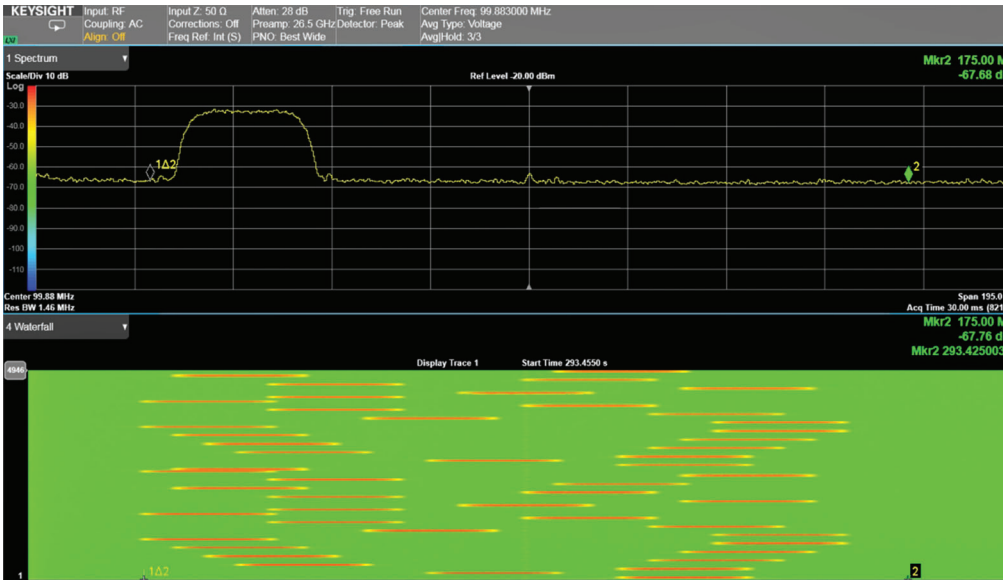

Figure 5.2: Using the WASP generator to hop a 20MHz QPSK signal around the available 150MHz spectrum.

the waveform could then be filtered with a user specified pulse shaping filter, as well as any additional interpolation filtering to create a target bandwidth. While the exact functional limits were not explored in this work, this system easily produced multiple simultaneous LDAPM waveforms on a single FPGA device with each modulator running in real-time at 20MHz bandwidth, capable of arbitrary bandwidths, frequency offsets, and power control. In order to provide a means for pure in-system simulations, the ability to add the signals to a background noise of AWGN was designed in as well. The final performance test showed that not only could the system operate multiple signals, but was also able to perform frequency hopping across the full operational bandwidth with dwell durations on the order of milliseconds or less.

The most useful aspect of this system is that while it can generate signals at high rates and produce large volumes of data at rates servers couldn't keep up with, it also produced signals for metadata tracking, allowing for precision timing to be provided in real-time. This means that a system capable of generating large quantities of data for developing ML datasets that hopefully will help alleviate some of the costs and strain associated with curating new datasets is well underway.

5.4 Commercial Whitespace

For the realm of commercial whitespace, the best point of reference is looking to the structure and results of DARPA's Spectrum Collaboration Challenge [217]. Here the use of the Colosseum, 21 servers consuming 65 kilowatts of power, to simulate as closely as possible the actual EME environment as the different transceivers move through an environment [217] and still collaborate on how to use the spectrum is a critical undertaking in making strides in this application space. Given the challenges of the hidden and exposed node problems for a single transceiver, the problem is one that likely can only truly be addressed in a distributed and collaborative way by making use of the actual observations while online in the channel. While it may still be possible to manage this level of task as a distributed, uncooperative approach, the challenge will come down to how any system is regulated, and how feedback is provided in order to minimize hidden and exposed node vulnerabilities to the observation space. A critical path forward is being able to simulate the real world with enough accuracy in real-time, or faster than real-time, in order to continue to make the strides in reaching a final product.

5.5 SEI

SEI is inherently a difficult problem as the ultimate desire for the problem space is to be able to say there is an approach that is invariant to the information, the modulation, the channel, and the receiver. This means the entire goal is to understand the transmitter such that

regardless of every other variable in the transmission path, the characteristics of the transmitter alone make the decision. Unlike modulations, being able to simulate the full scope of effects a transmitter imbues into the waveform requires detailed analysis and modeling that would be extremely difficult to do at scale, and therefore limit the applicability of simulated approaches to model these imperfections, resulting in trained models without any guarantee of transferability to the real-world problem set desired. This is further complicated due to the fact that real devices' characteristics are both bandwidth and carrier frequency dependent making one-configuration-handles-all assumptions highly unlikely [66]. ML paired with real-world captures, consisting of as diverse a set of transmitters as can be managed, across all bands of operation while varying the bandwidth, will likely provide the best opportunity to address this problem space. One more benefit to using ML approaches is due to the non-linear perturbations the transmitter induces on the waveform, which are potentially better handled by the non-linear nature of the layers and activations. By allowing the ML routine to learn from this real-world data, the correlations between degradations induced in the transmitter can better be leveraged, as the model will learn features of importance rather than rough isolated features from an expert selection.

SEI will continue to remain a difficult problem space as long as the taken approaches are so heavily swayed by the state of the channel [65]. One of the breakouts from the DARPA RFMLS program was the work performed at Expedition Technologies, where focus was put on algorithmically finding approaches whose computation structure is invariant to the channel [218]. Even with approaches designed to be invariant, the performance of systems trained with channels applied to the signals still resulted in better performance than those applied to the signals alone. Without a doubt, the way forward for this body of work will rely heavily on expanding the techniques designed for channel invariance and structures of the model that can better make use of such techniques.

Chapter 6

Conclusions and Future Research

The quality of data used during training drives a ML system's performance. This concept of data quality is tied directly to the application space under which the system is expected to perform, with higher quality data coming directly from that application space. In the case of whether or not a certain routine is feasible, the use of synthetic data alone provides the simplest and most affordable means of moving toward a conclusion in that regard. However, when the artifacts present in real-world observations are not well understood and accounted for in synthetic data, the end result of models trained on the synthetic data is models that rarely provide utility in the field, yet this is not a criticism of synthetic data as a whole, but of synthetic data that does not contain sufficient consideration of the application space. In order to fully explore what previously had been assumptions about this relation between synthetic data used during training and performance on data observed in the field, a large scale collection was designed and executed, providing an adequate quantity of observations in order to vet the assumptions and provide concrete evidence about the relationship. Additionally, a large scale experiment is conducted in this work showing the underlying nature of data origins, the role of augmentation, and the resulting effect such origins have on performance. This approach should lead to better understanding of the nature of the dataset's role in producing high quality models that can be used in the field successfully within the RFML domain.

By designing the experiment in a way that parametrically varies the available data, the combinations of datasets, and changes in the complexity of the problem space, underlying conditions are able to be discovered. This experiment allowed for examining dataset quality and doing a regression analysis to help better understand how both quantity and quality of available datasets can lead to more effective systems to be used in the field. The concept of dataset quality is then further extended to provide a quantitative means for comparing different sources of data that can better inform future data generation and collection efforts. For example, in the effort to improve synthetic generation through expert means or otherwise through GAN approaches, the quality of the networks trained can be compared quantitatively by their resulting performance relationship to distinguish higher quality creations.

By regressing the functional relationship between performance and data quantity, a means is designed for estimating the data quantity needed to achieve a significantly high performance for a system, with the assumption that it can actually reach that performance level. This approach is then extended by finding metrics correlated with performance that project the

models' affinity for the data space through use of measures associated with transfer learning. By then combining these results, a more reliable estimate regarding the needed data quantity for a given dataset quality to achieve high performing results. While this approach still has room for improvement, it is a critical need to allow for planning and forecasting data needs in order to balance both the time and the cost associated with procuring data at scale.

This work is derived from the specific application space of AMC, and while the quantitative results should be considered unique to the problem and data space, the techniques cultivated in this work have a general application to machine learning problems. In the absence of noisy or poisoned labels, the ability of a system to learn the problem should only be limited by the capacity of the technique to learn, and the quality of data available. While the augmentations applied (FO, SRM and SNR) will likely only be reasonable within the limited domain problem spaces (RFML and audio processing for example), understanding and applying useful augmentations will continue to be a significant capability to ML systems in order to increase generalization of models over the base available dataset alone.

One fundamental assumption to keep in mind is that there exists a test set from against which any trained model can be compared. In this way, there must be some initial data that can be trusted to represent the problem of interest in a meaningful manner. Failure to have some meaningful data could result in systems similar to the synthetic models that performed well on synthetic tests, which is to say, nearly useless when put into the field. For problems that have hard to observe phenomena, such as anomaly detection routines, this approach wouldn't work, the possibility of collecting large amounts of observations on an anomaly would be impractical. Rather, focusing on a surrogate dataset that alters real observations for test would be advised. One such example could be looking for a failing transceiver in the 2.4GHz band, and generating a dataset that encompassed all possible ways the transceiver could fail would be a daunting task to perform in a real data collection. The expectation is that data collections of the counter state, systems working well without anomaly, would be more beneficial to better understand what is proper and create a system whose predictive state would deviate from observations when anomalies are present.

Another domain where direct application of this work would be expected to struggle are in systems that are not passive observers. When a system is designed to be interactive with what is being observed, there is a feedback loop that is not included within a static observation space. Works where this is desired will require additional conditions place on the observation dataset that exceed both the assumptions on the dataset and the scope of this work.

For best results in applying this work, the problem space should be one in which passive analysis is desired. Meaning that the goal is to better understand what is present in an observation space, and can be observed either naturally or through controlled collection (the ability to force it to occur). Given that such a problem is the focus of the ML routine, and/or that a trusted good dataset is available, the techniques developed here can help distinguish whether dataset manipulations are beneficial or detrimental as well as helping to understand

the quality of the data generated through such manipulations.

The work here enables future research:

Improving synthetic dataset quality. By making use of the relationship between data quantity and performance, synthetic dataset generation could create benchmarking for different approaches of generation, along with different parameter ranges. This could help for those with access to the open source tools that can generate data, but might not have the ability to extract relevant characteristics from available data. By trial and error, the measure of quality regressed could help determine better parameters sets for the problem at hand. Similarly, this approach provides an external means to justify the data generated through GANs that might show improvements that might even exceed captured data germane to the problem space. For any approach the generates data for learning a problem space over direct capture, this approach can be applied and the quality can be compared between the generated and captured datasets.

Cost modeling for dataset curation. Data curation is typically a blind endeavor in terms of estimating the cost needed to fully procure. Underestimating these costs can result in inadequate or irrelevant data being collected, as has happened many times on funded research when timelines pressure a collection event before adequate failsafe checks are put in place to ensure proper data is being collected. In addition, by understanding just how much data is expected to be needed, this can allow for proper collection timelines to be designed, rather than a simple triage approach of collecting what is able within a specified timeline and hoping that is enough. The techniques within this work can allow for a smaller collection of data to be performed and then estimate the total data needed better than accuracy alone when regressing the relationship between performance and data quantity.

Study the effects of label noise. The primary focus in this work was in understanding the relation between performance and data quantity under the assumptions that the *Correctness* was absolute. But as more work goes into developing RFML techniques, the more likely it is that collection events will occur not in nicely maintained environments, but in contested, congested, and potentially adversarial environments. Taking time to research how the analysis here is affected by label noise would be vital to better understanding the level of care that needs to go into dataset curation. By applying this dissertation's data quantification techniques while intentionally adding in label noise, an investigation of how susceptible the problem space is to the problem of 'double descent' can be studied while examining how such controlled noise reduces the quality of the dataset.

Similar studies in other RFML applications. The work performed here is expected to translate to the other applications spaces within RFML; however, application-specific

dependencies are also expected. Developing a large enough dataset that contains the effects of the real world over large enough time scale could highlight other deficiencies in the domain that are not as significant in the problem space of AMC, but without robust datasets to corroborate or reject, this hypothesis it is left purely to conjecture. One such example is how SEI algorithms have a higher anecdotal dependence on the channel conditions and background noise conditions than in AMC applications.

Study different architectures/training methods. This approach isolates the architecture and training routine such that the effect of the dataset can be studied as something independent of them. However, as networks improve, significant nuisance parameters such as the effect the channel has on the problem space might in time have invariant approaches devised. For those attempts and studies, understanding and proving their invariance can be done through a study such as this, where the synthetic dataset has a higher quality and no longer stagnates as it has done in this work. For the same datasets being available, different architectures, or training routines, might have significant effect on the quantity and performance relationship, so contrasting the techniques here with a different architecture could provide insights into the compatibility with the problem or data space.

Co-channel and adjacent channel observations. For the most part, the literature focuses on problem spaces where signals can be isolated in time/frequency/space such that a network only needs to understand what is present in those terms. However, signals are often not so nicely isolated and co-channel and adjacent channel signals cannot be perfectly separated leading to significant interference when compared to the isolated signals used in this work. Works that can identify the number of signals present as well as characteristics of all signals present would be invaluable to regulatory agencies in order to help protect vested interests as well as fairness in spectrum usage. Datasets and test cases where all of these parameters can be adjusted would be powerful tools when it comes to characterizing the performance region of any system hoping to operate in the real world under a regulated system. By continuing to expand the available systems designed and geared toward metadata tracking, more intricate data problems like these can better be studied.

Transfer Learning across radios. The receiver in a RFML system introduces a fair share of distortions upon the observation. As radios are replaced for newer models, swapped out for cheaper devices, and changed on a whim, the understanding that a RFML model that went through fine tuning on the original radio might see performance loss overall on the new system should be expected. Being able to consider and distribute such models to any other transceiver is a study in itself. The use of a model fine tuned for one radio and then used on another is not well understood. Examining how different radios' observations are perceived by the model can allow for transfer learning to leverage minimal datasets to overcome the receiver difference.

Evaluation of non-Gaussian background environments with additive superimposed synthetic signals. The augmentation within this work focused on what can be done to an observation once it's been collected or created, but it is important to note that this isn't the only style of augmentation. For real-world observations with wide-bandwidth observers, there might be hundreds to thousands of signals present in an observation making up the full EME background of the problem space. In order to better understand how an approach would function, if at all, in these congested spaces, a more reasonable approach is to perform background captures that provide the realistic EME for the problem and enabling a synthetic additive placement of the application in question. Creating datasets that use this approach, it enables observations when no physical transmission system is available to test with. Exploring the characteristics necessary to reliably perform synthetic drop-in of a signal in such a manner and the effects when performing it in the real-world would be worth investigating. Designing a problem where the detection of a known and easy to generate signal can be constructed and then performing a collection of this can serve as the real-world test dataset. By then also collecting the spectral data when the signal is absent, this can allow for studies of how to best perform synthetic drop-in while still using the more complex and harder to simulate non-Gaussian background environments. The quality of different approaches can then be regressed and the relationship seen for different techniques in order to determine better methods of performing synthetic drop-in.

Development of HW/SW tools for high speed aggregate spectrum generation. While work done in this dissertation has started to create a system to this effect, having a variety of generation techniques would be ideal to help prevent any potentially unseen biases that might be contained in the approach. Having multiple techniques and tools for doing large dataset generation with hardware effects and software's precision and flexibility will undoubtedly allow for more competition and advancements in this field of study. A hope for this work is that the issue with dataset generation within RFML is sufficiently motivated, such that different groups might see it as worthwhile to create dataset generation platforms that will allow for larger and more diverse datasets while simplifying the ability to retain useful metadata.

Study individual augmentation effects on performance and quality. By separating the applied augmentations and training models based on their independent contributions to learning the understanding of the utility of each augmentation can better be studied. Through studying the augmentations in this manner, the overall quality of an augmented dataset might be improved over the results shown in this work as more directed intent can go into creating augmented data rather than just knowing that augmented data is desired. Studying individual augmentation techniques and comparing their relative qualities on a known good test set will potentially lead to better understanding how, and how much augmentation is needed to improve a model's generalization capabilities.

Bibliography

[1] L. J. Wong, W. H. Clark, IV, B. Flowers, R. M. Buehrer, W. C. Headley, and A. J. Michaels, "An RFML Ecosystem: Considerations for the Application of Deep Learning to Spectrum Situational Awareness," *IEEE Open Journal of the Communications Society*, vol. 2, pp. 2243–2264, 2021.

[2] W. H. Clark, IV, and A. J. Michaels, "Quantifying Dataset Quality in Radio Frequency Machine Learning," in *MILCOM 2021 Track 1 - Waveforms and Signal Processing (MILCOM 2021 Track 1)*, San DIego, USA, Nov. 2021.

[3] W. Clark, W. Headley, L. Wong, A. Michaels, and R. Buehrer, personal communication.

[4] S. C. Forouzan, Behrouz A.and Fegan, *Data communications and networking*, 4th ed. New York: McGraw-Hill Higher Education, 2007.

[5] G. R. M. Garratt, *The Early History of Radio: from Faraday to Marconi*, ser. History of Technology. Institution of Engineering and Technology, 1994. [Online]. Available: https://digital-library.theiet.org/content/books/ht/pbht020e

[6] "The Wireless Industry - Industry Data," CTIA, May 14, 2022. [Online]. Available: https://www.ctia.org/the-wireless-industry/infographics-library

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

[8] "The Radio Frequency Spectrum + Machine Learning = A New Wave in Radio Technology," Aug 2017. [Online]. Available: https://www.darpa.mil/news-events/2017-08-11a

[9] J. Mitola, "Cognitive radio," Ph.D. dissertation, Royal Institute of Technology (KTH), 2000.

[10] S. Blust, *Software Based Radio*. John Wiley & Sons, Ltd, ch. 1, pp. 1–22. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/0470846003.ch1

[11] M. Shimura, *A Mixed-Type Non-Parametric Learning Machine Without a Teacher*. Plenum Press, p. 42–55.

[12] P. Tilghman. (2017) DARPA Spectrum Collaboration Challenge. [Accessed: 24-November-2019]. [Online]. Available: https://www.darpa.mil/program/spectrum-collaboration-challenge

[13] L. Berlemann and S. Mangold, "Cognitive Radio and Dynamic Spectrum Access," Chichester, UK, 2009. [Online]. Available: https://onlinelibrary.wiley.com/doi/book/10.1002/9780470754429

[14] "Joint Chiefs of Staff, Joint Publication 3-85: "Joint Electromagnetic Spectrum Operations," 22 May 2020."

[15] Y. LeCun and Y. Bengio, *Convolutional Networks for Images, Speech, and Time Series*. Cambridge, MA, USA: MIT Press, 1998, p. 255–258.

[16] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, Oct 1986. [Online]. Available: https://doi.org/10.1038/323533a0

[17] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[18] P. Virtue, S. X. Yu, and M. Lustig, "Better than real: Complex-valued neural nets for MRI fingerprinting," in *2017 IEEE International Conference on Image Processing (ICIP)*, 2017, pp. 3953–3957.

[19] C. N. Brown, E. Mattei, and A. Draganov, "ChaRRNets: Channel Robust Representation Networks for RF Fingerprinting," 2021. [Online]. Available: https://arxiv.org/abs/2105.03568

[20] T. O'Shea and N. West, "Radio Machine Learning Dataset Generation with GNU Radio," *Proceedings of the GNU Radio Conference*, vol. 1, no. 1, 2016. [Online]. Available: https://pubs.gnuradio.org/index.php/grcon/article/view/11

[21] W. H. Clark, IV, S. Hauser, W. C. Headley, and A. J. Michaels, "Training Data Augmentation for Deep Learning Radio Frequency Systems," *The Journal of Defense Modeling and Simulation*, 2021. [Online]. Available: https://doi.org/10.1177/1548512921991245

[22] S. C. Hauser, W. C. Headley, and A. J. Michaels, "Signal detection effects on deep neural networks utilizing raw IQ for modulation classification," in *MILCOM 2017 - 2017 IEEE Military Communications Conference (MILCOM)*, Oct 2017, pp. 121–127.

[23] A. Paszke et al, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[24] M. Abadi et al, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[25] E. Frank, M. A. Hall, and I. H. Witten, "The WEKA Workbench," in *Data Mining: Practical Machine Learning Tools and Techniques*, 4th ed. Morgan Kaufmann, 2016.

[26] W. H. Clark and A. J. Michaels, "Quantifying and extrapolating data needs in radio frequency machine learning," 2022. [Online]. Available: https://arxiv.org/abs/2205.03703

[27] M. R. Williamson *et al*, "Multi-antenna pre-processing for improved rfml in congested spectral environments," in *2021 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, 2021, pp. 288–295.

[28] W. H. Clark, IV et al, "Developing RFML intuition: An automatic modulation classification architecture case study," in *MILCOM 2019 Track 5 - Big Data and Machine Learning (MILCOM 2019 Track 5)*, Norfolk, USA, Nov. 2019.

[29] M. O. Moore, W. H. Clark IV, R. M. Beuhrer, and W. C. Headley, "When is Enough Enough? "Just Enough" Decision Making with Recurrent Neural Networks for Radio Frequency Machine Learning," in *2020 IEEE 39th International Performance Computing and Communications Conference (IPCCC) (IEEE IPCCC 2020)*.

[30] W. Headley, W. Clark IV, and L. Wong, "General Dynamics (TREX) CogRF Phase III Final Review," Virginia Tech, Tech. Rep., 2018.

[31] W. Clark IV and Z. Leffke, "Cyborg Phase II Final Report," Virginia Tech, Tech. Rep., 2019.

[32] W. H. Clark IV, "Blind Comprehension of Waveforms Through Statistical Observations," M.S. thesis, ECE, Virginia Tech, Blacksburg, VA, 2015.

[33] W. H. Clark, IV, "Efficient waveform spectrum aggregation for algorithm verification and validation," in *GNU Radio Conference 2016*. https://github.com/gr-vt/gr-signal_exciter, 2016.

[34] "SignalEye AI Software for Automated Signal Classification - General Dynamics." [Online]. Available: https://gdmissionsystems.com/products/electronic-warfare/signaleye

[35] J. Schmidhuber, "Deep learning in neural networks: An overview," *CoRR*, vol. abs/1404.7828, 2014. [Online]. Available: http://arxiv.org/abs/1404.7828

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

[37] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *CoRR*, vol. abs/1502.03167, 2015. [Online]. Available: http://arxiv.org/abs/1502.03167

[38] Y. Gal, "Uncertainty in Deep Learning," Ph.D. dissertation, University of Cambridge, September 2016.

[39] Y. M. Asano, C. Rupprecht, A. Zisserman, and A. Vedaldi, "PASS: An ImageNet replacement for self-supervised pretraining without humans," 2021. [Online]. Available: https://arxiv.org/abs/2109.13228

[40] F. Zhuang et al, "A comprehensive survey on transfer learning," *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, 2021.

[41] L. J. Wong and A. J. Michaels, "Transfer Learning for Radio Frequency Machine Learning: A Taxonomy and Survey," *Sensors*, vol. 22, no. 4, 2022. [Online]. Available: https://www.mdpi.com/1424-8220/22/4/1416

[42] M. O. R. Prates, P. H. Avelar, and L. C. Lamb, "Assessing gender bias in machine translation: A case study with Google Translate," vol. 32, no. 10, pp. 6363–6381. [Online]. Available: https://doi.org/10.1007/s00521-019-04144-6

[43] K. Crawford and T. Paglen. (2019, Sept) Excavating AI: The Politics of Training Sets for Machine Learning. [Online]. Available: https://excavating.ai

[44] S. U. Noble, *Algorithms of Oppression : How Search Engines Reinforce Racism*. New York University Press, 2018, *ProQuest Ebook Central*. [Online]. Available: https://ebookcentral.proquest.com/lib/vt/detail.action?docID=4834260

[45] H. Chen, J. Chen, and J. Ding, "Data evaluation and enhancement for quality improvement of machine learning," *IEEE Transactions on Reliability*, vol. 70, no. 2, pp. 831–847, June 2021.

[46] M. E. Morocho-Cayamcela, H. Lee, and W. Lim, "Machine learning for 5G/B5G mobile and wireless communications: Potential, limitations, and future directions," *IEEE Access*, vol. 7, pp. 137 184–137 206, 2019.

[47] T. J. O'Shea, T. Roy, and T. Erpek, "Spectral detection and localization of radio events with learned convolutional neural features," in *2017 25th European Signal Processing Conference (EUSIPCO)*, 2017, pp. 331–335.

[48] T. O'Shea, T. Roy, and T. C. Clancy, "Learning robust general radio signal detection using computer vision methods," in *2017 51st Asilomar Conference on Signals, Systems, and Computers*, 2017, pp. 829–832.

[49] Z. Ye, A. Gilman, Q. Peng, K. Levick, P. Cosman, and L. Milstein, "Comparison of neural network architectures for spectrum sensing," in *2019 IEEE Globecom Workshops (GC Wkshps)*, 2019, pp. 1–6.

[50] Z. Ye, Q. Peng, K. Levick, H. Rong, A. Gilman, P. Cosman, and L. Milstein, "A neural network detector for spectrum sensing under uncertainties," *arXiv preprint arXiv:1907.07326*, 2019.

[51] Q. Peng, A. Gilman, N. Vasconcelos, P. C. Cosman, and L. B. Milstein, "Robust deep sensing through transfer learning in cognitive radio," *IEEE Wireless Communications Letters*, vol. 9, no. 1, pp. 38–41, 2020.

[52] P. D. White, R. M. Buehrer, and W. C. Headley, "FHSS signal separation using constrained clustering," in *IEEE Military Communications Conference (MILCOM)*, 2019, pp. 159–164.

[53] Q. Cheng, Z. Shi, D. N. Nguyen, and E. Dutkiewicz, "Non-cooperative OFDM Spectrum Sensing Using Deep Learning," in *2020 International Conference on Computing, Networking and Communications (ICNC)*, Feb 2020, pp. 704–708.

[54] X. Meng, H. Inaltekin, and B. Krongold, "End-to-end deep learning-based compressive spectrum sensing in cognitive radio networks," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, June 2020, pp. 1–6.

[55] Army Rapid Capabilities Office (RCO). (2018) Army signal classification challenge. [Accessed: 24-November-2019]. [Online]. Available: https://www.challenge.gov/challenge/army-signal-classification-challenge/

[56] B. Urrego. (2018) Grcon18 - army signal classification challenge. GNU Radio. [Online]. Available: https://www.gnuradio.org/grcon/grcon18/presentations/army_signal_classification_challenge/4-Bill_Urrego-Army_Signal_Classification_Challenge.pdf

[57] Paul Tilghman. DARPA Radio Frequency Machine Learning Systems. [Accessed: 24-November-2019]. [Online]. Available: https://www.darpa.mil/program/radio-frequency-machine-learning-systems

[58] P. Kolb, "Securing Compartmented Information with Smart Radio Systems (SCISRS)." [Online]. Available: https://www.iarpa.gov/index.php/research-programs/scisrs

[59] D. Deasy, "Department of Defense Statement on Mid-Band Spectrum," Aug 2022. [Online]. Available: https://www.defense.gov/News/Speeches/Speech/Article/2307288/department-of-defense-statement-on-mid-band-spectrum/

[60] "SPECTRUM REALLOCATION FINAL REPORT - Response to Title VI - Omnibus Budget Reconciliation Act of 1993," Aug 2022. [Online]. Available: https://www.ntia.doc.gov/legacy/osmhome/EPS/openness/final.html

[61] B. Clark, "The fall and rise of Russian Electronic Warfare," Jul 2022. [Online]. Available: https://spectrum.ieee.org/the-fall-and-rise-of-russian-electronic-warfare

[62] "Introduction To The Privacy Act - U.S. Department of Defense." [Online]. Available: https://dpcld.defense.gov/Portals/49/Documents/Privacy/2011%20DPCLO_Intro_Privacy_Act.pdf

[63] O. Dobre, A. Abdi, Y. Bar-Ness, and W. Su, "Survey of automatic modulation classification techniques: classical approaches and new trends," *Communications, IET*, vol. 1, no. 2, pp. 137–156, April 2007.

[64] 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. 33 544–33 555, 2019.

[65] K. Chowdhury, S. Ioannidis, and T. Melodia, "Deep Learning for RF Signal Classification and Fingerprinting," *IEEE Military Communications Conference (MILCOM)*, 2019.

[66] L. J. Wong, W. C. Headley, S. Andrews, R. M. Gerdes, and A. J. Michaels, "Clustering Learned CNN Features from Raw I/Q Data for Emitter Identification," in *MILCOM 2018 - 2018 IEEE Military Communications Conference (MILCOM)*, 2018, pp. 26–33.

[67] J. M. McGinthy, L. J. Wong, and A. J. Michaels, "Groundwork for neural network-based specific emitter identification authentication for iot," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 6429–6440, Aug 2019.

[68] A. C. Polak, S. Dolatshahi, and D. L. Goeckel, "Identifying wireless users via transmitter imperfections," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 7, pp. 1469–1479, 2011.

[69] A. Zubow, S. Bayhan, P. Gawłowicz, and F. Dressler, "Deeptxfinder: Multiple transmitter localization by deep learning in crowdsourced spectrum sensing," in *2020 29th International Conference on Computer Communications and Networks (ICCCN)*, Aug 2020, pp. 1–8.

[70] J. Yu, H. M. Saad, and R. M. Buehrer, "Centimeter-level indoor localization using channel state information with recurrent neural networks," in *2020 IEEE/ION Position, Location and Navigation Symposium (PLANS)*, 2020, pp. 1317–1323.

[71] R. Elbakly, H. Aly, and M. Youssef, "TrueStory: Accurate and robust RF-based floor estimation for challenging indoor environments," *IEEE Sensors Journal*, vol. 18, no. 24, pp. 10 115–10 124, 2018.

[72] A. Bacak and H. Çelebi, "Practical considerations for RSS RF fingerprinting based indoor localization systems," in *2014 22nd Signal Processing and Communications Applications Conference (SIU)*, 2014, pp. 497–500.

[73] M. I. AlHajri, N. T. Ali, and R. M. Shubair, "Indoor Localization for IoT Using Adaptive Feature Selection: A Cascaded Machine Learning Approach," vol. 18, no. 11, pp. 2306–2310.

[74] N. Tandiya, A. Jauhar, V. Marojevic, and J. H. Reed, "Deep Predictive Coding Neural Network for RF Anomaly Detection in Wireless Networks," in *2018 IEEE International Conference on Communications Workshops (ICC Workshops)*, pp. 1–6.

[75] T. J. O'Shea, T. C. Clancy, and R. W. McGwier, "Recurrent neural radio anomaly detection," 2016. [Online]. Available: https://arxiv.org/abs/1611.00301

[76] P. Tilghman, "Will rule the airwaves: A DARPA grand challenge seeks autonomous radios to manage the wireless spectrum," *IEEE Spectrum*, vol. 56, no. 6, pp. 28–33, 2019.

[77] T. J. O'Shea, T. Roy, N. West, and B. C. Hilburn, "Physical layer communications system design over-the-air using adversarial networks," in *2018 26th European Signal Processing Conference (EUSIPCO)*, 2018, pp. 529–532.

[78] T. J. O'Shea, T. Roy, and N. West, "Approximating the Void: Learning Stochastic Channel Models from Observation with Variational Generative Adversarial Networks," in *2019 International Conference on Computing, Networking and Communications (ICNC)*, Feb 2019, pp. 681–686.

[79] K. Davaslioglu and Y. E. Sagduyu, "Generative adversarial learning for spectrum sensing," in *2018 IEEE International Conference on Communications (ICC)*, May 2018, pp. 1–6.

[80] M. Ezuma, F. Erden, C. Kumar Anjinappa, O. Ozdemir, and I. Guvenc, "Detection and classification of UAVs using RF fingerprints in the presence of Wi-Fi and Bluetooth interference," *IEEE Open Journal of the Communications Society*, vol. 1, pp. 60–76, 2020.

[81] A. P. Arechiga and A. J. Michaels, "The effect of weight errors on neural networks," in *IEEE Annual Computing and Communication Workshop and Conference (CCWC)*, Jan 2018, pp. 190–196.

[82] ——, "The robustness of modern deep learning architectures against single event upset errors," in *IEEE High Performance extreme Computing Conference (HPEC)*, Sep. 2018, pp. 1–6.

[83] G. Li, S. K. S. Hari, M. Sullivan, T. Tsai, K. Pattabiraman, J. Emer, and S. W. Keckler, "Understanding error propagation in deep learning neural network (DNN) accelerators and applications," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis.*

New York, NY, USA: Association for Computing Machinery, 2017. [Online]. Available: https://doi.org/10.1145/3126908.3126964

[84] E. Altland, J. Castellanos, J. Detwiler, P. Fermin, R. Ferrá, C. Kelly, C. Latoski, T. Ma, T. Maher, J. M. Kuzin, A. Mohammadian, A. S. Abdalla, W. C. Headley, and A. J. Michaels, "Quantifying degradations of convolutional neural networks in space environments," in *IEEE Cognitive Communications for Aerospace Applications Workshop (CCAAW)*, June 2019, pp. 1–7.

[85] B. Reagen, U. Gupta, L. Pentecost, P. Whatmough, S. K. Lee, N. Mulholland, D. Brooks, and G. Wei, "Ares: A framework for quantifying the resilience of deep neural networks," in *ACM/ESDA/IEEE Design Automation Conference (DAC)*, June 2018, pp. 1–6.

[86] Z. Yan, Y. Shi, W. Li-ao, M. Hashimoto, X. Zhou, and C. Zhuo, "When single event upset meets deep neural networks: Observations, explorations, and remedies," *ArXiv*, vol. abs/1909.04697, 2019.

[87] M. A. Neggaz, I. Alouani, P. R. Lorenzo, and S. Niar, "A reliability study on CNNs for critical embedded systems," in *IEEE International Conference on Computer Design (ICCD)*, Oct 2018, pp. 476–479.

[88] E. Ozen and A. Orailoglu, "Sanity-Check: Boosting the reliability of safety-critical deep neural network applications," in *IEEE Asian Test Symposium (ATS)*, Dec 2019, pp. 7–75.

[89] D. Roy, T. Mukherjee, and M. Chatterjee, "Machine Learning in Adversarial RF Environments," vol. 57, no. 5, pp. 82–87.

[90] National Instruments, "Enabling AI research for 5G networks with NI SDR," *Whitepaper*, 2019. [Online]. Available: https://www.ni.com/en-us/innovations/white-papers/19/enabling-ai-research-for-5g-with-sdr-platform.html

[91] E. Balevi and R. D. Gitlin, "Unsupervised machine learning in 5G networks for low latency communications," in *IEEE International Performance Computing and Communications Conference (IPCCC)*, Dec 2017, pp. 1–2.

[92] T. Ma, F. Hu, and M. Ma, "Fast and efficient physical layer authentication for 5G HetNet handover," in *International Telecommunication Networks and Applications Conference (ITNAC)*, Nov 2017, pp. 1–3.

[93] V. P. Kafle, Y. Fukushima, P. Martinez-Julia, and T. Miyazawa, "Consideration on automation of 5G network slicing with machine learning," in *ITU Kaleidoscope: Machine Learning for a 5G Future (ITU K)*, Nov 2018, pp. 1–8.

[94] M. Alhajri, N. Alsindi, N. Ali, and R. Shubair, "Classification of indoor environments based on spatial correlation of rf channel fingerprints," in *Antennas and Propagation (APSURSI), 2016 IEEE International Symposium on*, 2016, pp. 1447–1448.

[95] B. Chatterjee, D. Das, S. Maity, and S. Sen, "RF-PUF: Enhancing IoT security through authentication of wireless nodes using in-situ machine learning," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 388–398, Feb 2019.

[96] A. Guerra-Manzanares, H. Bahsi, and S. Nõmm, "Hybrid feature selection models for machine learning based botnet detection in IoT networks," in *International Conference on Cyberworlds (CW)*, Oct 2019, pp. 324–327.

[97] Y. Liu, Y. J. Morton, and Y. Jiao, "Application of machine learning to the characterization of GPS L1 ionospheric amplitude scintillation," in *IEEE/ION Position, Location and Navigation Symposium (PLANS)*, April 2018, pp. 1159–1166.

[98] G. Liu, R. Zhang, C. Wang, and L. Liu, "Synchronization-free GPS spoofing detection with crowdsourced air traffic control data," in *IEEE International Conference on Mobile Data Management (MDM)*, June 2019, pp. 260–268.

[99] D. I. Moody, D. A. Smith, T. E. Light, M. J. Heavner, T. D. Hamlin, and D. M. Suszcynsky, "Signal classification of satellite-based recordings of radio-frequency (RF) transients using data-adaptive dictionaries," in *Asilomar Conference on Signals, Systems and Computers*, Nov 2013, pp. 1291–1295.

[100] K. Sankhe, M. Belgiovine, F. Zhou, S. Riyaz, S. Ioannidis, and K. Chowdhury, "Oracle: Optimized radio classification through convolutional neural networks," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2019, pp. 370–378.

[101] D. Adesina, J. Bassey, and L. Qian, "Practical radio frequency learning for future wireless communication systems," in *IEEE Military Communications Conference (MILCOM)*, Nov 2019, pp. 311–317.

[102] D. M. Le Vine, "Review of measurements of the RF spectrum of Radiation from lightning," Sep 2013. [Online]. Available: https://ntrs.nasa.gov/citations/19870001225

[103] L. H. Pederick and M. A. Cervera, "Modeling the interference environment in the HF band," *Radio Science*, vol. 51, no. 2, pp. 82–90, 2016. [Online]. Available: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1002/2015RS005856

[104] A. N. Mody *et al.*, "Recent advances in cognitive communications," *IEEE Communications Magazine*, vol. 45, no. 10, pp. 54–61.

[105] NVidia, "DGX-2 datasheet," 2018. [Online]. Available: https://www.nvidia.com/content/dam/en-zz/Solutions/Data-Center/dgx-2/dgx-2-print-datasheet-738070-nvidia-a4-web-uk.pdf

[106] F. Altiparmak, F. C. Akyon, E. Ozmen, F. Cogun, and A. Bayri, "Towards cognitive sensing: Radar function classification using multitask learning," in *Signal Processing and Communications Applications Conference (SIU)*, April 2019, pp. 1–4.

[107] R. M. Bowen, F. Sahin, A. Radomski, and D. Sarosky, "Embedded one-class classification on RF generator using mixture of Gaussians," in *IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Oct 2014, pp. 2657–2662.

[108] V. Camus, L. Mei, C. Enz, and M. Verhelst, "Review and benchmarking of precision-scalable multiply-accumulate unit architectures for embedded neural-network processing," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 9, no. 4, pp. 697–711, Dec 2019.

[109] S. Fox, J. Faraone, D. Boland, K. Vissers, and P. H. W. Leong, "Training deep neural networks in low-precision with high accuracy using FPGAs," in *International Conference on Field-Programmable Technology (ICFPT)*, Dec 2019, pp. 1–9.

[110] I. Colbert, K. Kreutz-Delgado, and S. Das, "AX-DBN: An approximate computing framework for the design of low-power discriminative deep belief networks," in *International Joint Conference on Neural Networks (IJCNN)*, July 2019, pp. 1–9.

[111] Y. Gwon, S. Dastangoo, C. Fossa, and H. T. Kung, "Fast online learning of antijamming and jamming strategies," in *IEEE Global Communications Conference (GLOBECOM)*, Dec 2015, pp. 1–6.

[112] L. H. Nguyen and T. D. Tran, "Separation of radio-frequency interference from SAR signals via dictionary learning," in *IEEE Radar Conference (RadarConf18)*, April 2018, pp. 0908–0913.

[113] M. A. Hannan, M. M. Hoque, A. Hussain, Y. Yusof, and P. J. Ker, "State-of-the-art and energy management system of lithium-ion batteries in electric vehicle applications: Issues and recommendations," *IEEE Access*, vol. 6, pp. 19 362–19 378, 2018.

[114] K. Vinsen, S. Foster, and R. Dodson, "Using machine learning for the detection of radio frequency interference," in *URSI Asia-Pacific Radio Science Conference (AP-RASC)*, March 2019, pp. 1–4.

[115] E. Strubell, A. Ganesh, and A. McCallum, "Energy and policy considerations for deep learning in NLP," *arXiv preprint arXiv:1906.02243*, 2019.

[116] P. Nakkiran et al, "Deep Double Descent: Where Bigger Models and More Data Hurt," *CoRR*, vol. abs/1912.02292, 2019. [Online]. Available: http://arxiv.org/abs/1912.02292

[117] USRP B210 USB Software Defined Radio (SDR). [Accessed: 04-September-2022]. [Online]. Available: https://www.ettus.com/all-products/ub210-kit/

[118] E. Blossom. (2004) GNU Radio: Tools for Exploring the Radio Frequency Spectrum. Linux Journal. [Online]. Available: https://www.linuxjournal.com/article/7319

[119] J. Gaeddart, "Liquid DSP," (Date last accessed 7-January-2020). [Online]. Available: https://github.com/jgaeddert/liquid-dsp/

[120] Redhawk. [Accessed: 04-September-2022]. [Online]. Available: https://redhawksdr.org/

[121] The GNU Radio Foundation, Inc. (2018) The Signal Metadata Format (SigMF). [Online]. Available: https://sigmf.org

[122] E. Hoffer, T. Ben-Nun, I. Hubara, N. Giladi, T. Hoefler, and D. Soudry, "Augment your batch: Improving generalization through instance repetition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[123] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2818–2826.

[124] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=r1Ddp1-Rb

[125] V. Verma, A. Lamb, C. Beckham, A. Najafi, I. Mitliagkas, D. Lopez-Paz, and Y. Bengio, "Manifold mixup: Better representations by interpolating hidden states," in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 09–15 Jun 2019, pp. 6438–6447. [Online]. Available: https://proceedings.mlr.press/v97/verma19a.html

[126] G. Jocher *et al.*, "ultralytics/yolov5: v6.2 - YOLOv5 Classification Models, Apple M1, Reproducibility, ClearML and Deci.ai integrations," Aug. 2022. [Online]. Available: https://doi.org/10.5281/zenodo.7002879

[127] R. D. Miller, S. Kokalj-Filipovic, G. Vanhoy, and J. Morman, "Policy based synthesis: Data generation and augmentation methods for rf machine learning," in *2019 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2019, pp. 1–5.

[128] N. Jaipuria, X. Zhang, R. Bhasin, M. Arafa, P. Chakravarty, S. Shrivastava, S. Manglani, and V. N. Murali, "Deflating dataset bias using synthetic data augmentation," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2020, pp. 3344–3353.

[129] B. Lewis, O. DeGuchy, J. Sebastian, and J. Kaminski, "Realistic SAR data augmentation using machine learning techniques," in *Algorithms for Synthetic Aperture Radar Imagery XXVI*, E. Zelnio and F. D. Garber, Eds., vol. 10987, International Society for Optics and Photonics. SPIE, 2019, p. 109870C. [Online]. Available: https://doi.org/10.1117/12.2518452

[130] H. Le, M. Nguyen, and W. Q. Yan, "Machine learning with synthetic data - a new way to learn and classify the pictorial augmented reality markers in real-time," in *2020 35th International Conference on Image and Vision Computing New Zealand (IVCNZ)*, 2020, pp. 1–6.

[131] A. Bochkovskiy, C. Wang, and H. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," *CoRR*, vol. abs/2004.10934, 2020. [Online]. Available: https://arxiv.org/abs/2004.10934

[132] R. Baweja, "FPGA Implementation of a Pseudo-Random Aggregate Spectrum Generator for RF Hardware Test and Evaluation," M.S. thesis, ECE, Virginia Tech, Blacksburg, VA, 2020.

[133] A. Nandi and E. Azzouz, "Modulation recognition using artificial neural networks," *Signal Processing*, vol. 56, no. 2, pp. 165 – 175, 1997. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S016516849600165X

[134] Namjin Kim, N. Kehtarnavaz, M. B. Yeary, and S. Thornton, "DSP-based hierarchical neural network modulation signal classification," *IEEE Transactions on Neural Networks*, vol. 14, no. 5, pp. 1065–1071.

[135] A. Fehske, J. Gaeddert, and J. H. Reed, "A new approach to signal classification using spectral correlation and neural networks," in *First IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005.*, pp. 144–150.

[136] F. Ge, Q. Chen, Y. Wang, C. W. Bostian, T. W. Rondeau, and B. Le, "Cognitive Radio: From Spectrum Sharing to Adaptive Learning and Reconfiguration," in *2008 IEEE Aerospace Conference*, pp. 1–10.

[137] L. Bixio, M. Ottonello, H. Sallam, M. Raffetto, and C. S. Regazzoni, "Signal classification based on spectral redundancy and neural network ensembles," in *2009 4th International Conference on Cognitive Radio Oriented Wireless Networks and Communications*, pp. 1–6.

[138] T. C. Clancy and A. Khawar, "Security threats to signal classifiers using self-organizing maps," in *2009 4th International Conference on Cognitive Radio Oriented Wireless Networks and Communications*, pp. 1–6.

[139] M. M. Ramón, T. Atwood, S. Barbin, and C. G. Christodoulou, "Signal classification with an SVM-FFT approach for feature extraction in cognitive radio," in *2009 SBMO/IEEE MTT-S International Microwave and Optoelectronics Conference (IMOC)*, pp. 286–289.

[140] J. J. Popoola and R. van Olst, "Application of neural network for sensing primary radio signals in a cognitive radio environment," in *IEEE Africon '11*, pp. 1–6.

[141] Shan Kang, Naiwen Chen, Mi Yan, and Xiaoxiao Chen, "Detecting identity-spoof attack based on BP network in cognitive radio network," in *Proceedings of 2011 Cross Strait Quad-Regional Radio Science and Wireless Technology Conference*, vol. 2, pp. 1603–1606.

[142] D. Pu, Y. Shi, A. V. Ilyashenko, and A. M. Wyglinski, "Detecting Primary User Emulation Attack in Cognitive Radio Networks," in *2011 IEEE Global Telecommunications Conference - GLOBECOM 2011*, pp. 1–5.

[143] F. He, X. Xu, L. Zhou, and H. Man, "A learning based cognitive radio receiver," in *2011 - MILCOM 2011 Military Communications Conference*, pp. 7–12.

[144] M. M. T. Abdelreheem and M. O. Helmi, "Digital Modulation Classification through time and frequency domain features using Neural Networks," in *2012 IX International Symposium on Telecommunications (BIHTEL)*, pp. 1–5.

[145] S. Li, X. Wang, and J. Wang, "Manifold learning-based automatic signal identification in cognitive radio networks," *IET Communications*, vol. 6, no. 8, pp. 955–963.

[146] K. M. Thilina, K. W. Choi, N. Saquib, and E. Hossain, "Machine Learning Techniques for Cooperative Spectrum Sensing in Cognitive Radio Networks," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 11, pp. 2209–2221, 2013.

[147] J. J. Popoola and R. van Olst, "The performance evaluation of a spectrum sensing implementation using an automatic modulation classification detection method with a Universal Software Radio Peripheral," vol. 40, no. 6, pp. 2165 – 2173. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0957417412011712

[148] S. Kim and G. B. Giannakis, "Dynamic learning for cognitive radio sensing," in *2013 5th IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, pp. 388–391.

[149] A. Tsakmalis, S. Chatzinotas, and B. Ottersten, "Automatic Modulation Classification for adaptive Power Control in cognitive satellite communications," in *2014 7th Advanced Satellite Multimedia Systems Conference and the 13th Signal Processing for Space Communications Workshop (ASMS/SPSC)*, pp. 234–240.

[150] T. Chen, J. Liu, L. Xiao, and L. Huang, "Anti-jamming transmissions with learning in heterogenous cognitive radio networks," in *2015 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, pp. 293–298.

[151] G. J. Mendis, J. Wei, and A. Madanayake, "Deep learning-based automated modulation classification for cognitive radio," in *2016 IEEE International Conference on Communication Systems (ICCS)*, pp. 1–6.

[152] T. J. O'Shea, L. Pemula, D. Batra, and T. C. Clancy, "Radio transformer networks: Attention models for learning to synchronize in wireless systems," in *2016 50th Asilomar Conference on Signals, Systems and Computers*, pp. 662–666.

[153] T. J. O'Shea, S. Hitefield, and J. Corgan, "End-to-end radio traffic sequence recognition with recurrent neural networks," in *2016 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pp. 277–281.

[154] T. J. O'Shea, J. Corgan, and T. C. Clancy, "Convolutional Radio Modulation Recognition Networks."

[155] N. E. West, K. Harwell, and B. McCall, "DFT signal detection and channelization with a deep neural network modulation classifier," in *2017 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, pp. 1–3.

[156] N. E. West and T. O'Shea, "Deep architectures for modulation recognition," in *2017 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, March 2017, pp. 1–6.

[157] K. Karra, S. Kuzdeba, and J. Petersen, "Modulation recognition using hierarchical deep neural networks," in *2017 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, pp. 1–3.

[158] S. Peng, H. Jiang, H. Wang, H. Alwageed, and Y. Yao, "Modulation classification using convolutional Neural Network based deep learning model," in *2017 26th Wireless and Optical Communication Conference (WOCC)*, pp. 1–5.

[159] K. P. K. Reddy, Y. Yeleswarapu, and S. J. Darak, "Performance evaluation of cumulant feature based automatic modulation classifier on USRP testbed," in *2017 9th International Conference on Communication Systems and Networks (COMSNETS)*, pp. 393–394.

[160] T. Nawaz, L. Marcenaro, and C. S. Regazzoni, "Stealthy jammer detection algorithm for wide-band radios: A physical layer approach," in *2017 IEEE 13th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pp. 79–83.

[161] T. Nawaz, L. Marcenaro, and C. S. Regazzoni, "Cyclostationary-based jammer detection for wideband radios using compressed sensing and artificial neural network." vol. 13, no. 12, p. 1. [Online]. Available: http://login.ezproxy.lib.vt.edu/login?url=http://search.ebscohost.com/login.aspx?direct=true&db=iih&AN=127039926&scope=site

[162] A. B. Ambaw, M. Bari, and M. Doroslovački, "A case for stacked autoencoder based order recognition of continuous-phase FSK," in *2017 51st Annual Conference on Information Sciences and Systems (CISS)*, pp. 1–6.

[163] A. Ali, F. Yangyu, and S. Liu, "Automatic modulation classification of digital modulation signals with stacked autoencoders," vol. 71, pp. 108 – 116. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1051200417302087

[164] D. Hong, Z. Zhang, and X. Xu, "Automatic modulation classification using recurrent neural networks," in *2017 3rd IEEE International Conference on Computer and Communications (ICCC)*, pp. 695–700.

[165] R. G. Yelalwar and Y. Ravinder, "Artificial Neural Network Based Approach for Spectrum Sensing in Cognitive Radio," in *2018 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, pp. 1–5.

[166] S. M. Hiremath, S. Deshmukh, R. Rakesh, and S. Kumar Patra, "Blind Identification of Radio Access Techniques Based on Time-Frequency Analysis and Convolutional Neural Network," in *TENCON 2018 - 2018 IEEE Region 10 Conference*, pp. 1163–1167.

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

[168] 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," vol. 6, pp. 18 484–18 501.

[169] Y. Wu, X. Li, and J. Fang, "A Deep Learning Approach for Modulation Recognition via Exploiting Temporal Correlations," in *2018 IEEE 19th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pp. 1–5.

[170] Z. Li, R. Liu, X. Lin, and H. Shi, "Detection of Frequency-Hopping Signals Based on Deep Neural Networks," in *2018 IEEE 3rd International Conference on Communication and Information Systems (ICCIS)*, pp. 49–52.

[171] A. Subekti, H. F. Pardede, R. Sustika, and Suyoto, "Spectrum Sensing for Cognitive Radio using Deep Autoencoder Neural Network and SVM," in *2018 International Conference on Radar, Antenna, Microwave, Electronics, and Telecommunications (ICRAMET)*, pp. 81–85.

[172] S. K. Jayaweera and M. A. Aref, "Cognitive Engine Design for Spectrum Situational Awareness and Signals Intelligence," in *2018 21st International Symposium on Wireless Personal Multimedia Communications (WPMC)*, pp. 478–483.

[173] T. J. O'Shea, T. Roy, and T. C. Clancy, "Over-the-air deep learning based radio signal classification," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 168–179, Feb 2018.

[174] Y. ZHANG, T. LIU, L. ZHANG, and K. WANG, "A Deep Learning approach for Modulation Recognition," in *2018 IEEE 23rd International Conference on Digital Signal Processing (DSP)*, pp. 1–5.

[175] Y. Sang and L. A. Li, "Application of novel architectures for Modulation Recognition," in *2018 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*, pp. 159–162.

[176] G. Vanhoy, N. Thurston, A. Burger, J. Breckenridge, and T. Bose, "Hierarchical Modulation Classification Using Deep Learning," in *MILCOM 2018 - 2018 IEEE Military Communications Conference (MILCOM)*, pp. 20–25.

[177] S. A. Shapero, A. B. Dill, and B. O. Odelowo, "Identifying Agile Waveforms with Neural Networks," in *2018 21st International Conference on Information Fusion (FUSION)*, pp. 745–752.

[178] K. Yashashwi, A. Sethi, and P. Chaporkar, "A Learnable Distortion Correction Module for Modulation Recognition," *IEEE Wireless Communications Letters*, vol. 8, no. 1, pp. 77–80.

[179] S. Peng, H. Jiang, H. Wang, H. Alwageed, Y. Zhou, M. M. Sebdani, and Y. Yao, "Modulation Classification Based on Signal Constellation Diagrams and Deep Learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 3, pp. 718–727.

[180] Y. Wang, M. Liu, J. Yang, and G. Gui, "Data-Driven Deep Learning for Automatic Modulation Recognition in Cognitive Radios," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 4, pp. 4074–4077.

[181] H. Liu, X. Zhu, and T. Fujii, "Cyclostationary based full-duplex spectrum sensing using adversarial training for convolutional neural networks," in *2019 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, pp. 369–374.

[182] S. Zheng, P. Qi, S. Chen, and X. Yang, "Fusion Methods for CNN-Based Automatic Modulation Classification," *IEEE Access*, vol. 7, pp. 66 496–66 504.

[183] P. Wang and M. Vindiola, "Data augmentation for blind signal classification," in *MILCOM 2019 Track 5 - Big Data and Machine Learning (MILCOM 2019 Track 5)*, Norfolk, USA, Nov. 2019.

[184] K. Merchant and B. Nousain, "Toward Receiver-Agnostic RF Fingerprint Verification," in *2019 IEEE Globecom Workshops (GC Wkshps)*, pp. 1–6.

[185] Q. Cai, S. Chen, X. Li, N. Hu, H. He, Y. Yao, and J. Mitola, "An integrated incremental self-organizing map and hierarchical neural network approach for cognitive radio learning," in *The 2010 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–6.

[186] D. Pu and A. M. Wyglinski, "Primary user emulation detection using frequency domain action recognition," in *Proceedings of 2011 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*, pp. 791–796.

[187] J. Torres-Sospedra, R. Montoliu, A. Martínez-Usó, J. P. Avariento, T. J. Arnau, M. Benedito-Bordonau, and J. Huerta, "UJIIndoorLoc: A new multi-building and multi-floor database for WLAN fingerprint-based indoor localization problems," in *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2014, pp. 261–270.

[188] K. A. A. Kumar, "SoC implementation of a modulation classification module for cognitive radios," in *2016 International Conference on Communication Systems and Networks (ComNet)*, pp. 87–92.

[189] M. Schmidt, D. Block, and U. Meier, "Wireless interference identification with convolutional neural networks," in *2017 IEEE 15th International Conference on Industrial Informatics (INDIN)*, pp. 180–185.

[190] M. R. Vyas, D. K. Patel, and M. Lopez-Benitez, "Artificial neural network based hybrid spectrum sensing scheme for cognitive radio," in *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pp. 1–7.

[191] N. Bitar, S. Muhammad, and H. H. Refai, "Wireless technology identification using deep Convolutional Neural Networks," in *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pp. 1–6.

[192] S. S. Fernandes, M. R. Makiuchi, M. V. Lamar, and J. L. Bordim, "An Adaptive Recurrent Neural Network Model Dedicated to Opportunistic Communication in Wireless Networks," in *2018 International Joint Conference on Neural Networks (IJCNN)*, pp. 01–08.

[193] E. Testi, E. Favarelli, and A. Giorgetti, "Machine Learning for User Traffic Classification in Wireless Systems," in *2018 26th European Signal Processing Conference (EUSIPCO)*, pp. 2040–2044.

[194] S. Yi, H. Wang, W. Xue, X. Fan, L. Wang, J. Tian, and R. Matsukura, "Interference Source Identification for IEEE 802.15.4 wireless Sensor Networks Using Deep Learning," in *2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, pp. 1–7.

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

[196] S. Mohammed, R. El Abdessamad, R. Saadane, and K. A. Hatim, "Performance Evaluation of Spectrum Sensing Implementation using Artificial Neural Networks and Energy Detection Method," in *2018 International Conference on Electronics, Control, Optimization and Computer Science (ICECOCS)*, pp. 1–6.

[197] G. J. Mendis, J. Wei, and A. Madanayake, "Deep Learning based Radio-Signal Identification with Hardware Design," *IEEE Transactions on Aerospace and Electronic Systems*.

[198] S. M. Hiremath, S. Behura, S. Kedia, S. Deshmukh, and S. K. Patra, "Deep Learning-Based Modulation Classification Using Time and Stockwell Domain Channeling," in *2019 National Conference on Communications (NCC)*, pp. 1–6.

[199] S. S. Chawathe, "Indoor-location classification using RF signatures," in *IEEE International Symposium on Network Computing and Applications (NCA)*, 2019, pp. 1–4.

[200] K. Merchant and B. Nousain, "Enhanced RF Fingerprinting for IoT Devices with Recurrent Neural Networks," in *MILCOM 2019 - 2019 IEEE Military Communications Conference (MILCOM)*, pp. 590–597.

[201] P. Y. Simard, D. Steinkraus, and J. C. Platt, "Best practices for convolutional neural networks applied to visual document analysis," in *Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings.*, pp. 958–963.

[202] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., pp. 1097–1105.

[203] L. S. Yaeger, R. F. Lyon, and B. J. Webb, "Effective training of a neural network character classifier for word recognition," in *Advances in Neural Information Processing Systems 9*, M. C. Mozer, M. I. Jordan, and T. Petsche, Eds. MIT Press, pp. 807–816.

[204] B. Urrego. Grcon18 - army signal classification challenge. Youtube. [Online]. Available: https://www.youtube.com/watch?v=7yJvVi6yEaE

[205] B. Flowers and W. C. Headley, "Adversarial Radio Frequency Machine Learning (RFML) with PyTorch, MILCOM 2019 - 2019 IEEE Military Communications Conference (MILCOM)," 2019.

[206] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," San Diego, CA, United states, 2015.

[207] C. R. Harris et al, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020. [Online]. Available: https://doi.org/10.1038/s41586-020-2649-2

[208] P. Virtanen et al, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[209] A. Swami and B. Sadler, "Hierarchical digital modulation classification using cumulants," *Communications, IEEE Transactions on*, vol. 48, no. 3, pp. 416–429, Mar 2000.

[210] B. Flowers, R. M. Buehrer, and W. C. Headley, "Evaluating Adversarial Evasion Attacks in the Context of Wireless Communications," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 1102–1113.

[211] B. L. Welch, "The generalization of student's' problem when several different population variances are involved," *Biometrika*, vol. 34, no. 1/2, pp. 28–35, 1947.

[212] C. V. Nguyen, T. Hassner, C. Archambeau, and M. W. Seeger, "LEEP: A new measure to evaluate transferability of learned representations," *CoRR*, vol. abs/2002.12462, 2020. [Online]. Available: https://arxiv.org/abs/2002.12462

[213] A. Tran, C. Nguyen, and T. Hassner, "Transferability and hardness of supervised classification tasks," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 1395–1405.

[214] K. You, Y. Liu, J. Wang, and M. Long, "LogME: Practical Assessment of Pre-trained Models for Transfer Learning," in *Proceedings of the 38th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. Meila and T. Zhang, Eds., vol. 139. PMLR, 18–24 Jul 2021, pp. 12 133–12 143. [Online]. Available: https://proceedings.mlr.press/v139/you21b.html

[215] M. G. Kendall, "A new measure of rank correlation," *Biometrika*, vol. 30, no. 1/2, pp. 81–93, 1938. [Online]. Available: http://www.jstor.org/stable/2332226

[216] R. Schwarz. (2022) Understanding Pulse Signal Generation. [Accessed: 18-September-2022]. [Online]. Available: https://www.youtube.com/watch?v=0GPp6ncHaJw

[217] DARPAtv. (2019) Spectrum collaboration challenge championship event. [Accessed: 30-November-2019]. [Online]. Available: https://www.youtube.com/watch?reload=9&v=-a-N8MiMqz0

[218] C. N. Brown, E. Mattei, and A. Draganov, "ChaRRNets: Channel Robust Representation Networks for RF Fingerprinting," 2021. [Online]. Available: https://arxiv.org/abs/2105.03568