Lightly Supervised Machine Learning for Wireless Signals

Josh Sanz

Technical Report No. UCB/EECS-2024-227
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2024/EECS-2024-227.html

December 20, 2024

Acknowledgement

The work contained herein was funded by the UC Berkeley ML4Wireless center member companies, the DARPA Spectrum Collaboration Challenge, NSF grants AST-144078 and ECCS-1343398, and SpectrumX, the Spectrum Innovation Center, NSF Award 2132700. I also wish to thank my co-authors and members of the BLISS lab for their helpful discussions and ideas.

Lightly Supervised Machine Learning for Wireless Signals

by

Joshua Sanz

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy in Engineering

in

Electrical Engineering and Computer Science

in the

Graduate Division

of the

University of California, Berkeley

Fall 2024

Lightly Supervised Machine Learning for Wireless Signals

Abstract

Lightly Supervised Machine Learning for Wireless Signals

by

Joshua Sanz

Doctor of Philosophy in Engineering in Electrical Engineering and Computer Science

University of California, Berkeley

Professor Anant Sahai, Chair

Modern wireless communication systems face unprecedented challenges in managing finite spectrum resources while meeting growing demands for data and connectivity. This dissertation explores how machine learning techniques with reduced supervision requirements can address these challenges through three complementary approaches. First, I demonstrate that two radio agents with minimal shared assumptions can learn compatible modulation schemes through cooperative interaction, enabling communication without explicit protocol design. Careful experimentation, including simulation and implementation on software-defined radios, shows that while reduced supervision increases learning time, agents can still achieve near-optimal performance. Second, I develop techniques for automatic calibration and metadata generation in distributed spectrum sensing networks using signals of opportunity as a form of environmental supervision. These techniques enable verification of sensor characteristics like field of view and location without manual intervention, facilitating trustworthy large-scale deployments. Finally, I propose using generative models to allow the sharing of wireless datasets while preserving privacy, addressing a key barrier to advancing wireless machine learning research. The unifying theme is the development of techniques that minimize required human supervision while maintaining robust performance, enabling more autonomous and scalable wireless systems. This research is a step toward cognitive radio networks that can adaptively and cooperatively manage spectrum resources with reduced human oversight.

To my advisor, Professor Anant Sahai,

Thank you for your invaluable guidance and support throughout my time at Berkeley. I am humbled by your unfailing positivity and incisive comments and questions. I hope to emulate your example throughout my career. Thank you also to my wife, family, and friends who were essential to keeping me happy, healthy, and sane through the ups and downs of a Ph.D.

Contents

Contents ii

List of Figures iv

List of Tables vii

1 Introduction 1
- 1.1 Why ML for Wireless Communication? 2
- 1.2 Employing Machine Learning for Wireless PHY 4
- 1.3 Beyond Protocols: Exploiting the Physical World 7
- 1.4 Bringing Wireless Comms into the Age of Big Data 10

2 Interactive Supervision for Learning Wireless PHY 12
- 2.1 Introduction 12
- 2.2 Related Work 15
- 2.3 Overview 17
- 2.4 Levels of Information Sharing 20
- 2.5 Alienness of Agents 26
- 2.6 Experiments 29
- 2.7 Results 32
- 2.8 Implementation in Software-Defined Radios 40
- 2.9 Conclusion 45

3 Environmental Supervision for Automatic Sensing and Calibration 51
- 3.1 Introduction 51
- 3.2 Related Work 56
- 3.3 Signal Sources 58
- 3.4 Automatic Sensor Evaluation and Calibration 68
- 3.5 Automatic Metadata Generation 79
- 3.6 Conclusion 89

4 Towards Synthetic Datasets for Data-Scarce Domains 92
- 4.1 Introduction 92

4.2 Related Work 98
4.3 Methods 99
4.4 Results 104
4.5 Conclusions 108

5 Concluding Thoughts **111**

Bibliography **113**

A Code for Echo Protocol **150**

B Detailed Agent Descriptions **151**
B.1 Classic 151
B.2 Neural 151
B.3 Polynomial 153

C Additional Results **156**
C.1 Effect of Modulation Order and Training Signal to Noise Ratio 156
C.2 Polynomial Agent Experiments 158

D Unsuccessful Constellation Centering Methods **165**

E Classic Modulation Schemes **167**

F Simulation Settings **168**

G Simulation Training Hyperparameters **169**
G.1 GP Training Hyperparameters 170
G.2 LP, ESP, and EPP Neural Training Hyperparameters 171
G.3 LP, ESP, and EPP Polynomial Training Hyperparameters 174

H GNU Radio Training Hyperparameters **176**

I Common EEG Channels **177**

J Spectrogram Model Hyperparameters **179**

List of Figures

1.1 A timeline of the development of wireless communications. 1
1.2 Block diagram of a typical PHY processing chain. Matching colors represent complementary operations. 2

2.1 Visualization of the Echo protocol. 19
2.2 Echo with Private Preamble: Round-Trip. 22
2.3 An example modulation scheme learned by agents using the EPP protocol to demonstrate ambiguity of communication after a half-trip, but coherence after a round-trip exchange. 24
2.4 Echo with Shared Preamble: Round-Trip. 25
2.5 Loss Passing: Half-Trip. 27
2.6 Gradient Passing: Half-Trip. 28
2.7 Example dB-off-optimal calculation. 31
2.8 Effect of Information Sharing, Neural-fast-and-Neural-fast. 35
2.9 Learning to communicate with a fixed agent, Neural-fast-and-Classic. 37
2.10 Learning with clones (Neural-fast-and-Neural-fast, Neural-slow-and-Neural-slow) compared to learning with self-alien (Neural-fast-and-Neural-slow) using the EPP protocol. 38
2.11 Learning with clones (Neural-slow-and-Neural-slow, Poly-fast-and-Poly-fast) compared to learning with an alien (Neural-slow-and-Poly-fast) using the EPP protocol. 39
2.12 Learning with clones (Neural-fast-and-Neural-fast, Poly-slow-and-Poly-slow) compared to learning with an alien (Neural-fast-and-Poly-slow) using the EPP protocol. 40
2.13 USRP radio connections. 43
2.14 Round-trip median bit error curves for Neural-and-Clone python simulation and GNU Radio agents learning QPSK under the ESP protocol at training SNRs corresponding to 1% BER. 44
2.15 Convergence of 50 simulation and 20 GNU Radio trials to be within 3 dB-off-optimal (at testing SNR corresponding to 1% BER) of the corresponding baseline for ESP trials of neural agent and clone at training SNR corresponding to 1% BER for QPSK modulation. 45
2.16 Round-trip median bit error curves for Neural-and-Clone python simulation and GNU Radio agents learning QPSK under the EPP protocol at training SNRs corresponding to 1% BER. 46

2.17 Convergence of 50 simulation and 20 GNU Radio trials to be within 3 dB-off-optimal (at testing SNR corresponding to 1% BER) of the corresponding baseline for EPP trials of Neural agent vs clone at training SNR corresponding to 1% BER for QPSK modulation. 47

3.1 ADS-B performance for measuring directionality. 59
3.2 The number of flights and sectors covered around points in North America at their busiest time during one day. 60
3.3 Six orbits of the NOAA 19 weather satellite. The red area shows the footprint of the satellite at a given location. 62
3.4 Transmitter frequency deviation of NOAA satellites. 63
3.5 Changes of the Doppler shift during a satellite pass. 64
3.6 NOAA satellite signal in the frequency domain. 65
3.7 Theoretical versus observed Doppler shift error, and the time period when the Doppler offset is tracked. This is the same pass as Figure 3.5. 66
3.8 Mobile network experiment testbed. 67
3.9 Measurement locations for frequency sensitivity experiments. 68
3.10 Cellular reception for nearby towers, ordered by increasing carrier frequency. . . 69
3.11 Broadcast TV reception: different frequency bands 70
3.12 Experiment hardware setup. 71
3.13 Example processing steps from aircraft detections to final probability of detection estimate. 72
3.14 ADS-B probability of detection by elevation. At higher elevations, the horizon and other obstacles have reduced impact on the field of view. 73
3.15 ADS-B FoV estimates for the experiment location. 73
3.16 ADS-B FoV estimates with increasing number of time samples. As the number of samples increases, the size and accuracy of the estimated FoV increases. . . . 74
3.17 Satellite FoV estimate using only angle measurements. 75
3.18 Merged FoV estimate using ADS-B and satellite detection data. Accuracy improves at long range. 76
3.19 Simulated antenna pattern based on experimental data. The antenna pattern estimates range dependence in the FoV of the experiment data then applies it isotropically in angle. 77
3.20 ADS-B FoV accuracy for 24 time samples at 20 km with “best” sampling, averaged across 100 locations in the U.S. 78
3.21 ADS-B FoV accuracy for one time sample at 20 km with “best” sampling, averaged across 100 locations in the U.S. 79
3.22 The difference in RSSI behavior when the receiver is indoor and outdoor. 80
3.23 The impact of antenna diversity on RSSI. 81
3.24 The size of cluster in each window as a metric to quantify RSSI variations. . . . 83
3.25 ROC curves for bi-level classifier. 87

3.26 Verifying the location of a sensor by comparing measured and expected Doppler shift. The colors represent the MSE. 88
3.27 Localization error versus number of satellite passes used as input to the localization algorithm. 89

4.1 PSDs of example FM radio and EEG recordings demonstrating the difference in bandwidth occupied. 97
4.2 A spectrogram of an audio clip that sounds like a dog barking and looks like a corgi [376]. 98
4.3 Plots of data snippets from the chosen datasets, showing the first 10 seconds of a random subject and the first 20 channels in the original ordering. 100
4.4 An example spectrogram after preprocessing. 102
4.5 Original and VAE reconstructed versions of an EEG spectrogram, using the original SDv1 VAE. 105
4.6 Original and VAE reconstructed versions of an EEG spectrogram, using the updated SDXL VAE. 105
4.7 MMD metrics using CLIP and DINOv2 features for synthetic and reconstructed datasets. 106
4.8 Classification accuracy for last-iteration and EMA models on trained on real or synthetic datasets, and tested on all. 107

C.1 Neural agents learning with a clone using the EPP protocol for different modulation orders. 159
C.2 Neural agents learning with a clone using EPP for different training signal to noise ratios. 160
C.3 Effect of Information Sharing while learning to communicate with a clone, Poly-fast-and-Poly-fast. 161
C.4 Learning to communicate with a fixed agent, Poly-fast-and-Classic. 162
C.5 Learning with clones (Poly-fast-and-Poly-fast, Poly-slow-and-Poly-slow) compared to learning with self-alien (Poly-fast-and-Poly-slow) under the EPP protocol. . . 163
C.6 Learning under ESP protocol. 164

D.1 An example of a pseudo-BPSK constellation reached during one training run with the GNU Radio EPP implementation. Two pairs of constellation points exist antipodally, just like a BPSK constellation. There is not enough separation between constellation points within the pairs to reliably demodulate the correct bit sequences. 166

E.1 Figures (a) through (c) show the fixed, optimal modulation used for Classic models; (d) through (f) show the corresponding demodulation boundaries. 167

List of Tables

0.1 List of Acronyms ix

2.1 SNRs corresponding to round-trip BER values. 30
2.2 Number of symbols exchanged before $\geq$ 90% of trials reached 3 dB-off-optimal BER, GP and LP. 33
2.3 Number of symbols exchanged before $\geq$ 90% of trials reached 3 dB-off-optimal BER, ESP and EPP. 34
2.4 Average execution times for Neural agent training update and GNU Radio wrapper processing. 41

3.1 Experimental FoV Estimation Error. 76
3.2 Simulated FoV Estimation Error (°). 77
3.3 ADS-B measurement locations and environments. 82
3.4 The expected cluster mean distance to center, or inertia, for different variation ranges and number of clusters. 83
3.5 Outdoor/indoor prediction for model-based method versus ground truth. 85
3.6 Outdoor/indoor prediction for model-based method per-sector versus ground truth. 85
3.7 Results for the bi-level classifier. Accuracy is the maximum accuracy achievable by adjusting the classification threshold. 86
3.8 Fractions of sectors labeled and predicted as being outdoor for experiments with the sensor located indoors and outdoors. 87

4.1 Datasets found on RFDataFactory and OpenRANGym containing IQ data. . . . 93
4.2 Gender classification CNN model architecture. 104
4.3 Accuracy scores for gender classifiers trained and tested on the real and synthetic datasets. 107

C.1 Number of symbols exchanged before $\geq$ 90% of trials reached 3 dB off of optimal BER for the ESP and EPP protocols with Neural agents and varying bits per symbol (BPS) and SNR. 157

F.1 Experiment settings for the different protocols and modulation orders. (* Because Neural-fast-and-Neural-fast converged so fast, we only trained for 500 iterations to adequately sample the convergence curve.) 168

G.1 Descriptions of hyperparameters used in experiments. 169
G.2 Neural hyperparameters used for GP QPSK Simulation experiments. There are no exploration-related parameters because GP does not use a Gaussian policy. . 170
G.3 Poly hyperparameters used for GP QPSK Simulation experiments. There are no exploration-related parameters because GP does not use a Gaussian policy. (* Due to the fact that inputs to modulators are bits, a unique max degree polynomial can be used.) 170
G.4 **Neural-fast** hyperparameters used for LP, ESP, and EPP QPSK Simulation experiments. 171
G.5 **Neural-slow** hyperparameters used for ESP and EPP QPSK Simulation experiments. 172
G.6 Neural hyperparameters used for ESP and EPP 8PSK Simulation experiments . 172
G.7 Neural hyperparameters used for ESP and EPP 16QAM Simulation experiments 173
G.8 Neural hyperparameters used for EPP QPSK Training SNR Simulation experiments. 173
G.9 **Poly-fast** hyperparameters used for LP, ESP, and EPP QPSK Simulation experiments. (* Due to the fact that inputs to modulators are bits, a unique max degree polynomial can be used.) 174
G.10 **Poly-slow** hyperparameters used for EPP QPSK Alien Simulation experiments. (* Due to the fact that inputs to modulators are bits, a unique max degree polynomial can be used.) 175

H.1 Neural hyperparameters used for GNU Radio experiments 176

I.1 Common channels and original indexes from datasets. 178

J.1 Training hyperparameters for the SDv1 diffusion model. 179
J.2 Training hyperparameters for the DiS diffusion model. 180
J.3 Training hyperparameters for the gender classification model. 180

Table 0.1: List of Acronyms

Acronym	Definition
ADS-B	Automatic Dependent Surveillance-Broadcast
AI	Artificial Intelligence
AMC	Automatic Modulation Classification
AMR	Automatic Modulation Recognition
AWGN	Additive White Gaussian Noise
BER	Bit Error Rate
BPS	Bits Per Symbol
BS	Base Station
CBRS	Citizens Broadband Radio Service
CFO	Carrier Frequency Offset
CNN	Convolutional Neural Network
CSI	Channel State Information
DFE	Decision Feedback Equalizer
DFT	Discrete Fourier Transform
DL	Deep Learning
DNN	Deep Neural Network
DPM	Diffusion Probabilistic Model
DSP	Digital Signal Processing
EEG	Electroencephalogram
EMA	Exponential Moving Average
EPP	Echo with Private Preamble (Protocol)
ESP	Echo with Shared Preamble (Protocol)
FEC	Forward Error Correction
FFT	Fast Fourier Transform
FID	Frechet Inception Distance
FIR	Finite Impulse Response
FLOP	Floating Point Operation
FM	Frequency Modulation
FoV	Field of View
GAI	Generative Artificial Intelligence
GAN	Generative Adversarial Network
GP	Gradient Passing (Protocol)
HSI	Hyperspectral Imaging
IIR	Infinite Impulse Response
IoT	Internet of Things
KID	Kernel Inception Distance
KNN	K-Nearest Neighbors

LEO	Low Earth Orbit
LLM	Large Language Model
LMS	Least Mean Squares
LNA	Low Noise Amplifier
LOS	Line of Sight
LP	Loss Passing (Protocol)
MAE	Mean Absolute Error
MIMO	Multiple-Input Multiple-Output
ML	Machine Learning
MLP	Multi-Layer Perceptron
MMD	Maximum Mean Discrepancy
NF	Normalizing Flow
NRDZ	National Radio Dynamic Zone
OpenRAN	Open Radio Access Network
PD	Probability of Detection
PHI	Protected Health Information
PHY	Physical Layer
PSD	Power Spectral Density
RF	Radio Frequency
RL	Reinforcement Learning
RNN	Recurrent Neural Network
RSRP	Reference Signal Received Power
RSSI	Received Signal Strength Indication
SDR	Software-Defined Radio
SER	Symbol Error Rate
SGD	Stochastic Gradient Descent
SNR	Signal to Noise Ratio
SoOp	Signal of Opportunity
SOTA	State of the Art
SWAP	Space Weight and Power
UE	User Equipment
VAE	Variational AutoEncoder

Acknowledgments

The work contained herein was funded by the UC Berkeley ML4Wireless center member companies, the DARPA Spectrum Collaboration Challenge, NSF grants AST-144078 and ECCS-1343398, and SpectrumX, the Spectrum Innovation Center, NSF Award 2132700. I also wish to thank my co-authors and members of the BLISS lab for their helpful discussions and ideas.

Chapter 1

Introduction

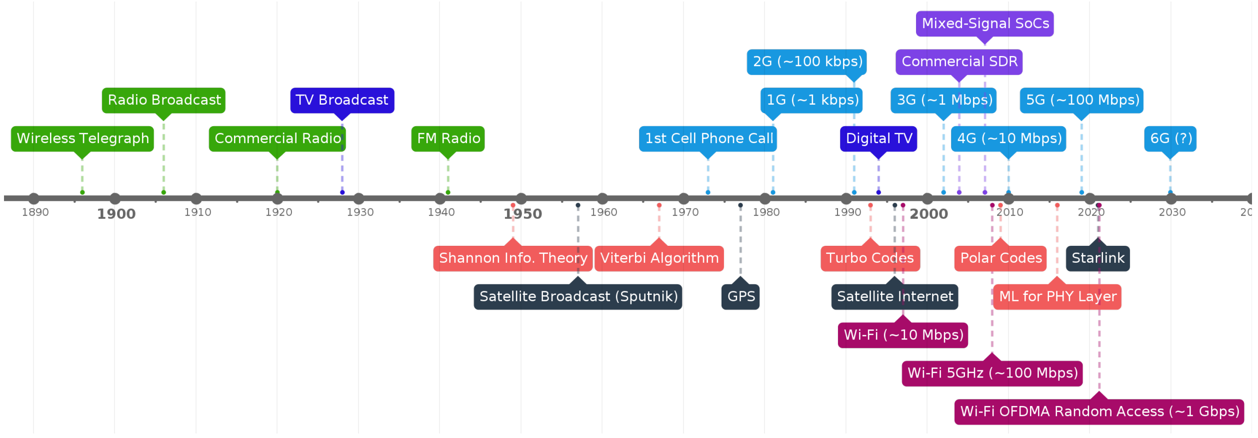


Figure 1.1: A timeline of the development of wireless communications.

The field of wireless communication has seen multiple epochs of change, introducing new capabilities and efficiencies that, in turn, serve as a foundation for services that we have come to depend on, invisibly shaping the world in which we exist. From the early days of radio broadcasts to the groundbreaking work of Claude Shannon on information theory in the 1940s, to modern wireless standards and the always-connected information age we live in now, these advancements shape how we communicate and connect. Figure 1.1 illustrates the history of wireless communications and the increasingly rapid progress leading into now and the future. Looking toward the future, the integration of machine learning (ML) in wireless communication presents an exciting prospect for adaptive and intelligent systems that will continue to shape how we interact with the world around us. These new systems will be able to, and must, operate in more crowded, more complex, and more constrained environments than ever before. This dissertation will explore some of the technologies and techniques enabling new capabilities, focusing particularly on adaptation and exploitation of the physical, or PHY layer of the larger wireless communication stack (see Figure 1.2) through ML techniques and what I dub "light supervision".

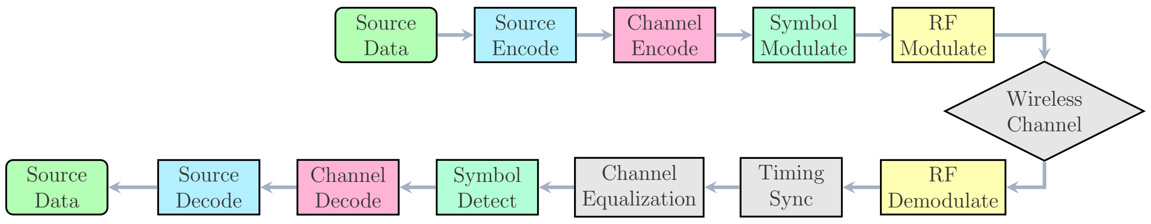


Figure 1.2: Block diagram of a typical PHY processing chain. Matching colors represent complementary operations.

Light supervision here refers to techniques that, rather than full and near-perfect knowledge of the world or a dataset provided by a human for optimization, use a limited subset of data or data obtained by observing or interacting with the environment to improve or adapt with the minimum amount of human involvement possible. Reducing the requirement for human input is crucial to move towards autonomously adapting systems that can be deployed in the real world and function without requiring the continual intervention of a domain expert. In the following chapters, I will expand on different views of "supervision" in each segment of the wireless communication ecosystem I explore and the enabling techniques for light supervision.

1.1 Why ML for Wireless Communication?

Before diving into the technical details, I will explain why ML should be the tool of choice for building the next epoch of wireless communications. After all, for most of the last 100-plus years, wireless communication engineers and researchers have built and improved systems, with a few exceptions such as Least Mean Squares (LMS) channel equalization [1], without relying explicitly on ML techniques, particularly in the PHY layer. This has been possible for several reasons. First, simple models such as additive white Gaussian noise and binary symmetric channels, which lend themselves to simple yet powerful analysis, have been good facsimiles for the conditions experienced by communication networks in the wild [2]. Second, wireless communication systems have historically been designed as independently operating layers, only interacting when data is directly passed between them. This design allowed optimizations and design changes to each layer without impacting those above or below. Under some assumptions, this separation of blocks is even optimal [3]–[5]. One justification for independent layers, even at the cost of performance, is simply to allow humans to design systems without an explosion of special corner cases.

Most operations in each block are also linear operations like a Discrete Fourier Transform (DFT) or finite impulse response (FIR) filter. Again, this design choice makes analysis and implementation more straightforward and often optimal under the modeled conditions. However, the continually increasing demand for wireless data, in quantity and availability, requires system designers to eke out every bit of performance available within the current protocols, adapt to changing conditions, and operate in new conditions where old assumptions no longer hold. Each of these cases represents an opportunity for ML to supplement the techniques and models that underpin modern communication systems.

One way to meet increasing demand is to increase the performance of systems using existing resource allocations, primarily power, spectrum, and time. Under the non-linear and non-stationary conditions of the real world, the assumption of independence between blocks of the PHY processing chain (see Figure 1.2) no longer holds and linear operations may no longer be optimal. This results in a loss of performance, but also an opportunity to regain performance by optimizing across layers and introducing non-linear operations. An example of both cross-layer and non-linear processing without introducing ML is Decision Feedback Equalization (DFE) [6]. In DFE, the output of the channel demapper (see Figure 1.2) is fed back to the equalizer to update the channel correction taps. This breaks the barrier between processing blocks, violating independence, and results in a non-linear feedback control loop. Despite the increased difficulty of analysis and implementation, DFE is widely employed because it produces real gains in system performance and because designers have the tools from control theory to understand DFE's behavior and limits - and herein lies one of the key differences between traditional wireless communication systems design and ML.

Designers, especially of safety-critical or mission-critical systems, want guarantees of system performance and behavior. Most ML techniques, particularly modern deep learning models, operate mostly as block boxes with unknown behavioral boundaries and failure modes [7]–[10]. Establishing and enforcing boundaries on the output of a model and understanding failure modes is an ongoing topic for state-of-the-art research, particularly in the context of reinforcement learning (RL) models for controlling robots and vehicles [11], [12]. Even the behavior of the most basic Multi-Layer Perceptron (MLP) networks is still being explored, for example, with differentially noisy overparametrized data [13]–[15]. Along with these difficulties, however, ML provides opportunities to eke out better performance from wireless communication systems.

Historically, space, weight, and power (SWAP) limited computing required PHY layer algorithms to be implemented directly in dedicated hardware, or at least digital signal processing (DSP) accelerators, to meet latency and power requirements. Fortunately, the ever-decreasing cost of computing power has enabled the creation of so-called software-defined radios (SDRs). Beginning from efforts at Raytheon in the 1980s and first commercially available as Ettus USRPs in 2004, it is now possible to program (and reprogram) radios without designing an entirely new hardware package each time. The open-source community has even implemented a functional WiFi station from hardware to firmware to software [16]. The advent of reconfigurable SDRs opens up the possibility of bringing ML algorithms to the PHY layer.

This ability to adapt is a key enabler for the concept of cognitive radio [17]. Cognitive radios adapt their bands, interfaces, protocols, and spatial and temporal usage patterns to better utilize the finite RF spectrum resource. To accomplish this, they must sense their environment, coordinate with other users, and adapt their behavior autonomously, repeatedly solving a high dimensional optimization and reasoning about their own and others' behaviors. Machine learning is realistically the only way to solve this problem, particularly in the face of changing usage demands and patterns.

1.2 Employing Machine Learning for Wireless PHY

One of the earliest places ML techniques found purchase is resource allocation for cellular systems, initially in power and user allocation [18]–[20], then continuing onto more fine-grained channel usage [21], [22]. ML techniques have several advantages in this area that led to their early adoption. System behavior was not already standardized, meaning designers did not have to force models to fit within arbitrary constraints. This kind of high-dimensional optimization problem maps well to already understood ML and optimization problems. In the same way that designers could take tools from control theory to fit DFE into the PHY processing chain, tools from (mixed) integer optimization and RL lent themselves to the cellular resource allocation problem. Finally, cellular base stations have guaranteed connectivity and large SWAP budgets, meaning they can spend more compute and energy on solvers and centralize optimizations.

As computation becomes cheaper, it has become possible to introduce ML algorithms to lower layers of the stack. Work on replacing PHY layer components with ML algorithms kicked off with work that replaces the entire PHY layer for two radios with an autoencoder that compresses data bits to channel symbols on one end and decompresses them back to data bits on the other [23], [24]. This end-to-end autoencoder style training has been expanded upon with [25] and without [26]–[31] channel models. Active research continues with improving generalization and fast adaptation [32], online adaptation [33], and resource reduction for Internet of Things (IoT) devices [34]. Other recent work includes [35]–[44]. Besides replacing the entire PHY layer with an ML algorithm, other work focuses on replacing specific blocks or sets of blocks from the processing chain in Figure 1.2.

There has been interest in applying ML techniques to equalization since the 1990s [45], [46]. Equalization lends itself naturally to ML techniques since it is a supervised (or semi-supervised in the case of DFE) regression problem, a well-studied task in ML literature, and has many overlapping aspects. The LMS algorithm is precisely a special case of Stochastic Gradient Descent (SGD), a fundamental optimization algorithm for modern deep learning. FIR and IIR (infinite impulse response) filters map directly to convolutional (CNN) and recurrent neural networks (RNNs), respectively. Recent efforts to apply ML to equalization include more modern deep learning architectures such as CNNs and RNNs [47] and techniques such as meta-learning [48]–[50] and Bayesian techniques for uncertainty estimation [49], [50], generating information useful higher up the processing chain. The equalization block is often combined with the symbol demapper [51], [52]. Another advantage of equalization with ML techniques is that protocols specify the pilot symbols that a transmitter must send for equalization, but not how the receiver processes them, letting designers drop in whatever algorithm they prefer. Other equalization via ML work, particularly on blind equalization among other special cases, includes [53]–[64]. The Dual Path Network architecture in [64] is an interesting special case, where the ML components only perform feature extraction to feed parameters such as frequency offset or channel taps into traditional signal processing algorithms that actually modify the received signal.

Along with equalization, symbol mapping/demapping can greatly benefit from taking advantage of non-linearities in real-world systems. Systems with access to a side channel, which allows for feedback from receiver to transmitter, can learn a symbol constellation that deviates from the optimal regular pattern in an AWGN channel. This can be used to compensate for non-linearities in the transmitter's amplifier [65] or in long-haul fiber optics which face primarily multiplicative noise rather than additive [66]. Other work which does not require feedback to the transmitter includes ViterbiNet [67] which embeds a neural network inside the Viterbi algorithm to estimate symbol log-likelihoods in place of a conventional soft-demapping block. This allows the system to function in complex, unknown, or varying channels where a conventional demapper would break down. This approach can be extended to BCJR decoding for turbo codes [68] or trained with meta-learning to adapt rapidly to non-stationary channel conditions [48]. Symbol demapping is often combined with equalization as in ViterbiNet or [49], [54], [57], [58], [69]–[71]. This is useful since intersymbol interference means the demapping decision for one symbol may impact the decision for the next. Non-linear neural networks can take advantage of non-linear relations between symbols which linear equalization followed by demapping may destroy. Similarly, demapping and decoding are often merged into a single operation to take advantage of intersymbol relations [56], [72]. Another line of work seeks to shape a symbol constellation independently, compensating for channel or system nonlinearities [66], [73]–[75].

A key insight to the approach in [67] and [68], which will come up again when discussing forward error correction (FEC), is the idea of embedding ML models within a conventional algorithm designed to take advantage of inherent characteristics of a system to reduce computational complexity. This embedding is, in a sense, the inverse of deep unfolding [76], where the sequential stages of an iterative algorithm are 'unfolded' into successive layers of a single deep neural network (DNN). Embedding a DNN, rather than unfolding an iterative algorithm, minimizes the number of potentially expensive non-linear operations and the number of trainable parameters, reducing the computation cost and the amount of data required to train the model. The structure of the original algorithm, often carefully designed to take advantage of underlying structure in the data, is also preserved. [64] employs a related design technique, only using neural nets to inform traditional algorithms which operate on the data. Some of the most significant advances in deep learning have come from taking advantage of structure hidden in data. CNNs make use of spatial locality and translational invariance in images to reduce parameter count and speed computation [77], [78], and attention in transformers makes use of the fact that most words in a sentence are only closely related, or attend, to a few others to achieve success in language modeling [79], [80]. It follows that introducing ML models to wireless communication will be most successful by similarly taking advantage of structure inherent to the problem.

Design of good codes, particularly decoding algorithms, is a space especially open to improvement with ML techniques. Optimal decoding generally depends on knowledge of channel characteristics [81], but traditional decoders must rely on equalization to remove intersymbol dependencies and ignore fading effects. The adaptivity of ML models enables channel-specific decoding for complex or poorly modeled channels [82]–[84] and new coding architectures like feedback codes [85]. Short block length codes especially underperform compared to theoretical limits as, until Polar codes [86], all codes guaranteed performance only as a limit with infinite block length. Some recent work focuses on short block-length codes [83], [87]–[90] using various code types and decoding architectures. Several papers take advantage of the common trait of ML models providing fast, approximately correct results compared to iterative solvers to decode polar-coded blocks faster than list decoding or belief propagation, with minimal performance lost [87], [91]–[94]. [83] designs low-latency codes with RNNs for encoding and decoding. KO codes [88] employ the technique of embedding NNs inside a traditional algorithm to great effect, generalizing linear Reed-Solomon codes to include non-linear encoding and decoding and outperforming linear codes even on AWGN channels with similar computational complexity. [85], [95] explore the possibilities of adaptive coding with feedback. Other recent work includes [72], [96]–[107].

The inherently high dimensional nature of coding poses a significant challenge to traditional ML methods since seeing every possible 128-bit codeword just once would require 2^{128} samples, something completely infeasible to train even for state-of-the-art cloud-based compute clusters. Instead, any successful ML model must be able to extract structure from the training data and learn to generalize correctly to unseen samples. It is thus no surprise that many of the most promising approaches to ML-assisted coding employ structured architectures and algorithms like RNNs and tree-structured decoding. At the same time, the near-limitless data, especially once noise is added, means that any model can be eventually trained to completion. The trick is then to figure out how to train a model rapidly, reliably, and robustly, whether through careful hyperparameter selection or techniques such as meta-learning [84], [108]. These features mean that even as ML provides a toolbox for tackling problems in the wireless communication domain, those same problems provide a lens for examining phenomena like robustness, generalization, and online adaptation of general interest to ML practitioners in a setting where there are comparisons like existing techniques and information-theoretic bounds to know how well a model should work and how well it possibly could work.

Massive MIMO (multiple-input multiple-output) is a topic that has recently grown in importance with multiplying antenna counts in 5g and beyond technologies. As the size of the problem of channel state estimation and antenna selection grows rapidly with the number of antennas, it quickly becomes infeasible to solve analytically in real-time. ML models can be employed to provide fast, approximate answers in place of expensive operations like matrix inverses for channel estimation and correction [65]. Alternatively, work like [109] replaces explicit channel estimation with direct extraction of symbols from IQ data, relying on the model to implicitly account for channel effects. Another common technique in literature is to use the compression ability of autoencoders [110] or predictive models [111] to reduce overhead for channel state information (CSI) feedback from user equipment (UE) to base stations. Unlike much other work, [111] was able to leverage training on high-quality (i.e., uncompressed) CSI data and careful selection of model architecture and prediction target to produce a model capable of being run in real-time on a 5G testbed system. Some recent work employs more modern ML techniques such as graph neural networks [112], transfer and meta-learning [113], and attention [114]. Other work on CSI estimation, antenna selection, and beamforming includes [70], [115]–[121].

One feature that all of the previously mentioned work shares in common is that the algorithms and techniques are predetermined and co-designed. Pilot symbols are known and fixed, prearranged side channels provide feedback for training autoencoders, the types of coding schemes are preset, interfaces for sharing CSI information are clearly defined, et cetera. In Chapter 2, I explore the effects of viewing co-design between learning agents (radios with ML algorithms) as a form of inherent supervision and how reducing supervision, or ‘light supervision,’ affects the performance of learning algorithms. To narrow the problem scope, I focus on the task of learning a mutually understood symbol modulation constellation. The Echo Protocol, introduced in [73], provides a minimal set of behaviors the two agents must obey as a baseline level of co-design (supervision). Adding additional shared information to the protocol acts as a tunable level of supervision with which to explore the problem space. Throughout the chapter, I show that two agents with even the minimum amount of co-design are able to learn a shared modulation scheme and that increasing supervision through co-design enables faster and more robust convergence to a final constellation, with minimal performance loss compared to an optimal constellation. Additionally, the results are verified on a testbed with two USRPs, experiencing all of the challenges faced by real-world systems, such as clock mismatch, channel effects, and quantization error. Overall, my results suggest that careful choice of a basic set of system behaviors can allow cognitive radios to bootstrap from nothing after being dropped in an unknown environment to a functional communication network. At the same time, having greater co-design provides a better guarantee of successful connection and better performance. The chapter will further discuss how this co-design might manifest in future systems.

1.3 Beyond Protocols: Exploiting the Physical World

Section 1.2 focuses on applying ML directly to the process of communication between two radios. However, a cognitive radio must do more than communicate with a partner, it must also sense and understand the world around it to make decisions about power, band selection, routing, and more. Some of this knowledge can come from communicating with partners, like CSI feedback or tables of visible nodes for routing. Other information can come from simply listening to other users of the same band, like received signal strength indication (RSSI) values to understand what other users are ‘close’ or ‘far’. Sensing band occupancy is arguably one of the most important problems to solve for cognitive radio, and smart selection of bands for communication accounting for other users is a key part of addressing increasing spectrum congestion [122].

As an important capability for modern and cognitive radios, band occupancy sensing has been well studied. [123] surveys the primary methods of occupancy detection using signal processing techniques and the a-priori information required to employ them:

- energy detection, requiring no knowledge of other users,
- waveform detection, searching for known preambles, pilot sequences, or other repeated signals,
- cyclostationary feature detection, which can find digital communication signals using periodicities inherent to the signal,
- matched filtering, which optimally detects any known signal,
- and others that can be tailored to specific use cases.

Since the instantaneous bandwidth of a single radio is typically much less than the amount of spectrum that it wants to sense, sensors can cooperatively combine measurements to produce a more accurate picture [124] or employ federated learning to achieve the same effect while maintaining privacy [125]. Even once a signal has been detected, it can be useful to further identify what type of signal it is, since different users have different usage patterns. A radar will likely repeat a signal at regular intervals while a WiFi station might transmit in short, dense bursts to save power. [126] surveys traditional and deep learning (DL)-based techniques for automatic modulation classification (AMC), a common step in identifying usage types.

Awareness of other users of shared spectrum is now a part of the regulatory environment for several bands. Regulatory efforts began with rules allowing unlicensed reuse of TV whitespace [127] and evolved into tiered licensing and dynamic access as part of the Citizens Broadband Radio Service (CBRS) [128] and WiFi 6E [129]. Non-incumbent (generally meaning non-radar) users of these bands must be aware and avoid interfering with higher priority users, either through coordinating with a central authority that performs its own spectrum sensing (CBRS) or listening themselves for other users (WiFi 6E). These users must also know their own location and indoor/outdoor status to avoid interfering. Other experimental efforts on spectrum cohabitation, such as the National Radio Dynamic Zones (NRDZ) [130], involve even finer spectrum occupancy sensing and coordination.

The ability to sense other signals in the environment provides more information than just who else is nearby. The proliferation of wireless devices, from WiFi and mobile phones to IoT devices and satellites provides a wealth of signals which can be exploited to learn more about a device and its surroundings. Sensor localization, especially indoors, is an important problem as wireless sensor networks grow in size meaning humans can't provide precise location information for each node, and GPS localization is not always feasible due to cost, power, or signal availability. [131] provides an excellent overview of various ML techniques applied in this domain using information from other nodes in the network, and [132] discusses signal processing approaches. However, there are often many signals from outside its network 'visible' to a radio. A question that remains relatively unexplored is, "What we can learn about a sensor and its environment from ambient signals?"

Ambient signals in the environment that a sensor node can exploit uncooperatively for some purpose, dubbed "Signals of Opportunity" (SoOps), are surprisingly powerful sources of knowledge. The first application was bistatic radar using AM broadcasts [133], LTE [134], WiFi [135] and other signals. [136] used SoOps to measure soil moisture, snow sensing, and sea surface height. Hobbyists exploit cell and TV signals to measure PPM offsets of cheap SDRs [137], [138]. SoOps are particularly useful for localization. Automatic Dependent Surveillance-Broadcast (ADS-B) signals are broadcast from all commercial aircraft with high power (250-500 W) and contain location information [139], [140]. They are also intentionally easy to receive and decode. Several works employ ADS-B for indoor localization where GPS signals cannot reach [141], [142]. Doppler shift measurements from orbiting satellites have been used since the 1960s for navigation [143], and have seen a resurgence in interest in the modern era of spoofed and jammed GPS signals [144]–[148]. None of these applications provide unique information that something like a calibrated test device or careful installation by a human could not. Instead, they provide a kind of supervision for otherwise unsupervised measurements, like oscillator frequency or indoor location.

Chapter 3 explores potential uses for several signals of opportunity, primarily ADS-B and transmissions from NOAA weather satellites, as a form of "automatic calibration" for system parameters, backed up by real measurements with SDRs. I claim that SoOps act as 'light supervision' in the sense that we have no control over which signals are present ahead of time or even when and where in the case of moving emitters like aircraft. They are also uncooperative, so we must extract information through careful consideration of the physical properties and publicly visible information in each signal. 'Full supervision' would consist of a calibrated measurement device or a trusted human user providing each value directly. Despite their limitations, SoOps can even be used to create new insights into a system, such as field of view (FoV) measurements based on moving emitters sweeping across a sensor's vicinity.

Viewing measurements from SoOps as calibration tools leads to a novel perspective as supervision for the human user. Location estimates can verify that a human installer did not fat finger the latitude of a node on their Nth install of the day or make it harder to maliciously claim an incorrect location to avoid quiet zones or exploit crowd-sensing reward systems. Measurements from SoOps can also provide ongoing insights into system performance and debugging tools that would be too expensive to have humans provide. For example, comparisons to known SoOps could diagnose the difference between a damaged RF sensor and a long-term change in characteristics of the primary sensing target.

Overall, the goal of the automatic calibration work in Chapter 3 is to explore a pathway toward autonomous cognitive radios. In the past, systems have been viewed as individually autonomous, relying on their own measurements of the world to act and react. Adding SoOps is a step towards a collective view of autonomy in which the collective produces and shares information to enable productive individual behaviors, such as the location embedded in ADS-B messages. As systems can obtain and share more information they are freed more and more from human supervision - but they are still supervising each other.

1.4 Bringing Wireless Comms into the Age of Big Data

The rise of very large-scale datasets combined with progress in AI/ML learning algorithms and training hardware has led to the rise of generative model 'killer apps' in ChatGPT [149] for language generation, and DALL-E and StableDiffusion [150] for image generation. Generative artificial intelligence (GAI) research has shown that generative models excel at discovering and reproducing complex data distributions across multiple areas beyond language and images such as motion planning and protein design [151]–[153]. Despite their power, the computational cost of both training and inference and complex or unstable training processes of many generative techniques limit how suitable they may be for typical physical layer applications [154]–[156]. In some cases where existing techniques are generally inadequate or very limited, such as underwater wireless optical communication, the additional cost of generative modeling may be worth paying [157]. Generative models also show promise for blind equalization of a communication signal with unknown pilot signals [53], [158], [159]. In this case, the probability distribution constraints imposed on generative models may act as a form of light supervision, forcing the models to learn to separate probabilistic noise and channel effects from signal content.

Generative models truly shine as aids for wireless communication when they are used 'on-label' to somehow augment available data. A common pitfall of DL approaches to physical layer problems is that they require large datasets and long training to obtain good performance [160], [161], and may overfit to one particular instance of a channel or fail to adapt as channel conditions change [162], [163]. Unaided DL methods often break down in non-gaussian and time-varying channels [164], just where they are needed most. In [163], a generative adversarial network (GAN) learns to reproduce channel effects and serves as a synthetic "bridge," allowing gradients to flow from the decoder through an unknown channel back to the encoder of an autoencoder. Other work uses the distributions learned by generative models more explicitly. In [162], a GAN generates synthetic variants of training data to augment the dataset and prevent overfitting. In [165], a variational autoencoder (VAE) translates the complex channel output to a simpler latent distribution for which a posterior probability can be more easily calculated and used for signal detection. [166] and [167] also employ VAEs to avoid calculating intractable posterior probabilities. To overcome a lack of diverse channel models for simulation of complex millimeter wave and urban environments, [168] and [169] generate plausible channel models based on a limited set of known models for use in further simulations. Non-generative approaches to dataset synthesis or augmentation generally start with purely synthetic data [170] or high-quality captures of radio transmissions, potentially with impairments applied for diversity [171].

In Chapter 4, I tackle a problem that, to my knowledge, has not been directly addressed via generative modeling in the wireless communication literature so far: overcoming barriers to releasing large recorded datasets due to privacy concerns. Some repositories of over-the-air IQ recordings exist, such as may be found in the RFDataFactory [172]. However, most of these datasets were recorded in research testbeds rather than in the wild and may not be representative of common or all channel environments a system in the wild will face. This is not solely due to the cost of taking measurements in the field - researchers fear liability for privacy breaches caused by sharing recorded data publicly, even if they never process the contents of their data themselves [173], [174]. One approach to sharing data while maintaining privacy is federated learning [125], [175], [176], but this requires coordination between participants who wish to use the data, complicating any open data or model releases. Instead, I propose training a conditional generative model on the real dataset, then releasing a synthetic dataset conditioned to replace any real private contents with a harmless replacement. The generative model learns to preserve and replicate distributional features of the real dataset while removing sensitive data. Some major hurdles to eventual deployment of this approach include challenges inherent to generative modeling like memorization [177], and a lack of provable privacy guarantees like differential privacy [178], [179]. The goal of this work is to enable researchers to eventually apply a “best effort” anonymization pass to sensitive wireless (or other) datasets and be able to release them without fear of liability.

Chapter 2

Interactive Supervision for Learning Wireless PHY

2.1 Introduction

Communication is a fundamentally cooperative action between two (or more) agents that exhibit behaviors designed to facilitate the exchange of information between them. Usually, these behaviors are solely to effect the passage of information from one user to another. However, in some cases such as feedback codes [85], [90], [95], these cooperative behaviors are instead intended to help a partner improve itself to communicate better. This chapter takes the concept of feedback enabling agents to better communicate and uses it as a lens to explore how varying levels of supervision impact learning and communication performance. Starting from a minimal feedback behavior of each agent 'echoing' what it hears from its partner blindly, successive levels of shared information, implying greater co-design, allow better supervision during the learning process and eventually build to completely supervised learning. This spectrum of supervision provides insights into how unrelated, or non-co-designed, agents can learn cooperatively and into how supervision affects the difficulty of some learning tasks. The work in this chapter was originally published in [74].

As communication is necessarily a cooperative activity, communication itself can be viewed as both a special case of cooperation and a building block that can be leveraged to permit more effective cooperation. The fundamental limits to learning how to cooperate with a stranger have been studied in an abstract theoretical setting in [180]–[184]. By asking how two intelligent agents might understand and help each other without a common language, a basic theory of goal-oriented communication was developed in these papers. The principal claim is that for two agents to robustly succeed in the task of learning to collaborate, the goal must be explicit, verifiable, and forgiving. However, the approach in these works took a fundamentally *semantic* perspective on cooperation. As Shannon pointed out in [185], the arguably simpler cooperative problem of communicating messages can be understood in a way that is decoupled from the issue of what the messages mean. To see whether existing machine learning paradigms can be adapted to achieve cooperation with strangers, this chapter considers the concrete problem where two agents learn to communicate in the presence of a noisy channel. Each agent consists of a modulator and demodulator and must learn compatible modulation schemes to communicate with each other.

This problem of learned communication has been tackled using learning techniques under different assumptions on the information that the agents are allowed to share and how tightly coordinated their interaction is. Early work in this area [23], [186], where gradients are shared among agents, demonstrated the success of training a channel autoencoder using supervised learning when a stochastic model of the channel is known. Subsequent works relax the assumption on the known channel model by learning a stochastic channel model by using GANs as in [187]–[189] or by approximating gradients across the channel and using that for training. However, these approaches cannot be said to represent communication with strangers, and instead represent a way of having co-designed systems learn to communicate. If instead of sharing gradients, agents can only share scalar loss values, then with access to a shared preamble, reinforcement learning can be used to train the system as demonstrated in [73] and [190] without having access to a stochastic channel model.

Moving closer to minimal co-design, if we further restrict ourselves to the case where the two agents only have access to a shared preamble, the "Echo" protocol, where an agent hears, understands, and repeats (echoes) back the message received from the other agent, as specified in [73] has been shown to work. By comparing the original message with the received echo, a learning agent can obtain feedback about how well the two agents understand each other[1]. The work in [73] considered a neural network-based modulator that was trained using reinforcement learning via policy gradients [191], but the demodulator was nearest neighbors based and required no training — it used small-sample-based supervised learning. The work in this chapter builds on this foundation and studies the truly "blind" case where agents do not have access to a shared preamble.

We dub this "blind interactive learning" to acknowledge the motivational connection with the well-known and traditional problems of blind equalization and blind system identification — where systems have to deal with a channel and implicitly learn a model for it without knowing the actual input to the channel. (See, for example, the book [192] for a survey of well-understood approaches.) Such blind approaches are fundamentally motivated by the desire for universality and the resulting robust modularity. Traditional blind approaches in signal processing are intellectually akin to what are called unsupervised learning approaches in machine learning. Reinforcement learning has always occupied a middle ground between

[1]Round-trip stability is not by itself a sufficient condition to guarantee mutual comprehension. After all, one agent might only be doing simple mimicry — repeating back the raw analog signal value received with no attempt to actually demodulate. However, in [73], the key insight was that intelligent agents, though strangers, are believed to be cooperative and so wish to understand and communicate with each other. They do not need to coordinate with another designer to realize that sheer mimicry would not necessarily advance their goal of cooperation. Consequently, the Echo protocol can rely on good intentions to eliminate the possibility of agents just mirroring what has been heard instead of trying to understand what was sent and repeating it back.

supervised and unsupervised learning, because in a sense, it is self-supervised and carried out via interaction with an environment. For the purposes of this chapter, "blind interactive learning" involves agents interacting blindly via an environment — where the individual agents might be self-supervised but there is no explicit joint self-supervision. They are blind in the traditional sense of not really knowing what exactly went into the channel whose output they are observing, and in particular, not sharing a known training sequence.

It is important to differentiate here between this work on blind *interactive* modulation learning, and automatic modulation recognition (AMR) as in [193]. AMR seeks to take an unknown signal present in the environment and classify it as one of a set of known modulation schemes for subsequent demodulation. No interaction with the signal source occurs — indeed, for surveillance applications, interaction might destroy all value! In contrast, this work requires interaction to learn how to demodulate any possible signal, even ones never before seen or imagined, and further to learn to modulate in a similarly arbitrary way understandable by an agent on the other side of a communication channel. This interaction takes the form of round-trip training so that both the modulation and demodulation functions can be updated. Although AMR-based techniques could play a role in the demodulation half of this process by speeding up learning for known signals, they are insufficient on their own to complete the circle. This work introduces a new problem, blindly learning an entire modulation and demodulation scheme, rather than introducing a new technique for classifying existing modulation schemes.

This chapter also introduces the concept of "alienness" among agents. After all, if our goal is to understand learning of communication between strangers, we need to be able to test with strangers. A natural question is what it means for two agents to be alien. This work in this chapter considers agents to be alien if they differ in their learning architecture or their hyperparameters. This work examines modulators and demodulators represented using different types of function approximators such as neural networks and polynomials.

Our main contribution is to investigate whether the Echo protocol is universal, i.e. does it allow two agents to learn to communicate irrespective of their inner workings, **and what level of information sharing, or supervision through co-design, is necessary for successful learning**. Although we do not have a formal proof of universality yet, we provide some empirical evidence by pairing agents with different levels of "alienness" based on the hyperparameters, architectures, and techniques used in their modulators and demodulators. By doing so, we wish to separate the effects of the agents' implementations, such as those owed to specific function approximators, from the meta protocols (specifically the Echo protocol) used to do the interactive learning. To both connect to the literature and explore the spectrum between complete co-design and cooperative learning among strangers, we look at different levels of information sharing: shared gradients, shared loss information, shared preamble, and finally, the case where only the overall protocol is shared.

Machine learning scholarship is notorious for producing results that are not easily reproducible and failure to identify the source of and explain the reasoning behind performance gains [194]. Keeping this in mind, to evaluate the ease, speed, and robustness of the learning task under various levels of alienness and information sharing, we conduct repeated trials for each setting using different random seeds and multiple sets of hyperparameters. We report the fraction of trials that succeeded as a function of the number of symbols exchanged, as well as aggregate statistics about the bit error rate achieved at different signal-to-noise ratios (SNRs) by the learned modulation schemes. We compare experimental bit error rates to those achieved by optimal modulation schemes for AWGN channels to provide a baseline for comparison. The code used to generate the results is available in [195]. From our experiments, we observe and conclude that the Echo protocol does enable two agents to learn a modulation scheme even under minimal shared information and that as the amount of shared information decreases, the learning task becomes harder, i.e., a lower fraction of trials succeed, and the agents take longer to learn.

It appears that learning to communicate with "alien" agents is not necessarily more difficult than learning to communicate with agents of the same type. However, it is significantly easier to learn to communicate if one of the agents already uses a good modulation scheme, for example a hand-designed scheme like QPSK. Finally, as the modulation order for communication increases, the learning task becomes harder, especially in settings with little information sharing.

Although a majority of the results reported in this chapter were performed purely in simulation, we replicate the main results using USRP radios and observe similar results — two agents can learn to communicate in a decentralized fashion even using real hardware.

2.2 Related Work

Deep learning has shown great success in tasks that historically relied on multi-stage processing using a series of well-designed, hand-crafted features such as computer vision, natural language processing, and more recently robotics. Wireless communication is another area that historically uses hand-crafted schemes for various processing stages such as modulation, equalization, demodulation, and encoding and decoding using error correcting codes. Thus, as alluded to in [196] and [197], one might believe that bringing deep learning into wireless communication is a worthwhile endeavor. In fact, learning and deep learning have been present in the sub-field of AMR since at least the 1980s. AMR has undergone a similar transition from hand-crafted features, such as phase difference and amplitude histograms as detailed in [193], to modern deep learning techniques [198]–[201].

Beyond modulation recognition, the pioneering work in [23], [186] demonstrated the promise of the channel auto-encoder model by using supervised learning techniques to learn an end-to-end communication scheme, including both transmission and reception. This approach assumes the knowledge of an analytical (differentiable) model of a channel and the ability to share gradient information between the receiver and transmitter. This approach was a natural first step given the known connection between auto-encoders and compression (see e.g. [202]) as well as the well-known duality between source-coding (compression) and channel-coding (communication) [185].

Building on this foundation, other works deal with the case where the channel model is unknown, as is the case when performing end-to-end training over the air. In [190], a stochastic model that allows backpropagation of gradients to approximate the channel is used with a two phase training strategy. Phase one involves auto-encoder-style training using a stochastic channel model, whereas phase two involves supervised fine-tuning of the receiver part of auto-encoder based on the labels of messages sent by transmitter and the IQ-samples recorded at the receiver. This approach relies on starting out with a good stochastic channel model. Use of Generative Adversarial Networks to learn such models is explored in [187]–[189]. In [203], instead of estimating the channel model, stochastic approximation techniques are used to calculate the approximate gradients across the channel. The idea of approximating gradients at the transmitter has also been used in [31] to successfully perform end-to-end training.

In the absence of a known channel model, reinforcement learning can also be used to train the transmitter as demonstrated in [73] and [190]. In [73], the Echo protocol, a learning protocol where an agent hears, understands, and repeats (echoes) back the message received from the other agent was used to obtain a scalar loss that was used to train the neural-network based transmitter using policy gradients. Here the receiver used a lightweight nearest-neighbor based scheme that was trained afresh in each communication round. This work assumed that the agents have access to a shared preamble so I dub it *Echo with Shared Preamble* (ESP). In [190] both the transmitter and receiver were neural-network based. The receiver was trained using supervised-learning whereas the transmitter was trained using policy gradients by passing scalar loss values obtained at the receiver back to the transmitter. Reinforcement learning techniques have the added advantage of being implementable in software-defined radios to perform end-to-end learning over the air. To do this one must tackle the issue of time synchronization between the transmitted and received symbols as done in [204] and [205]. In [206], the general problem of synchronization in wireless networks is addressed via the use of attention models.

Other parts of the communication pipeline such as channel equalization and error correcting code encoding and decoding have also been studied using machine learning techniques. The use of neural networks for equalization is studied in [207] and [208]. Construction and decoding of error correcting codes is considered in [83], [84], [196], [209]. Joint source channel coding is an area where performance gains are possible through co-design as demonstrated in [210] for wireless communication, and in the application of wireless image transmission in [211]. End-to-end auto-encoder style training continues to be an area of interest in wireless communication. There has been recent work demonstrating the success of convolutional neural network based architectures and block based schemes in this setting in [212]–[215]. This approach has also been used successfully in OFDM systems [216] to learn the symbols transmitted over the sub-carriers. Deep learning techniques and auto-encoder style training have been used in the fields of fiber-optic [217], [218] and molecular communication [219], [220] to model the channel and to leverage the channel model to learn communication schemes that achieve low error rates.

A theoretical analysis of the general learning to cooperate problem is done in the works

[180]–[184]. This body of work investigates the possibility for two intelligent beings to cooperate when a shared context is absent or limited. In particular, this work also does not presume a pre-existing communication protocol. In asking how two intelligent agents might understand each other without a common language, a theory of goal-oriented communication is developed. The principal claim is that for two agents to robustly succeed in the cooperative task, the goal must be explicit, verifiable, and forgiving. Agents should have feedback about whether the goal is achieved or not, and it should be possible for the agents to achieve the goal from any state that is reached after a finite set of actions. The works [221]–[224] bring about these ideas in a limited setting.

From a psychological perspective, developmental psychology [225] provides a rich account of how human infants learn to communicate. How do babies come to understand sounds, words, and meaning? It begins in the development of 'categorical perception of sound' which creates discrete categories of sound perception, not unlike the task of demodulation. Later on, other tasks emerge such as word segmentation, attributed to statistical learning, where in the child grows increasingly aware of sounds and words that belong together. Soon after, the child engages in babbling as an exploration of language production, investigating rhythm, sound, intonation, and meaning, a task similar to modulation. Important to all the above processes, is social interaction and exchange, most often between child and caretaker, which provides the rich information required for learning to be successful.

2.3 Overview

Problem Formulation

Consider the setting where two agents communicate in the presence of a discrete-time additive white Gaussian noise (AWGN) channel. Each agent comprises an encoder (modulator) and a decoder (demodulator). We treat the modulator as an abstract (black-box) object that converts bit symbols to complex numbers, i.e., a mapping $M : B \to \mathbb{C}$ where B refers to the set of bit symbols and $\mathbb{C}$ refers to the set of complex numbers. Similarly, we treat the demodulator as an abstract object that converts complex numbers to bit symbols, i.e., a mapping $D : \mathbb{C} \to B$. The set of bit symbols B is specified by the modulation order (bits per symbol). For instance, when bits per symbol is 1, $B = \{0, 1\}$, and when bits per symbol is 2, $B = \{00, 01, 10, 11\}$. For the case where bits per symbol is 1, the classic[2] BPSK (binary phase shift keying) modulation scheme is given by:

$$M^{BPSK}(0) = 1 + 0j, \tag{2.1}$$

$$M^{BPSK}(1) = -1 + 0j. \tag{2.2}$$

[2]Here, we use classic to refer to a modulation scheme that is fixed and specified identically for all communicating agents by a certain standard. One example of such a scheme is BPSK signaling, as described here.

The corresponding demodulator performs the demodulation as,

$$D^{BPSK}(c) = \begin{cases} 0, & Re(c) \geq 0 \\ 1, & Re(c) < 0 \end{cases}. \tag{2.3}$$

In addition to agents that use fixed modulation schemes, we also consider 'learning' agents. These agents use function approximators to learn the mappings performed by a modulator and demodulator, and we denote these as $M(\cdot;\theta)$ and $D(\cdot;\phi)$ where θ and ϕ denote the parameters of the underlying function approximators and are updated during training. The specifics of the learning agents and their update methods can be found Appendix B.

The main focus of this work is on learning modulation schemes; thus, to make it easier to conduct experimental simulations, we make the following simplifying assumptions:

1. There are at most two agents, and they engage in perfect turn-taking.
2. The two agents are separated by a unit gain AWGN channel. There is no carrier frequency offset, timing offset, or phase offset.
3. Both agents encode and decode data using the same fixed number of bits per symbol (i.e., the modulation order is preset). Section 2.6 describes the modulation orders and their reference modulation schemes used in this paper.
4. The environment is stationary and non-adversarial during the learning process.

Motivation and Approach – Echo with Private Preamble Protocol

The main objective of this work is to specify a robust communication-learning protocol that allows two independent agents to learn a modulation scheme under minimal assumptions on information sharing beyond shared knowledge of the learning protocol and the ability to take turns. No other information is shared *a priori* or via a side channel during training. We name this protocol *Echo with Private Preamble* (EPP). Details about the EPP protocol are provided in Section 2.4. The EPP protocol is a special variant of the Echo protocol described in Fig. 2.1.

The underlying premise of the Echo protocol is that an echo of the message — originating from one agent and repeated back to them by another agent — provides sufficient feedback for an agent to learn expressive modulation schemes. Under the Echo protocol, one agent (the "speaker") broadcasts a message and receives back an estimate of this message (preamble), an echo, from the other agent (the "echoer"). The passage of the original message from the speaker to the echoer and back to the speaker as an echo is denoted as a *round-trip*. (A *half-trip* goes only from speaker to echoer.) After a *round-trip*, the speaker compares the original message to the echo and trains its modulator and demodulator to minimize the difference (usually measured in bit-errors) between the two messages. The two agents then switch roles

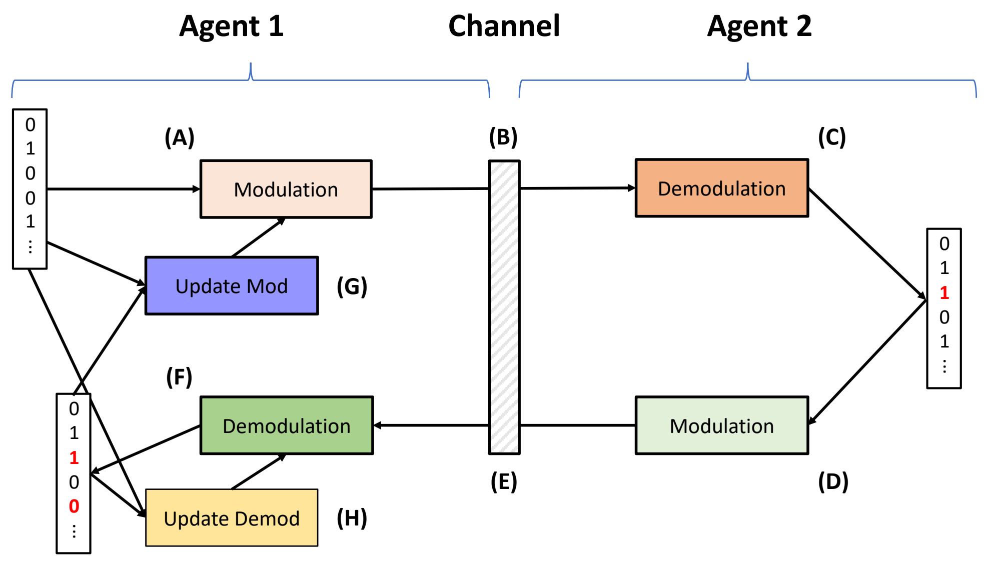


Figure 2.1: Visualization of the Echo protocol.
(A) Speaker Agent (A1) modulates a bit sequence and (B) sends it across a (AWGN) channel. (C) Echoer Agent (A2) receives the sequence and demodulates it. (D) A2 then modulates the recovered sequence, and (E) sends it back over the channel. (F) A1 receives this echoed version of its original sequence and demodulates it. (G, H) Then A1 uses the received echo to update its modulator and demodulator. The agents switch roles and repeat until convergence. Details of the protocol are elaborated in Fig. 2.2 and Sec. 2.4.

and repeat. When the difference between the original message and the demodulated echo is small, we infer that the agents can communicate with one another.

The ideas here are similar to the approach to solving image-to-image mapping, or style transfer, popularized by CycleGAN [226]. Both works solve the problem of learning mappings between domains with only weak supervision by introducing a round-trip and defining 'goodness' as how close the output of the round-trip is to the input. Having round-trip feedback from either the other radio agent or the other GAN crucially enables performance measurement and, hence, training.

In the *Echo with Shared Preamble* protocol from [73], the echo behavior is introduced only to train the modulator, and knowledge of a shared preamble between the two agents is assumed to facilitate directly supervised training of the demodulator after a half-trip exchange. By contrast, in the EPP protocol, the agents do not have access to a shared preamble and must learn to demodulate blindly without knowing what was actually sent by the other agent.

We believe the EPP protocol minimizes the information-sharing assumptions for learning modulation schemes for two reasons. First, some sort of feedback is required for learning and the echo provides this feedback. Second, the EPP protocol treats the environment as a regenerative channel, i.e., a channel that provides helpful feedback without requiring assumptions about the nature of the other communicating agent. As long as the other agent is cooperative (in the sense that it echoes back what is heard), the environment behaves like a regenerative channel.

Next, we argue that the EPP protocol is a plausible mechanism for learning modulation schemes when the channel is regenerative by considering the case of a learning agent communicating with an agent that uses fixed, classic schemes. In this setting, even random exploration would eventually find a modulation scheme that successfully interfaces with the fixed agent. Using feedback to guide exploration, we expect the EPP protocol to perform much better than random guessing and quickly converge to a suitable modulation scheme. One can think of such a fixed, friendly regenerative channel as a "game" that the learner plays where a positive reward is achieved if what the channel echoes back can be decoded as what the learner encoded and sent in. Reinforcement learning is good at optimizing behaviors for simple games like this [227]. One of this work's main contributions is to show that the EPP protocol works not only with fixed communication partners but even in the case where two agents are learning simultaneously.

To verify the universality of the EPP protocol and understand its performance relative to more structured or complex procedures, we run experiments with:

1. Different learning protocols based on varying amounts of information sharing as described in Section 2.4.

2. Different levels of "alienness"[3] among agents as described in Section 2.5.

3. Different modulation order and levels of training SNR as described in Section 2.6.

2.4 Levels of Information Sharing

The EPP protocol introduced in Section 2.3 is designed to be minimalist in the sense that agents share as little information as possible. However, using less information usually comes at the cost of performing worse. To quantify the value of shared information, this section describes the following protocols that allow an increasing amount of shared supervisory information:

1. *Echo with Shared Preamble* (ESP) protocol: Agents have access to shared preamble but can only get feedback via a round-trip during training,

2. *Loss Passing* (LP) protocol: Agents have access to a shared preamble and share scalar loss values directly (without using the channel) during training, and

[3] "Alienness" describes how different the models for the agents' modulators and demodulators are between the two agents. Factors that determine alienness include whether the agents are fixed or learning, the class of function approximators used by the learning agents, and the choice of hyperparameters and initializations.

3. *Gradient Passing* (GP) protocol: Agents have access to a shared preamble and share gradients directly (without using the channel) during training.

Note that by sharing gradients or loss information directly across the channel, it is possible to truncate the learning process at step C in Fig. 2.1 and still update the modulator of the speaker. Examples of algorithms that stop at this step are shown later in Figs. 2.5 and 2.6. In fact, traditional autoencoder-style training is like the gradient passing protocol described above. A reader who is already familiar with this concept may wish to read about the protocols in reverse order from how they are presented.

Our purpose in studying LP, GP, and ESP protocols is primarily to understand the effect of shared information, or increasing supervision, on learning since these are not new and have been studied independently in previous works such as [73], [190], [197]. The LP and GP protocols are not implementable in real-world systems without a side channel to pass losses and gradients — i.e., they can be used in simulation at design-time but not really used at run-time among distributed agents without depending on some existing communication infrastructure between them. ESP, however, is practical and can be implemented by mandating that every agent use a common fixed preamble. The major difference is that ESP requires agents to establish a shared preamble through some other mechanism before they can learn to communicate, whereas EPP removes this requirement. Section 2.7 reports the results of experiments comparing the performance of these protocols and quantifying the value of shared information.

The following subsections describe the learning protocols for EPP, ESP, LP, and GP in detail, highlighting the important differences between them.

Echo Protocol With Private Preamble

The EPP protocol is the main contribution of this chapter. It is described in detail in Alg. 1 and Fig. 2.2. The key details when comparing to ESP, LP, and GP are the natures of the modulator and demodulator updates. For EPP, the demodulator updates use supervised learning relying on noisy feedback because only p is known, but the demodulator actually receives $\hat{p}$. The modulator updates use reinforcement learning based on the round-trip feedback. Because the preamble is known only to the speaker, only the speaker's modulator and demodulator can be updated during a round-trip. The choice of when to terminate training is arbitrary, but we choose to halt training after a fixed number of training iterations. Other implementations might halt training after a BER target is reached.

One important consideration unique to the EPP protocol is that there is no way to ensure that the bit sequence sent by the modulator is interpreted as the same sequence after being demodulated. More formally, there is no way to ensure that

$$p = D_2\left(M_1(p;\theta_1);\phi_2\right). \tag{2.4}$$

For example, Agent 1 might modulate the sequence 11 as some symbol c_1, but Agent 2 might interpret c_1 as 00. After a round-trip, however, any incorrect bit sequence to modulated

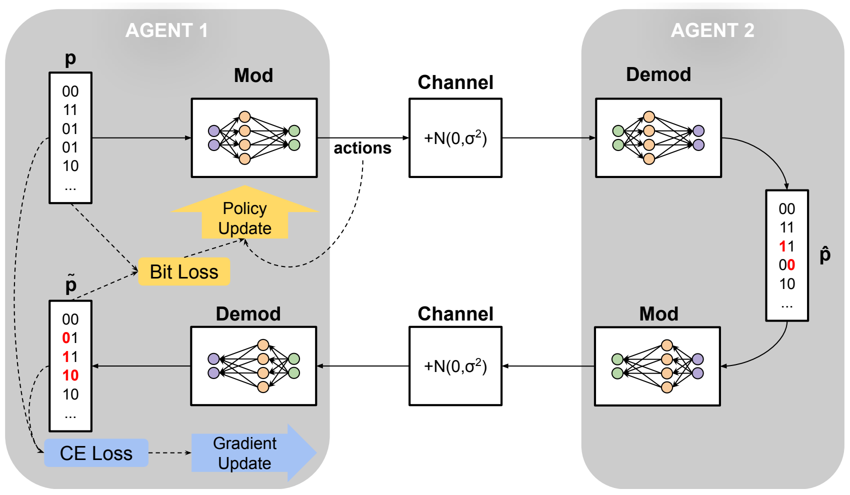


Figure 2.2: Echo with Private Preamble: Round-Trip.
In this diagram, the preamble p is modulated and sent from Agent 1 through a channel to Agent 2 to be demodulated as $\hat{p}$. Agent 2 has no information about the message it received, so it cannot update. It simply modulates and echos back the message it demodulated. Agent 1 then demodulates the echoed preamble as $\widetilde{p}$ and does a policy update of its modulator using a bit loss between the original preamble that Agent 1 sent, p, and echoed preamble that Agent 1 received, $\widetilde{p}$, as well as a supervised gradient update of its demodulator with cross-entropy loss. Agent 1 and Agent 2 then switch roles so that now Agent 2 is the speaker and Agent 1 is the echoing agent. All implementations for the modulator currently use a Gaussian policy with mean and variance estimated by a function approximator as described in Section 2.6.

symbol mappings will be reversed if the agents have trained properly. We can guarantee that

$$p = D_1\left(M_2\left(D_2\left(M_1(p;\theta_1);\phi_2\right);\theta_2\right);\phi_1\right). \tag{2.5}$$

Fig. 2.3 demonstrates how this might happen. We address how to evaluate agents when this mapping ambiguity is present in Section 2.6. In general, it would require a protocol higher up the communication stack to disambiguate symbol mappings without access to a shared preamble — some way of symmetry breaking is required, presumably requiring knowing more about the context of the communication.

Echo With Shared Preamble

The ESP protocol is described in Fig. 2.4 and Alg. 2. ESP was first explored in [73] where the modulator was neural network based and trained using policy gradients but where the demodulator used clustering methods[4] trained via supervised learning using the shared preamble.

[4]If the demodulator is using unsupervised clustering algorithms, acting cooperatively requires the clustering algorithm to be stable. If the label assigned to each cluster changes every iteration, the other agent

Algorithm 1 Echo Protocol with Private Preamble

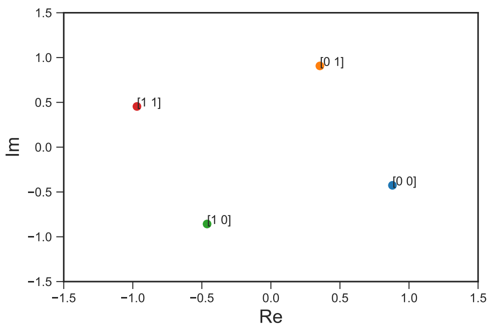
```
procedure EPP(Agent 1, Agent 2)
    Speaker ← Agent 1
    Echoer ← Agent 2
    while training do
        $p \leftarrow n$ random bits    ▷ $p$ is known only to Speaker
        $\mu, \sigma^2 \leftarrow M(p; \theta_s)$    ▷ Speaker generates parameters for its Gaussian policy using $p$
        $s \leftarrow \mathcal{N}(\mu, \sigma^2 I)$    ▷ Speaker modulates by sampling from this distribution
        $\hat{s} \leftarrow f_{\text{channel}}(s)$
        $\hat{p} \leftarrow D(\hat{s}; \phi_e)$    ▷ Echoer demodulates received symbols
        $\mu, \sigma^2 \leftarrow M(\hat{p}; \theta_e)$    ▷ Echoer generates parameters for its Gaussian policy using $\hat{p}$
        $\widetilde{s} \leftarrow f_{\text{channel}}\left(\mathcal{N}(\mu, \sigma^2 I)\right)$    ▷ Echoer modulates by sampling from this distribution
        $\widetilde{p} \leftarrow D(\widetilde{s}; \phi_s)$
        $\theta_s' \leftarrow \theta_s + \Delta_{\theta_s}(s, \widetilde{p}, p)$    ▷ Policy gradient update for speaker's mod
        $\phi_s' \leftarrow \phi_s + \Delta_{\phi_s}(\widetilde{s}, \widetilde{p}, p)$    ▷ Cross-entropy loss gradient update for speaker's demod
        Speaker ⟷ Echoer    ▷ Agents switch Speaker and Echoer roles
    end while    ▷ Only the Speaker updates each round-trip
end procedure
```

In this work, we use the ESP protocol to train agents whose modulators and demodulators both use function approximators. (See Appendix B for more information.)

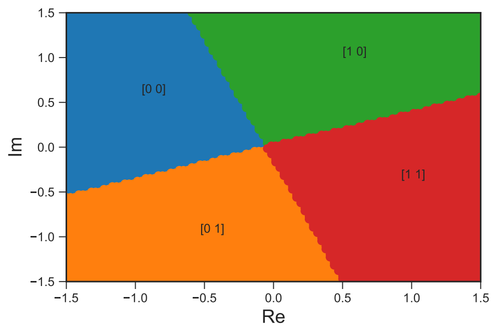
ESP is similar to the EPP protocol, but now both the speaker and echoer know the preamble p that is transmitted. This allows the echoer to update its demodulator after the first half-trip since it knows exactly what it was supposed to have received. This demodulator update is typically of higher quality than the updates in EPP since those updates only have access to symbols based on the (possibly incorrect) estimate of the original preamble sent back by the echoer. The speaker agent does not bother to update its demodulator after the round-trip is complete since it will receive higher-quality feedback on the next training iteration after the speaker and echoer roles are switched.

Importantly, the speaker's modulator still requires a full round-trip before it can receive feedback and be updated. In the next Sections 2.4 and 2.4, this will no longer be the case. The consequence of round-trip feedback is that the speaker's modulator is actually optimizing for the performance of the *speaker's* demodulator since that is the only loss it has access to. Our presumption is that improving the round-trip performance of the speaker's demodulator will indirectly improve the half-trip performance of the echoer's demodulator since the half-trip BER limits the round-trip BER. The consequences of this indirection are illustrated in Section 2.7.

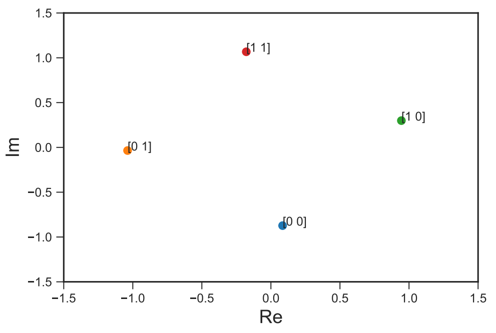
will not be able to converge on a modulation scheme.



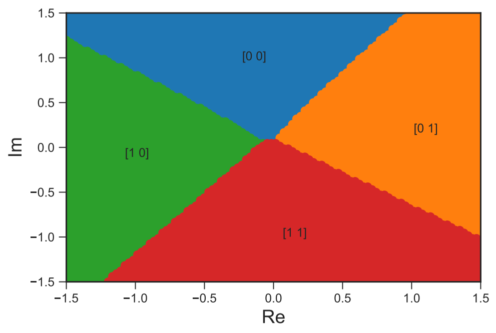
(a) Agent 1 modulation scheme



(b) Agent 2 demodulation scheme



(c) Agent 2 modulation scheme



(d) Agent 1 modulation scheme

Figure 2.3: An example modulation scheme learned by agents using the EPP protocol to demonstrate ambiguity of communication after a half-trip, but coherence after a round-trip exchange.
In this scheme, Agent 1 maps the bit sequence ‘00’ to the complex number $1 - 0.5j$, i.e., M_1(‘00’) = $1 - 0.5j$. Agent 2 demodulates this as the bit sequence ‘11’, i.e., $D_2(M_1(\text{‘00’}))$ = ‘11’ $\neq$ ‘00’. However, this mismatch is reversed when the round-trip is completed. Agent 2 modulates ‘11’ as $0 - 1j$, and Agent 1 demodulates this as ‘00’. Thus $D_1(M_2(D_2(M_1(\text{‘00’}))))$ = ‘00’.

Loss Passing: Half-Trip

Now, we remove the restriction that information can only be shared over the channel during training and allow the agents to magically pass losses back and forth. The loss passing protocol, as used in previous work such as [190], is detailed in Fig. 2.5 and Alg. 3. There is no longer a need for an echo from the second agent since the speaker’s modulator receives a loss value directly from the second agent’s demodulator. This results in two major changes: a full training update can be completed after only a half-trip, and the speaker’s modulator is optimizing for the echoer’s demodulator performance directly.

In the EPP and ESP protocols, the speaker’s modulator has to optimize for the performance of the speaker’s demodulator, only indirectly addressing the performance of the echoer’s demodulator. The LP protocol allows the speaker’s modulator to directly optimize

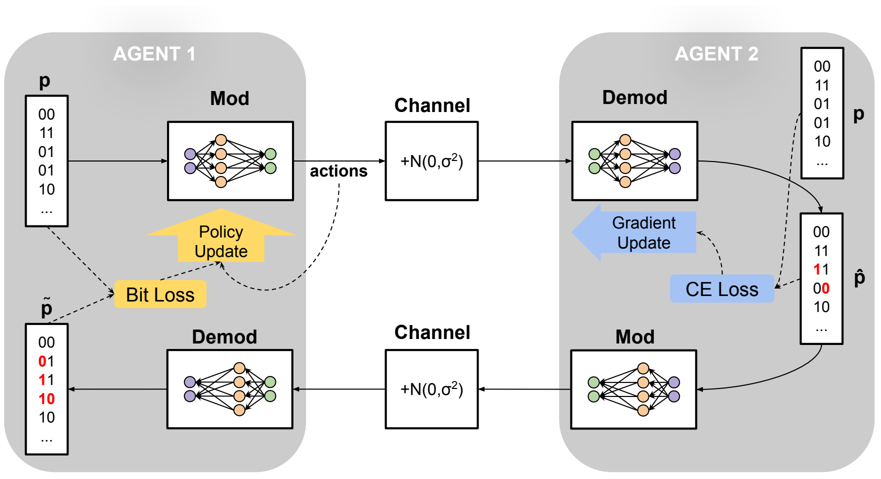


Figure 2.4: Echo with Shared Preamble: Round-Trip.
In the ESP protocol, the preamble p is modulated and sent from Agent 1 across the channel to Agent 2 and is demodulated as $\hat{p}$. Using the shared preamble, Agent 2 performs a gradient update on its demodulator and also modulates and sends back an echo, an estimate of the preamble it received, $\hat{p}$, through the channel back to Agent 1. Agent 1 then demodulates the echo as $\widetilde{p}$ and does a policy update of its modulator using the bit loss between the original preamble p and estimate of the echo $\widetilde{p}$. Agent 1 and Agent 2 then switch roles and repeat the process. All implementations for the modulator currently use a Gaussian policy with mean and variance estimated by a function approximator as described in Section 2.6.

for the performance of the echoer's demodulator since the speaker has access to the relevant loss values. Although the speaker's modulator still has to use reinforcement learning rather than supervised learning to perform parameter updates, we expect the agents to be able to train much faster when using loss passing.

Gradient Passing: Half-Trip

If we further allow the agents to share gradients during training, the system can naturally be treated as an end-to-end autoencoder[5] with channel noise introduced between the encoding and decoding sections. This method was employed successfully in [23]. Our version of such an autoencoder-based training protocol, which we call the GP protocol, is explained in detail in Fig. 2.6 and Alg. 4.

As in the LP protocol, the speaker's modulator can be trained after only a half-trip because it has access to feedback from the echoer's demodulator. Instead of using reinforcement learning to train a Gaussian policy, however, the speaker in GP trains its modulator to

[5]For classic modulators/demodulators, we were able to generate gradient updates by treating/approximating the modulator or demodulator as a differentiable function.

Algorithm 2 Echo Protocol with Shared Preamble

```
procedure ESP(Agent 1, Agent 2)
    Speaker ← Agent 1
    Echoer ← Agent 2
    while training do
        $p \leftarrow n$ random bits    ▷ $p$ is known to Speaker and Echoer
        $\mu, \sigma^2 \leftarrow M(p; \theta_s)$    ▷ Speaker generates parameters for its Gaussian policy using $p$
        $s \leftarrow \mathcal{N}(\mu, \sigma^2 I)$    ▷ Speaker modulates by sampling from this distribution
        $\hat{s} \leftarrow f_{\text{channel}}(s)$
        $\hat{p} \leftarrow D(\hat{s}; \phi_e)$    ▷ Echoer demodulates received symbols
        $\phi'_e \leftarrow \phi_e + \Delta_{\phi_e}(\hat{s}, \hat{p}, p)$    ▷ Cross-entropy loss gradient update for echoer's demod
        $\mu, \sigma^2 \leftarrow M(\hat{p}; \theta_e)$    ▷ Echoer generates parameters for its Gaussian policy using $\hat{p}$
        $\widetilde{s} \leftarrow f_{\text{channel}}(\mathcal{N}(\mu, \sigma^2 I))$    ▷ Echoer modulates by sampling from this distribution
        $\widetilde{p} \leftarrow D(\widetilde{s}; \phi_s)$
        $\theta'_s \leftarrow \theta_s + \Delta_{\theta_s}(s, \widetilde{p}, p)$    ▷ Policy gradient update for speaker's mod
        Speaker ⟷ Echoer    ▷ Agents switch Speaker and Echoer roles
    end while    ▷ The Echoer's demodulator updates, not the Speaker's
end procedure
```

encode bits directly as complex numbers, and the gradients from the echoer's demodulator are used for supervised learning updates.

2.5 Alienness of Agents

How can we determine if the EPP is universal? We need to determine if it allows us to learn to communicate with strangers. There are, in principle, three kinds of agents (strangers) that we might encounter with which we might wish to learn to communicate:

1. A fixed agent that knows how to communicate;
2. A learning agent that does not know how to communicate yet but is cooperative and willing to learn; or
3. An agent that does not know and will not learn how to communicate.

The *Classic* agent uses a fixed modulation scheme known to be optimal for AWGN channels for the given modulation order, e.g., QPSK for 2 bits per symbol, 8PSK for 3 bits per symbol, and 16QAM for 4 bits per symbol [6]. This is an example of an agent of the first kind. An example agent of the second kind is one that uses a function approximator for its modulator and demodulator, which can be trained. We consider *Neural* agents, which use neural networks as function approximators, and *Poly* agents, which use polynomials as

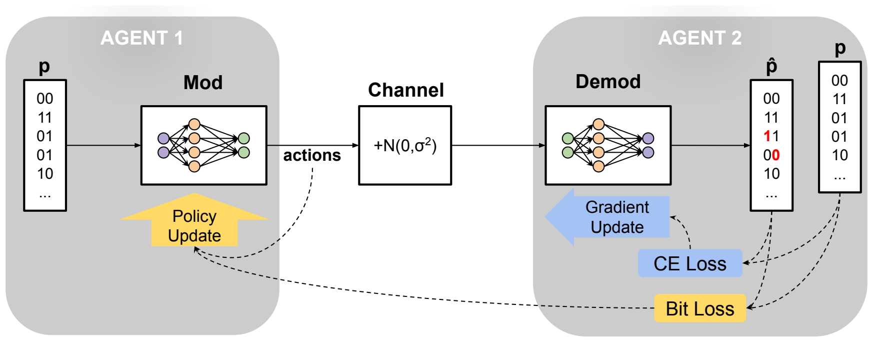


Figure 2.5: Loss Passing: Half-Trip.

In the LP protocol, the preamble p is modulated and sent from Agent 1 across the channel to Agent 2, where it is demodulated as $\hat{p}$. Using the shared preamble, Agent 2 performs a gradient update for its demodulator and shares a scalar bit loss value with Agent 1. Agent 1 then uses this bit loss to perform a policy update of its modulator. Note that the loss is not passed through the channel. All implementations for the modulator currently use a Gaussian policy with mean and variance estimated by a function approximator as described in Section 2.6.

Algorithm 3 Loss Passing: Half-Trip

```
procedure LP(Agent 1, Agent 2)
    Speaker ← Agent 1
    Echoer ← Agent 2
    while training do
        $p \leftarrow n$ random bits                        ▷ $p$ is known to Speaker and Echoer
        $\mu, \sigma^2 \leftarrow M(p; \theta_s)$      ▷ Speaker generates parameters for its Gaussian policy using $p$
        $s \leftarrow \mathcal{N}(\mu, \sigma^2 I)$          ▷ Speaker modulates by sampling from this distribution
        $\hat{s} \leftarrow f_{\text{channel}}(s)$
        $\hat{p} \leftarrow D(\hat{s}; \phi_e)$                ▷ Echoer demodulates received symbols
        $\phi'_e \leftarrow \phi_e + \Delta_{\phi_e}(\hat{s}, \hat{p}, p)$       ▷ Cross-entropy loss gradient update
        $L \leftarrow \hat{p} \oplus p$
        $\theta'_s \leftarrow \theta_s + \Delta_{\theta_s}(s, L, p)$              ▷ Policy gradient update
        Speaker ⟷ Echoer                 ▷ Agents switch Speaker and Echoer roles
    end while                              ▷ Only a half-trip is required for updates
end procedure
```

function approximators. We ignore the agents of the third kind since it is impossible to learn to communicate with such agents.

Note that there are several other examples of agents. A learning agent that has been pre-trained and frozen behaves like a fixed agent. We can, in principle, have learning agents with decision tree or nearest neighbor-based function approximators. However, this work is restricted to Classic, Neural, and Poly agents. Details about these agents, including the

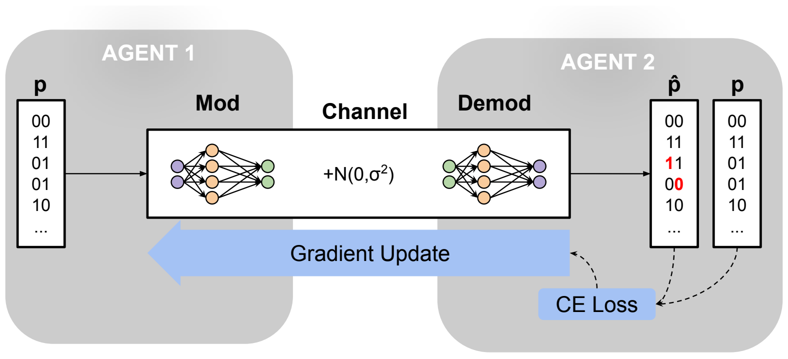


Figure 2.6: Gradient Passing: Half-Trip.
In the GP protocol, the preamble p is modulated and sent from Agent 1 through the channel to Agent 2 and is demodulated as $\hat{p}$. Using the shared preamble, the modulator of Agent 1 and the demodulator of Agent 2 are updated using the cross entropy loss.

Algorithm 4 Gradient Passing: Half-Trip

```
procedure GP(Agent 1, Agent 2)
    Speaker ← Agent 1
    Echoer ← Agent 2
    while training do
        $p \leftarrow n$ random bits                    ▷ $p$ is known to Speaker and Echoer
        $s \leftarrow M(p;\theta_s)$                     ▷ Speaker modulates $p$ directly
        $\hat{s} \leftarrow f_{\text{channel}}(s)$
        $\hat{p} \leftarrow D(\hat{s};\phi_e)$                ▷ Echoer demodulates received symbols
        $\phi_e' \leftarrow \phi_e + \Delta_{\phi_e}(\hat{s},\hat{p},p)$      ▷ Cross-entropy loss gradient update
        $\theta_s' \leftarrow \theta_s + \Delta_{\theta_s}(s,p,\Delta_{\phi_e})$    ▷ Gradient update
        Speaker ⟷ Echoer                         ▷ Agents switch Speaker and Echoer roles
    end while                                    ▷ The Speaker performs a gradient-loss update
end procedure
```

hyperparameters used and training methods employed, are provided in Appendix B.

WE perform experiments by pairing two agents with different levels of alienness, where alienness is determined by:

1. Whether they are fixed agents or learning agents (e.g., Neural-and-Classic matchup)

2. The class of function approximators used by the learning agents. We denote such agents as “Aliens” (e.g. Neural-and-Poly).

3. The random initialization and hyperparameters used by two learning agents using the same class of function approximators. WE denote agents that use the same class of

function approximators but different random initialization and hyperparameters as "Self-Aliens" (e.g. Neural-and-Self-Alien).

4. The random initialization used by two learning agents using the same class of function approximators and the same hyperparameters. WE denote agents that differ only in random initialization as "Clones" (e.g. Neural-and-Clone).

Results for these experiments that portray the effect of different levels of alienness on the performance of the EPP protocol are provided in Section 2.7.

2.6 Experiments

In addition to the effects of different levels of information sharing and alienness, modulation order, training SNR, and modulator constellation power constraints are other factors that affect the performance of our learning protocols.

Modulation Order and Training Signal to Noise Ratio

Modulation order, determined by the bits per symbol (*bps*) used, determines the number of unique symbols that can be sent and received. A *bps* of b corresponds to 2^b unique symbols. For instance, for $bps = 2$, we have 4 unique symbols: '00', '01', '10', and '11'. We consider settings where bits per symbol is either 2, 3, or 4. For Classic agents, *bps* determines the fixed scheme, optimal for AWGN channels, used as a baseline. These are provided in Table 2.1 and visualized in appendices of the published work. For Neural and Poly agents, *bps* determines the size of the inputs and outputs of the modulator and demodulator. Details about this are provided in Appendix B.

Since higher modulation orders have higher bit error rates (BERs) at the same SNR, we must determine an appropriate SNR to use for training and testing to provide a fair comparison between different modulation orders. We do this by selecting the SNR based on the round-trip BER achieved using the baseline (classic) schemes. Most experiments use a training SNR corresponding to a BER of 1%, and all experiments test on SNRs corresponding to BERs ranging from 0.001% to 10% as described in Table 2.1. We explore the effect of modulation order and training SNR on the performance of EPP protocol in Appendix C.1.

Constellation Power Constraints

As described in Section 2.3, the modulator maps symbols (bits) into complex numbers, i.e., points on the complex plane. Due to the presence of the AWGN channel, it is optimal to place these points as far away as possible to minimize the likelihood of an error. Thus, to get non-degenerate solutions we must impose a constraint on how far these points can be from the origin. Note that this is similar to a real-world constraint on power used by a radio system.

	Bits Per Symbol	# Constell. Points	SNR Corresponding to Round-trip BER of				
			0.001%	0.01%	0.1%	1%	10%
QPSK	2	4	13 dB	12 dB	10.4 dB	8.4 dB	4.2 dB
8PSK	3	8	18.2 dB	17 dB	15.4 dB	13.2 dB	8.4 dB
16QAM	4	16	20 dB	18.8 dB	17.2 dB	15.0 dB	10.4 dB

Table 2.1: SNRs corresponding to round-trip BER values.
The SNR-to-BER mappings are used to set test and train SNRs for performance measurements. The SNR corresponding to 1% BER (shaded column) is the default training SNR for all experiments.

we introduce a hard power constraint by requiring that the modulator outputs have an average power of less than 1. we experimented with other soft power constraints by including a penalty term in the loss function based on the power used while training but chose not to use it in the end. For simulations, we observed that a hard power constraint was sufficient and, more importantly, did not require tuning the hyperparameter corresponding to the weight of the power penalty.

Training

For the Neural and Poly learning agents, the demodulator is trained using supervised learning with cross-entropy loss. In the GP protocol, the modulator output is equal to the output of the underlying function approximation, and its parameters are updated using supervised learning. In the EPP, ESP, and LP protocols, the modulator employs a Gaussian policy. The modulator output is sampled from a Gaussian distribution with mean and variance determined by the output of the underlying function approximation, whose parameters are updated using vanilla policy gradients. More details about the update procedure are provided in Appendix B.

We conduct multiple trials using different random seeds for each experiment to accurately estimate the performance of the protocols and agents. An experiment fixes the learning protocol, the agent types, training SNR, and modulation order. Each trial is run for a maximum number of training iterations that we determine empirically for each experiment. Easier learning tasks are run for fewer iterations to speed up the simulations.

Note that instead of measuring training iterations, we can also measure the number of preamble symbols transmitted. These two measurements are related via the preamble length - the number of symbols in the preamble. For all experiments, we set the preamble length to 256 symbols to allow for a fair comparison. This also reduces the relative cost of overheads in the implementation on real hardware radios. Certain protocols and modulation orders require fewer transmitted symbols to achieve good performance. Details about the maximum iterations (and thus maximum number of preamble symbols transmitted) can be found in Appendix F, and in the code itself at [195].

Evaluation

How do we determine the metrics that should be used to measure the performance of a learning protocol? These metrics should allow for a fair comparison across different protocols (GP, LP, ESP, and EPP) and must be informative in determining the effect of different levels of information sharing, alienness, and modulation order on the learning task. We are primarily interested in quantifying 'efficiency', how long the protocol takes to learn a modulation scheme, and 'robustness', how reliably the protocol learns this scheme.

First, we must decide on a metric to determine if the learned modulation scheme is 'good.' Bit error rate (BER) is a natural choice in communication settings, but since we have two agents, we must determine whether to measure cross-agent BER (half-trip BER) or round-trip BER. In the GP, LP, and ESP protocols, both cross-agent and round-trip BER are indicative of performance. In the EPP protocol, since the two agents have no shared preamble, measuring cross-agent BER is not a good indicator of performance since the two agents may have different bit interpretations of the same modulated symbol, as described in Section 2.4. However, round-trip BER is a valid measure of performance in this case. Consequently, we choose the round-trip BER to allow for a fair comparison between different protocols. Note that when measuring the BER to *evaluate* performance of agent(s), instead of sampling from the Gaussian policy, the modulators deterministically use the mean of the Gaussian policy. This avoids introducing additional errors from "exploration."

Next, we must determine the SNR that we measure the round-trip BER at and whether the measured BER is indicative of good performance. As discussed in Section 2.6, we decide on the test SNRs based on the modulation order depending on the performance of the baseline.

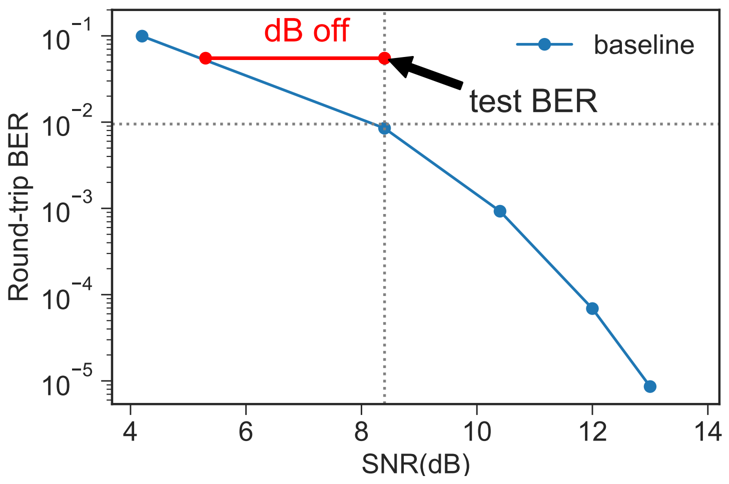


Figure 2.7: Example dB-off-optimal calculation.

To determine if the measured BER is indicative of good performance, we measure the metric 'dB-off-optimal,' illustrated in Fig. 2.7. To compute this metric, we first measure the test BER achieved by a protocol at the SNR where the corresponding baseline scheme achieves 1% BER. Then, we compute the difference between this SNR(dB) and the minimum SNR(dB) required for the baseline scheme to achieve the measured BER. We measure this at different stages of the learning process corresponding to different numbers of preamble symbols transmitted.

Using the round-trip BER and dB-off-optimal metrics, we look at the following two graphs:

1. **Round-trip BER vs SNR**
   Here, we plot order statistics of the round-trip BER achieved by the learning protocol after it has converged or reached the maximum number of iterations allowed in the setup alongside that achieved by the baseline. This graph measures the limit of BER performance of our protocols subject to the maximum number of training iterations symbols we allow. Fig. 2.10a is a representative example.

2. **Fraction of trials that are 3 dB off optimal**
   Here, we plot the fraction of trials that achieve dB-off-optimal values of less than 3 vs the number of preamble symbols transmitted. This shows how robustly and how fast the agents learn the modulation scheme. Fig. 2.10b is a representative example.

Neither of these metrics is novel on its own; however, we are unaware of works that report both metrics. Our contribution is to combine these metrics to understand the performance of a learning protocol.

2.7 Results

In our experiments we consider the following agents: Classic, Neural-fast, Neural-slow, Poly-fast, and Poly-slow. The Neural-fast and Neural-slow agents (similarly Poly-fast and Poly-slow) are Self-Aliens; they share the same architecture but differ in learning rates and exploration parameters. They are named with -fast or -slow depending on their relative modulation scheme learning times using the EPP protocol when paired with a clone. To choose hyperparameters for the agents, we performed coarse hand-tuning for the EPP protocol in the Agent-and-Clone setting and used the same hyperparameters for the ESP and LP protocols. The hyperparameters chosen this way were sufficient to obtain performance close to the baseline in terms of BER for all of these protocols in the Agent-and-Clone setting. However, for the GP protocol, we observed that using the same hyperparameters as EPP led to sub-optimal performance, and thus, we tuned parameters separately for the GP case. While tuning hyperparameters, we further ensured that Poly-fast-and-Poly-fast had similar convergence times to Neural-slow-and-Neural-slow. It is known that hyperparameters matter, and finding good hyperparameters is hard (see, for example, [228]). However, we trust our conclusions about the relative performance of protocols in these experiments because we see order of magnitude differences across many trials. Hyperparameters for each agent are listed in Appendix G.

We categorize the different pairings based on the different levels of alienness introduced in Section 2.5 as follows:

1. Agent learning to communicate with its clone (Agent-and-Clone): Neural-fast-and-Neural-fast, Neural-slow-and-Neural-slow, Poly-fast-and-Poly-fast, Poly-slow-and-Poly-slow.

2. Agent learning to communicate with its self-alien (Agent-and-Self-Alien): Neural-fast-and-Neural-slow, Poly-fast-and-Poly-slow.

3. Agent learning to communicate with an alien: Neural-fast-and-Poly-slow, Neural-fast-and-Poly-fast, Neural-slow-and-Poly-slow, Neural-slow-and-Poly-fast, and any agent with a Classic.

The experiments in this section use 2 bits per symbol and train at 8.4 dB SNR, corresponding to 1% BER for the QPSK baseline. Tables 2.2 and 2.3 contain numerical results for the experiments on the effects of information sharing and alienness on the performance of modulation learning schemes. The following subsections present additional figures and discuss the meaning of these results. Experiments on the effect of modulation order and training SNR on the performance of the ESP and EPP protocols can be found in Appendix E.

	Neural Mod, Classic Demod	**Poly Mod, Classic Demod**	**Neural Mod, Neural Demod**	**Poly Mod, Poly Demod**
Gradient Passing	2048	3328	2048	5632
Loss Passing	5888	9984	2816	20736

Table 2.2: Number of symbols exchanged before $\geq 90\%$ of trials reached 3 dB-off-optimal BER, GP and LP.
Using 8.4 dB test SNR with various combinations of modulator and demodulator type and 2 bits per symbol. As expected, the results show learning using the GP and LP protocols to be fast (compared to ESP and EPP in Table 2.3). Furthermore, the GP protocol converges faster than equivalent experiments using LP. Gradients can carry more supervisory information than scalar loss values, so it makes sense for GP to be a more effective learning protocol.

Effect of Information Sharing

In this first set of experiments, we explore the effect of information sharing on our learning protocols. We seek to quantify the value of shared information and understand the performance trade-off incurred by reducing shared information, or supervision, in the ESP

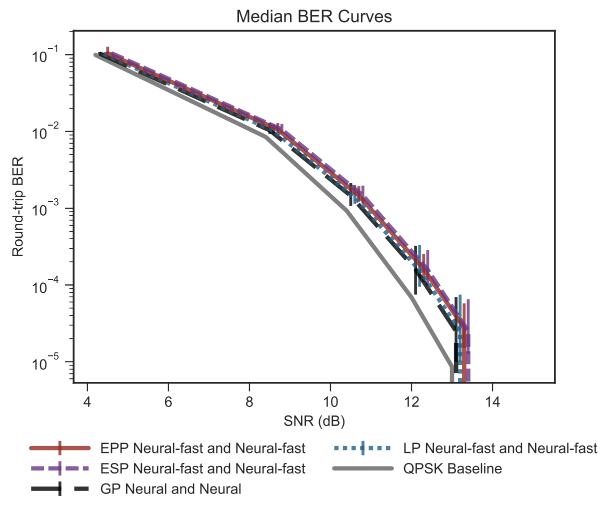
Agent 2 / Agent 1	**Classic**	**Neural-fast**	**Neural-slow**	**Poly-fast**	**Poly-slow**
ESP					
Neural-fast	19456	25600	-	206336	-
Poly-fast	105472	-	-	347648	-
EPP					
Neural-fast	18432	115200	528384	404480	528384
Neural-slow	137728	-	528384	768000	690176
Poly-fast	107520	-	-	528384	690176
Poly-slow	176640	-	-	-	690176

Table 2.3: Number of symbols exchanged before $\geq 90\%$ of trials reached 3 dB-off-optimal BER, ESP and EPP.
Using 8.4 dB test SNR with various agent types and 2 bits per symbol. An order of magnitude more symbols have to be exchanged before the learning agents converge compared to the GP and LP protocols in Table 2.2. Learning with a fixed agent (Classic) is much easier than a clone learning agent, with convergence happening $1.3 - 3.3\times$ faster for ESP and $3.8 - 6\times$ faster for EPP. The extra shared information in ESP seems to compensate for the increased difficulty of learning with another learning agent.

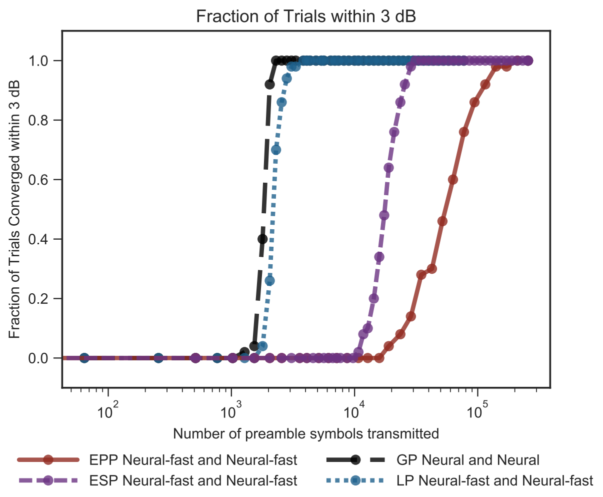
protocol. For this, we primarily consider the case of an agent learning to communicate with its clone. We choose this case because, to succeed at learning to communicate with others, one must first be able to communicate with (a copy of) oneself.

Figure 2.8 compares the performance of the GP, LP, ESP and EPP protocols for a Neural agent learning to communicate with its clone. From the round-trip BER curves in Figure 2.8a, we observe that all protocols achieve similar values for the median BER and upper and lower percentiles. Furthermore, the median BER is close to the QPSK baseline. This is one of the main results of this work. **EPP can perform as well as ESP, LP and GP and achieve performance close to an optimal baseline.** From Figure 2.8b, we observe that all protocols are robust, with the fraction of trials that converge going to 1 after sufficient preamble symbols are exchanged. The EPP protocol needs the most preamble symbols to converge, followed by the ESP protocol, and both these protocols take a much larger number of preamble symbols to converge than the GP and LP protocols. Thus, we conclude that with decreasing amounts of information sharing, it takes longer to learn to communicate, highlighting the value of shared information.

Tables 2.2 and 2.3 tabulate the number of preamble symbols that have to be exchanged for more than 90% of trials to converge within 3 dB-off-optimal for the different protocols. From these tables, we see that there is an order of magnitude or more difference in the number of preamble symbols required between the protocols that use a side channel (GP and LP) and ones that don't (ESP and EPP). we performed similar experiments using Poly agents and observed the same behavior, included in Appendix C.



(a) Round-trip median BER. The error bars reflect the 10th to 90th percentiles across 50 trials. All agents are evaluated at the same SNR, but error bars have been dithered for readability.



(b) Convergence of 50 trials to be within 3 dB-off-optimal at testing SNR 8.4 dB.

Figure 2.8: Effect of Information Sharing, Neural-fast-and-Neural-fast.
The BER plot (a) shows that all protocols achieve BER close to the QPSK baseline. From the convergence plot (b), we observe that EPP is much slower than ESP, which in turn is an order of magnitude slower than LP and GP. Protocols with greater information sharing (thus supervision) lead to faster convergence.

In the rest of the experiments, we determine the effect of alienness on the EPP protocol to address the universality of the Echo protocol.

Effect of Alienness

We explore the effects of alienness with the following cases:

1. Learning with fixed: An agent learning to communicate with a fixed agent. (Neural-fast, Neural-slow, Poly-fast, Poly-slow and Classic.)

2. Learning with clone: An agent learning to communicate with its clone. (Neural-fast and Neural-fast, Neural-slow and Neural-slow, Poly-fast and Poly-fast, Poly-slow and Poly-slow.)

3. Learning with self-alien: An agent learning to communicate with its self-alien. (Neural-fast and Neural-slow, Poly-fast and Poly-slow.)

4. Learning with alien. An agent learning to communicate with an alien (Neural-slow and Poly-fast, Neural-fast and Poly-slow.)

We are primarily interested in answers to the following questions:

1. Is it possible to learn to communicate with self-aliens and alien agents using the EPP protocol?
2. Is it intrinsically more difficult to learn to communicate with aliens or self-aliens than to learn with clones?
3. Can we say something about the performance of the EPP protocol with alien agents based on the individual performances when trained with clones? (e.g Can we say something about the performance of Neural-slow-and-Poly-fast by looking at the performances of Neural-slow-and-Neural-slow and Poly-fast-and-Poly-fast?)

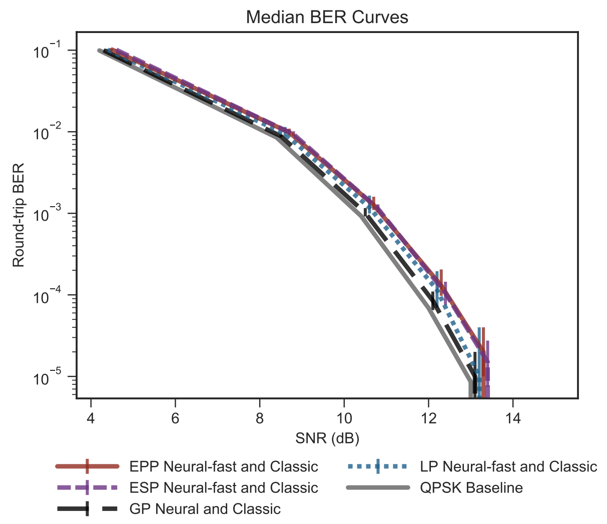
Learning with Fixed Agents

we first address the question of whether it is possible for learning agents to learn to communicate with fixed agents. This is important since we are likely to encounter agents that use fixed modulation schemes in the real world, and our learning agent must be compatible with them. To do this, we run experiments with a Neural agent learning to communicate with a fixed Classic agent, Neural-fast-and-Classic. After confirming that learning agents can work with fixed Classic agents, we compare this to the case when a Neural agent trains with another learning agent. In particular, we examine whether learning to communicate with a clone is harder than learning to communicate with a fixed agent and whether increasing alienness (self-alien and completely alien) further increases the difficulty of the task.

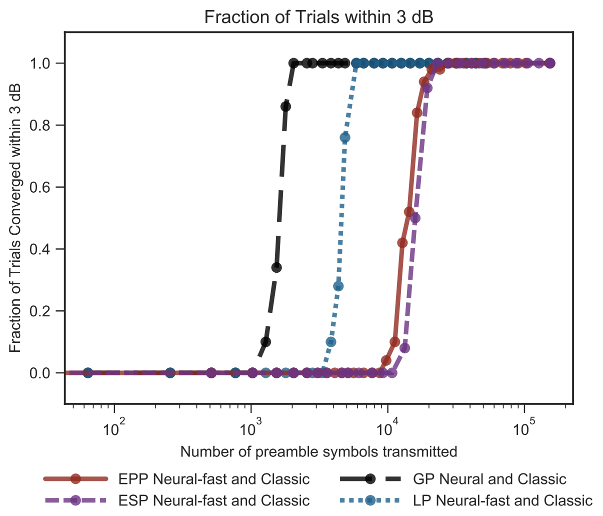
Figure 2.9 compares the performance of the GP, LP, ESP and EPP protocols for a Neural agent learning to communicate with a Classic agent. From the round-trip BER curves in Figure 2.9a, we observe that all protocols achieve round-trip BER close to the QPSK baseline. Figure 2.9b shows the EPP and ESP protocols have similar convergence behavior but are an order of magnitude slower than the GP and LP protocols. All protocols lead to robust convergence. Furthermore, comparing against Figure 2.8b, we see that learning to communicate with a Classic agent is much easier than learning to communicate with a clone learning agent. Table 2.3 shows a difference in convergence speed of up to 5.5$\times$ between these two cases when using the EPP and ESP protocols. This matches what we expect intuitively since when both agents are learning, each agent is trying to improve its own behavior and *simultaneously* track the behavior of the other agent. When one agent is fixed, the learning agent only has to match a static behavior. Graphical results for a Poly agent learning to communicate with a Classic agent can be found in Appendix C.

Learning with Self-aliens

Figure 2.10 compares learning with clones to learning with a self-alien for Neural agents using the EPP protocol. Here, we have one “fast” agent and one “slow” agent, defined based on speed of convergence when paired with a clone. From the BER curves in Figure 2.10a,



(a) Round-trip median BER. The error bars reflect the 10th to 90th percentiles across 50 trials. All agents are evaluated at the same SNR, but error bars have been dithered for readability.



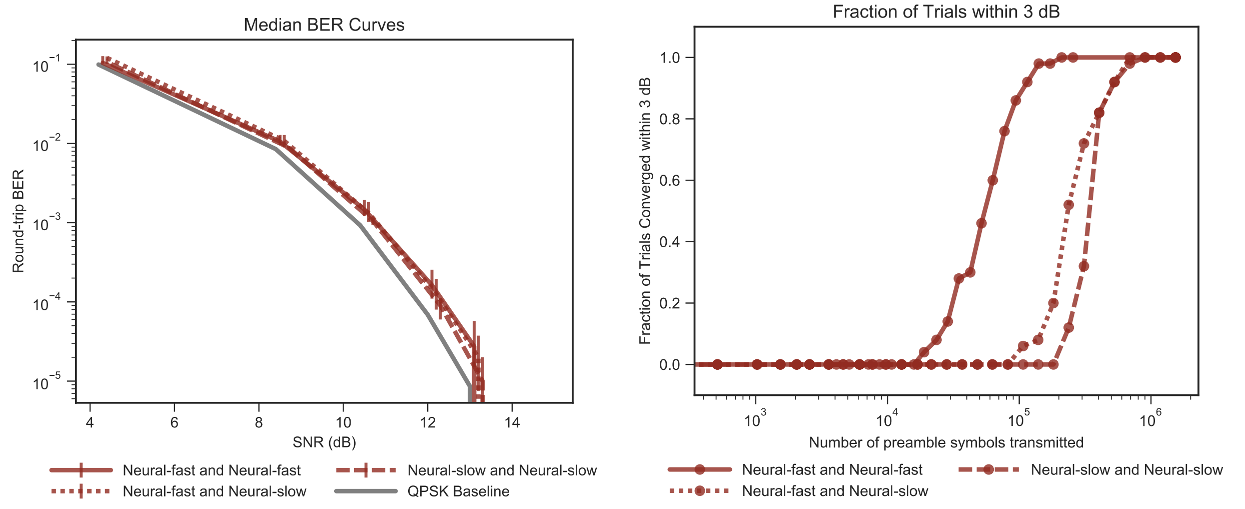
(b) Convergence of 50 trials to be within 3 dB-off-optimal at testing SNR 8.4 dB.

Figure 2.9: Learning to communicate with a fixed agent, Neural-fast-and-Classic. The BER plot (a) shows that all protocols achieve BER close to the QPSK baseline. From the convergence plot (b), we observe that EPP and ESP have similar convergence behavior and are an order of magnitude slower than LP and GP. For the ESP and EPP protocol, convergence is much faster than when learning with a clone (Figure 2.8).

we see that all cases achieve round-trip accuracy close to the QPSK baseline. From the convergence plot in Figure 2.10b, we make an interesting observation. The fast agent helps the slower agent to learn more quickly, resulting in convergence times for the self-alien pairing in between those of the clone pairings. This is very encouraging since it suggests that not only is learning with self-aliens possible, it can also be faster than learning with clones for a slow agent. We repeated this experiment using Poly agents and found similar behavior, shown in Appendix C.2.

Learning with Aliens

Next, we compare learning with aliens to learning with clones. Figure 2.11 depicts the results for the case where a Neural and Poly agent that show similar convergence behavior when learning with a clone are paired with each other. From the BER curve in Figure 2.11a, we see that the Neural agent when paired with a clone has slightly lower BER than the Poly agent paired with a clone. More importantly, the BER for the alien pairing is very similar to the others and close to the QPSK baseline. From the convergence plot in Figure 2.11b, we see that when the two clone pairings show similar convergence behavior, the alien pairing



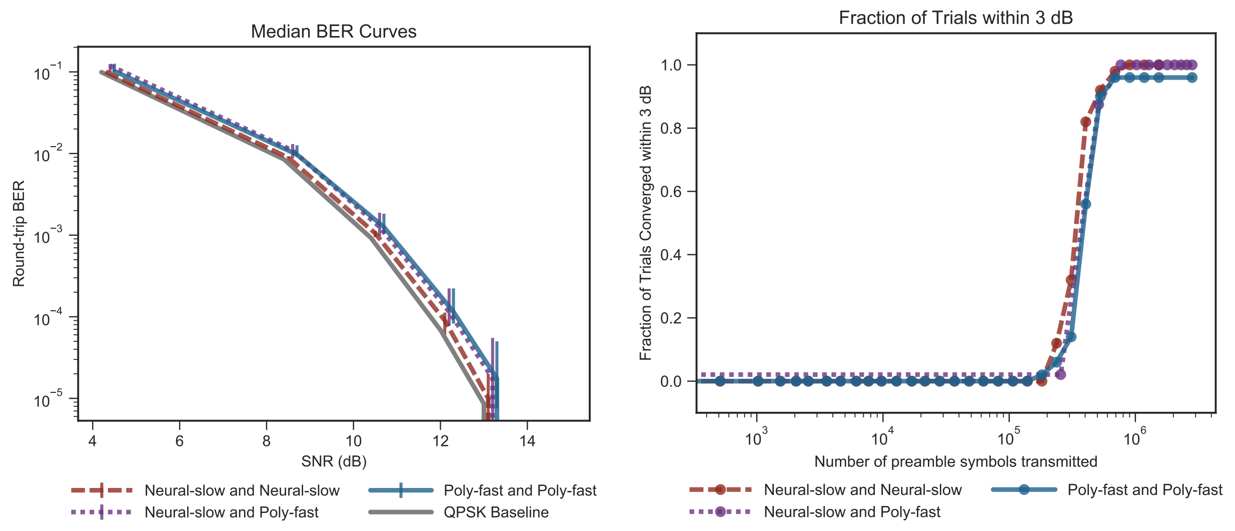
(a) Round-trip median BER. The error bars reflect the 10th to 90th percentiles across 50 trials. All agents are evaluated at the same SNR, but error bars have been dithered for readability.

(b) Convergence of 50 trials to be within 3 dB-off-optimal at testing SNR 8.4 dB.

Figure 2.10: Learning with clones (Neural-fast-and-Neural-fast, Neural-slow-and-Neural-slow) compared to learning with self-alien (Neural-fast-and-Neural-slow) using the EPP protocol.
The BER plot (a) shows that the round-trip BER for Neural-slow-and-Neural-slow is slightly lower than the others, but all BERs are close to the QPSK baseline. From the convergence plot (b), we observe that Neural-fast-and-Neural-fast is much faster than Neural-slow-and-Neural-slow. However, pairing Neural-fast with Neural-slow helps the latter learn faster, resulting in a convergence time for Neural-fast-and-Neural-slow between those of the clone pairings.

does not deviate. This is another main result of this work. **Learning to communicate with an alien agent using EPP is not intrinsically more difficult than learning with a clone**.

In the next experiment we investigate whether it is possible for an agent to learn to communicate with an alien agent when the two agent types have vastly different convergence behaviors while learning with clones. From the BER curve in Figure 2.12a, we see that the two clone pairings have round-trip error rates close to the QPSK baseline. Interestingly, Figure 2.12b shows that the alien pairing always learned a good modulation scheme even though the Poly-slow-and-clone pairing occasionally failed. The alien pairing convergence speed lies in between the two clone pairings. These results suggest that a fast, reliable agent can help a slow agent, alien or not, learn more quickly and more robustly. This phenomenon is not unique to the EPP protocol, as can be seen from results with the ESP protocol in Appendix C. Another interesting result from this experiment found in Table 2.3 is that for ESP the difference between convergence times for Neural-and-Classic and Neural-and-Clone



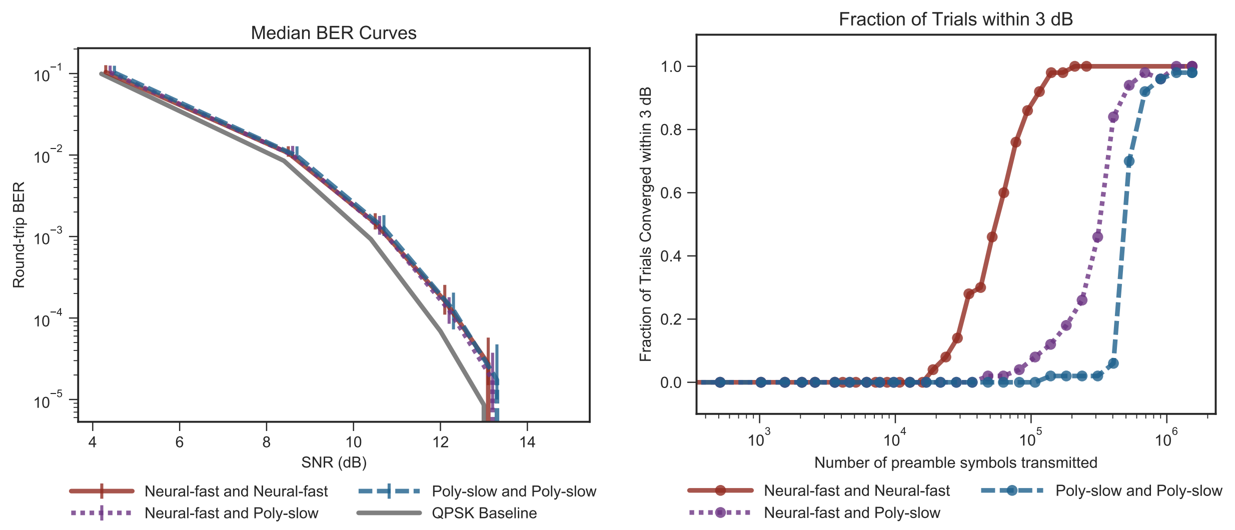
(a) Round-trip median BER. The error bars reflect the 10th to 90th percentiles across 50 trials. All agents are evaluated at the same SNR, but error bars have been dithered for readability.

(b) Convergence of 50 trials to be within 3 dB-off-optimal at testing SNR 8.4 dB.

Figure 2.11: Learning with clones (Neural-slow-and-Neural-slow, Poly-fast-and-Poly-fast) compared to learning with an alien (Neural-slow-and-Poly-fast) using the EPP protocol. Here, the Neural-slow and Poly-fast agents have similar convergence behavior when paired with a clone. The BER plot (a) shows that in all cases, the BER achieved is close to the QPSK baseline. From the convergence plot (b), we observe that when both clone parings show similar convergence behavior, the alien pairing also has the same behavior. Learning to communicate with an alien is not intrinsically more difficult than learning to communicate with a clone.

(and similarly for Poly-and-Classic and Poly-and-Clone) is smaller than for EPP. This could be partially because the hyperparameters were tuned for the EPP agent-and-clone setting, but also suggests that as we increase information sharing, the performance gap between learning with fixed agents and other learning agents shrinks.

We can now provide answers to the questions we raised at the start of this subsection. It is possible to learn to communicate with self-aliens and aliens using the EPP protocol. In fact, neither of these tasks is intrinsically more difficult than learning with a clone. Furthermore, a self-alien/alien pairing shows convergence behavior between the two clone pairings. A fast agent paired with a slower agent can help the slow agent learn faster. An interesting experiment would be to map out the conditions where these observations continue to hold. Can learning become impossible if the difference in convergence behavior of the two agents is large enough? Can two agents have convergence behaviors that are similar, but differ so fundamentally in the way they learn that they fail to learn in the alien setting? We leave this as an area for future research.



(a) Round-trip median BER. The error bars reflect the 10th to 90th percentiles across 50 trials. All agents are evaluated at the same SNR, but error bars have been dithered for readability.

(b) Convergence of 50 trials to be within 3 dB-off-optimal at testing SNR 8.4 dB.

Figure 2.12: Learning with clones (Neural-fast-and-Neural-fast, Poly-slow-and-Poly-slow) compared to learning with an alien (Neural-fast-and-Poly-slow) using the EPP protocol. Here, the Neural-fast and Poly-slow agents have vastly different convergence behavior when paired with a clone. The BER plot (a) shows that in all cases, the BER achieved is close to the QPSK baseline. From the convergence plot (b), we observe that when the clone pairings have vastly different convergence behavior, the alien pairing shows convergence behavior somewhere in between. All alien pairings converged to within 3 dB-off-optimal, even though the Poly-slow with clone failed at least once. The robustness of the Neural-fast learner may have helped the Poly-slow agent converge more reliably.

2.8 Implementation in Software-Defined Radios

To corroborate our simulation results, we implement the ESP (2.4) and EPP (2.3) protocols on Ettus USRP software-defined radios (SDRs) using GNU Radio [229]. The goal of this implementation is not to provide a real-time implementation of the Echo protocol since, in general, the real-time components of radio communications are implemented in ASICs, and even software components are run in special real-time operating systems to achieve deterministic or bounded latencies. The focus of this work is to learn modulation schemes, so the primary goal of the GNU Radio implementation is to demonstrate that the learning protocols work not only in simulations but also when trained in real, physical systems with all the imperfections that implies. Other work such as [204] and [205] has also demonstrated that end-to-end learning of communication schemes is possible over the air in real radio systems. Other components of radio communications, such as channel equalization and error correction coding, are left for future work. Only after all of these processing components have been addressed will it be necessary to have real-time hardware implementations of components such as modulation learning.

Additional Processing

The GNU Radio implementation attempts to abstract away the details of packet transmission, reception, and non-AWGN channels to provide as close an approximation as possible to the training environment of the previous sections. The implementation corrects for carrier frequency offset (CFO), multitap channels, and arbitrary packet arrival times using several algorithms implemented with NumPy [230]. We detect packets using correlation against a fixed prefix and constant false alarm rate detection [231]. CFO and channel effects are corrected using the same prefix. We perform coarse sample timing synchronization by upsampling to two samples per symbol for transmission, then downsampling after the start of the packet has been detected.

The additional processing adds significant overhead to each round-trip training cycle. The results from a typical run with a 50-unit single hidden layer modulator and demodulator and 256 symbols per preamble are shown in Table 2.4. As shown in the table, the packet wrapper comprises more than one-third of the execution time during a run. In addition to the computation time, the GNU Radio implementation introduces latency by sending data between packet processing blocks and modulator or demodulator blocks.

	Neural Modulator	Neural Demodulator	Packet Processing
Median Processing Time (ms)	13.4	26.1	20.6
Percent of Execution Time	22.3	43.3	35.8

Table 2.4: Average execution times for Neural agent training update and GNU Radio wrapper processing.
The additional processing required for transmission over USRP radios is about 1/3 of the total execution time. These times do not account for the additional latency of moving data between components of the GNU Radio processing chain.

Training Procedure Modifications

Constraints introduced by running on physical radios required several changes to the Neural agent training procedure before we could successfully train these agents. The constraints and the modifications necessary to overcome them are detailed in Sections 2.8 and 2.8.

Maximum Transmit Amplitude

Signals sent through USRP radios cannot exceed a maximum amplitude, and any signals sent to the radio that exceed this amplitude are silently clipped to the maximum amplitude. However, the EPP implementation in our simulations only restricts the average energy of a constellation. This means that any individual constellation point can have almost arbitrarily large amplitude, and exploration can drive the amplitude of a transmitted symbol even higher. It turns out that clipping a significant number of transmitted symbols breaks the training process for neural modulators, and they never converge to a reasonable constellation. To prevent clipping, we restrict the average power of a constellation during training to significantly less than the radio's cap and rely on the vast majority of symbols that are not clipped to produce good training feedback.

Because we control the environment for the tests, we can ensure that we train and test at the desired SNRs for any given constellation. However, in the real world, a system may need to use all of its transmit power to achieve a usable SNR. In such a case, restricting the average power of a constellation to less than the maximum would prevent learning from taking place. we hope to address the problem of exploring out to a bounding box, without exceeding it, while maintaining training performance in future work. Empirically, we have observed that allowing transmit symbols to exceed the bounding box but applying a penalty when outside the box results in oscillation about the limit as the penalty 'strobes' on and off.

DC Offset Correction

USRP radios use an adaptive DC offset canceler in the receive chain, which causes the IQ that the demodulator eventually receives to be centered around the origin, regardless of the originally transmitted constellation. However, the base Echo implementation does not place any restriction on the mean of a constellation. The most energy-efficient constellation possible is always centered at the origin, so the constellations achieved after training are approximately centered at the origin as well. Unfortunately, the constellation center commonly moves far from the origin during the training process before being forced back as the constellation is optimized. This causes a significant DC offset in the transmitted signal. The receive chain DC offset corrections change the round-trip feedback that a modulator receives significantly enough that neural modulators fail to train.

The adaptive DC offset cancellation can be disabled, but this would require a calibration period at the start of each run, or even after each received packet, to measure the true DC offset and set the DC offset canceler manually. Instead, we explored methods of forcing the constellations to be approximately centered while training. We settled on a loss term for the squared magnitude of the constellation center — this was done individually at each agent and so did not violate the spirit of the problem. See Appendix D for more details.

Experiments

The radio experiments were conducted using two Ettus USRP X310 SDRs connected to each other with SMA cables, as shown in Figure 2.13. 75 dB of attenuation was added between the radios both to simulate path loss and to be able to achieve desired SNRs with the available internal transmit and receive gains.

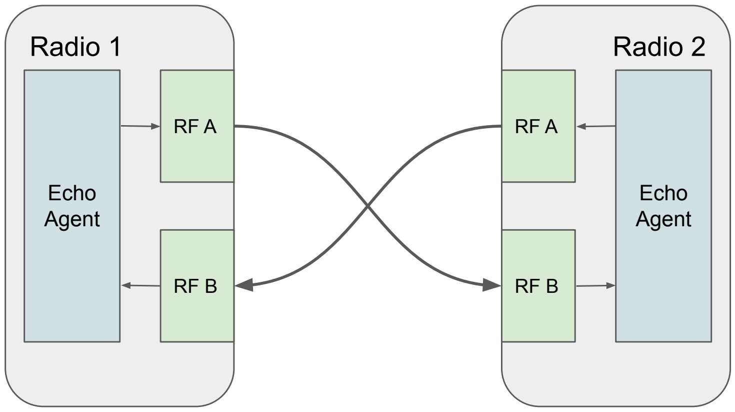


Figure 2.13: USRP radio connections.

We tuned hyperparameters for the radio experiments separately from the main simulation hyperparameters because of the extra hyperparameter introduced for DC offset correction. We use these same hyperparameters in simulations when comparing with the radio experiment results. After coarse hand-tuning, we achieved performance similar to Neural-slow-and-clone. The hyperparameters are listed in the Appendix H. Each experiment was run 20 times with random seeds at an SNR, which resulted in 1% round-trip BER for two classic agents.

Echo with Shared Preamble Comparison

Figures 2.14 and 2.15 compare the performance of the GNU Radio implementation to simulations for ESP neural-clone training. Figure 2.14 shows that the additional processing required to handle channel equalization and CFO correction requires 2 dB additional empirical SNR to achieve the same baseline BER performance for classic agents. In Figure 2.14, the agents trained on SDRs perform slightly worse relative to the baseline than agents trained in simulation. Figure 2.15 shows that learning agents train at approximately the same rate on SDRs as in simulation. Although the simulation curve comes from sampling one set of seeds over time as they train, each data point on the software radio curve comes from a separate set of seeds trained for a given amount of time. There is some variance in how many seeds eventually converge, which causes the droop in the curve around 600,000 symbols transmitted. For the ESP case with neural agents, the simulated performance is similar to that obtained while using SDRs. This is evidence in support of Echo style protocols being practically implementable procedures for learning to communicate.

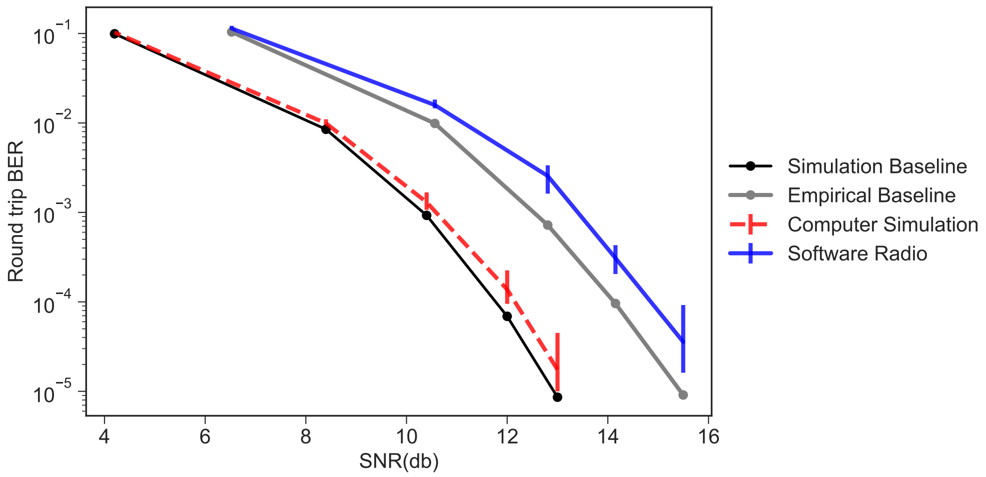


Figure 2.14: Round-trip median bit error curves for Neural-and-Clone python simulation and GNU Radio agents learning QPSK under the ESP protocol at training SNRs corresponding to 1% BER.
Alongside the bit error curves of the learned modulation schemes is the baseline. In all cases, modulation constellations are constrained as detailed in Section 2.8. Although 2 dB SNR extra is required to achieve the same baseline performance due to processing losses in the GNU Radio implementation, the trained agents show only slightly greater loss in performance against the baseline than the pure simulation agents.

Echo with Private Preamble Comparison

Figures 2.16 and 2.17 compare the performance of the GNU Radio implementation to simulations for EPP neural-clone training. Apart from the additional SNR required to achieve the same baseline performance, the trained neural agents show a similar spread in final BER performance across SNRs. This is another main result of this work: **EPP successfully learns modulation schemes over the wire using SDRs.**

Figure 2.16 compares the convergence rate for many trials with training time for the GNU Radio implementation to simulation. Clearly, it takes longer for the GNU Radio agents to converge to 3 dB off of optimal BER than the simulation agents, but the final proportion of successful trials is similar. We speculate that there may be more noise in the feedback given to agents during the GNU Radio training process than in the simulation training. One potential source is quantization noise, which we had to address in the experimental setup by introducing additional channel attenuation, then increasing receive gain. This could slow down convergence by reducing the consistency of feedback without reducing its average quality, i.e., some very good feedback mixed with poor feedback. Over time the

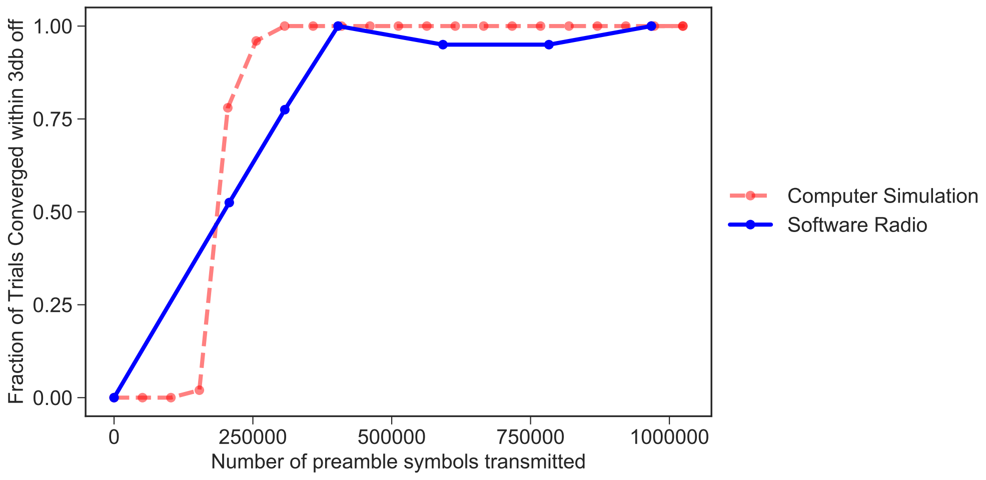


Figure 2.15: Convergence of 50 simulation and 20 GNU Radio trials to be within 3 dB-off-optimal (at testing SNR corresponding to 1% BER) of the corresponding baseline for ESP trials of neural agent and clone at training SNR corresponding to 1% BER for QPSK modulation.
The GNU Radio agents were only trained at 1% BER SNR, equivalent to SNR_dB=8 among the simulation curves. Unlike the simulation curve, which is sampled over time for one batch of agents, the GNU Radio data points come from separate batches with different seeds. The dip in performance around 600,000 symbols is a result of variance in how many seeds converge, not agents losing performance after they have initially reached the performance threshold.

good feedback would prevail, since it will be self-consistent, whereas the poor feedback will not be consistent and will eventually be averaged away. We will address this discrepancy further in future work.

2.9 Conclusion

In this work, we studied whether the Echo protocol enables two agents to learn modulation schemes with minimal information sharing and the impact of increasing supervision through information sharing via co-design. We proposed a variation of the generic Echo protocol, denoted EPP (Echo with private preamble), that assumes no shared knowledge apart from knowledge of the echo protocol and the ability to perform turn-taking. We believe Echo is the interactive counterpart to self-supervision. Self-supervision is a form of unsupervised learning where data modalities such as image orientation or sentence content are used to create labels for a pretext task, like ‘predict the image’s rotation’ or ‘fill in the missing word’ [232], [233]. In Echo’s case, the task is to recover your own message.

To evaluate the cost of minimal information sharing, we explored a range of protocols varying in the amount of information shared. We observed that reduced supervision via

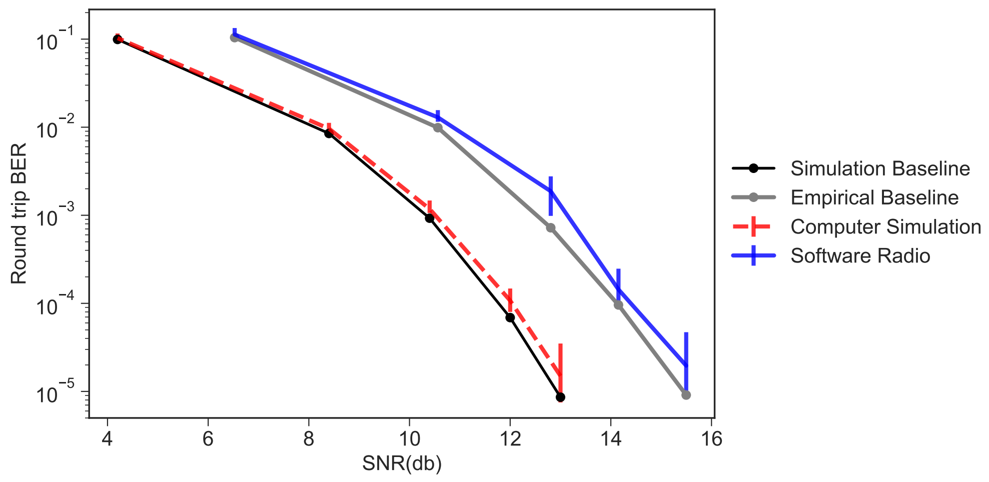


Figure 2.16: Round-trip median bit error curves for Neural-and-Clone python simulation and GNU Radio agents learning QPSK under the EPP protocol at training SNRs corresponding to 1% BER.
Alongside the bit error curves of the learned modulation schemes are the baselines. In all cases, modulation constellations are constrained as detailed in Section 2.8 to constrain the average signal power. Although 2 dB SNR extra is required to achieve the same baseline performance due to processing losses in the GNU Radio implementation, the trained agents show similar loss in BER performance compared to the baseline.

information sharing comes at the cost of slower convergence, meaning more symbols need to be exchanged before a good modulation scheme is learned. A learning agent paired with a clone can robustly learn a two bits per symbol modulation scheme in 2×10^3 symbols if we allow gradient passing and in 3×10^3 symbols if we allow loss passing. Restricting information sharing further, the number of symbols required to learn a scheme robustly goes up exponentially. Allowing only sharing of preambles takes 2.5×10^4 symbols while the case without shared preambles takes 10^5 symbols. It is important to remember that symbol transmissions take time. Even if learning occurs instantaneously, at one megasymbol per second, these values range from two milliseconds to a tenth of a second. Including learning and inter-packet spacing will inflate these values accordingly. As long as learning times remain short to human perception, there are many domains where a system can afford to replace fixed standards with learning components at the cost of an imperceptible one-time delay[6].

Despite the increase in sample complexity, our results show that even with minimal supervision through basic assumptions, agents can learn to communicate. The EPP protocol is universal, in that it allows agents of diverse types to learn to communicate with each other, and also works when one of the agents uses a fixed communication scheme.

[6]Even though the delay can be made short to human perception, it is large enough that we can not afford it for every packet, unlike channel sounding preambles.

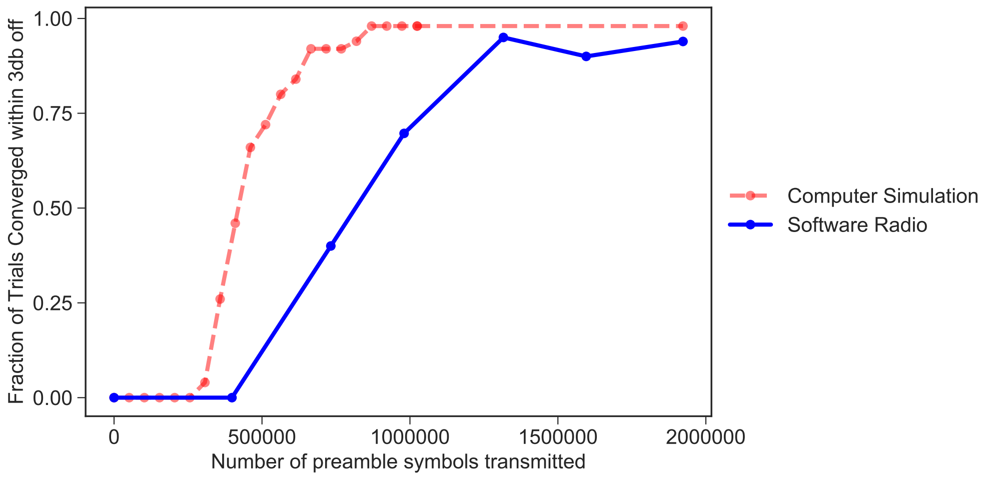


Figure 2.17: Convergence of 50 simulation and 20 GNU Radio trials to be within 3 dB-off-optimal (at testing SNR corresponding to 1% BER) of the corresponding baseline for EPP trials of Neural agent vs clone at training SNR corresponding to 1% BER for QPSK modulation.
The GNU Radio agents were only trained at 1% BER SNR, equivalent to SNR_dB=8 among the simulation curves. Note that the GNU Radio agents take longer to converge to within 3 dB of optimal, but after sufficient time a similar proportion of trials converge.

These results suggest that learning with "alien" agents is not intrinsically more difficult than learning with agents of the same type. For instance, with the learning agents Neural-slow and Poly-fast, the clone pairings (Neural-slow and Neural-slow, Poly-fast and Poly-fast) as well as the alien pairing (Neural-slow and Poly-fast) require a very similar number of training symbols of around 7×10^5 to robustly learn a modulation scheme. However, learning to communicate with an agent that uses a fixed modulation scheme is much easier with Neural-fast and Classic requiring only 10^4 symbols before a good scheme is learned.

In Appendix C, we investigated the performance of the learning protocols for higher modulation orders and noticed that the difficulty of the learning task increases substantially with modulation order. The number of preamble symbols that must be transmitted before a good scheme is learned increases exponentially. Confirming the results of others ([23], [204]), we observed that moderate levels of noise have a regularizing effect and facilitate learning but too much noise can be detrimental to the learning process.

Overall, learned modulation schemes have a high cost in complexity and some cost in loss of optimality for AWGN channels, relative to a human-designed optimal scheme. For simple channels, it is possible to design a scheme that is provably optimal and has no cost in time spent converging to a common method. Future work should extend this learning protocol to more complex channels for which optimal schemes are not known. If it can achieve near-optimal performance for a simple channel, there is hope it will also perform well on a harder channel.

This work raises intriguing questions and opens up several exciting new avenues of research. On the universality of the learning process, one might wonder if it is always possible for two alien agents to learn to communicate with each other when each has the ability to learn to communicate with a clone. What happens when these two agents have vastly different convergence behaviors in terms of how fast they learn, measured in terms of the number of preamble symbols transmitted? Is learning still possible? Or is there something fundamental to the learning process that determines whether two agents can learn when paired together and two agents with seemingly similar convergence behavior can fail to learn to communicate with each other because their inherent learning behavior is different?

Future work will involve relaxing some of the most restrictive assumptions of this work. Although the EPP protocol aims to share as little information as possible, currently, we assume a fixed and known number of bits per symbol. Removing this assumption would be a further step towards a complete learning protocol. In this work, a single pair of agents take turns in perfect order, but in real-world environments there are likely to be many agents with imperfect turn-taking. It will be important to explore how the Echo protocol works with multiple agents, and when agents do not always echo the most recent message or even echo at all.

Meta-Learning and Foundation Models for Inititalization

Meta-learning techniques have shown promise for decreasing the test-time sample complexity of learning tasks and have enabled few shot learning in several applications [84], [108], [234]. In PHY applications, meta-learning has been used for equalization [48], [49], demodulation [50], predict channel fading [235], adapt FEC codes [84], and learn end-to-end communication models [32]. Can we apply meta-learning techniques to initialize learning agents in a favorable state that allows them to learn to communicate with others much faster than when initialized to a random state? Alternately, foundation models are trained to be able to reliably perform many tasks when prompted. They are a counterpart to meta-learning, where most training happens before test time, and prompts induce task-specific outputs.

Indeed, the possibilities of meta-learning and foundation models open up an entire line of research related to initialization and collaboration between devices in the modern edge-cloud paradigm. Imagine a system where devices “booting up” in a new environment can query the cloud for a good initial model to use with other devices in the local area so they can vastly reduce the time required to optimize their communication behaviors. Then, the device can publish its fine-tuned model back to the cloud infrastructure for use in a federated or distributed learning algorithm to improve the base initialization for that local area. Perhaps a pool of contributed fine-tuned models is maintained and used with a set of simulated channels to update a “master” model that works with everyone in the pool instead. Such a system would require answers to many questions. How do we learn a good initialization? How specialized does an initialization need to be, i.e., can we use one global version or many versions specialized for specific environments? What is the fallback if a device loses access to the cloud infrastructure? How do we include devices with varying amounts of processing power and adaptability?

Can other parts of the communication pipeline, such as equalization and error-correcting codes, be integrated into the learning process? Can all these processing stages be learned end-to-end, and does that provide a benefit in terms of training time or communication performance? End-to-end training might allow us to discover new communication strategies for certain types of channels that beat the current best-known strategies for the channel. Alternately, are there certain portions of the processing chain that simply work better with pre-determined standard-based behaviors?

Potential Impact on Future Standards

Historically, communications standards are created by large committees involving players from the government, industry, and academia. The standards have to specify *nearly* every single behavior of a radio and every corner case to ensure interoperability — the true goal driving the creation of a standard. The few aspects of a communication standard that are left unspecified are components that, first, do not affect interactions with other agents and, second, have incentives for anyone implementing the standard to get it right. One example is equalization algorithms for the receiver. Pilot symbols are completely specified so everyone knows what to expect, but the exact equalization algorithm (e.g., zero forcing, MMSE equalization, or something more complicated) is left open to the implementer. As a device maker, poor equalization translates to poor reception and, thus, worse performance than a competing product. Thus, every device maker will implement a working equalization algorithm because they face market incentives to do so.

Now that AI agents are nearly everywhere, assisting in coding, writing, creating images, and more, a natural question is how AI can be integrated into standards and the standards creation process. I envision a future of "light standards," where standards provide only a framework of behaviors for successful communication, and many of the details are left to be learned by radios in situ. This would allow for faster updates of standards themselves, which normally take years per iteration, and potentially faster iterations of devices because fewer operations would be baked permanently into hardware. As mentioned previously, some algorithms are left unspecified in standards already. A major direction for future work in this area is what other algorithms can now be left unspecified for an AI component to learn, relying on the implicit shared goal of cooperative communication to produce a successful outcome. How much performance is gained or lost by introducing learning components at each part of the PHY and higher layers? Is it more effective to have a fully specified base protocol that bootstraps more powerful learned components, essentially enabling a feedback path more like LP instead of ESP?

Integrating the cloud into the learning parts of a standard also offers new opportunities for accelerating learning. Even if each device maker operates their own siloed cloud backend for their devices, this opens up questions on what kinds of cooperative multi-agent exploration and learning with centralized components can help versus the fully decentralized model in this work. Perhaps cloud-based cooperative learning can become another component of light AI-enabled standards to let all participants share training data or models for mutual benefit.

Looking ahead, integrating AI-driven learning protocols into communications standards represents a fundamental paradigm shift in how we approach wireless system design and implementation. The demonstrated success of the Echo protocol, particularly in its EPP variant, suggests that future communications systems could effectively balance predetermined frameworks with adaptive learning capabilities, potentially revolutionizing how devices interact and optimize their behavior in diverse operational environments. This evolution toward "light standards" and learning-enabled protocols may ultimately lead to more resilient, adaptable, and efficient communication systems that can autonomously optimize their performance across varying channel conditions and operational scenarios. Combined with sensing and sharing information about the wireless environment (covered in the next chapter), this line of work is a significant step towards true cognitive radios [17]. However, realizing this vision requires careful consideration of the tradeoffs between flexibility and interoperability, as well as systematic investigation of the optimal balance between fixed protocols and learned behaviors across different layers of the communication stack. The results presented here serve as a foundation for this broader research agenda, demonstrating both the feasibility and challenges of implementing learning-based approaches in next-generation communication systems.

Chapter 3

Environmental Supervision for Automatic Sensing and Calibration

3.1 Introduction

The dynamic nature of modern radio spectrum usage presents unprecedented challenges and opportunities in wireless networking. At the macro level, spectrum is a limited resource, and it can be quite costly to relocate a large installed base of existing systems, leading to increased interest in spectrum sharing [122], [236]. At the micro level, modern wireless protocols such as 5G New Radio [237] include unprecedented control over frequency and time allocations per user to meet the growing demands for data and user capacity with limited infrastructure. The cognitive radio concept [17], involving both communication and cooperation between radios about usage and non-cooperative sensing of the environment, is well-positioned to address the challenges and take advantage of the opportunities the modern spectrum sharing paradigm represents. At a high level, all of the decisions a cognitive radio or wireless network makes rely on an accurate and precise understanding of the environment around a node, both physically and with regard to other users and demands for capacity.

The most fundamental limitation to the performance of a radio is the channel through which the signal is transmitted [185]. With perfect knowledge of the channel, a radio could make optimal decisions about spectrum usage and modulation, coding, and other algorithm selections. Alas, we live in an imperfect world and thus must rely on our own observations of the world around us, feedback from our communication partners such as CSI measurements [110], [120], [121], [237], and higher level measurements of channel usage and conditions from spectrum monitoring networks. A cognitive radio would use each of these data sources to supervise its decision-making process. The techniques introduced in this chapter are primarily intended as enablers for practical distributed spectrum monitoring networks (e.g. [238]–[240]) and radio nodes operating in regulated spectrum sharing bands such as CBRS and Wi-Fi 6E [128][1].

[1]This chapter is based on work published in [241]–[243] and additional unpublished efforts.

The key insight motivating the work in this chapter is that signals of opportunity, despite being non-cooperative, provide insight into both the environment around a sensor (radio) node and the performance or characteristics of the node itself. Using signals of opportunity requires taking a step beyond the initial "spectrum sensing" style of measurement, where no or very minimal assumptions are made about the signals present in an environment. This is sufficient to measure characteristics like occupancy [123] but throws away much information. By taking advantage of information contained in publicly broadcast signals like ADS-B, we can gain additional information like the transmitter's location useful for further processing [244]. Even signals with unknown modulations, proprietary encodings, or encrypted data will have physical properties common to all communication signals, such as cyclostationarity or regularly spaced frequency content, that can be used to extract useful information [146], [147], [243]. I view this as a form of 'light supervision' in the sense that SoOps are not designed or broadcast to help in the purpose we, as opportunists, are using them for. Thus, they require additional processing to be useful or provide imperfect assessments of a measurement of interest. SoOps, by nature, may not be present at all times or locations desired, providing only sparse (read: light) supervisory signals.

Combining information extracted from signal content or properties with outside sources, such as databases of FM, TV, and cellular transmitter locations and frequencies, allows new follow-on applications and new perspectives on SoOps and distributed sensing networks. Two topics of particular interest are automatic sensor evaluation and calibration, and automatic metadata generation. Automatic sensor evaluation and calibration uses SoOps to provide reference signals, which can provide information on sensor frequency ranges and sensitivities, location and pointing, and field of view [241], [245], [246]. These figures are primarily of interest in distributed spectrum sensing networks, especially crowd-sourced networks where untrusted participants operate nodes. Automatic metadata generation employs SoOps to measure and track system metadata which is useful at startup and during operation. Examples include oscillator frequency offsets [137], [138], presence indoor or outdoor [242], [247], [248], and location without access to GPS [141], [142], [146], [147], [243], [244]. These values are more broadly applicable to any network node with a radio receiver. The measurements provided by automatic sensor evaluation and calibration are also metadata, but they are more specialized to spectrum sensing applications. The common term between these two topics is *automatic* because SoOps are used to replace or supplement human input — another aspect of light supervision. Replacing human input reduces the cost of deploying sensor networks, a major factor when an application such as spectrum sensing requires coverage of a large geographic area. Supplementing human input leads to a novel viewpoint on user-supplied metadata as just another noisy input that requires supervision, as discussed in Section 1.3. Both of these topics are explored and validated through the experiments in the rest of this chapter.

Automatic Sensor Evaluation and Calibration

"This radio node appears not to be working, but it has power, and the software and networking seem to be operational. Is the antenna broken? Has it been moved inside a closet? Do we have to send a technician to see?" This is a practical problem that any distributed system involving radios might face, and it only worsens with scale. Fortunately, it turns out that SoOps are well-positioned to automatically evaluate radios vis-a-vis their relationship to the external environment. Not only is this capability likely to be practically useful for many generic distributed wireless systems, it is especially useful in the context of radio nodes dedicated to spectrum monitoring.

Spectrum monitoring will play a crucial role in tomorrow's wireless networking landscape. By continuously monitoring spectrum, regulatory authorities, service providers, and businesses can ensure the quality and reliability of their wireless networks. Spectrum monitoring also facilitates the detection of unauthorized or illegal transmissions, aiding in the enforcement of regulations and the prevention of harmful interference. Some recent examples are the potential for interference from 5G cellular networks in airplanes' radio altimeters [249] and GPS systems [250]. Data on true interference levels provided by spectrum monitoring is crucial for understanding and resolving such situations. By providing valuable insights into spectrum usage patterns, spectrum monitoring supports the planning and deployment of future wireless networks, enabling the seamless operation of various applications and services that rely on wireless connectivity, such as telecommunications, IoT, and public safety.

Covering large geographical areas continuously presents significant challenges when performing spectrum monitoring [251], [252]. The sheer scale requires a substantial deployment of monitoring equipment and resources, which can be costly and time-consuming. As a result, distributed monitoring has been a long-standing problem in this area. Crowd-sourcing holds significant potential in spectrum monitoring due to its ability to engage many participants to cover large geographical areas [253]. In this approach, participants set up a spectrum sensor node, such as a Software-Defined Radio (SDR), that captures and transmits spectrum-related information to the cloud. We envision a distributed system in which node operators offer spectrum sensing as a service, and users pay to rent these services from operators. Spectrum sensing in this system would not necessarily be limited to band occupancy. Users could ask for IQ streams from a node to perform their own processing or temporarily extend reception of their wireless networks.

A key problem hindering the realization of this idea is how users can trust the quality of data offered by each operator. One important factor that impacts the quality of such sensors is potential obstruction of the antenna in relation to signal sources. Since random people around the world set up sensors, no assumptions can be made about the quality of the setup. Furthermore, since node operators are paid for services, there is a potential incentive to provide fabricated or incorrect data to receive reimbursement. For instance, a provider can claim that the sensor antenna has an unobstructed view of the horizon while the antenna is sitting on a desk indoors or that they can receive frequencies that their antenna does not cover. Now imagine that a user requires multiple sensors to cover a region. Manually testing each individual node quickly becomes impractical, presenting a significant scaling issue.

Section 3.4 details experimental results for automatic frequency sensitivity and FoV evaluation, relying only on IQ data captured by a sensor node without any human supervision. These capabilities provide an automatic mechanism for sensor node operators to verify that their setup receives signals it should after installation. It can also be used for system health monitoring during operation. For example, all software may show correct operation, but if the system suddenly indicates that FoV has reduced to zero, there may be a loose or damaged cable. System operating health monitoring is also useful for organizations that provide equipment that is installed by third parties and distributing sensing networks in general [254], [255]. FoV estimates also provide valuable information for spectrum sensor networks intended to monitor geographic areas, showing gaps or overlaps in coverage that may not be obvious from propagation simulations. A motivating use case for this feature is a government agency that wishes to monitor for illegal transmissions or transmissions interfering with critical infrastructure.

The key idea behind this automatic evaluation system is processing SoOps received by a sensing node from mobile sources with known locations in various frequency bands alongside signals from stationary broadcast emitters with known locations. Mobile signals sweep across large swathes of the environment around a sensor, while stationary transmitters are generally higher power and easy to receive but geographically sparse. The mobile signals currently processed are ADS-B messages transmitted by nearby airplanes and wireless signals transmitted by LEO satellites.

ADS-B messages inform air traffic controllers about the location, speed, and identity of aircraft alongside other information. ADS-B operates at a frequency of 1090 MHz and relies on line-of-sight communication. Consequently, any obstruction significantly degrades the signal. Since airplanes fly in all directions, we can assess the reception capability of a stationary sensor node from various angles. LEO satellites are becoming very popular with large constellations being launched by companies such as StarLink. LEO satellites use a variety of frequency bands to transmit their data back to the Earth. In this paper, we utilize satellite transmissions in the 137 MHz band. The precise location of these satellites can be computed at any point in time; therefore, they are an excellent candidate as a mobile signal source with a known location. The advantage of satellites over airplanes is that they cover every point on Earth. To establish trust in the quality of data provided by a sensor node, this work compares the received signal with ground truth information from various perspectives. We match the aircraft identifier obtained from ADS-B messages against the ground truth acquired from a flight tracking website to assess the accuracy of the sensor data. Similarly, we compare satellite data, such as their location and speed and the theoretically induced Doppler shift, with empirical results to validate data fidelity.

The primary contributions of this portion of work are:

- Designing and implementing an end-to-end system that utilizes existing wireless signals to automatically evaluate the quality of data provided by nodes in a network of spectrum sensors without human supervision.

- Leveraging ADS-B messages from nearby airplanes and weather image data transmitted by NOAA weather satellites to determine the presence of obstructions around a sensor node.
- Utilizing signals across various frequency bands to assess the impact of obstructions on a sensor node's reception in a frequency-specific manner.

Automatic Metadata Generation

The rapid evolution of wireless communication systems and increasing spectrum congestion have created an urgent need for sophisticated metadata verification mechanisms in distributed sensing networks. As regulatory authorities, service providers, and businesses deploy spectrum monitoring solutions to ensure network reliability and prevent harmful interference, the traditional paradigm of manually verified metadata becomes increasingly inadequate. Recent incidents involving potential interference between 5G networks and aircraft radio altimeters [249], as well as GPS systems [250], underscore the critical importance of accurate, verifiable spectrum monitoring data.

The challenge of metadata verification becomes particularly acute in the context of spectrum sharing frameworks, where new applications must coexist with incumbent users while minimizing harmful interference. Contemporary approaches to spectrum sharing, exemplified by the Citizens Broadband Radio Service (CBRS) and Wi-Fi 6E deployments, rely heavily on centrally managed environment sensing and spectrum allocation and self-reported parameters to manage transmission power limits and prevent interference. Wi-Fi 6E systems currently depend on primitive mechanisms such as user self-reporting and physical weatherization restrictions, which are likely insufficient for ensuring reliable operation and regulatory compliance.

We propose a paradigm shift toward automatic metadata generation and verification, treating human-reported parameters as potentially noisy inputs rather than canonical truth. This approach leverages SoOps to verify and generate crucial system metadata automatically. The fundamental concept draws from wireless reciprocity principles, utilizing outdoor signals as proxies to understand propagation characteristics and environmental conditions, even in the absence of perfect reciprocity between transmitters and receivers.

This methodology exploits multiple signal sources, including ADS-B transmissions from aircraft and signals from LEO satellites, to enable comprehensive metadata verification. ADS-B signals, operating at 1090 MHz, provide particularly valuable information due to their ubiquity and line-of-sight propagation characteristics. The rich multipath environment inside buildings creates distinctive temporal patterns in ADS-B signal strength, enabling accurate indoor/outdoor classification with 85% accuracy and an ROC AUC of 0.72. Similarly, LEO satellite signals, particularly from NOAA weather satellites, exhibit unique Doppler signatures that enable precise location verification through cross-referencing with claimed positions. The proof-of-concept Doppler location estimates in this work do not meet the

±50 meter requirement for verifying regulatory compliance [128], but other work on Doppler positioning achieves accuracy in the 10s of meters with additional signal processing [146].

The feasibility of this approach has been significantly enhanced by advances in software-defined radio technology and reduced costs in RF data acquisition and processing. Modern edge devices and cloud computing infrastructure enable sophisticated signal processing chains and machine learning techniques that were impractical when many current systems were designed. This technological evolution allows for the replacement of expensive specialized hardware, such as GPS receivers and calibrated antennas, with software-defined solutions that leverage ambient signals for metadata verification.

This research demonstrates that signal strength variations and propagation characteristics can reveal subtle environmental features that transcend simple binary classifications. For instance, traditional indoor/outdoor categorizations prove inadequate when considering scenarios such as window-adjacent installations, where a technically "indoor" device may still cause significant external interference. This observation suggests the need for more nuanced classification schemes that consider directional attenuation patterns and complex propagation environments.

This work represents a fundamental advancement in distributed spectrum sensing systems, building toward scalable, trustworthy deployments while minimizing human intervention in the verification process. The proposed methodology not only reduces operational costs but also enhances system reliability by providing continuous, autonomous verification of critical metadata parameters. As spectrum-sharing frameworks continue to evolve, these capabilities will become increasingly essential for ensuring efficient spectrum utilization while maintaining robust interference protection for incumbent users.

The methodology presented here opens new avenues for research in autonomous system calibration and verification. Future work might explore additional signals of opportunity across diverse frequency bands, enabling more comprehensive metadata verification and enhanced environmental classification capabilities. This approach holds particular promise for emerging applications in spectrum sharing, regulatory compliance, and distributed sensing networks, where traditional manual verification methods prove increasingly impractical at scale.

3.2 Related Work

The evolving landscape of wireless communications has rendered traditional approaches to system verification and calibration increasingly inadequate. This section examines relevant prior work across multiple domains that inform our approach to automatic metadata generation and verification using signals of opportunity.

The emergence of tiered access systems like CBRS [256] and TV whitespace [257] has heightened the importance of cooperative and distributed spectrum sensing. However, much existing work. such as [253] and [258]. focuses on detecting or preventing malicious actors but does not address the limitations of poor siting or equipment setup. Prior work in distributed spectrum sensing networks has progressed from early efforts with specialized hardware [238] to contemporary architectures employing software-defined radios and cloud infrastructure [259], enabling access to IQ data and true open spectrum monitoring networks [239], [260]. Notable implementations include Electrosense [240] and RadioHound [261], which leverage inexpensive SDR front-ends for widespread deployment. However, these systems lack robust automatic calibration mechanisms, requiring manual verification that proves impractical at scale.

The concept of signals of opportunity, introduced by Hall [133], has emerged as a powerful paradigm for system characterization and environmental understanding. SoOps have been successfully employed across diverse applications, including passive bistatic radar illumination [134]–[136], sensor localization [145], [148], [262], phased array calibration [245], and oscillator error correction [137], [138]. These applications share foundational similarities with blind calibration techniques [263], where system parameters are derived from spatial or temporal relationships in a priori unknown signals [264], [265].

Within a decade of the first artificial satellites, the U.S. Navy developed a navigation system using Doppler measurements to achieve tens of meters accuracy [143]. [266] provides a historical overview of the development of dedicated satellite navigation systems which moved away from Doppler measurement toward precise time-based ranging. The advent of large constellations of LEO communications satellites, such as OneWeb, OrbComm, and StarLink, along with concerns over denial of GNSS signals, have resulted in renewed interest in Doppler-based positioning taking advantage of these signals of opportunity [262]. One challenge in taking advantage of signals of opportunity is accurately measuring their Doppler shift using only features inherent to the physical transmission, which is addressed in work such as [144], [145], [147] for digital signals. These demonstrations achieve down to 10 meter accuracy by compensating for ionospheric and atmospheric effects with precise and relatively expensive ($1000s) SDRs. This accuracy is well within the position reporting requirement of ± 50 m for CBRS stations [128]. Even more recent work has achieved similar accuracy for navigation on moving platforms with [262] and without [148] the aid of inertial navigation system measurements. Our work does not achieve comparable accuracy but is a proof-of-concept showing that low-cost hardware and simple processing can achieve useful accuracy, which more advanced signal processing would only improve.

The indoor/outdoor classification problem intersects multiple domains. Computer vision researchers have developed sophisticated techniques ranging from low-level feature extraction [267] to deep learning approaches [268], [269], with the resulting classification often used for downstream tasks [270]. GPS-based classification methods [271]–[273] face limitations in urban environments due to weak signal strength. [274] presents a case for using co-located signals found at other frequencies to understand the shadowing environment (what indoor/outdoor is trying to get at) at a frequency of interest. More recent work on indoor/outdoor classification using terrestrial signals of opportunity focuses on mobile devices that need not be co-located. There is a strong motivation for this work in enabling network operators to increase quality of service through seamless handovers to indoor femtocells [275], [276]. The ubiquitous nature of LTE, Bluetooth, and Wi-Fi networks, the primary networks mobile devices are equipped to sense, also facilitates their use here. [277] used LTE network performance metrics and ensemble learning methods to classify indoor/outdoor for an urban area. [248] includes both Wi-Fi and cellular metrics as inputs to tree-based ensemble classifiers. The authors also discuss the difficulty of classifying locations with large windows, as they can display outdoor signal propagation characteristics despite being technically indoors.

ADS-B signals are particularly valuable for system characterization, with research demonstrating their utility for indoor localization with accuracies of 20-30 meters [141], [142]. Wi-FLY [278] innovatively uses ADS-B broadcasts as control signals for opportunistic relay nodes. The mobile nature of aircraft provides diverse spatial sampling that static reference signals cannot achieve, enabling detailed reception pattern characterization across varied environments [241].

Wireless sensor networks have traditionally employed techniques relying on shared observations [279], [280] or opportunistic reference signals [281]. Modern participatory sensor networks often include trust and reputation mechanisms [282], [283], which could benefit from our proposed evaluation methodology, particularly in deployments where sensor coverage is unlikely to overlap.

3.3 Signal Sources

This section describes the SoOps received from airplanes, LEO satellites, and ground infrastructure to evaluate a sensor automatically without human supervision. The following sections describe how these signals are used to produce the measurements and metadata of interest.

Airplanes as a signal source

ADS-B is widely used everywhere in the world and, in fact, is mandatory in most airspace [284]. ADS-B equipped airplanes broadcast their position and velocity at least two times per second when airborne [139]. These messages are not encrypted. Therefore, they can be received and decoded by any receiver within range. We exploit this open architecture to receive and decode ADS-B messages on the sensor under evaluation. These messages reveal which directions have an unobstructed view and which directions are occluded.

We use the dump1090 program [285] to decode the signal received on the SDR. The fields in a packet of particular importance are ICAO aircraft address (ICAO) and Message, containing location information. The ICAO aircraft address identifies the airplane that transmitted a given ADS-B message for comparison to ground truth. Receiving ADS-B messages from a distant airplane strongly indicates that the field of view is open in that direction. However, not receiving any messages from a direction does not necessarily indicate blockage. It could be the case that no aircraft were in that direction at the time of measurement. Therefore, the data received on a node is combined with the data retrieved from another source to see if

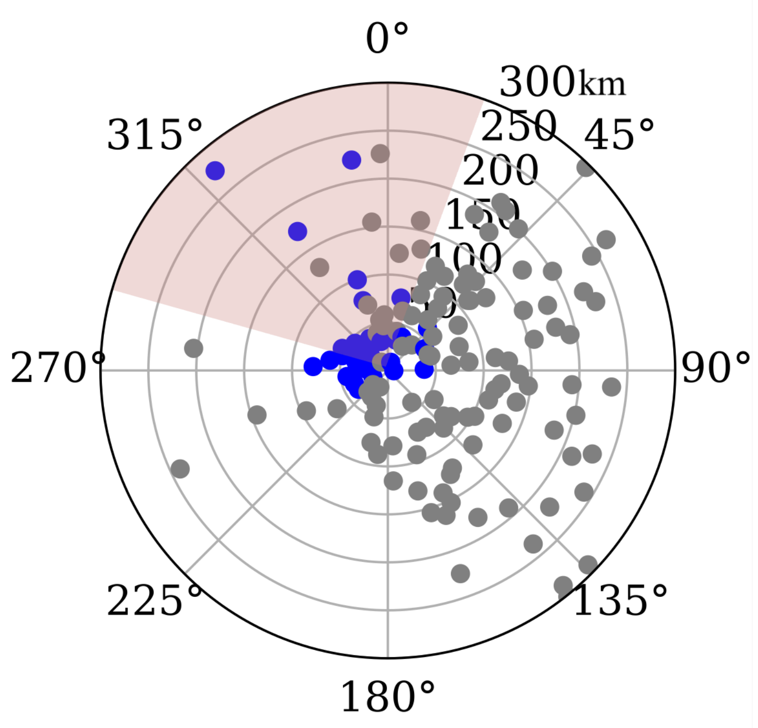


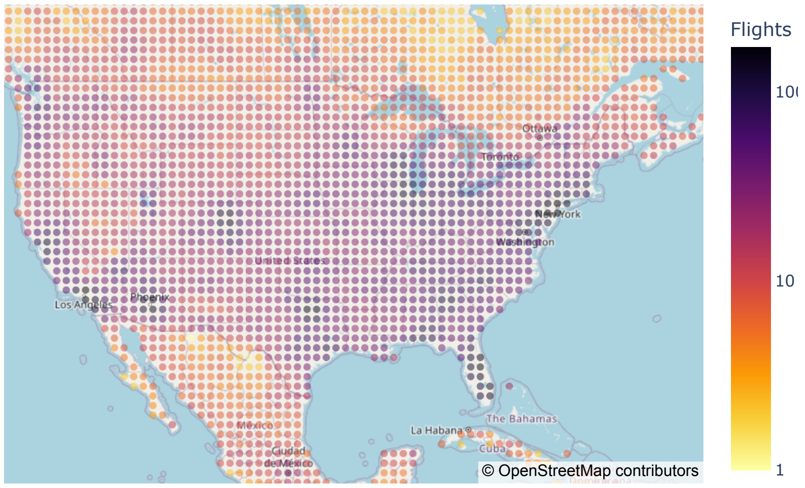
Figure 3.1: ADS-B performance for measuring directionality.
● message received for airplane, ● message was not received. The shaded area depicts the actual FoV.

there is any airplane in the direction with no received messages. we query the FlightRadar24 website[2] through an API to acquire the ground truth when evaluating a node.

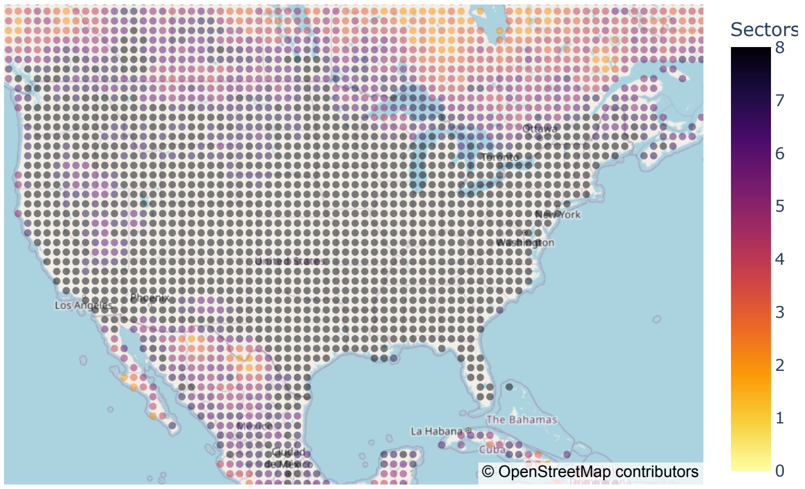
Figure 3.1 shows an example ADS-B measurement. Each point on these plots represents an airplane within 300 km of the sensor. The shaded area depicts the actual field of view. Blue points represent airplanes the sensor successfully received at least one ADS-B message from during the 30 second measurement period. Gray points are airplanes that no message was received from. We know there is an aircraft in a gray location despite not receiving any message from it based on ground truth data from FlightRadar24[3]. The figure shows a clear correlation with their respective fields of view. When an aircraft is relatively close, the ADS-B signal is very strong and might be received even through an indirect path. However, when a signal is received from a faraway plane, it is a strong indicator of unobstructed direction. In Section 3.4, this data is used to estimate the sensor's field of view.

[2]FlightRadar24.com is a popular flight tracking website that provides real-time information on flight-related data. It utilizes a crowd-sourced network of ADS-B receivers to gather data from airplanes.

[3]FlightRadar24 has more data than this single node by aggregating ADS-B messages from multiple sensors in an area.



(a) Number of flights in a 100 km radius



(b) Number of sectors in a 100 km radius

Figure 3.2: The number of flights and sectors covered around points in North America at their busiest time during one day.

ADS-B coverage

This section explores the density of flights in different regions for potential in evaluating sensor nodes. It is based on flight information data obtained from FlightRadar24 with a 10-minute resolution for a day. Each measurement is instantaneous and resembles a snapshot of flight traffic at that moment. Starting with a uniform grid covering the entire Earth, with points separated by 100 km, for every point and in every measurement, we calculate how many flights are within a radius of 100 km. Additionally, we divide each circle into eight sectors and measure how many of them contain at least one flight. This is a crucial metric for the field of view estimation algorithm in Section 3.4. Figure 3.2 depicts the maximum number of flights observed around each point on the map and the number of sectors covered at that time for North America. Note that each point may reach its maximum flight count at different times of the day. These plots showcase the potential for using ADS-B data in evaluating sensor nodes. Observe that most points can simultaneously detect several flights at their busiest times. The areas with a low number of flights generally have low populations; therefore, it is less likely that a sensor is installed in these regions. Section 3.4 shows that even if the number of flights is insufficient, a node can solely rely on satellite data to operate.

LEO satellites as a signal source

The next SoOp is signals transmitted by NOAA's polar-orbiting weather satellites. These satellites monitor and predict worldwide weather patterns and conditions. Currently, NOAA 15, 18, and 19 are active. These satellites orbit the Earth at a relatively low altitude of 850 km and transmit high-power signals (i.e., 5 Watts) within the 137 MHz band, making them an excellent reference for evaluation. We demonstrate that a spectrum sensor can receive signals from NOAA LEO satellites, even when using an antenna not specifically tuned for 137 MHz. In contrast, NOAA GOES satellites orbit at an altitude of 36,000 km, resulting in signals too weak to be detected by a sensor node equipped with a typical wide-band antenna.

We require a model to predict and track the movement of NOAA satellites to determine from which direction a signal is received. The Simplified General Perturbations 4 (SGP4) model [286] tracks satellite movements. SGP4 is implemented in the PyEphem [287] Python library, allowing precise prediction of satellite orbits.

LEO satellites converge

Figure 3.3 shows a simulation of six consecutive orbits of the NOAA 19 satellite. The blue lines depict the ground track of these orbits, and the red curve represents the footprint of the satellite, spanning approximately 6000 km in diameter. NOAA 15, 18, and 19 follow polar orbits, meaning their path crosses over both the North and South poles. Consequently, NOAA satellites pass over every point on Earth due to the rotation of Earth around its axis. It is important to note that the figure illustrates only six passes over a 10-hour period. Each orbit has a duration of 102 minutes, resulting in approximately 14 orbits completed by a NOAA satellite each day.

As depicted in Figure 3.3, the radius of the footprint exceeds the distance between two consecutive orbits, even at the equator where the lines are farthest apart. Therefore, if a specific location on Earth falls within the footprint of the current pass, the satellite will become visible again from that location during the next pass (sometimes even during the next two passes). In the subsequent experiments, each NOAA satellite passed over the region both in the morning and in the evening, covering at least two orbits each time. Considering the three active NOAA satellites, this totals up to 12 passes every day.

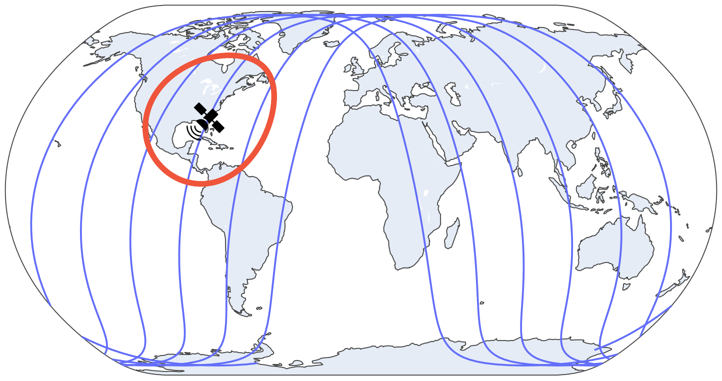


Figure 3.3: Six orbits of the NOAA 19 weather satellite. The red area shows the footprint of the satellite at a given location.

Satellite Signal Processing Methodology

The first and most important question to answer is how to determine if we have successfully received any signal from a satellite when capturing IQ data. One approach is to decode the signal to see if it yields any meaningful output. For instance, NOAA APT (Automatic Picture Transmission) is a data transmission system employed by NOAA weather satellites to transmit high-resolution images of Earth's weather conditions. Decoding the APT signal can help ascertain reception if it generates a meaningful image [4].

However, this methodology is tied to the specific satellite and encoding scheme it uses, rendering it less applicable to other satellites, especially when proprietary encoding techniques are employed. Furthermore, APT requires a relatively strong signal (i.e., > 20 dB SNR) for reasonable decoding. In contrast, this application requires a technique capable of detecting the presence of any satellite signal, even in low SNR scenarios, since sensor nodes may lack a suitable antenna for the 137 MHz band.

We designed a methodology that measures Doppler shift during a satellite pass to differentiate between satellite signals and noise or interference in the captured IQ data. Low-orbit satellites typically travel at speeds ranging between 7 and 8 km/s. This speed can induce a Doppler shift of up to $\pm 3,000$ Hz for a stationary observer on Earth which varies during the satellite pass. In this signal detection methodology, we compute the expected Doppler shift during a satellite pass and compare it with the signal captured by the software-defined radio. If the frequency of the detected signal changes according to the theoretical model, we recognize it as the signal coming from the intended satellite. This methodology is covered in detail in the rest of Section 3.3

[4]APT is an analog decoding scheme, and it consistently produces an image, even when the input is noise.

Satellite Transmission Frequency Deviation

While processing recorded data, it became clear that the Doppler shift of signals from the NOAA 18 satellite almost perfectly matched the theoretical model. However, the Doppler shift had a constant deviation for NOAA 15 and 19. After measuring the frequency of the satellite signal when each satellite was at its maximum elevation, so its radial relative speed to the observer is zero and no Doppler shift is expected, NOAA 15 and 19 always had an almost constant deviation from their center frequency. Initially, we suspected that the error might be coming from the SDR. We repeated the experiment with a different SDR and observed that the deviation values were roughly the same. Figure 3.4 shows the frequency deviation measured for all three NOAA satellites using two different SDRs. Observe that the deviations are different for each satellite but approximately the same across the two SDRs. We conclude that NOAA 15 and 19 have a slight deviation, approximately 650 and 200 Hz, from the expected center frequency. Therefore, for the rest of the experiments in this paper, we update the center frequencies using these values. Note that this satellite-specific deviation should not be confused with the Doppler shift.

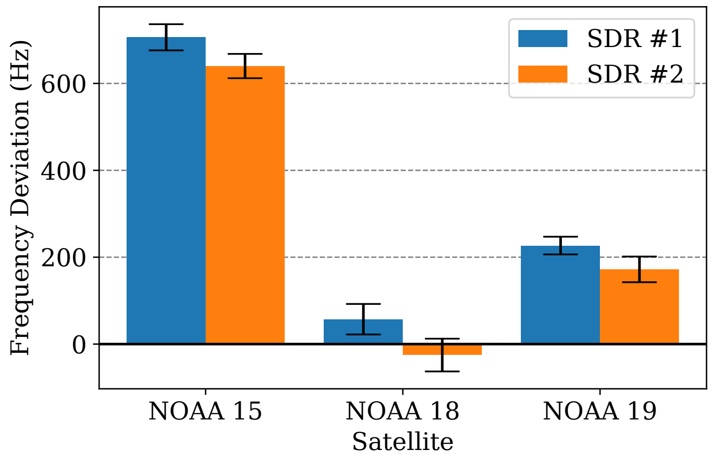


Figure 3.4: Transmitter frequency deviation of NOAA satellites.

Doppler shift during a satellite pass

When a satellite rises above the horizon, it attains its maximum speed relative to the ground station. As the satellite reaches its maximum elevation, the relative speed becomes zero. Note that this point is not necessarily directly overhead. During the first half of the pass, the received signal's frequency is higher than that of the transmitted signal, resulting in a positive Doppler shift. After the satellite passes its maximum elevation point, the Doppler shift becomes negative and reaches its lowest value when the satellite leaves the observer's field of view. Figure 3.5 illustrates this process, showcasing both theoretical and experimental measurements for a NOAA 15 satellite pass over the experimental sensor node. The derivation of the empirical curve is detailed in the following section.

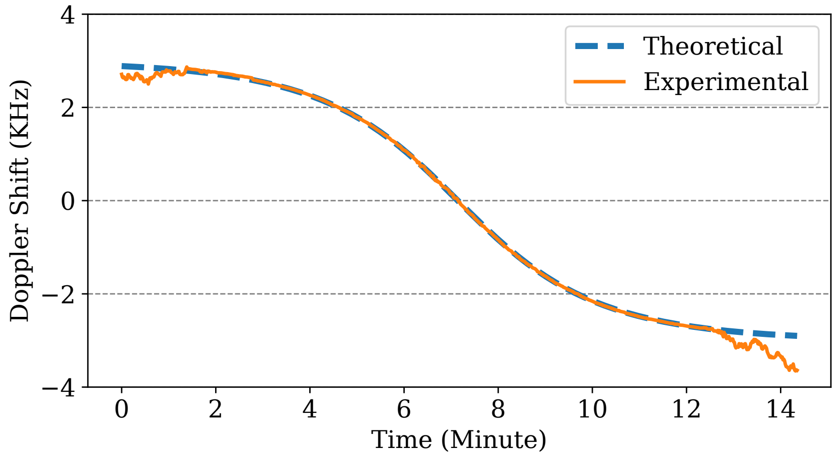


Figure 3.5: Changes of the Doppler shift during a satellite pass.

Doppler shift measurement

To detect the satellite's signal, we need to calculate both the theoretical and empirical Doppler shift of the satellite signal. The theoretical Doppler shift is computed using the SGP4 model. To calculate the empirical Doppler shift, we begin with the raw IQ data captured by the software-defined radio. The transmission frequencies of NOAA 15, 18, and 19 are 137.62 MHz, 137.9125 MHz, and 137.1 MHz, respectively. The SDR is configured to a center frequency of 137.5 MHz, which is the midpoint of the 137 MHz band. The sampling rate is set to 1.2 mega-samples per second to encompass the entire 137 MHz band, with an additional 100 kHz on each side to ensure we capture signals from all NOAA satellites.

The initial step involves computing the Fast Fourier Transform (FFT) of the IQ data to convert the data into the frequency domain, using a 5-second window. Given the potential for satellite signals to experience fading due to the satellite's speed, a 5-second window is empirically needed to allow the satellite signal to accumulate enough energy to distinguish it from interfering signals that may occur during data collection. This 5-second window then shifts by 1 second, and the process is repeated until all the data is processed. Subsequently, we apply a 40 kHz band-pass filter centered around the specified NOAA satellite's center frequency to eliminate unwanted signals. Figure 3.6 shows the output of the FFT step at an arbitrary time during a satellite pass. The NOAA APT signal is amplitude modulated (AM) onto a 2400 Hz carrier with a bandwidth of 4160 Hz. That signal is then frequency modulated (FM) onto a 137 MHz signal with a 17 kHz deviation. This is why the frequency domain signal has multiple spikes separated by 2400 Hz. The figure shows the frequency difference between the satellite transmission frequency (i.e., the green dashed line) and the received signal frequency (i.e., the center spike). In this example, the satellite has passed the maximum elevation thus the Doppler shift is negative.

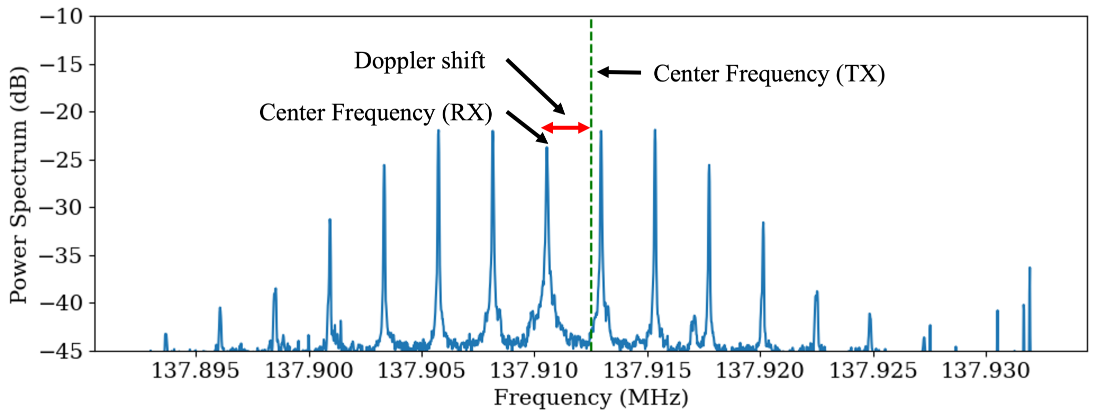


Figure 3.6: NOAA satellite signal in the frequency domain.

To calculate the Doppler shift empirically, subtract the center frequency of the transmitted signal from the center frequency of the received signal. Finding the center frequency of the received signal is not trivial, given the fluctuation of the signal amplitude and nearby interference regularly observed during measurements. To find the center frequency, we calculate the FFT at the time when the satellite is at its maximum elevation angle. At this point the expected Doppler shift is zero, therefore, the center spike should be very close to the transmission frequency. We consider a small 1000 Hz window centered at the transmission frequency (i.e., known for each satellite) and find the FFT bin where the signal has the highest energy. The algorithm also finds the location of two previous and two succeeding spikes. The center frequency is then calculated as the average frequency of these five symmetrically distributed spikes. The algorithm then moves to the next (and previous) window and tries to find the center frequency of the received signal. This process continues until the entire trace is processed.

Confirming satellite signal reception

The final step verifies whether the received signal was indeed from the intended satellite by comparing the empirically obtained Doppler curve with the theoretical model. If the empirical data moves along the theoretical curve, the signal is "locked" to the expected Doppler curve. We determine the time period in which the Doppler tracking algorithm is locked in several stages. First, set a maximum absolute deviation between the observed and theoretical Doppler shift of 300Hz. Next, iteratively reject outlier samples, defined as greater than three standard deviations from the mean of remaining samples, recalculating the mean and standard deviation each iteration. This is similar in principle to RANSAC [288], but replaces a linear regressor estimate with the sample mean. We also apply this outlier rejection to the first difference of the Doppler shift error to remove segments of high variance relative to the locked period, even if they share the same mean. Finally, find the two largest contiguous segments of locked samples. If they are of sufficiently similar length and sufficiently close in time, merge them and the samples in between to produce the final tracked period. Otherwise, use the longest contiguous segment. The result of this process is shown in Figure 3.7.

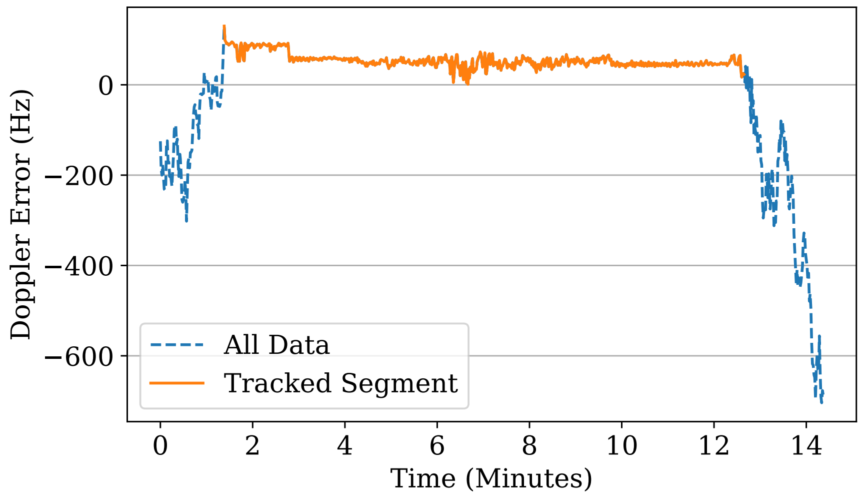


Figure 3.7: Theoretical versus observed Doppler shift error, and the time period when the Doppler offset is tracked. This is the same pass as Figure 3.5.

Cellular Networks

SDRs offer the advantage of supporting a wide frequency range, making them versatile enough for many applications. In the context of renting a sensor node, it becomes crucial for users to assess the node's performance within frequency bands relevant to their needs, given differing propagation and reception characteristics. Therefore, the automatic evaluation technique aims to effectively characterize the node's performance at all frequency bands supported by the node. Note that ADS-B messages characterize a node at the 1090 MHz frequency band only. In other words, if FoV estimation determines a node is fully or partially obstructed, we would like to know how the obstruction impacts its capability in other bands.

This can be accomplished by utilizing known signals in a variety of frequency bands. One excellent candidate source of such signals is 4G/5G cellular networks because they operate in a wide range of spectrum. Moreover, cellular coverage is nearly ubiquitous and the locations of cell towers and the exact frequency bands each uses are known. Mobile networks in North America can operate from as low as 617 MHz all the way to 4499 MHz in 4G networks. In addition, 5G also supports millimeter-wave bands from 24 to 48 GHz.

The LTE standard includes a reference signal designed to allow signal strength estimation even for devices not attached to a particular cell tower [237]. Information on cellular signals can be captured with srsUE [289] as software client user equipment. This tool is an integral component of the open-source srsRAN project, which provides a complete software stack for both 4G and 5G networks. srsUE is able to scan for nearby cellular networks and measure their Reference Signal Received Power (RSRP). RSRP quantifies the strength of the received signal from the base station, serving as a crucial indicator of the signal quality experienced by mobile devices. Databases such as cellmapper.net show cellular towers in a region with their exact channel (i.e., ARFCN). This information can be used to configure srsUE to scan frequency bands of interest. Figure 3.8 illustrates the location of cellular towers used in these experiments with respect to the experiment site.

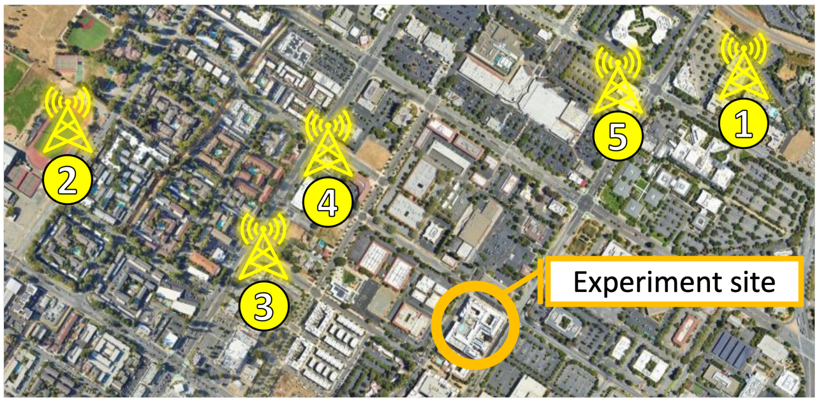


Figure 3.8: Mobile network experiment testbed.

Broadcast TV

Broadcast TV signals extend the frequency coverage from cellular networks (commonly 600 MHz to 2500+ MHz) down to 85 MHz across much of the world. However, measuring signal quality required development of a custom program using the GNU Radio software environment [290]. The SDRs were configured with a fixed gain to prevent measurement differences from automatic gain control. The received power was measured by bandpass filtering a desired ATSC channel, then applying Parseval's identity to measure the band's power by running the magnitude-squared time-domain samples through a very long moving average filter for a stable measurement across time.

3.4 Automatic Sensor Evaluation and Calibration

Frequency Sensitivity Estimation

The first requirement for a spectrum sensor node is that it must be able to receive signals at the frequencies of interest. Although some SoOps like ADS-B and LEO satellite signals are very useful for spatial inference, they are extremely sparse in frequency space. RF signals propagate very differently at various frequencies commonly employed. Signals in the 10s to low 100s of MHz penetrate buildings easily and can even diffract around geographic features like hills. Signals in the multiple GHz, on the other hand, struggle to penetrate even a single wall, while millimeter-wave (10s of GHz) signals will be completely blocked by a human body. Fortunately, SoOps like LTE and broadcast TV signals span wide frequency ranges. This enables evaluation of a sensor's frequency response across many bands commonly of interest.

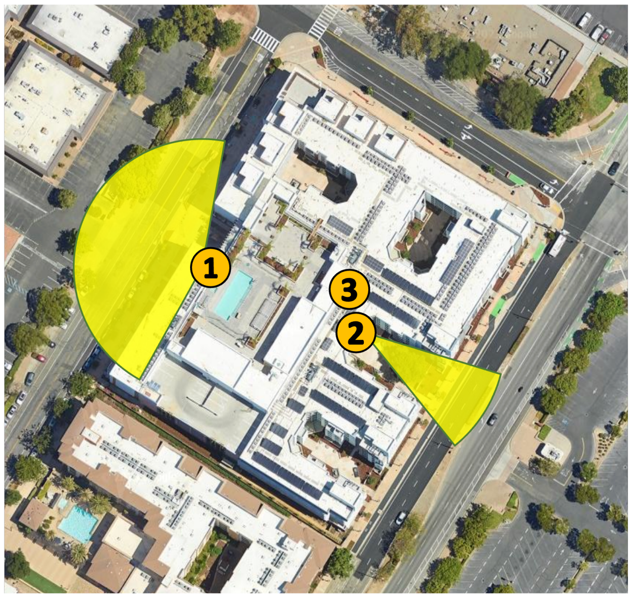


Figure 3.9: Measurement locations for frequency sensitivity experiments.

The frequency sensitivity experiments used a BladeRF xA9 SDR to a laptop host machine. The SDR was connected to a wide-band antenna with a frequency range of 700 MHz to 2700 MHz. Measurements were taken at three locations with widely varied reception capabilities, depicted in Figure 3.9. Location ① was located on the rooftop of an apartment building on the 6th floor. It has an open field of view to the west as indicated by the yellow shaded area in the figure. Some building structures on the rooftop obscure its view in other directions. Location ② was behind a window that faces southeast on the 5th floor. Because of the buildings to the left and right, this location has a narrow field of view. Finally, Location ③ was inside the building on the 5th floor at least 8 meters away from windows, with no field of view to the outside.

Cellular Networks

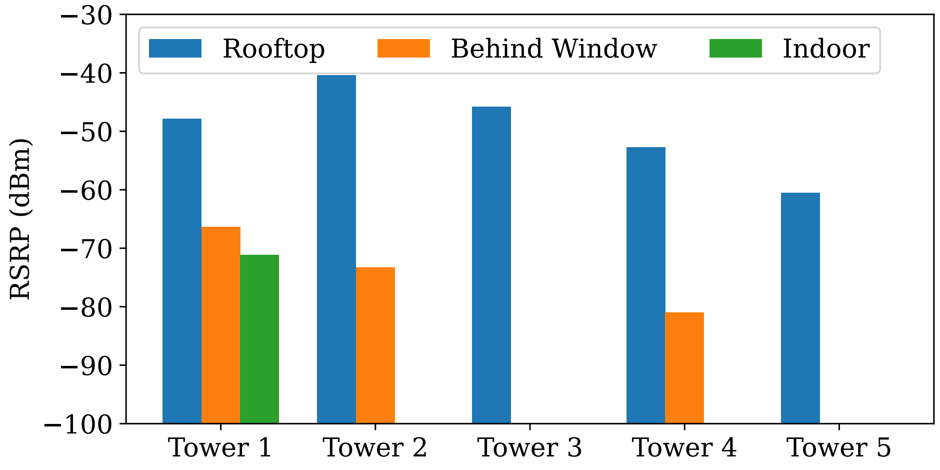


Figure 3.10: Cellular reception for nearby towers, ordered by increasing carrier frequency.

Figure 3.10 shows the performance of a node when placed on a rooftop, behind a window, and inside the building far from windows (i.e., locations ①, ②, and ③ as described above). This experiment measures the RSRP from five nearby 4G/5G cellular base stations as illustrated in Figure 3.8. The downlink frequencies of the base stations at towers 1 to 5 are 731, 1970, 2145, 2660, and 2680 MHz, respectively. The coverage range in the low-band (i.e., tower 1) is up to 40 km, while the range is 1.6 to 19 km for the mid-band (i.e., towers 2-5). All of these towers are 500 to 1000 meters from the experiment site; therefore, reception should be excellent for these towers in the absence of obstructions.

Figure 3.10 shows that RSRP is very high, indicating excellent reception for all five towers when the sensor is placed on the rooftop. The sensor either has a line of sight to the tower or is partially obstructed. However, Figure 3.10 reveals significant signal attenuation when the sensor is not installed outside. A missing bar indicates that the signal was too weak for srsUE to decode successfully. When the sensor is placed inside a building at location ③ it can only decode wireless packets from tower 1. This is because tower 1 operates in the 700 MHz band. 700 MHz signals can penetrate buildings much better than mid-band signals from towers 2 through 5, although the difference varies based on building materials. When the sensor is placed behind a window at location ② the signal is attenuated significantly, but it can still see the signals coming from Towers 1, 2, and 3. However, the obstructions around location 3 kill the signal completely at higher frequencies. These findings are in line with cellphone reception at these locations. A phone shows only one or two bars when placed at these locations, and its connection is very weak.

Broadcast TV

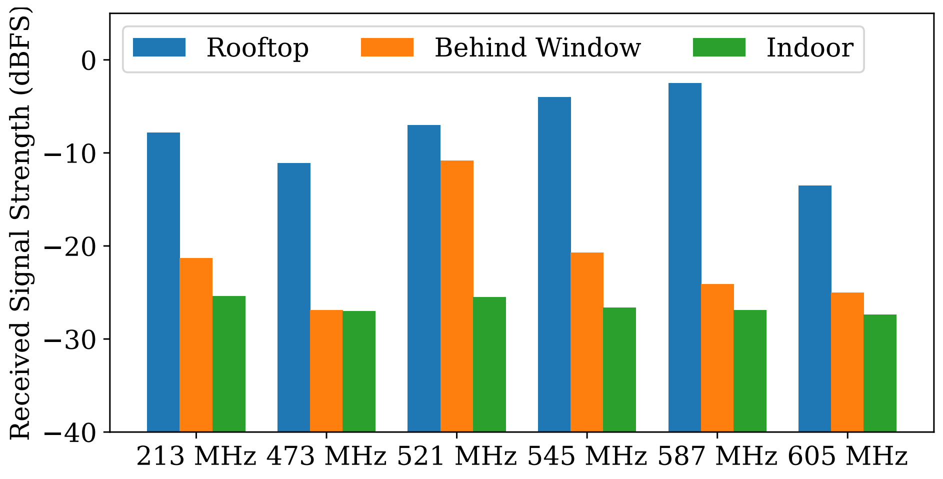


Figure 3.11: Broadcast TV reception: different frequency bands

This experiment measured the received signal strength from multiple TV broadcast towers up to 50 km away from the experiment site. Figure 3.11 shows the results for the 3 experiment locations. The experiment shows that at locations ② and ③ the building structure degrades reception, but the SDR is still able to receive a relatively strong signal. The exception to this behavior is the very strong signal at 512 MHz when the sensor is placed behind a window. This is because the tower broadcasting at this frequency is in the field of view of the sensor and the building structure minimally impacts the signal. This experiment shows that despite some attenuation at locations ② and ③ they can be used for sub-600 MHz spectrum measurements.

Combining the results from multiple experiments, including ADS-B, cellular networks, and broadcast TV, can provide additional insights, such as determining whether an installation is indoor or outdoor. For instance, if the sensor consistently receives all signals with high quality when placed on a rooftop, it can be inferred that the sensor is installed outdoors. Conversely, if there is significant signal degradation observed at specific locations, such as ② and ③, at higher frequencies, it suggests that the sensor is located inside a building. By analyzing the combined experimental data, valuable information about the installation environment and the placement of the sensor can be deduced. These deductions can be used to independently verify claims about a node installation.

Field of View Estimation

One crucial metric for evaluating a spectrum sensor node is the geographic region it covers. This section describes a method for building a map of coverage for a sensor node, then from the coverage map extracting a single field of view (FoV) metric describing the width and direction of the largest visible region for a sensor. Experimental data in this section was recorded on the roof of a five-story academic building.

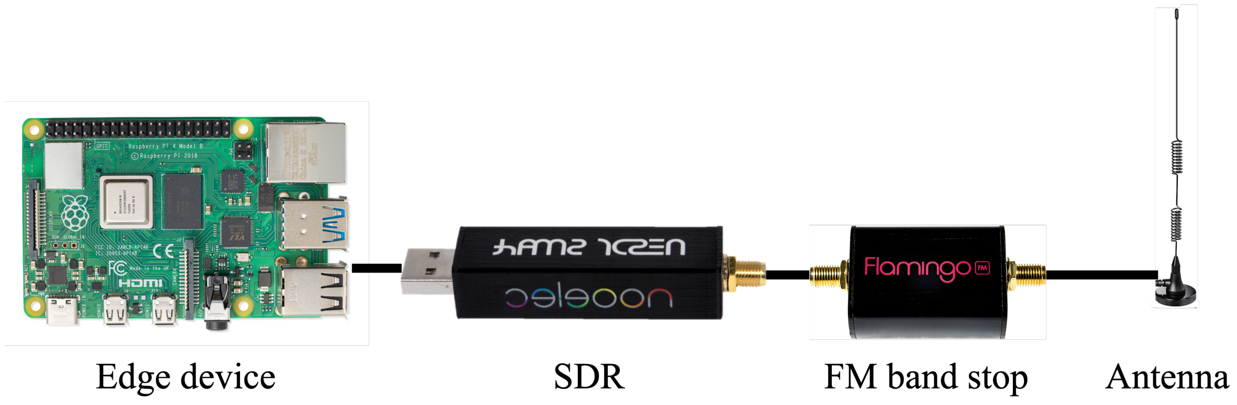


Figure 3.12: Experiment hardware setup.

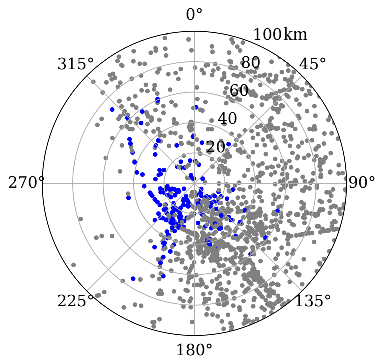
A significant barrier to widespread participation in a crowdsourced sensor network is the cost of initial setup. One commercially successful crowdsensing effort is the air pollution monitoring service Purple Air [291], whose sensors cost $200-300. We aimed to keep the cost of the experimental hardware to a similar range. For the following experiments, the hardware setup included

- a $60 Tram 1410 25 MHz - 1300 MHz antenna or
  a $16 Bingfu 20-1300MHz scanner antenna,
- a $33 RTL-SDR software defined radio,
- a $22 Flamingo FM bandstop filter,
- a $45 Raspberry Pi 4 single board computer,
- and miscellaneous RF connectors and cabling,

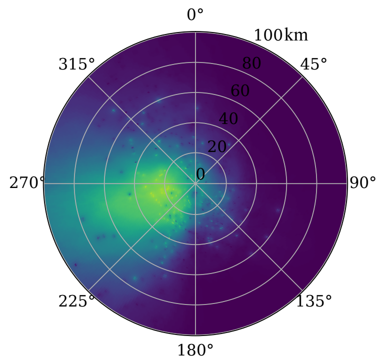
keeping total system cost to less than $200. The overarching goal in keeping system cost down is to demonstrate that it is possible to achieve useful calibration measurements without the expensive equipment used in other spectrum monitoring deployments such as [238], [239], [260], or custom components as used in [261]. This will enable widespread deployment among hobbyists and others who are not spectrum researchers. Figure 3.12 illustrates the hardware setup. This hardware setup was used for the remaining experiments in this section.

Field of view from ADS-B data

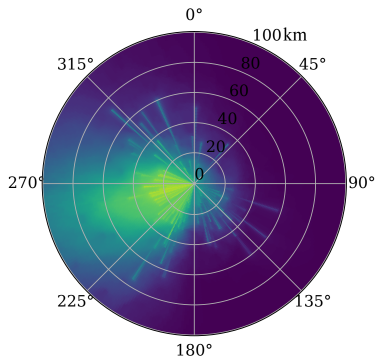
Figure 3.13 demonstrates the processing stages used to produce the FoV metric. Figure 3.13a shows a plot of all aircraft in an experiment, colored by whether they were detected (blue) or not (grey) and transformed from latitude and longitude to bearing and range relative to the SDR. We feed this data into a non-parametric classifier to estimate the probability of



(a) Aircraft location relative to receiver, colored by detected (blue) or missed (grey).



(b) Probability of detection predicted by KNN model.



(c) Radial-max probability of detection from KNN model.

Figure 3.13: Example processing steps from aircraft detections to final probability of detection estimate.

detection (PD) for an aircraft at any given location. Here, we use the K-Nearest Neighbors (KNN) algorithm [292] implemented by scikit-learn [293]. Fitting and prediction are performed in Cartesian coordinates, then the output is converted back to polar coordinates for plotting. The number of neighbors K is adaptively and automatically selected through K-fold cross validation [292] with a classification accuracy metric. Figure 3.13b shows PD predicted by KNN using the data from Figure 3.13a. Finally, we make the assumption that, barring receiver saturation, reception should only improve as an emitter gets closer to the sensor. Thus, we apply a "radial-max" operation to the KNN output in which samples along each radial direction are set to the maximum value seen so far, from far to near. The result of applying radial-max is shown in Figure 3.13c.

For a spectrum sensor monitoring terrestrial emitters, only coverage near the horizon in terms of elevation angle matters. Since ADS-B data includes altitude, it can be filtered for aircraft within specific elevation bins. Figure 3.14 shows sensor coverage for three elevation bins: near horizon, slightly up, and the rest of the way to vertical. For this use case, most ($> 90\%$) flights are near the horizon, with fewer flights but more visibility as elevation increases. The following figures will only include near-horizon ($< 5°$) data unless otherwise specified.

The FoV is calculated at multiple ranges since the predicted visible regions vary with distance based on previously seen aircraft. FoV is calculated as the longest unbroken arc with PD greater than a threshold (i.e., 0.1). We calculate the expected FoV for comparison by measuring the surrounding terrain's elevation using HORAYZON [294], then similarly find the largest unbroken arc where the terrain horizon never exceeds the upper limit of the near-horizon elevation bin, $\leq 5°$. Figure 3.15 plots the horizon and the FoV for several ranges.

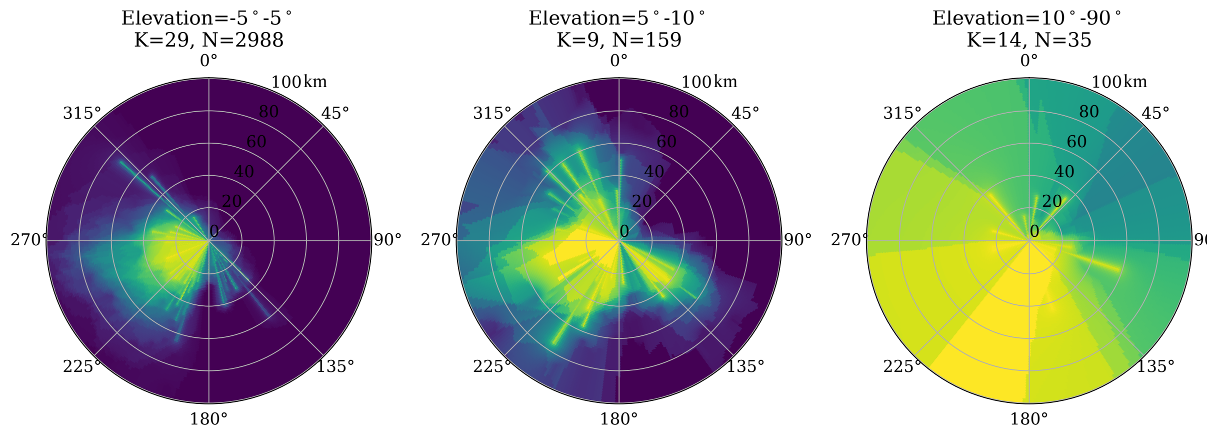


Figure 3.14: ADS-B probability of detection by elevation. At higher elevations, the horizon and other obstacles have reduced impact on the field of view.

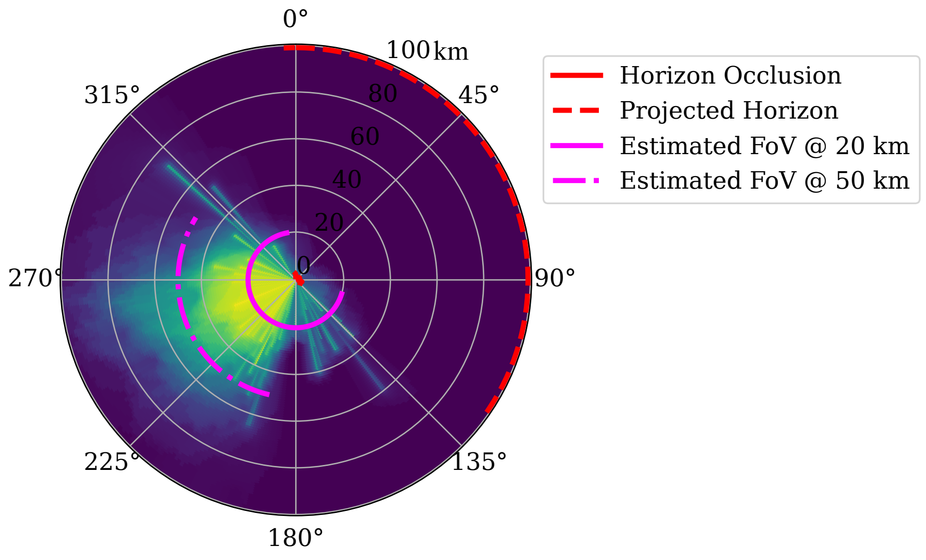


Figure 3.15: ADS-B FoV estimates for the experiment location.

Even at the primary experiment location, which is in a high air traffic region, the number of nearby aircraft at one time is often insufficient to cover the entire region of potential visibility. This problem is only exacerbated in low-traffic regions. Fortunately, aircraft are mobile, so waiting a few minutes then collecting another sample will likely provide data in new regions of the map. Figure 3.16 demonstrates the additive effect of taking multiple samples in time. In particular, the sector to the North-North-West lacks any positive examples until all 24 time samples are included. Table 3.2 shows the quantitative accuracy impact of additional samples for multiple locations, and the section on simulation discusses quantitative impacts further. Figures 3.20 and 3.21 show the aggregate accuracy impact of

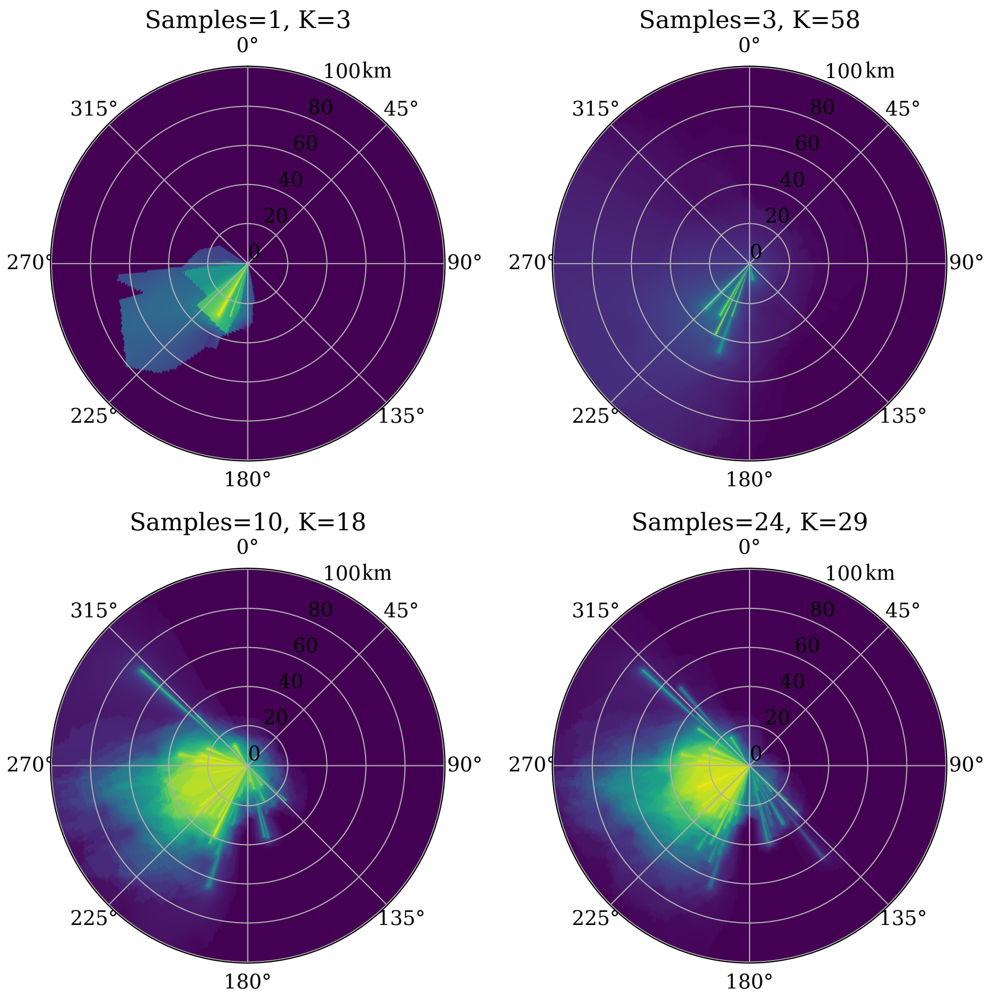


Figure 3.16: ADS-B FoV estimates with increasing number of time samples. As the number of samples increases, the size and accuracy of the estimated FoV increases.

additional samples for multiple locations and are discussed further later on.

Field of view from satellite data

The FoV for data from satellites is calculated similarly to ADS-B data. For each satellite pass, samples from the time period during which the Doppler offset can be reliably tracked are defined as detections. Samples from the period on either end of the pass where Doppler tracking fails but the signal can still be detected are defined as missed detections. By the nature of their orbits, satellites are extremely far from the sensor while at low elevation, so we remove range data and perform KNN estimation on bearing angle only. From this point, estimating the FoV proceeds exactly as for ADS-B data except that there is no dependence on range. Figure 3.17 shows the outcome of estimating FoV from one day of satellite passes

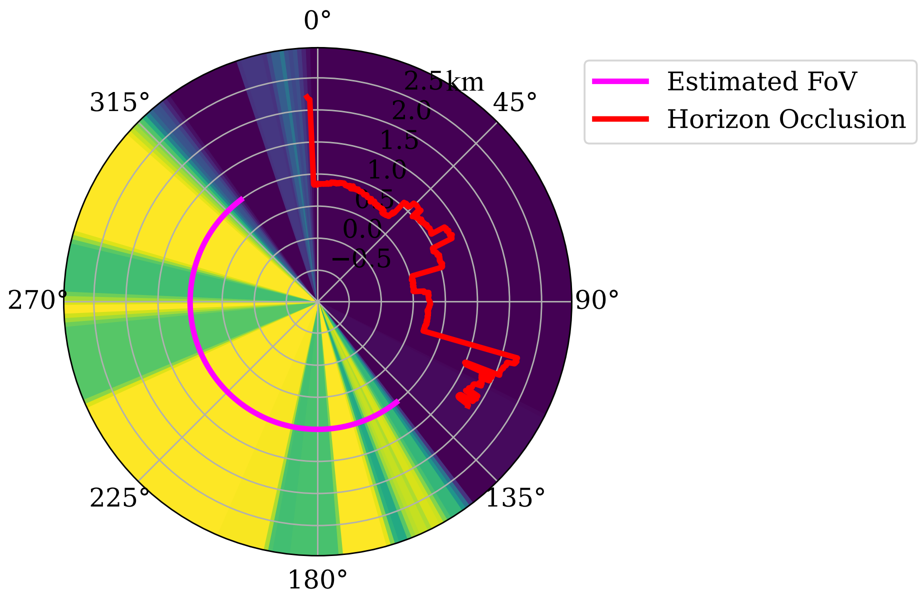


Figure 3.17: Satellite FoV estimate using only angle measurements.

at the primary experiment location.

The natural next step once both satellite and ADS-B FoV estimates are available is to merge them. The estimates are merged by sampling at identical azimuths and broadcasting the satellite PD estimate to all ranges, then taking the maximum of the two at each point. The result is shown in Figure 3.18, and a comparison of FoV estimation accuracy for all three of ADS-B, satellite, and merged estimates is available in Table 3.1. At 20 km, the ADS-B estimate is better than the satellite estimate in width and similar in bearing, and merging the two does not reduce accuracy. At 50 km where a paucity of flights reduces the accuracy of the ADS-B estimate, merging the two sources maintains similar accuracy to the solely satellite estimate. Merging the estimates provides the best of both worlds, at least for this experimental site. We believe that this relationship should hold up for other locations as well. Satellites in polar orbits, like the ones used here, sweep large continuous arcs from North to South, providing dense estimates of FoV to the East and West of any sensor. They have poor coverage to the North and South of a sensor except in the rare case of a pass directly overhead. Aircraft, depending on local flight patterns, are less likely to leave a specific direction uncovered but provide sparser measurements. Thus, combining the two should provide better overall estimates of FoV.

FoV estimation accuracy

In order to understand FoV estimation performance across many possible terrain and air traffic conditions, we selected 100 random locations in the US and simulated the outcome of estimating FoV from ADS-B broadcasts at each location with a virtual sensor. The virtual

Table 3.1: Experimental FoV Estimation Error.

Data Source	Range (km)	Brng. Err. (°)	Width Err. (°)
ADS-B	20	-11.7	18.0
	50	16.4	-104.0
Satellite	20	-8.0	-51.0
	50	-8.0	-51.0
Merged	20	-10.7	20.0
	50	-9.2	-49.0

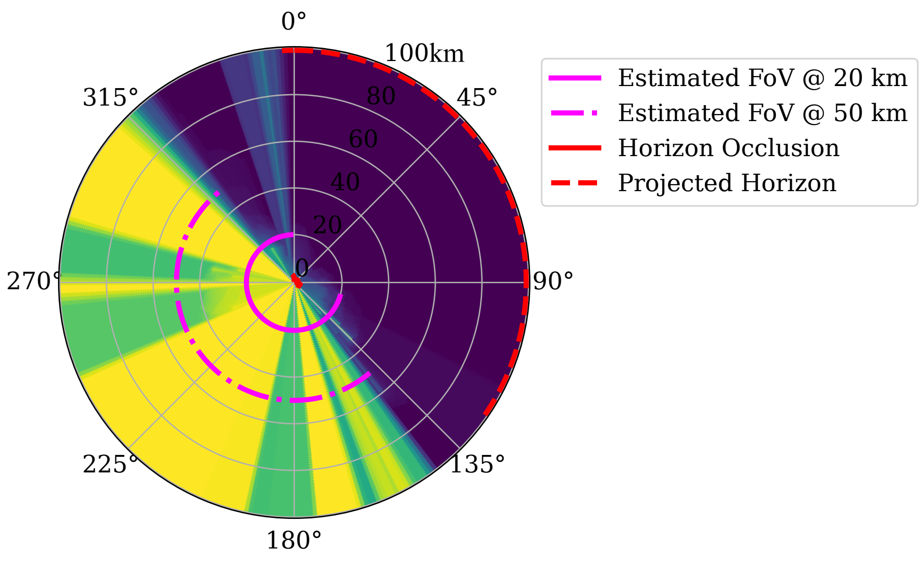


Figure 3.18: Merged FoV estimate using ADS-B and satellite detection data. Accuracy improves at long range.

sensor starts with an isotropic antenna pattern estimated from averaging PD across the 20 km range FoV of the experiment system. The antenna pattern is shown in Figure 3.19. Then, the horizon is used to mask out impossible detections from aircraft within 100 km of the virtual sensor's location. The remaining aircraft are then randomly chosen as detections or misses based on the probability of detection given by the antenna pattern.

After simulating sensor coverage at 100 locations, we calculated the mean error and mean absolute error (MAE) in FoV direction and width across five trials for varying ranges, number of time samples, and methods of choosing times to sample. Table 3.2 contains the results for ranges of 20 and 50 km, 1 and 24 time samples (about four hours if recorded sequentially at 10 minute intervals), and the "best" and "random" time sampling methods. The sampling

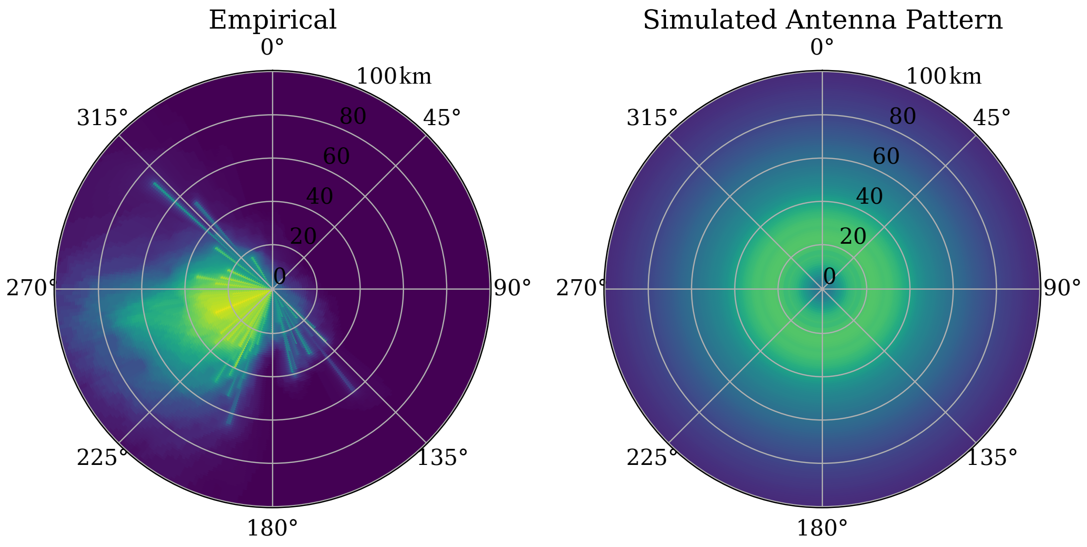


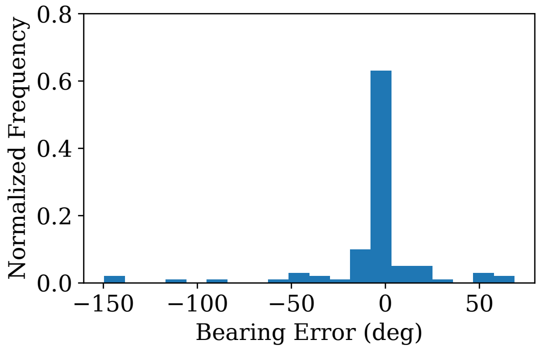
Figure 3.19: Simulated antenna pattern based on experimental data. The antenna pattern estimates range dependence in the FoV of the experiment data then applies it isotropically in angle.

Table 3.2: Simulated FoV Estimation Error (°).

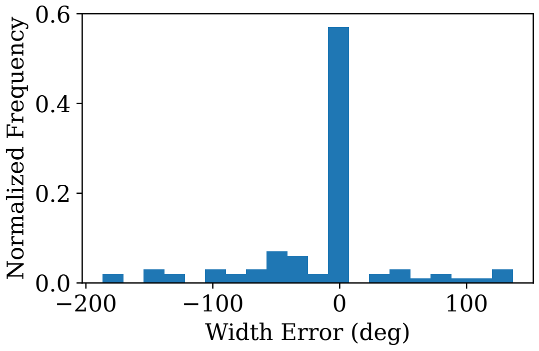
Method	N Samp.	Range (km)	Brng. Err.	Width Err.	Brng. MAE	Width MAE
best	1	20	-36.0	-82.6	49.6	91.9
		50	-35.6	-119.5	51.5	124.4
	24	20	**-4.8**	**-12.7**	**14.2**	**33.3**
		50	-5.5	-58.8	23.3	65.0
random	1	20	-62.8	-155.3	76.2	160.2
		50	-65.2	-179.7	80.2	183.5
	24	20	-13.8	-23.5	27.9	41.8
		50	-13.4	-71.8	33.3	76.2

methods are:

- **Random**: random sampling consists of selecting N unique time samples from a one week period with samples recorded every 10 minutes.
- **Best**: the number of planes tracked by FlightRadar24 in each 10 minute interval and 100 km by 100 km region spanning the US was averaged over a separate week of recordings to determine the highest traffic periods for an average day. The N highest



(a) Bearing accuracy for 20 km. Bin width is 10.9°.



(b) Width accuracy for 20 km. Bin width is 16.2°.

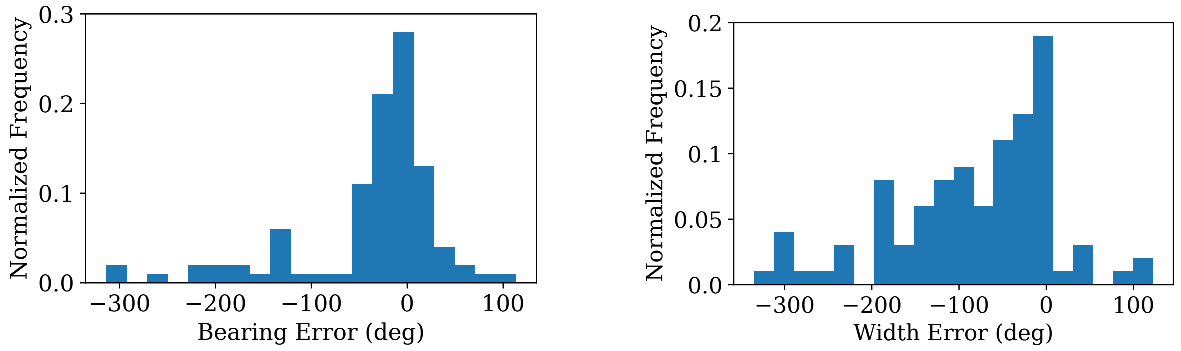
Figure 3.20: ADS-B FoV accuracy for 24 time samples at 20 km with "best" sampling, averaged across 100 locations in the U.S.

traffic periods for the 100 km by 100 km region containing a location were then selected and extracted from random days to construct the simulation data.

Figure 3.20 shows the FoV bearing and width accuracies for 20 km range and 24 time samples with "best" sampling. Approximately 60% of locations estimate the FoV to within 10°, with thin but long tails for locations with larger errors. The mean bearing error is −4.8° and mean width error is −12.7°, slightly underestimating the size of the covered area. The MAEs are 14.2° and 33.3°, respectively. At 50 km, the bearing accuracy remains nearly the same, while the width becomes significantly underestimated with an error of −58.8° and an absolute error of 65°. The "random" sampling method performs about half as well for mean error, with about 50% larger absolute error.

Figure 3.21 shows the accuracy when only one time sample is chosen. Performance degrades significantly with large tails for bearing and width errors. The FoV width is greatly underestimated, with a mean error of −82.6° and MAE of 91.9°, and the bearing error is also greater with an MAE of 49.6°. The width is underestimated so severely because there are generally fewer detections than misses, so when few flights are available there can be large regions where only misses are seen due to chance, despite the sensor having line of sight. In low traffic regions or times, there may be less than 10 flights in a 100 km radius, making it very unlikely to obtain an accurate estimate of the FoV. This disparity illustrates the importance of taking multiple samples over time and the benefit that intelligently selecting sampling times can provide. The ability to improve estimates by sampling multiple times demonstrates the advantage that mobile emitters like aircraft or satellites provide since they can sweep large areas, providing a more accurate final estimate than fixed emitters like cell or radio towers.

Due to the global nature of satellite coverage, we expect to see similar improvements in



(a) Bearing accuracy for 20 km. Bin width is 10.9°.

(b) Width accuracy for 20 km. Bin width is 16.2°.

Figure 3.21: ADS-B FoV accuracy for one time sample at 20 km with "best" sampling, averaged across 100 locations in the U.S.

FoV accuracy at long range as observed in the experimental data. Future work will entail developing a similar model for simulating satellite observations based on orbital data and simulating the accuracy of merged FoV estimates.

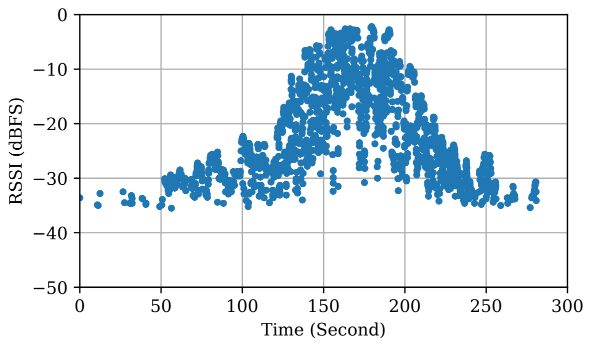
3.5 Automatic Metadata Generation

Indoor-Outdoor Classification

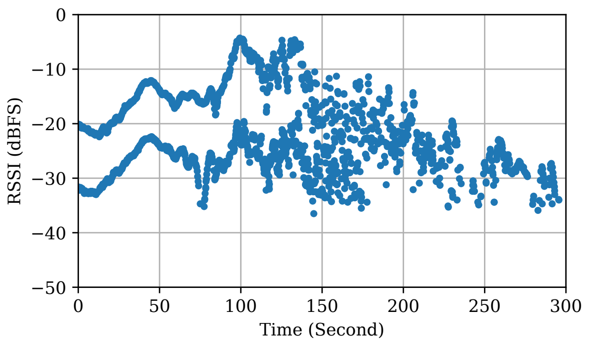
The key idea behind this indoor-outdoor classification method is to use certain characteristics of signals received from nearby airplanes to indicate if the receiver is indoors or outdoors. Intuition suggests the multipath field differs in indoor and outdoor environments. Indoors, there are typically many obstructions and reflectors near the receiver. Conversely, in outdoor environments, there might be a line-of-sight path between an aircraft and the receiver, with reflectors typically farther away from the receiver. As a result of these differences, the Received Signal Strength Indicator (RSSI) of the signal tends to be more stable in outdoor scenarios.

Figure 3.22 illustrates an example of such behavior in the RSSI received from an aircraft over time. Figure 3.22a shows how the RSSI received from an aircraft changes over a 300-second experiment when the receiver is indoors. As the aircraft moves, the best signal path between the aircraft and the receiver changes rapidly due to surrounding obstructions. However, Figure 3.22b paints a different picture for an outdoor receiver in the first 100 seconds. The figure indicates that the RSSI is very stable during this time because there is an unobstructed line of sight between the aircraft and the receiver. Between 100 and 150 seconds, the aircraft starts to leave the line of sight (LOS) path, and after 150 seconds, there is no LOS. During this time, the RSSI fluctuates similarly to the indoor receiver case.

This RSSI behavior forms the foundation of a system for automatically detecting indoor and outdoor environments.



(a) Indoor receiver, recorded at Albany bedroom (Table 3.3).



(b) Outdoor receiver, recorded at Ottawa backyard (Table 3.3).

Figure 3.22: The difference in RSSI behavior when the receiver is indoor and outdoor.

Another important insight gained from this experiment is that the range of RSSI variation or their absolute values cannot reliably differentiate indoor and outdoor receivers. This is because most SDRs are not calibrated and, therefore, do not report the absolute received power. Even if they do, the transmission power of the ADS-B system is not known for a given aircraft, making it challenging to determine if the signal is attenuated only due to LOS path loss or the building structure (or other obstacles). As shown in Figure 3.22, in both indoor and outdoor experiments, the RSSI variation range is from below -30 to near 0 dBFS. Therefore, the high RSSI values in the outdoor case cannot be conclusively associated with being outdoors. The transmission power of ADS-B on an airliner is between 125 to 500 Watts; hence, the signal strength is very high when the airplane is in the vicinity, even if the receiver is indoors [140]. Although commercial aircraft fly at altitudes of $30,000$ ft or more, they are still valuable proxies for interference between ground stations because they are observable from 200+ km away. At 55 km, an airplane at $32,000$ ft is $< 10°$ above the horizon, and at 100 km it is $5°$ above.

Interestingly, the RSSI measurements show that in some cases (often outdoors), there are two distinct bands in RSSI values. Figure 3.23 displays an example of such measurements for an aircraft during a two minute capture. This effect is due to the fact that commercial airliners use two antennas (one on top and one on the bottom of the aircraft) to transmit ADS-B messages to simultaneously facilitate robust air-to-ground and air-to-air communication. The dual-antenna setup enhances signal reception and transmission capabilities, ensuring a more reliable and comprehensive surveillance data exchange between aircraft and ground stations, other nearby airplanes, and satellites. The ADS-B system alternates between the top and bottom antennas when broadcasting messages. As a result of the antenna diversity at the transmitter, measured RSSIs can change significantly (typically 10 to 20 dB)

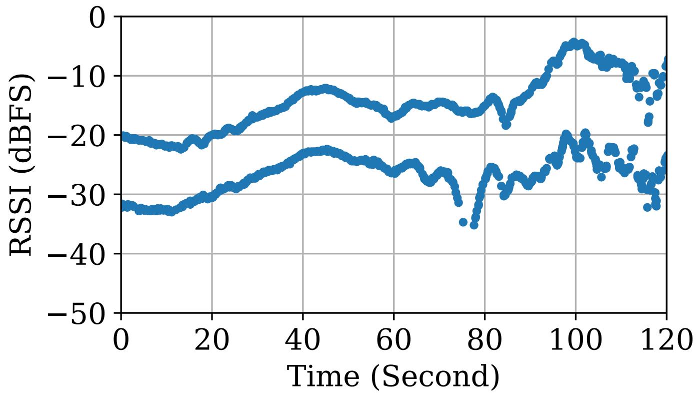


Figure 3.23: The impact of antenna diversity on RSSI.

between two consecutive packets. This change in RSSI affects the proposed indoor/outdoor detection system; therefore, the system design takes advantage of it.

Experiment Setup

The experiments were conducted in and around four locations in Ottawa, Ontario, Canada and the San Francisco Bay Area. The locations include freestanding homes, a townhouse, an apartment, and a multistory academic building. Signals were collected at multiple points throughout these buildings, ensuring a varied mix of nearby windows, completely enclosed spaces, unobstructed rooftops, and partially obscured outdoor locations. Table 3.3 describes all 27 measurement locations. Each measurement was 15 minutes long. For each location, we estimate, for each of eight directional sectors, whether the environment was obscured by a wall or neighboring building or unobstructed, either by being outdoors or near a window.

Each 15-minute measurement was conducted using the setup illustrated in Figure 3.12. The setup comprises an SDR connected to a host machine. To ensure the independence of the indoor/outdoor detection mechanism from specific hardware, we performed measurements with different SDRs, including the Nooelec Smart V5, RTL-SDR V3, and LimeSDR Mini. The Nooelec Smart V5 and RTL-SDR V3 SDRs feature an 8-bit ADC, while the LimeSDR is equipped with a 12-bit ADC. These varying dynamic ranges influence the RSSI measurements. However, the system is robust against these variations in the hardware used for the experiments.

The edge device was either a Raspberry Pi 4 or an Intel UP Element i12 edge device. Instead of employing a dedicated ADS-B antenna, which could have enhanced the results by capturing more messages, we opted for a wideband antenna. The rationale behind this antenna choice is that we envision integrating the indoor/outdoor detection mechanism into existing devices. Therefore, it would be ideal if it could utilize the existing antenna. All measurements were collected using this non-optimized ADS-B antenna. Despite using a wideband antenna, we could still receive ADS-B messages even inside buildings. To mitigate interference from a nearby FM tower's powerful signals saturating the Low Noise Amplifier (LNA) at 1090 MHz, we inserted an FM band stop between the antenna and the SDR.

Building	City	Locations	Indoor	Outdoor	Total
Townhouse (multi-floor)	Ottawa, ON, Canada	kitchen, dining area, bedrooms, entrance, hallway, backyard, driveway	6	2	8
Townhouse (flat)	Albany, CA, USA	bedroom, behind window	2	0	2
Townhouse (multi-floor)	Berkeley, CA, USA	bedroom, closet, stairs, kitchen, deck, driveway	8	4	12
Academic building	Berkeley, CA, USA	ground floor, 4th floor window, 4th floor interior, roof	4	1	5

Table 3.3: ADS-B measurement locations and environments.

Model-based Classification

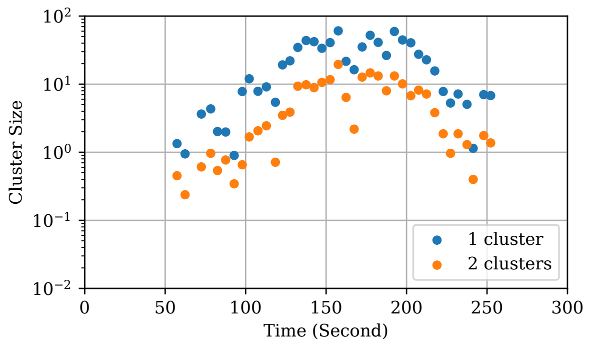
Quantifying RSSI variations is the first step in constructing a model capable of distinguishing between indoor and outdoor environments. Subsequently, this metric will be integrated into a model to deduce the environment in which the signal was recorded. Given the anticipation of unique channel characteristics for each aircraft, the initial phase involves grouping packets received from individual aircraft together. While time-series analysis over all packets received from an aircraft might seem suitable, we focus on local variations in the RSSI over time rather than long-term changes. Consequently, we divide the RSSI data for each aircraft into small windows to assess the variation within each window. This approach prevents the mixing of different channel conditions during analysis, as the long-term RSSI behavior changes when a receiver may have a line of sight to an airplane for a few minutes and then loses the direct path due to obstructions. The experiments indicated that a window size of 5 to 10 seconds is optimal —– long enough to capture sufficient samples yet short enough to be immune to long-term changes.

Airplanes typically broadcast about 10 ADS-B messages per second, including required location announcements and responses to air-to-air and ground-to-air interrogations. Therefore, a window size of 5 to 10 seconds yields 50 to 100 samples. Central dispersion metrics, such as the standard deviation, cannot effectively quantify RSSI variation due to the two-band behavior detailed earlier. In Figure 3.22b, for instance, at time 50s, despite the signals being stable in each band, the standard deviation is high because the samples jump between the two bands. Therefore, a metric that accounts for this special pattern in the data is required.

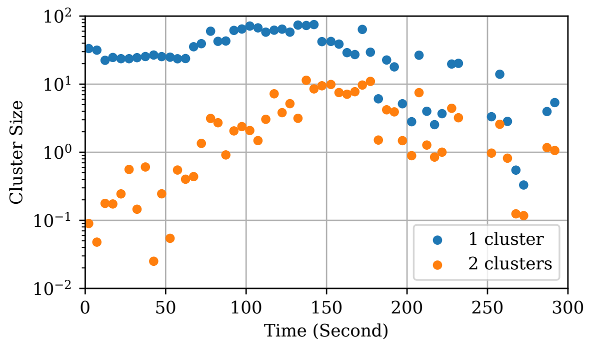
Using the K-Means clustering algorithm in each window is an effective method to quantify RSSI variation. We compute K-Means for just one cluster and two clusters and determine the cluster size. If the signal is stable in two distinct bands, the cluster size is large for one cluster and small for two clusters. In other words, if we group all samples into one cluster, all samples are far from the center, which is somewhere between the two bands. However, when clustering the data points into two clusters, the data points will be close to their respective cluster center, resulting in a small cluster size. When the data does not exhibit the two-band behavior, using 2 clusters does not significantly reduce the cluster size (only the expected reduction in cluster size for uniformly distributed data). Note that increasing the number of clusters almost always decreases the cluster size due to the nature of clustering. However, when the data is in two bands, increasing the number of clusters from one to two decreases the cluster size significantly. Table 3.4 shows the expected cluster size if the data points are scattered uniformly in a range of 10, 20, and 30 dB. It also shows the cluster size for a perfect two-band distribution with no variation for reference. The table shows that when the points are scattered randomly, the cluster size is reduced. However, due to variation, the cluster size is still large in the case of two clusters compared to the two-band case.

Range	10 dB		20 dB		30 dB	
Clusters	1	2	1	2	1	2
Two band	25.0	0.0	100.0	0.0	255.0	0.0
Uniform	8.3	2.1	33.3	8.3	75.0	18.8

Table 3.4: The expected cluster mean distance to center, or inertia, for different variation ranges and number of clusters.



(a) Indoor receiver data, from Figure 3.22a.



(b) Outdoor receiver data, from Figure 3.22b.

Figure 3.24: The size of cluster in each window as a metric to quantify RSSI variations.

Figure 3.24 illustrates the cluster size for 1 and 2 clusters when applying the K-Means algorithm to the data from the experiments explained earlier in Figure 3.22. The data is divided into windows of size 5 seconds, with K-Means clustering applied to the data in each window. We utilize the Python `scikit-learn` [293] library to implement the K-Means algorithm on the RSSI data. The clustering algorithm is run 10 times with random cluster center locations. To calculate the cluster size, we compute the sum of squared distances to the nearest cluster center divided by the number of data points. Figure 3.22b illustrates that, for the outdoor receiver, during the initial 50 seconds when the RSSI is very stable (in two bands), the cluster size significantly reduces when the data points are clustered into two clusters. However, throughout the rest of the experiment, as the fluctuations of RSSI increase, the distance between the cluster sizes of $k = 1$ and $k = 2$ decreases. This pattern is also observed for the indoor receiver, as depicted in Figure 3.22a.

Next, we employ this metric to ascertain whether a receiver is located indoors or outdoors solely from the Received Signal Strength Indicator (RSSI) of ADS-B packets. The method utilizes a threshold-based approach to assess the stability of RSSI values within a window, thereby determining if a LOS exists between the airplane and the receiver. If the signal is deemed stable, the algorithm predicts that the receiver is positioned outdoors. The algorithm follows this procedure:

1. For every airplane, the system divides the data into small windows (i.e., 5 seconds),
2. For every window, it clusters the data into 1 and 2 clusters and calculates the corresponding cluster sizes.
3. If the cluster size with 1 and 2 clusters are CS_1 and CS_2, respectively. If $CS_1 - CS_2$ is bigger than 10, then we consider CS_2 as the cluster size; otherwise, CS_1.
4. If the cluster size is smaller than a threshold (i.e., 0.1), the RSSI variations are considered *stable*; otherwise, *unstable.*
5. If there is at least one window across all airplanes with *stable* signal, the system associates it with the existence of a LOS path and estimates the environment to be outdoor. If no window with *stable* signal is detected, the environment is estimated as indoor.

Running this algorithm over all RSSI traces collected in a variety of environments using different SDRs, as described above, results in predictions summarized in Table 3.5. The results indicate that the algorithm correctly detects all outdoor receivers with no false negatives. However, approximately one-third of indoor receivers are categorized as outdoor "in error," resulting in an overall accuracy of 78%. Upon investigating these indoor cases classified as outdoor, a striking pattern emerges. It turns out that most of these cases had a LOS to an airplane through a nearby window. **A challenging question arises: is it really incorrect to classify these cases as outdoor?** It is essential to recall that these classifications aim to automatically detect indoor and outdoor environments for the express purpose of adjusting the transmission to avoid interference with other tenants in the same band. Therefore, if there is a direction from which the victim receiver could receive the signal, much like an outdoor transmission (with the exception of slight attenuation caused by the window glass), it might be appropriate to classify the radio as outdoor to prevent interference.

A natural follow-up question based on these results is *why do we see many false positives for outdoor, but no false negatives*? The authors of [248] discuss the confounding effect

		Predicted	
		Outdoor	Indoor
Actual	Outdoor	0.26	0.00
	Indoor	0.22	0.52

Table 3.5: Outdoor/indoor prediction for model-based method versus ground truth.

of windows on their results, suggesting that some directions may be more "outdoor" than others, even for an indoor radio. The spatial diversity of ADS-B signals, along with the fact that they include the location of the transmitter, allows attribution of each received packet to one of the eight sectors mentioned in Section 3.5 and hence to predict outdoor or indoor status per sector. Because the eight sectors are relatively coarse and aircraft can cross between sectors as they travel, we count an outdoor prediction for a windowed segment of data as correct as long as either the sector it is assigned to or one of the neighboring sectors is labeled as outdoor. A prediction of indoor is only marked correct if the sector it is assigned to is labeled indoor as well. This flexibility in assigning outdoor labels also reflects the desire of any incumbent system operator to ensure that other users err on the side of safety when setting their transmit powers to prevent interference.

Table 3.6 shows the results for per-sector predictions. The true positive rate falls from 100% to 57%, but the false positive rate also shrinks from 30% to 5%. To determine indoor/outdoor per sector in a real system, we need a way to tune the true positive rate. The next section details an ML-based bi-level classification method that allows a system to select the true positive rate.

		Predicted	
		Outdoor	Indoor
Actual	Outdoor	0.14	0.11
	Indoor	0.04	0.71

Table 3.6: Outdoor/indoor prediction for model-based method per-sector versus ground truth.

ML-based Classification

To improve predictions and to produce a probability score for outdoor vs. indoor for each sector, rather than a single yes or no prediction per window, we introduce a random forest-based bi-level classifier and additional data processing. We start by splitting the dataset into five chunks by recording and sector to prevent cross-contamination between train and test data. These chunks are used to perform 5-fold cross-validation with the following data processing, independently training five classifiers on four chunks and testing on the fifth. For each fold, the bi-level classifier performs the following steps:

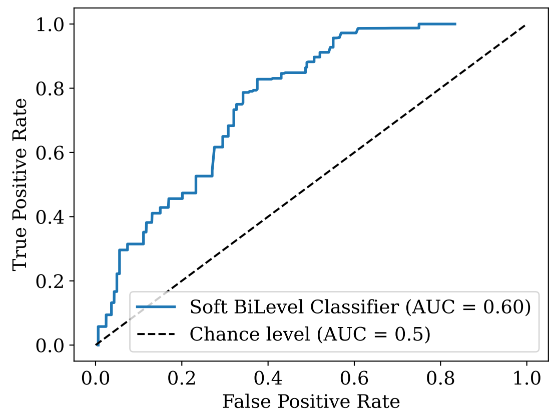
1. Append time series features produced by TSFRESH [295] to the 1- and 2-cluster sizes described in the previous section.
2. Use feature importances calculated using an Extremely Randomized Trees classifier [296] to perform feature selection for each training set.
3. With the downselected features, train a random forest classifier [297] with ensemble size 100 and maximum tree depth 8 to predict indoor/outdoor per 10 second window. The output of the random forest classifier constitutes the first level of the bi-level classifier.
4. For the second level, collect the predictions for all 10 second windows belonging to a specific sector and recording, either as a yes/no prediction (hard decision) or a probability (soft decision).
5. Combine predictions either through voting (hard decision) or combining log-likelihoods (soft decision) to produce a final probability of outdoor for a sector as the second level of the bi-level classifier.

Figure 3.25 shows the ROC curves for the bi-level classifier with soft and hard decisions. Despite the fact that soft decisions preserve additional information from the first level to use in the second level of classification, the classifier performs better with hard decisions (AUC= 0.72) than soft (AUC= 0.6). This suggests that there is high variance in the predictions from the first level. This is unsurprising since this version of the system does not distinguish between nearby aircraft with high SNR and distant aircraft with low SNR. Table 3.7 shows numerical results for hard- and soft-decision classification. The accuracy values are the best achievable by tuning the classification threshold.

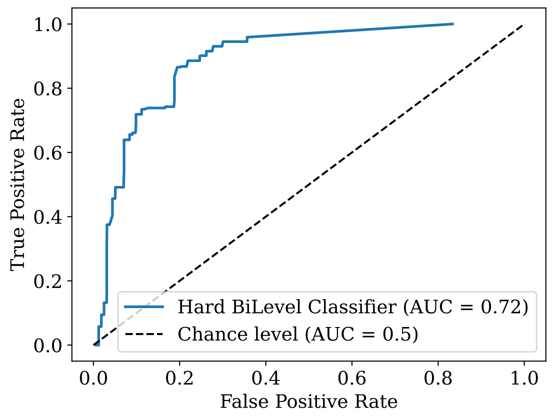
	Train Acc. (%)	Test Acc. (%)	AUC
Soft Decision	83	78	0.60
Hard Decision	96	85	0.72

Table 3.7: Results for the bi-level classifier. Accuracy is the maximum accuracy achievable by adjusting the classification threshold.

For the hard decision bi-level classifier, to achieve a true positive rate of $> 90\%$ requires a false positive rate of 25%. Diving into the false positives, however, reveals a particular pattern: one location, inside a windowless closet on the top floor of a home, experienced false positives for seven of eight sectors. **This suggests building materials play a large role**



(a) ROC curve for the bi-level classifier with soft decisions.



(b) ROC curve for the bi-level classifier with hard decisions.

Figure 3.25: ROC curves for bi-level classifier.

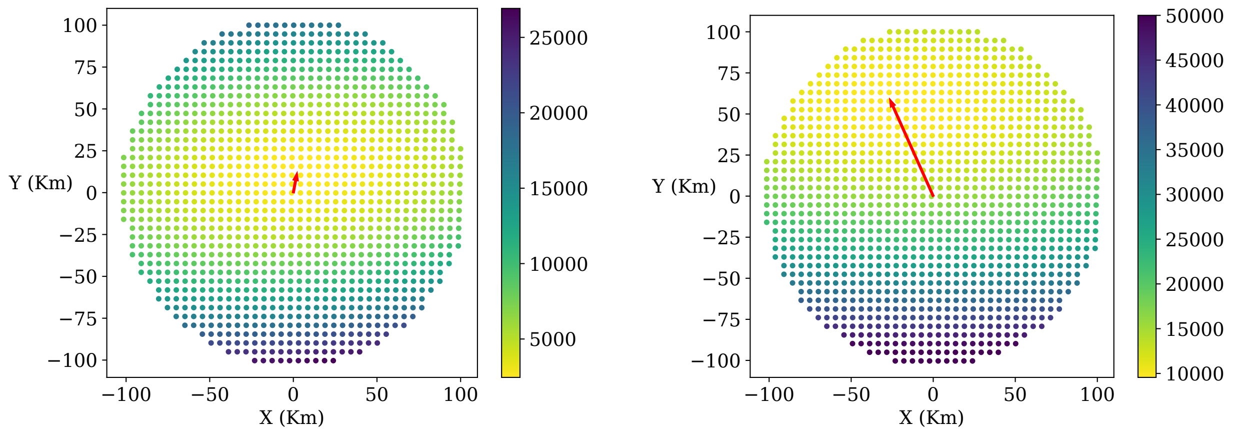
in whether the radio environment appears to be outdoors even when a human perceives a radio as inside. Many of the remaining false positives occurred in the sectors with the highest levels of aircraft traffic, especially where traffic patterns brought aircraft close to the sensor. This implies the performance of the system can likely be improved by including information about the range of an aircraft during classification as well as accounting for the density of traffic, perhaps by including a ground truth estimate of how many aircraft are present obtained from a source like FlightRadar24.

	Indoor Expts.		Outdoor Expts.	
	Predicted	Labeled	Predicted	Labeled
Outdoor Sectors (%)	39	17	52	50

Table 3.8: Fractions of sectors labeled and predicted as being outdoor for experiments with the sensor located indoors and outdoors.

Location Verification

An important factor in trusting the data from a sensor node is the node's location. Can a crowdsourced node provider claim to be in a nearby city just tens of kilometers away? If so, how do we detect it? Doppler shift is very sensitive to the location of the observer and even deviations on the order of kilometers can cause a mismatch between the theoretical model and empirical results with only basic processing. We use this fact to automatically verify the location of a radio sensor only using signals of opportunity. As discussed in Section 3.2, with additional processing it may be possible to determine node location down to 10s of meters which can be beneficial in applications that require higher accuracy, such as regulatory compliance in the CBRS band. Note that the goal of this work is not to design a localization system but rather to showcase the potential of SoOps in labeling radio sensor metadata and how automatic labeling can increase trust in distributed systems.



(a) Correct reported location. (b) Incorrect reported location.

Figure 3.26: Verifying the location of a sensor by comparing measured and expected Doppler shift. The colors represent the MSE.

As illustrated in Figure 3.26, we consider a circle around the claimed location (with a radius of 100 km in this case) and generate a grid of points separated by a fixed distance (i.e., 5 km). For a given satellite pass, we calculate the expected Doppler shift for each point on this grid. We then compare the empirically obtained Doppler shift curve with each of these candidate locations. We compute the mean squared error (MSE) between the theoretical and experimental Doppler curves. This entire process is repeated for several satellite passes, and, calculating the average MSE at each point. The color at each location in Figure 3.26a represents the average MSE obtained for that location over 13 satellite passes. We then identify the location with the minimum average MSE. The red arrow in the figure extends from the claimed location to the estimated location. In this case, the claimed location is indeed correct, so the red arrow indicates the error in our estimation, 10.1 km in this test.

Figure 3.26b replicates the same experiment, but this time, the claimed location is incorrect (approximately 50 km Southeast of the actual location). Specifically, the claimed location is Stanford University, while the radio sensor was physically at the University of California, Berkeley. The figure vividly illustrates the deviation, with the arrow's length in the experiment measuring 58.1 km, aligning with the difference between the claimed and actual locations.

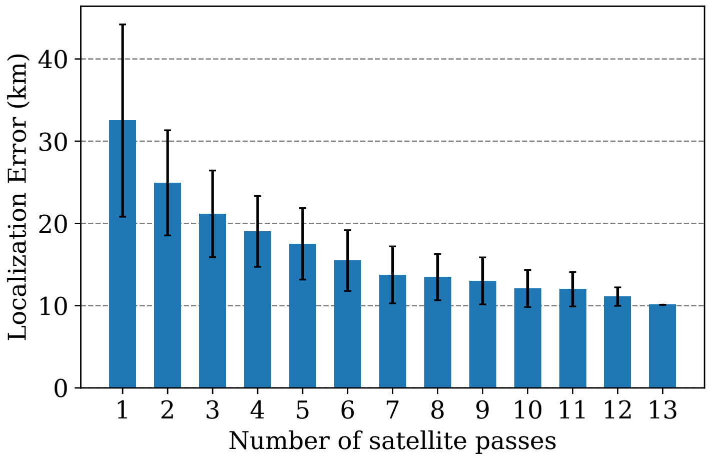


Figure 3.27: Localization error versus number of satellite passes used as input to the localization algorithm.

The previous experiments estimated the location using 13 satellite passes over the sensor node. Next, we assess how reducing the number of passes impacts the accuracy of estimation. We repeated the experiment with the correct location of the node, but instead of using all the data, we performed localization using a subset of passes. This process was repeated 100 times for each number of satellite passes, ranging from 1 to 13. Each of the 100 runs randomly selected a subset from the total 13 traces and calculated the error. We then determine the average and standard deviation for each number of passes. Figure 3.27 illustrates the results of this experiment. As the number of satellite passes for estimating the node's location reduces, the accuracy decreases. However, a noteworthy finding is that even with just a few passes, an average error of under 20 km can still be achieved. In fact, even a single pass can provide valuable information, revealing the approximate location of the sensor node.

3.6 Conclusion

This chapter demonstrated how environmental supervision through signals of opportunity (SoOps) enables a new paradigm for automatic metadata generation and verification in distributed sensing systems. The work established two key capabilities: 1) the ability to automatically and continually evaluate sensor coverage and performance characteristics without human intervention and 2) the ability to automatically generate and verify crucial metadata like location and installation environment.

The research showed that by combining observations from mobile SoOps, like aircraft ADS-B signals and LEO satellites with stationary sources, such as cellular networks and broadcast TV, a system can automatically:

- Evaluate frequency sensitivity across multiple bands.
- Determine effective field of view with 10-15 degree accuracy.
- Classify indoor/outdoor installation environments with 85% accuracy.
- Verify location claims to within 10-20 km using satellite Doppler measurement, with potential for 10 m accuracy.
- Detect installation issues and environmental changes during operation.

Importantly, these capabilities work with low-cost commercial hardware (under $200) and don't require specialized equipment, making them practical for widespread deployment. This enables a fundamental shift from manual metadata entry and verification to automatic, continuous evaluation and updates. Systems that learn about their environment using signals of opportunity can detect mismatches between human-provided and automatically measured metadata to identify improper installations or changes in operating conditions that software alone cannot detect.

Several challenges remain for future work:

- Developing end-to-end systems that optimize when to perform measurements and inform users if important values change during operation.
- Establishing comprehensive trust frameworks while preventing data fabrication.
- Incorporating additional RF sources for more robust calibration.
- Extending calibration techniques for specific application requirements.
- Protecting privacy when using SoOps for metadata generation.

As spectrum sharing becomes more prevalent and distributed sensing networks in general grow in importance, the ability to automatically evaluate and verify sensor capabilities becomes crucial. The environmental supervision paradigm demonstrated here provides essential building blocks for trustworthy, distributed systems that can scale while maintaining data quality and reliability. Future work can build on these foundations to enable new forms of metadata that humans cannot practically provide, like vector representations of sensor coverage areas and dynamic environmental characteristics. This research represents an important step toward practical, trustworthy spectrum monitoring at scale. The techniques developed here could extend beyond spectrum sensing to other distributed sensing applications where automatic metadata generation and verification are valuable for ensuring system reliability and performance.

The crucial re-conceptualization behind this work is to treat the densely occupied spectrum full of legacy systems as a feature, not a bug. These systems are designed to have high accuracy and reliability and can serve as precision references for the world around you, particularly when operating without trust or with low-precision components [137], [138]. After all, which is more trustworthy, a NOAA satellite or a backyard hobbyist? Each of these legacy signals carries valuable information in their physical representation. Doppler shifts, showing the relative motion between a sensor and another object, are one physical signal. Signal to noise ratio is another. Often, this information is valuable on its own. Each signal can also provide information through its content, like the identity or location of a transmitter. Combining these two sources of information from a single signal and across multiple signal types can provide insights no single source alone could provide. Legacy sources are relatively fixed, often changing type or location only on the scale of decades. This provides an opportunity for the open-source community to collaborate on 'plugins' that extract information from each signal type and provide it in a common format, similar to the plugin model of GNU Radio [290]. A library of such plugins would serve as a foundational resource for further research on SoOps and environmental supervision.

On a broader level, the goal of this work is to move towards an "AI RF module" that can be cheaply dropped into any radio and allow it to function as a full cognitive radio [17]. Such a module would sense and communicate with peers to learn about the state of the world, then make intelligent decisions about resources and behaviors to both meet its own communication needs and act as a good RF citizen. The work presented in this chapter demonstrates methods to obtain cheap situational awareness, an important component of a cognitive radio. The previous chapter points the way toward learning behaviors cooperatively. Future work, and work that already has much research published separately, involves communicating knowledge of the world to and receiving knowledge from peer actors and then using that knowledge to make intelligent and fair decisions.

Chapter 4

Towards Synthetic Datasets for Data-Scarce Domains

4.1 Introduction

The massive recent success of generative language and image models is driven by the power of the transformer architecture [79] and large-scale unsupervised pre-training followed by supervised fine-tuning for tasks [298]. Unsupervised pre-training allows designers to leverage extremely large datasets with limited or no labeling, larger than could be reasonably created with human supervision. Indeed, experiments on dataset size show that it directly impacts the potential accuracy of a model alongside computing FLOPs (floating point operations) and model parameter count. This combination of factors results in what are known as scaling laws, dictating the required dataset size to allow the most compute-efficient pre-training at a given model size [299]. Scaling laws have been observed in generative language [299], image [300], [301], and mixed-modality models [302]. The availability of large-scale datasets clearly both drives and limits the performance of generative models in the language and image spaces. Unfortunately, not every domain has ready access to large datasets researchers or industry can use to develop large generative models.

Large-scale datasets for pre-training primarily come from scraping the internet for publicly available text and images. GPT2, the first widely publicized large language model (LLM), was trained on 40 GB of text scraped from outbound Reddit links [303]. LLaMA, a recent open-source LLM, was trained on more than 4 TB of data from multiple large collections of web data [304]. Many companies maintain their own private datasets, but some public versions have also been released for the research community to take advantage of [305]–[307]. Although some images have descriptive captions, many do not and must be somehow labeled for text-to-image generation. These labels can be synthetic, generated by open foundation models such as CLIP (contrastive language-image pre-training), which themselves rely on large scraped datasets [308] and techniques such as self-supervised learning [309], contrastive methods [310], self-training approaches [311] and generative modeling [312]. Another area that benefits from large publically available datasets is audio generation, often speech or music [313]–[318]. These datasets vary in size from a few gigabytes each [318] to over a terabyte [316], [317]. I view the common enabling characteristic of these domains as the fact that the data can be both produced and consumed directly by the average human. Nearly anyone can listen to audio, view an image, or read text, and similarly record, capture, or write examples to be published for anyone to see. In the same vein, generative models for these domains produce outputs useful to everyone, driving interest and investment.

In contrast, scientific topics generally do not benefit from a planet's worth of internet citizens generating content that can be scraped and used for training. Despite this, there is a long history of using models to generate synthetic data for specific processes. Examples include neuron potentials in neuroscience [319], [320] and galaxy formation in astrophysics [321]. Others involve more specific tasks that condition generated data on class labels like healthy or diseased physiology [322], synthesizable or unsynthesizable molecules [323], or perform data augmentation on small EEG datasets [324] or imputation to replace missing or poor-quality channels in EEG recordings [325]. Some domains, such as chemistry, benefit from their own sources of data like the ChEMBL database of bioactive compounds [326]. Astronomy and particle physics have large international collaboration projects producing streams of data for analysis and training. But what about wireless signals?

A great deal of work goes into simulation packages for wireless systems. MATLAB has an entire toolbox dedicated to it [327], and NVidia created a GPU-accelerated library for link-level simulations [328]. No matter how much detail goes into a simulation, though, it can't accurately represent every detail and imperfection that exists in the real world. This is particularly important for generative models learning a distribution since they only reproduce what they are given. Existing work on generative wireless modeling, such as channel estimation and simulation, is often trained on synthetic data only [169], [329]. Other work attempts to improve real-world performance through distributed learning [330] or by training on simulated data and fine-tuning on a much smaller real dataset [168].

Dataset Content	Dataset Size (GB)
Wi-Fi traffic [331], [332]	55
LTE traffic [333]–[336]	501
LoRa traffic [337]	1000
RFID signals [338]	200
Traffic from hovering UAVs [339], [340]	9
Massive-MIMO signals [341]–[348]	12,161
Total	13,926

Table 4.1: Datasets found on RFDataFactory and OpenRANGym containing IQ data.

The natural follow-up question is why there is not more work using recorded real-world data. I surveyed every dataset linked from RFDataFactory [172] and the Open-

RANGym [349], looking for datasets with raw IQ data. Raw IQ data provides the most fundamental measurement of wireless signals that can either be directly trained on or processed to produce metrics for training indirectly. Other datasets include features like packet captures for CSI prediction, but they are not generally useful in the same way raw IQ is. The datasets include a mix of laboratory captures in an anechoic chamber or directly cabled SDRs and recordings of radios on campuses or in experimentally licensed testbeds such as POWDER [350] — no recordings of signals in the wild or so-called "wardriving" where sensors are mounted on a vehicle and capture data as they move around. These datasets are described in Table 4.1. Note that 12 of 14 TB of the data come from Massive-MIMO setups and another terabyte from a single LoRa radio fingerprinting dataset. That leaves just one terabyte of all other signals. For comparison, the LAION-5B-en dataset [1] [352] used to pre-train Stable Diffusion [353] contains 2.32 billion images, which at 384×384 resolution and with an average JPEG compression ratio of 10:1 would be 93 TB. There is a clear discrepancy between the availability of wireless signals data and other domains.

Given that it should be easy to set up an SDR listening on a known LTE frequency and record terabytes of traffic over the course of a day or week, there must be a reason for the paucity of raw IQ data. In fact, a confluence of legal privacy protections and court precedents combined leaves any researchers in legal limbo when publicly releasing wireless signal recordings. The Supreme Court case Kyllo v. United States ruled that using thermal imaging to search for growing marijuana is an unconstitutional invasion of expected privacy even though the "search" could be performed remotely off Kyllo's property [354]. Separately, the Electronic Communications Privacy Act of 1986 updated federal wiretapping laws to make recording electronic communications, such as Wi-Fi or cellular signals, illegal without a warrant. Combined, these create a legal risk that even if a researcher unintentionally records protected data as raw IQ but never processes it to reveal information and then releases the data publically, they could be held liable for someone else using that dataset to obtain protected information. Demonstrating the real risk, Google settled a case over Wi-Fi data its Street View cars collected while wardriving containing emails and other data for several million dollars [355]. Even the government itself faces hurdles in collecting and publishing wireless datasets. A pilot program where the FCC mounted spectrum sensors and cell signal monitors on USPS vehicles to map coverage ended partly due to the potential legal issues with the data [173], [174]. In the face of these legal hurdles, it may never be possible to release an unaltered recording of in the wild wireless data.

If progress on machine learning for wireless signals requires access to large datasets, we must turn to alternate solutions for obtaining data. One potential solution is federated learning, in which each participant maintains a separate private dataset and learns their own model, which is sent to a central aggregator to produce the final version [175]. There are two primary problems with federated learning for our purposes. It requires central coordination, agreement on methods, and ongoing participation as new data becomes available,

[1]The original LAION-5B dataset was taken down in December 2023 to remove links to CSAM material [351]. A new version with illicit links removed was released in August 2024.

an extremely difficult task for multiple unrelated research groups and organizations. More importantly, it means that data remains private and can't be used inventively by unrelated researchers to advance the field. Federated learning is unlikely to be the solution to data woes in the wireless field.

Differential privacy is another option that faces its own set of challenges. Differential privacy involves adding noise either to the original data or to operations on the data, such as means or gradients during machine learning, and releasing only artifacts that result from the noising process [356]. This prevents freeform exploration of data, a common activity in the early stages of research. Applying differential privacy to time series data adds additional challenges for preserving privacy due to the volume of data and correlations across data points. Protecting every data point individually reduces the utility of the data, while protecting higher-level features may introduce privacy breaches through correlations [357]. This is even more challenging in the space of wireless signals where the most sensitive information, user content, is encoded *specifically* to protect it from corruption by added noise, potentially rendering differential privacy techniques useless without enough noise to destroy any utility originally in the data.

The solution I employ for the work in this chapter is to generate synthetic data from a generative foundation model. Ideally, a foundation model specific to wireless signals could be used, but the unique set of challenges facing large-scale modeling and generative work in this domain means that no such model yet exists [358]. Instead, other models will have to be employed as a substitute. Gunasekar et al. [359] showed they could train a state-of-the-art code generation model on a small high-quality dataset and a larger synthetic dataset of imperfect but on-topic and diverse examples generated by a much larger foundation model, GPT3.5. Other work shows that synthetic data can perform a role similar to knowledge distillation [360], improving the performance of downstream models with limited datasets by feeding them synthetic samples from a larger teacher model [361]–[363]. Synthetic data created by generative models, rather than functions like noising, cropping, rotation, and superposition, is growing in popularity as a data augmentation tool to improve model performance in computer vision tasks [364]. Some tasks even achieve equivalent performance with purely synthetic datasets. Although synthetic data may be a useful tool to address data scarcity, it is important to be careful with its use. The growth of synthetic data on the web has introduced the risk of a phenomenon known as model collapse, where recursive training on synthetic data results in forgetting and reduced performance [365]–[368]. It will be important when producing and using synthetic datasets to ensure that they are not recycled as contaminants, reducing data quality.

There is a whole suite of startups providing synthetic data generation services. Some focus on specific domains like computer vision and robotics, including Lexset, Bifrost AI, and Synthesis AI. This subset focuses as much or more on providing synthetic data to augment expensive or small real datasets as on ensuring privacy. They may also have their own generative models which produce synthetic data from inputs like 3D CAD models. Other startups like ExactData, Gretel, and Accelario focus more on data privacy and compliance. Some don't focus on a specific aspect, providing a more general-purpose platform, such as

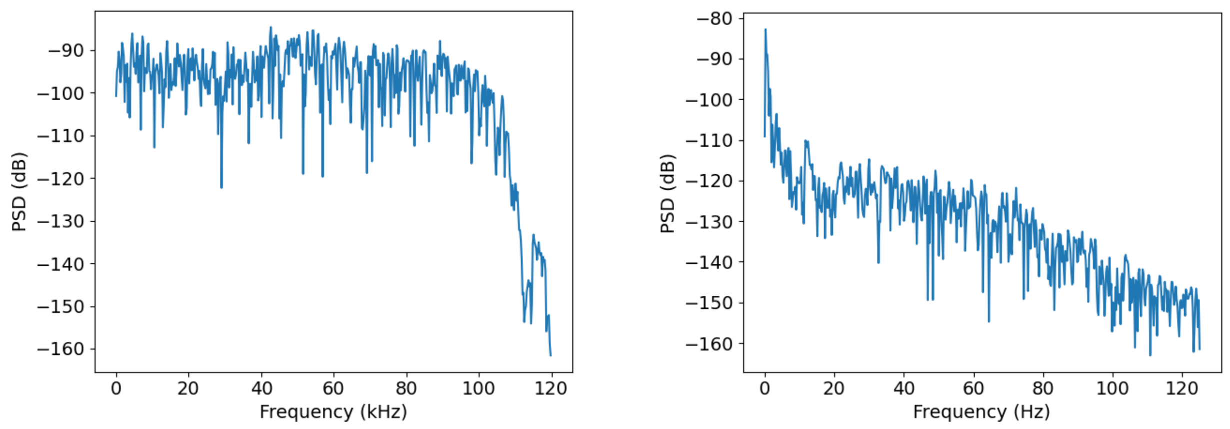
Tonic AI or GenRocket. These companies typically provide data storage and interfaces and either custom modeling or an interface to train your own model.

Inspired by the Riffusion project [369][2], in this chapter, I show the potential to repurpose foundation models trained in one domain, like images, by fine-tuning for use in generating synthetic data in another, like time series. Repurposing large foundation models in this manner has two advantages. First, fine-tuning is much cheaper computationally, and thus financially, than training a new model from scratch. Second, by representing data from one domain (time series) in a manner interpretable in another (spectrogram images), we can take advantage of the learned interpretive and predictive power a foundation model has in the original domain. We can also take advantage of the surrounding training infrastructure and associated models like CLIP image and text embeddings available in the more developed original domain. This allows actions like conditioning the image output of the fine-tuned model on a text input, which would require training further models from scratch in the original domain.

The eventual goal of this work is to be able to take data containing private information and replace it with data generated in such a way that it effectively anonymizes the original data while preserving any and all characteristics that may be important for downstream learning tasks. Example applications are changing private data contained in Wi-Fi packets or protected health information (PHI) that may be implicitly contained in a biosignal recording like an EEG (electroencephalogram). The first step, demonstrated in the body of this chapter, is to produce a synthetic version of a dataset with protected information controllable via conditioning during generation. The next step, which may or may not be required to begin releasing trusted synthetic datasets, would be to use image-to-image techniques to produce copies of each original private sample with protected information swapped out randomly by additional conditioning. Although generative models do not guarantee privacy, even potentially regurgitating memorized data [370], there is hope that researchers looking to publish data could receive legal protection by applying "best effort" anonymization through synthetic datasets generated with best practice techniques.

Applying anonymization through conditioned generation faces two major challenges in the domain of wireless signals. Wireless protocols are designed to be as information-dense and high dimensional as possible because this makes the most efficient use of limited bandwidth. The protected data in wireless signals is also very high dimensional because it consists of the sequences of near-random bits carried by the signal, in the case of digital communication schemes. Even older analog signals such as FM radio have very little bandwidth expansion between the audio signal and metadata they carry and the occupied bandwidth of the transmission, perhaps only 2x or less depending on optional components. In contrast, most of the energy and information is concentrated in the first 10% or so of the EEG spectrum. Put together, it seems to be a very difficult task to jump directly into synthesis of wireless datasets.

[2]Riffusion converts music snippets to spectrograms and uses a fine-tuned stable diffusion model to generate new snippets conditioned on text prompts.



(a) PSD of an FM radio broadcast (KPFA). Up to the downsampling filter cutoff at 100 kHz, all frequencies are equally occupied.

(b) PSD of an EEG channel. Only small sections of bandwidth around 3 Hz and 12 Hz have significant energy relative to the rest of the signal.

Figure 4.1: PSDs of example FM radio and EEG recordings demonstrating the difference in bandwidth occupied.

Instead, this work [371] selects a similar time series signal from a domain that also faces privacy concerns when wishing to publish raw data but mitigates some of the difficulties above: EEG recordings of brain activity. EEG signals can be used to determine personally identifying information such as age and gender [372], [373], and PHI such as the presence of Parkinsons disease [374], [375]. These protected features are low dimensional and can be represented as a human-understandable and hence text encoder-interpretable phrase like "a male 80-year-old subject with Parkinsons' disease," significantly reducing the difficulty of applying conditioning during synthesis. Additionally, EEG signals are much less dense spectrally than wireless communication signals. Figure 4.1 demonstrates example power spectral densities (PSDs) of an FM radio signal and an EEG recording. This should make it easier for a generative model to learn to reproduce the distribution of EEG source data during fine-tuning.

After training diffusion models to produce synthetic EEG spectrograms, this chapter evaluates the suitability of the synthetic data in two ways. First, by calculating distributional similarity metrics employed in the computer vision space, I evaluate how closely the synthetic data appears to match the original distribution in spectrogram image space. This also provides insights into where bottlenecks in improving generation performance may exist. Second, I train gender classifier models on real and synthetic data, then test performance on each dataset to evaluate how well the diffusion models can control for protected classes in the data. Overall, this work provides a proof-of-concept that repurposing foundation models from domains with abundant data to leverage and amplify smaller datasets in other areas has potential to address data woes.

4.2 Related Work

The concept of representing a 1D signal as spectrograms for generative modeling with image diffusion models is not unique. Apart from Riffusion [369], which originally inspired this work, [376] uses diffusion models to produce spectrograms of sound clips which also *look* like something related to the sound. For example, the sound of a dog barking may have a spectrogram that looks like a dog, as in Figure 4.2. Spectrograms depict the spectral energy content of a signal, removing phase information and preventing direct recovery of the original time domain signal. Although this chapter operates only on spectrograms after they have been produced from the EEG data, both [369] and [376] recover an estimate of the original signal using the Griffin-Lim algorithm [377].

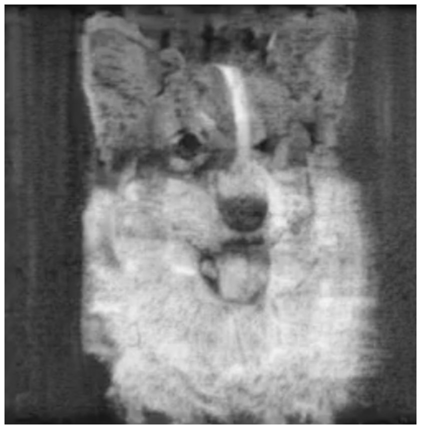


Figure 4.2: A spectrogram of an audio clip that sounds like a dog barking and looks like a corgi [376].

Generative modeling has also been used to address scarce data in other medical domains. [378] surveys the history and applications of synthetic data for medical purposes, primarily focused on direct statistical models implemented in R or Python. Deep generative models, namely GANs and diffusion models, have been used for synthetic medical imaging in chest x-rays [379]–[381], optical coherence tomography [382], and MRI [383]. [384] uses Mamba-based diffusion models [385] to correct endoscope images for exposure. For EEGs, [324] surveys non-generative data augmentation methods and concludes that no one method is best, depending on the downstream task. [386] demonstrates channel imputation for multi-channel EEG recordings using diffusion, similar to [387]. In [388], the authors generate EEG spectrograms with diffusion models and use them to augment training data for an emotion classifier.

One shortcoming present in much of the literature is that evaluation of the synthetic data tends to either report a distributional similarity measure such as FID (Frechet Inception

Distance) with no reference for how well diagnostically important features are represented, or performance of a classifier trained using the synthetic data without statistical similarity measures. Even the similarity measures presented tend to include FID and related scores, which have been shown to be biased, particularly with fewer than $\sim 50,000$ samples and to perform more poorly on images not represented in the ImageNet dataset [43], [389]. For distributional evaluation, I use the statistically unbiased Maximal Mean Discrepancy (MMD) metric [389], [390] with alternate kernel features provided by DINOv2 [391] since it generalizes better to rare or unseen image distributions than CLIP, the model proposed in [390].

[392] surveys the state of EEG classification algorithms up to 2018, covering common feature selections, metrics, and classification algorithms including linear classifiers, NNs, non-linear Bayesian classifiers, nearest neighbor and hybrid schemes. A common difficulty experienced by those working on EEG classification is high variance over time (non-stationarity) and especially between subjects. For this reason, some papers such as [393] train a classifier model per-user for tasks such as decoding speech perception or predicting and classifying seizures [394]. Since the advent of the attention operation and transformer architecture [79], some authors have used attention in EEG classification models for emotion recognition [395], sleep state [396], and age and gender [372], [397], [398]. [398] used recordings of speech as well as EEG signals to predict age and gender through a form of principle component analysis and support vector machine classifiers. [372] uses features from wavelet decomposition and a random forest classifier to achieve $> 90\%$ accuracy on their dataset. In contrast, [397] used transformer networks on raw EEG data without explicit feature extraction to achieve similar performance.

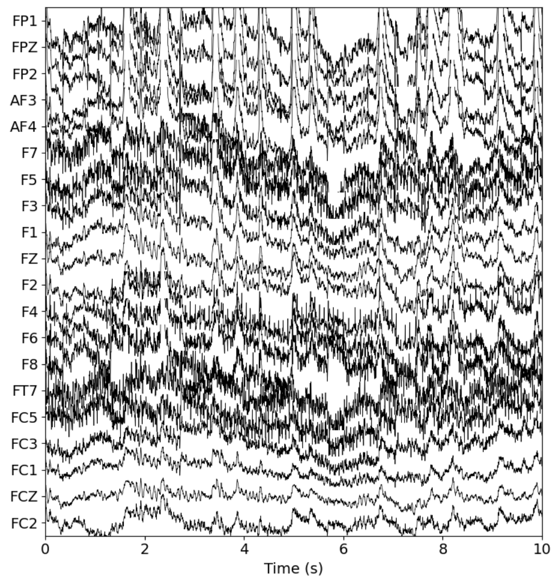
4.3 Methods

This section describes the reasoning for datasets used, preprocessing steps employed, and model selection and training. Ideally a comprehensive selection of datasets and model architectures and training methods could have been tested to better understand the interactions between pre-training, foundation models, and cross-domain knowledge transfer for synthetic data generation. Time constraints, however, dictated that only a small subset of available datasets relatively similar in format and content and two main model architectures could be tested. These results still point toward the potential of the technique and interesting future explorations.

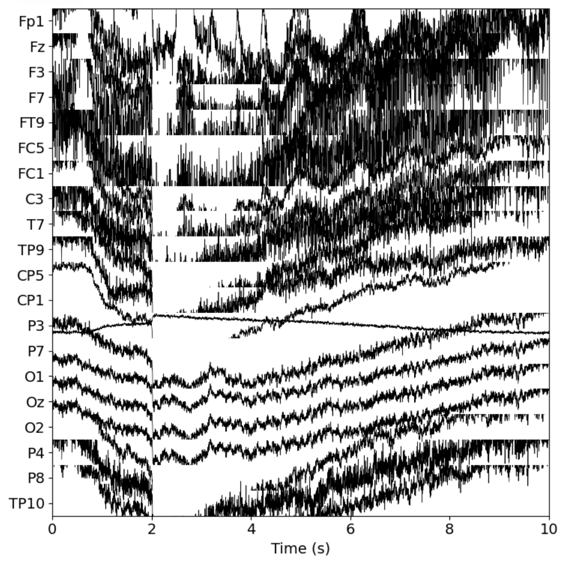
Dataset and Preprocessing

Just as wireless signal data has myriad potential variations in the number of antenna channels, bandwidth, SNR, protocol, and more, EEG data has similar complexity. Everything from the age of subjects (e.g., college students or those old enough to be at risk of Alzheimer's) to the number and placement of electrodes, contact quality, sampling rate and filtering, and whether the subject is at rest or active, healthy or potentially seizing at any given moment. Combined, these give each dataset a unique fingerprint that must be adjusted to produce common inputs for a model during training. To demonstrate both the difficulty and potential of using meta-datasets, formed from multiple unrelated source datasets, we chose to use the SEED-V [399] and Rest eyes open [400] datasets. SEED-V consists of 16 college-age subjects who were recorded watching video clips designed to elicit specific emotions. Rest eyes open consists of 149 subjects between age 50 and 90 at rest, 100 of whom had Parkinson's disease of various severities. These datasets were chosen because they are on the larger size at 38 GB and 3 GB, respectively, and because they contain a relatively even mix of genders and cover both young and old subjects when combined. They also contain a common subset of at least 20 channels measured at similar locations on the head, listed in Appendix I.

Figure 4.3 displays a plot of the raw EEG signals from the first few channels of a 10 second snippet of each dataset. The plots demonstrate differing noise levels, sampling rates, channels, and DC offsets. These features require some preprocessing before the results can be combined into the final spectrogram images. The preprocessing steps applied are enumerated in order below:



(a) Plot of a 10 second snippet from the SEED-V dataset.



(b) Plot of a 10 second snippet from the Rest eyes open dataset.

Figure 4.3: Plots of data snippets from the chosen datasets, showing the first 10 seconds of a random subject and the first 20 channels in the original ordering.

1. Extract a common subset of 20 channels in identical order.

2. Low-pass filter then decimate the data to 125 Hz.
3. Split each recording into 2000 sample chunks with 50% overlap.
4. Calculated a Hann windowed, overlapped, log-spaced frequency bin spectrogram per channel.
5. Convert the linear data to decibels and clip to a minimum power level to reduce dynamic range.
6. Concatenate the channels to form a 256×256 greyscale image.
7. Rescale the values to $0 - 255$, quantize to `uint8` values, and save as a JPEG image.
8. Store the gender, age, and health labels of the image to a metadata file.

Log-spaced frequency bins were empirically more useful for downstream classification tasks because much of the information in EEG signals is carried in very low frequencies, exemplified in Figure 4.1b. Log-spaced frequency sampling is similar to Mel cepstrum sampling common in audio signal processing [401], which aims to better represent how humans perceive audio signals. In this case, we aimed to better represent signals important for later classifiers. We used the adaptive windowing and overlapping methods from [402] to improve resolution and low frequencies and noise floor estimation at higher frequencies. To prevent contamination by the DC offset, we remove the offset while calculating the spectrogram and then reintroduce the offset in only the DC frequency bin. This allows a model to account for the DC offset without swamping data in near-zero bins by a much larger sidelobe from the DC signal. Figure 4.4 displays an example spectrogram after preprocessing.

One may ask why spectrograms would have any chance of conveying information about the EEG signals interpretable by a model pre-trained on images. Spectrograms were originally developed during World War II to provide a visual representation of sounds for analysis [403]. It follows that a technique designed for vision-oriented humans would be interpretable by image-generating models.

To prevent classification models from learning to identify individuals rather than useful features, the data is split by subject into 80% training and 20% test partitions. During training of generative models only, the data is reweighted by dataset and gender to reduce class imbalance resulting from different dataset sizes and gender ratios. At test time and while training classification models the data is left as-is.

Diffusion Models

We train two latent diffusion models to perform EEG spectrogram synthesis. The first is a fine-tuned Stable Diffusion v1.5 [353][3], referred to from here on as SDv1, trained on a 24

[3]This model was originally available under 'runwayml/stable-diffusion-v1-5' on Huggingface but has since been taken down. The 'CompVis/stable-diffusion-v1-4' model is a good substitute, since v1-5 only added additional fine-tuning to the UNet component.

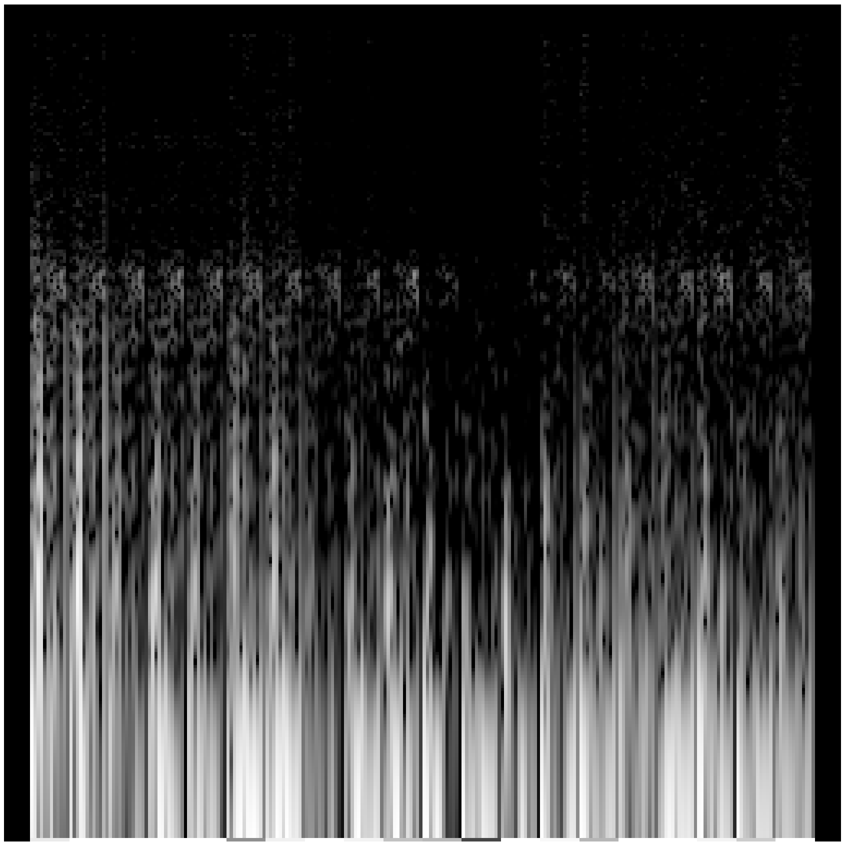


Figure 4.4: An example spectrogram after preprocessing.

GB Nvidia A10G GPU on AWS. This was the smallest GPU we could use without out-of-memory errors because training went unstable when using fp16 and bf16 mixed-precision training. We use SkyPilot to automatically manage environments and jobs [404]. We use a DDIM [405] scheduler during both training and sampling. Other training hyperparameters are listed in Appendix J. Conditioning for this model is provided by passing statements of the form

`an EEG spectrogram of a 80 year old, parkinsons disease diagnosed, male subject`, or
`an EEG spectrogram of a 66 year old, healthy, female subject`

with the portions corresponding to each label appropriately filled in. During training, approximately 23 GB of memory is used for batch size 4, with gradient accumulation employed to achieve larger effective batch sizes. Wall clock training time was 7.5 hours. We use an exponential moving average (EMA) [406] copy of the model for final results.

We recognize that not every researcher has the funding and confidence to use cloud computing infrastructure for training larger generative models, so we also trained a smaller diffusion model from scratch on a 10 GB Nvidia RTX3080 desktop GPU. The much smaller memory pool limited the size of the model we could use, so we chose to use a new architecture that performs linear attention and has been shown to achieve similar results as transformer models one weight class larger, Mamba [385]. After Mamba's release, several groups have adapted the Mamba architecture in various configurations for vision modeling rather than 1D data [407]–[411]. We adapt the DiS model and training script from [407], which uses DDPM [412] for training and sampling. Conditioning is provided by learned class embeddings for the four combinations of healthy/sick and male/female, and a fifth no-guidance embedding. Again, hyperparameters are provided in Appendix J. The DiS-B/2 model class can be trained using 9.9 GB of memory with batch size 4 on a desktop GPU. For our final results, we train a DiS-M/2 model with a batch size of 8 on a cloud GPU using 22 GB of memory and gradient accumulation to increase the effective batch size. Wall clock training time was 63 hours. We use an EMA copy of the model for final results.

For the SDv1 model we use the provided VAE model to produce and decode the latent distribution. For the DiS model we used the 'stabilityai/sdxl-vae' autoencoder, referred to from here on as SDXL, which shares its architecture with the SDv1 autoencoder but was fine-tuned using larger batch sizes and with EMA, increasing performance on reconstruction metrics. We tried to use the SDXL VAE model with the SDv1 UNet but found that the latent spaces did not align, producing strangely colored and blurred outputs, and further training did not help.

The synthetic datasets were generated using 50 sampling steps to produce 5,120 synthetic spectrograms each, approximately matching the size of the real training dataset. Classes for conditioning were selected uniformly randomly, and classifier-free guidance [413] with a guidance scale of 7.5 was applied. The MMD distributional similarity metric between real and synthetic datasets is calculated with `torch-fidelity` [414] using CLIP and DINOv2 features [391].

Classification Model

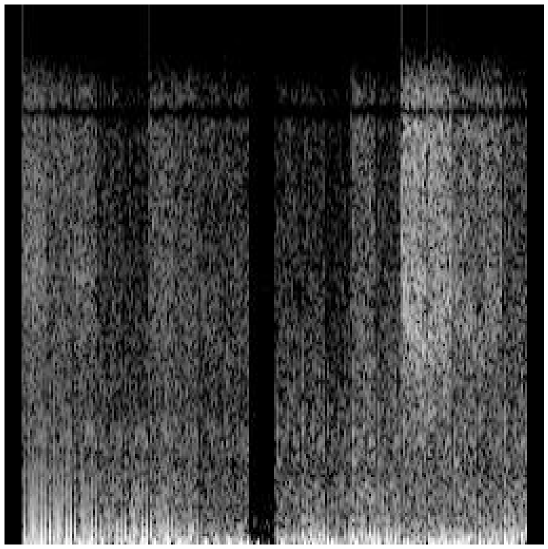
The gender classification model takes as input a grayscale (single channel) spectrogram image and outputs two logits, one for male and one for female. Applying a softmax operation would result in per-class probabilities. The model's architecture is presented in Table 4.2. Each convolutional layer is followed by a GELU [415] activation, and a dropout layer follows each pooling operation. The linear layer has no bias. The loss function is cross-entropy with label smoothing [416] to improve generalization. During training, we maintain an EMA copy of the weights and report results for both last-iteration and EMA versions of the classifier. Further hyperparameters selected through extensive search are available in Appendix J.

Layer Type	Kernel Size	Channels Out	Stride	Padding
Conv2d	5	8	1	2
AvgPool2d	2	-	2	-
Conv2d	3	16	1	1
AvgPool2d	2	-	2	-
Conv2d	3	32	1	1
AvgPool2d	2	-	2	-
GlobalAvgPool	-	-	-	-
Linear	-	2	-	-

Table 4.2: Gender classification CNN model architecture.

4.4 Results

If we are to have any hope of succeeding with *latent* diffusion models, we must first verify that the VAEs that map images to and from the latent space can accurately reconstruct the spectrograms from our dataset. Figures 4.5 and 4.6 show the original and reconstructed versions of an example spectrogram. To the human eye, both versions look nearly identical. There is very slight blurring and a small change in exposure, but no major differences. More quantitative results can be found in Figure 4.7. The 'self' columns represent the near-zero MMD score for two disjoint subsets of the training data. This suggests that if the VAEs could perfectly reconstruct each input, the resulting MMD would also be near-zero. The SDXL VAE produced better reconstructions according to both featurizers, as a lower MMD score represents a closer alignment between the data distributions. The MMD scores have error bars because the kernel matrices for MMD are computed over subsets of the data and averaged to avoid constructing and evaluating very large matrices. Otherwise, calculating the score would take hours to days for larger datasets. We use the default subset size of 1000.


(a) Original EEG spectrogram.

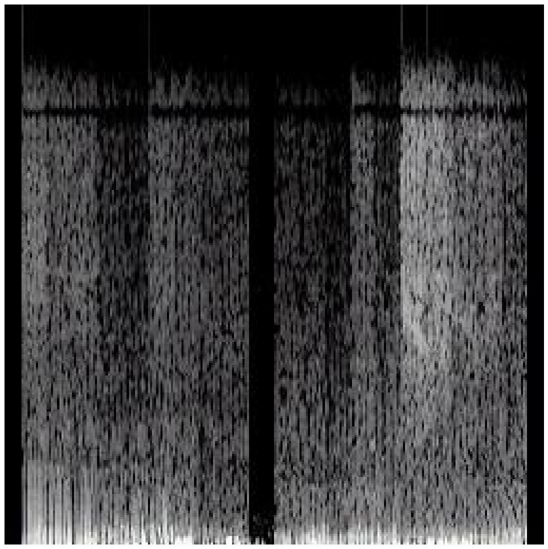
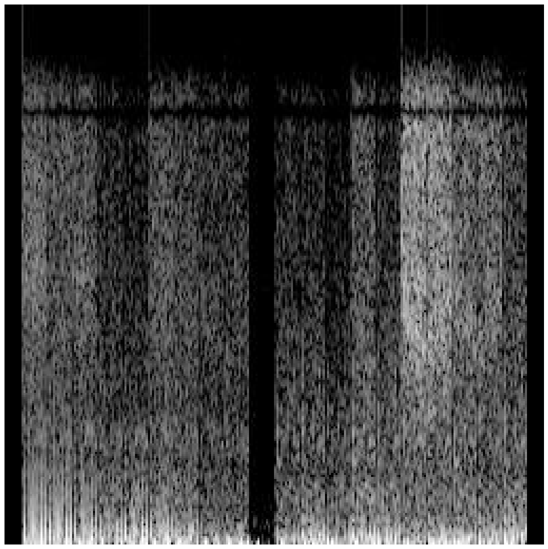

(b) SDv1 VAE reconstruction of the original spectrogram.

Figure 4.5: Original and VAE reconstructed versions of an EEG spectrogram, using the original SDv1 VAE.


(a) Original EEG spectrogram.

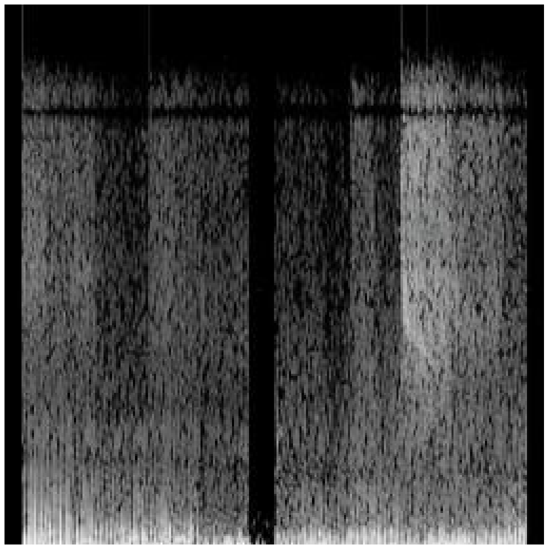

(b) SDXL VAE reconstruction of the original spectrogram.

Figure 4.6: Original and VAE reconstructed versions of an EEG spectrogram, using the updated SDXL VAE.

Now that we have a baseline for MMD using VAE-only reconstructions, we can evaluate synthetic datasets produced by the SDv1 and DiS diffusion models. As might be expected, the DiS model's output has a higher MMD than the SDXL VAE alone. It also loses out to SDv1 synthetic data. Since the SDv1 UNet is much larger than the DiS diffusion model and has much more pre-training, we expect SDv1 to reproduce the target distribution better. Surprisingly, the SDv1 synthetic data somehow has a lower MMD than the VAE used by the model can achieve on its own. The exact cause of this disparity is unclear. One possible explanation is that the encoder of the VAE loses or distorts information while compressing the spectrograms to the latent space. Then, somehow, the averaging process during training may smooth out the distortions, and the diffusion model learns to generate "good" latent representations that the decoder portion of the VAE can restore to accurate spectrograms. More investigation is required to produce a more confident explanation of this discrepancy.

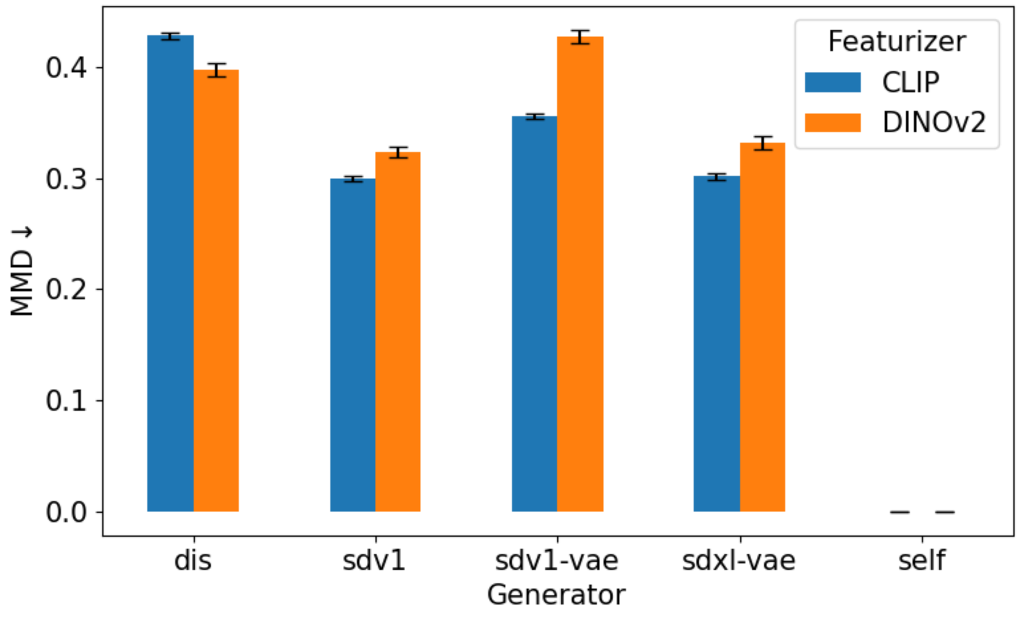


Figure 4.7: MMD metrics using CLIP and DINOv2 features for synthetic and reconstructed datasets.

The MMD scores are an important metric for understanding the distributional similarity of the synthetic data to real data. They can allow direct comparison between successive models and techniques applied to this domain. However, they do not represent the utility of the data for downstream learning tasks. After all, if we unconditionally produce a synthetic dataset, we lack labels for training a supervised classifier, leaving us no better off than before. Fortunately, we can apply CFG during dataset generation to produce data with associated class labels. We trained a gender classifier model per dataset on the real, DiS, and SDv1 synthetic datasets. We stored the last-iteration and EMA weights for each model to test whether models that generalize better do well on other datasets, i.e., whether the model easily overfits on a given dataset. Then, we measured the test accuracy on held-out data from all datasets. The results are shown in Figure 4.8. They are also in Table 4.3.

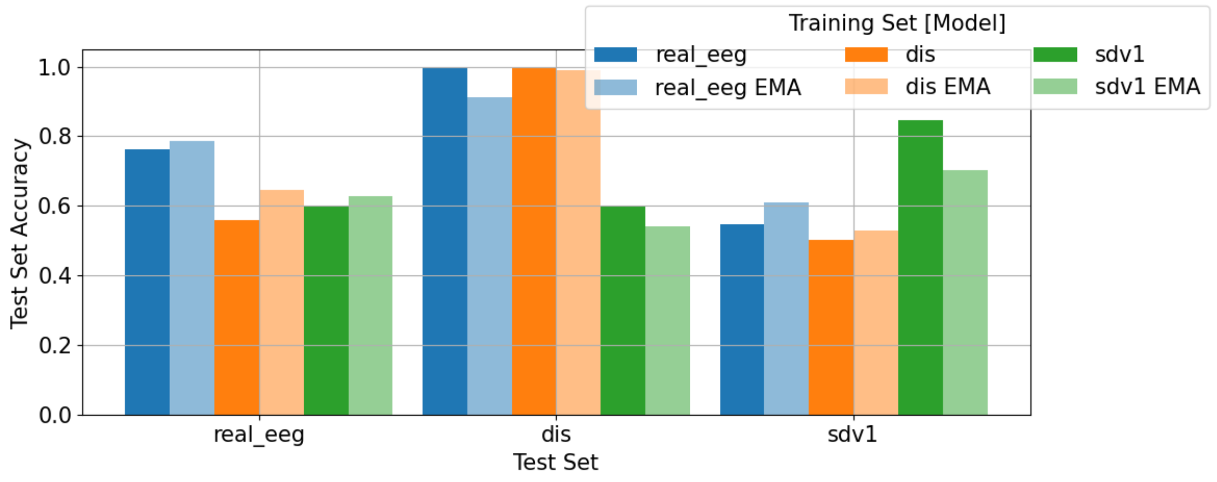


Figure 4.8: Classification accuracy for last-iteration and EMA models on trained on real or synthetic datasets, and tested on all.

Test Set Train Set	Real	DiS	SDv1
Real	0.76	0.99	0.55
Real EMA	0.79	0.91	0.61
DiS	0.56	1.00	0.50
DiS EMA	0.65	0.99	0.53
SDv1	0.60	0.60	0.85
SDv1 EMA	0.63	0.54	0.70

Table 4.3: Accuracy scores for gender classifiers trained and tested on the real and synthetic datasets.

As a baseline for comparison, the EMA classifier trained on real data achieves 79% accuracy. The precise value varies by a few percent depending on the seed of the run. The real EMA classifier achieves 91% accuracy on DiS data, suggesting that the DiS model learned to focus too heavily on features that correspond to gender, overfitting to the extremes of that part of the data distribution. Essentially, the DiS model can only produce outputs that are *for sure* male or *for sure* female. This conclusion is supported by the results from training an EMA model on DiS data. It achieves 99% accuracy on DiS data but only 65% accuracy on real data. The DiS data likely only represents the extremes of maleness and femaleness, so the model does not learn how to deal with spectrograms somewhere in the middle.

Interestingly, classifiers trained on SDv1 data only perform well on SDv1 data with 70% accuracy for the EMA model and 85% for the last-iteration model. They perform almost no better than chance on the other datasets. Classifiers trained on other datasets perform equally poorly or worse on SDv1 data. The SDv1 data seems to exist in its own reality. This contradicts the results from MMD scores where SDv1 outperformed DiS. This result implies that **ability to reproduce a distribution and ability to produce useful data for downstream tasks are not the same**. Neither is sufficient on its own — data that lets users train "accurate" models may be teaching models to look for features that do not generalize, and data that closely matches an overall distribution may be incorrectly labeled or have spurious features that a classifier picks up but do not represent real characteristics.

4.5 Conclusions

This preliminary exploration into generating synthetic time series datasets using repurposed image diffusion models demonstrates both promise and areas requiring further investigation. While we were able to successfully train classifiers on synthetic data that achieved strong performance on real data — with the DiS model's EMA classifier reaching 91% accuracy on real data compared to the 79% baseline — the results reveal important nuances in evaluating synthetic data quality. The apparent contradiction between distributional similarity metrics (MMD scores) and downstream task performance highlights that these metrics alone are insufficient for determining synthetic data utility. There is also work to be done to ensure data diversity within classes, not just across the whole dataset. Donoho's Frictionless Reproducibility phenomenon, an explanation for the steady flow of innovation in ML, requires competitive benchmarks as a target for practitioners [417]. MMD is a powerful pre-existing score in the world of generative modeling, but as it is insufficient on its own future work should involve designing a hybrid or all new metric for comparison across publications.

Future work will go to ensure proper isolation between training sets used for generative models and those used for evaluation. While care was taken in this study to separate subjects between training and test sets, another layer of segregation may be needed to guarantee that there is no cross-contamination of information between the data used to train generative models and the data used to evaluate downstream task performance. The small size of datasets in use combined with a large demand for training data for the diffusion models meant that splitting the data further would have severely impacted performance. This could be solved by adding additional datasets to the meta-dataset used in this work.

Several promising directions for future work emerge from this study. Rather than converting time series data to images and then performing a noisy reconstruction of the time domain, foundation models from the audio domain could enable direct synthesis of raw EEG and wireless signal data. This would eliminate potential information loss from the spectrogram conversion process and may better capture temporal dependencies. How best to address varying channel counts remains an open question. In this work, we include channels as concatenated columns in the spectrogram image and choose the spectrogram hop size to set the width of each channel's spectrogram appropriately. Audio foundation models do not have that extra dimension to play with, complicating the process of repurposing them for multi-channel or Massive-MIMO datasets. Hyperspectral imaging (HSI) faces a similar issue of mismatched channel depth when trying to use regular vision foundation models. Most papers on HSI foundation models simply use their own network trained from scratch to expect a certain number of channels beyond RGB [418]–[420]. One uses a 1D convolution adapter from hyperspectral channel depth to three channels and uses a standard vision backbone, but still trains their network from scratch [421]. A similar adapter or tricks with tokenization could let audio foundation models be used with high channel count data.

Extending conditioning mechanisms to handle higher-dimensional labels, such as seizure locations in EEG data or data bits in wireless signals, represents a crucial step toward practical applications. There are a growing number of techniques for higher dimensional conditioning of diffuser outputs. ControlNet [422] allows spatial conditioning to be applied, and Universal Guidance [423] extends the Classifier Guidance [424] technique to allow arbitrary guidance functions. CRS-Diff [425], again from the field of HSI, demonstrates simultaneous conditioning on text, images, and metadata using ControlNet techniques. It will take effort and experimentation to determine which of these methods is best for each application, but it will be an important component of future work.

An intermediate goal on the way to true foundation models for wireless or medical time series signals could be "pre-fine-tuning" repurposed foundation models on collections of public data and simulated data (not synthetic to avoid model collapse). This would shift the model towards the general distribution of EEG or wireless data, hopefully without destroying the interpretive power it learned from the original pre-training domain. Finally, the pre-fine-tuned model could be fine-tuned on a very small private dataset and expected to perform well. For example, the SDv1 model could first be trained on a broad collection of EEG recordings across research groups and intended uses before fine-tuning on a specific clinical dataset. Alternately, the model could be trained on wireless channel simulations using established modeling tools like MATLAB before specializing to a particular protocol or an urban versus rural environment. This approach might help the model learn general characteristics of the signal domain while still capturing dataset-specific features. It also mirrors the training process used for the most recent and powerful LLaMA 3 models [426]. LLaMA 3 pre-trained on data scraped from the web, then performed what the authors dub supervised fine-tuning using a mix of high-quality human-annotated datasets and synthetic data from the previous generation LLaMA 2 models. Finally, the model was optimized to produce outputs pleasing to users through Direct Preference Optimization [427], similar to optimizing for a single dataset.

The ultimate test of this approach to dataset anonymization will be its application to wireless signals, where the combination of high dimensionality, complex protocols, and privacy constraints creates unique challenges. Future work should investigate how well these techniques transfer to wireless domains such as Wi-Fi, cellular, and IoT protocols, potentially incorporating domain-specific knowledge into the model architecture or training process. Despite the open questions and challenges ahead, these initial results suggest that repurposing foundation models for synthetic dataset generation merits continued investigation as a potential solution to data scarcity challenges.

Chapter 5

Concluding Thoughts

This body of work has coincided with a fascinating and exciting transition in how researchers and engineers approach wireless communications. Prior to the release of the 5G NR standard in late 2017, just before I entered graduate school, machine learning and AI were mostly unaddressed in wireless protocols. The only places machine learning could commonly be employed in real systems were base stations for resource scheduling, a task that coincidentally was left open to device makers and network operators to implement as desired. Now, the 5G OpenRAN (Open Radio Access Network), a set of industry-wide standards for device makers and network operators to enable interoperability and flexibility, explicitly allows for components to be replaced or augmented by AI models up and down the communications stack [237]. The growth in papers exploring applications of ML in various components of the PHY and other layers reflects a wider change in attitude from desiring guaranteed behaviors specified in standards to exploring the possibilities for performance and adaptivity that learning components can provide.

In the conclusion of Chapter 2, I discuss what a world of "light standards" with learning components to replace unnecessary specifications might look like and how we might get there. Then, in Chapter 3, I present introductory work and a vision for how individual devices may observe and interact with the world around them to support light standards and a cognitive radio mode of operation at the individual level. Finally, in Chapter 4, I demonstrate early work towards solving the challenge of data scarcity in wireless communications and related domains. Large datasets will likely be required to support the deployment of learning radios in the real world, particularly in a hybrid edge-cloud model. Together, these chapters support the potential of machine learning to improve wireless communications and important paths of investigation to realize that potential.

One crucial consideration that has been unaddressed so far in this dissertation, and often in the body of literature as well, is the power and time cost of learning algorithms. Some work does compare the number of addition or multiplication operations to traditional algorithms but often ignores the difference in cost between integer or fixed-point operations and floating-point operations. Due to the early exploratory nature of this area, authors are often more concerned with whether some technique will work at all rather than whether it can fit onto a small chip in a mobile device. Even AI techniques designed to be run on resource-constrained edge devices are very costly compared to the operations performed by a dedicated Wi-Fi chip. Greater power draw reduces the battery life of mobile devices, a significant downside for many applications. Apart from power use, AI computations are often slower than the traditional algorithms they are intended to replace, particularly when implemented in hardware. Many standards, including Wi-Fi and cellular, have strict latency requirements for a device to interpret a received signal and respond appropriately. Any AI components will have to somehow meet these latency requirements to succeed. Much work has gone into techniques such as model quantization [428] and pruning [429] to reduce the storage, power, and latency requirements of models. However, there is no one-size-fits-all solution, and even models such as EfficientNet [430] that are designed to be maximally efficient require millions of parameters and on the order of a billion FLOPs for inference.

Most current AI chips are optimized for performing as many operations in parallel as possible to maximize the amount of data or size of model that can be trained or used for inference at one time. This results in server-scale accelerators that draw 100s of Watts while in operation — not a solution that can be packaged into a consumer phone. All high-end phones now contain accelerators for AI inference, used for photo post-processing and, increasingly, LLMs and generative models. For AI-enabled wireless devices to succeed, they will need both more efficient models and new hardware designed to exploit model architectures to reduce latency and power draw. Apple, Arm, Qualcomm, Nvidia, and Google all now produce AI accelerator chips for use in edge computing, either integrated in an SoC or standalone [431]–[435]. Some of these devices have their own SDKs (software development kits) while others can be used directly with tools such as LiteRT [436] or TinyML [437], [438]. Some work has already begun using these embedded AI tools to test ML techniques in edge devices for applications like MU-MIMO user grouping [439] and more [440]. Future work will have to address how to jointly design AI models and hardware for wireless communication.

Despite these challenges, the potential benefits of incorporating AI into wireless communication systems make pursuing this line of research worthwhile. The ability to adapt to changing environments, learn from experience, and optimize performance in ways not possible with traditional fixed algorithms could revolutionize how we build and deploy wireless networks. As research continues in model efficiency, hardware acceleration, and joint hardware-software design, we will likely see the first successful deployments of AI-enhanced wireless systems in areas with relaxed latency requirements or where the benefits clearly outweigh the power costs. These early successes can then inform the development of more efficient solutions suitable for resource-constrained mobile devices. The transition to learning-based wireless systems may be gradual, but the foundation laid by work like that presented in this dissertation suggests a promising path forward for realizing the full potential of AI in wireless communications.

Bibliography

[1] A. F. Molisch, “Wireless Communications: From Fundamentals to Beyond 5G,” in Wiley, 2023, ch. Equalizers.

[2] A. F. Molisch, *Wireless Communications: From Fundamentals to Beyond 5G*, 3rd ed. Chichester, UK: Wiley, 2023.

[3] V. Kawadia and P. Kumar, “A cautionary perspective on cross-layer design,” *IEEE Wireless Communications*, vol. 12, no. 1, pp. 3–11, 2005. DOI: `10.1109/MWC.2005.1404568`.

[4] H. Rutagemwa, L. Li, and P. Vigneron, “Cross-Layer Design and Performance Analysis of Tactical Radio Networks,” *IEEE Transactions on Vehicular Technology*, vol. 64, no. 1, pp. 339–355, 2015. DOI: `10.1109/TVT.2014.2320228`.

[5] Q. M. Salih, M. A. Rahman, M. Z. A. Bhuiyan, and Z. R. M. Azmi, “The Art of Using Cross-Layer Design in Cognitive Radio Networks,” in *Security, Privacy, and Anonymity in Computation, Communication, and Storage*, G. Wang, M. Atiquzzaman, Z. Yan, and K.-K. R. Choo, Eds., Cham: Springer International Publishing, 2017, pp. 544–556.

[6] B. Sklar, *Digital Communications: Fundamentals and Applications*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1988, ISBN: 0-13-211939-0.

[7] C. Rudin and J. Radin, “Why Are We Using Black Box Models in AI When We Don’t Need To? A Lesson From an Explainable AI Competition,” *Harvard Data Science Review*, vol. 1, no. 2, Nov. 2019, https://hdsr.mitpress.mit.edu/pub/f9kuryi8.

[8] K. Rasheed, A. Qayyum, M. Ghaly, A. Al-Fuqaha, A. Razi, and J. Qadir, “Explainable, trustworthy, and ethical machine learning for healthcare: A survey,” *Computers in Biology and Medicine*, vol. 149, p. 106 043, 2022, ISSN: 0010-4825. DOI: `10.1016/j.compbiomed.2022.106043`.

[9] E. Toreini, M. Aitken, K. Coopamootoo, K. Elliott, C. G. Zelaya, and A. van Moorsel, “The relationship between trust in AI and trustworthy machine learning technologies,” in *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, ser. FAT* ’20, Barcelona, Spain: Association for Computing Machinery, 2020, pp. 272–283, ISBN: 9781450369367. DOI: `10.1145/3351095.3372834`.

[10] R. Shwartz-Ziv and N. Tishby, "Opening the Black Box of Deep Neural Networks via Information," *CoRR*, vol. abs/1703.00810, 2017. arXiv: `1703.00810`.

[11] F. Berkenkamp, M. Turchetta, A. Schoellig, and A. Krause, "Safe Model-based Reinforcement Learning with Stability Guarantees," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, *et al.*, Eds., vol. 30, Curran Associates, Inc., 2017. [Online]. Available: `https://proceedings.neurips.cc/paper_files/paper/2017/file/766ebcd59621e305170616ba3d3dac32-Paper.pdf`.

[12] S. Gu, L. Yang, Y. Du, *et al.*, "A Review of Safe Reinforcement Learning: Methods, Theories and Applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[13] V. Subramanian, R. Arya, and A. Sahai, "Generalization for multiclass classification with overparameterized linear models," in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., vol. 35, Curran Associates, Inc., 2022, pp. 23 479–23 494. [Online]. Available: `https://proceedings.neurips.cc/paper_files/paper/2022/file/94b472a1842cd7c56dcb125fb2765fbd-Paper-Conference.pdf`.

[14] V. Muthukumar, A. Narang, V. Subramanian, M. Belkin, D. Hsu, and A. Sahai, "Classification vs regression in overparameterized regimes: Does the loss function matter?" *Journal of Machine Learning Research*, vol. 22, no. 222, pp. 1–69, 2021. [Online]. Available: `http://jmlr.org/papers/v22/20-603.html`.

[15] V. Muthukumar, K. Vodrahalli, V. Subramanian, and A. Sahai, "Harmless Interpolation of Noisy Data in Regression," *IEEE Journal on Selected Areas in Information Theory*, vol. 1, no. 1, pp. 67–83, 2020. DOI: `10.1109/JSAIT.2020.2984716`.

[16] open-sdr, *Openwifi*, version 239ba79, 2024. [Online]. Available: `https://github.com/open-sdr/openwifi`.

[17] J. Mitola and G. Q. Maguire, "Cognitive radio: making software radios more personal, volume = 6, year = 1999," *IEEE Personal Communications*, pp. 13–18, 4, ISSN: 10709916. DOI: `10.1109/98.788210`.

[18] G. Tesauro, N. Jong, R. Das, and M. Bennani, "A Hybrid Reinforcement Learning Approach to Autonomic Resource Allocation," in *2006 IEEE International Conference on Autonomic Computing*, 2006, pp. 65–73. DOI: `10.1109/ICAC.2006.1662383`.

[19] T. Han and N. Ansari, "On greening cellular networks via multicell cooperation," *IEEE Wireless Communications*, vol. 20, pp. 82–89, 1 2013, ISSN: 15361284. DOI: `10.1109/MWC.2013.6472203`.

[20] Y. Gao, J. Chen, Z. Liu, B. Zhang, Y. Ke, and R. Liu, "Machine Learning based Energy Saving Scheme in Wireless Access Networks," *2020 International Wireless Communications and Mobile Computing, IWCMC 2020*, pp. 1573–1578, Jun. 2020. DOI: `10.1109/IWCMC48107.2020.9148536`.

[21] Y. Shao, Y. Cai, T. Wang, *et al.*, "Learning-based Autonomous Channel Access in the Presence of Hidden Terminals," Jul. 2022. DOI: `10.48550/arxiv.2207.03605`.

[22] J. Gao, X. Yi, C. Zhong, X. Chen, and Z. Zhang, "Deep Learning for Spectrum Sensing," *IEEE Wireless Communications Letters*, vol. 8, pp. 1727–1730, 6 Dec. 2019, ISSN: 21622345. DOI: `10.1109/LWC.2019.2939314`.

[23] T. J. O'Shea, K. Karra, and T. C. Clancy, "Learning to Communicate: Channel Autoencoders, Domain Specific Regularizers, and Attention," *CoRR*, vol. abs/1608.06409, 2016. arXiv: `1608.06409`.

[24] S. Dorner, S. Cammerer, J. Hoydis, and S. T. Brink, "Deep Learning-Based Communication Over the Air," *IEEE Journal on Selected Topics in Signal Processing*, vol. 12, pp. 132–143, 1 Jul. 2017, ISSN: 19324553. DOI: `10.1109/jstsp.2017.2784180`.

[25] B. Karanov, M. Chagnon, F. Thouin, *et al.*, "End-to-End Deep Learning of Optical Fiber Communications," *Journal of Lightwave Technology*, vol. 36, pp. 4843–4855, 20 Oct. 2018, ISSN: 07338724. DOI: `10.1109/JLT.2018.2865109`.

[26] H. Ye, G. Y. Li, B. H. F. Juang, and K. Sivanesan, "Channel Agnostic End-to-End Learning Based Communication Systems with Conditional GAN," *2018 IEEE Globecom Workshops, GC Wkshps 2018 - Proceedings*, Feb. 2019. DOI: `10.1109/GLOCOMW.2018.8644250`.

[27] S. Li, C. Häger, N. Garcia, and H. Wymeersch, "Achievable Information Rates for Nonlinear Fiber Communication via End-to-end Autoencoder Learning," *European Conference on Optical Communication, ECOC*, vol. 2018-September, Nov. 2018. DOI: `10.1109/ECOC.2018.8535456`.

[28] A. Gupta and M. Sellathurai, "A stacked-autoencoder based end-to-end learning framework for decode-and-forward relay networks," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2020-May, pp. 5245–5249, May 2020, ISSN: 15206149. DOI: `10.1109/ICASSP40776.2020.9054512`.

[29] Q. Zhang, Z. Wang, S. Duan, *et al.*, "An Improved End-to-End Autoencoder Based on Reinforcement Learning by Using Decision Tree for Optical Transceivers," *Micromachines*, vol. 13, 1 Jan. 2022, ISSN: 2072666X. DOI: `10.3390/MI13010031`.

[30] B. Zhang and N. V. Huynh, "Deep Deterministic Policy Gradient for End-to-End Communication Systems without Prior Channel Knowledge," May 2023. arXiv: `2305.07448`.

[31] F. A. Aoudia and J. Hoydis, "Model-Free Training of End-to-End Communication Systems," *IEEE Journal on Selected Areas in Communications*, vol. 37, pp. 2503–2516, 11 Nov. 2019, ISSN: 15580008. DOI: `10.1109/JSAC.2019.2933891`.

[32] S. Park, O. Simeone, and J. Kang, "Meta-Learning to Communicate: Fast End-to-End Training for Fading Channels," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2020-May, pp. 5075–5079, May 2020, ISSN: 15206149. DOI: 10.1109/ICASSP40776.2020.9053252.

[33] M. E. Morocho-Cayamcela and W. Lim, "Accelerating wireless channel autoencoders for short coherence-time communications," *Journal of Communications and Networks*, vol. 22, pp. 215–222, 3 Jun. 2020, ISSN: 19765541. DOI: 10.1109/JCN.2020.000011.

[34] N. Soltani, H. Cheng, M. Belgiovine, *et al.*, "Neural Network-based OFDM Receiver for Resource Constrained IoT Devices," May 2022. DOI: 10.48550/arxiv.2205.06159. [Online]. Available: https://arxiv.org/abs/2205.06159v1.

[35] F. A. Aoudia and J. Hoydis, "End-to-End Learning for OFDM: From Neural Receivers to Pilotless Communication," *IEEE Transactions on Wireless Communications*, vol. 21, pp. 1049–1063, 2 Feb. 2022, ISSN: 15582248. DOI: 10.1109/TWC.2021.3101364.

[36] J. Du, L. Zhang, P. Tian, and X. Zhou, "Fast self-learning modulation recognition method for smart underwater optical communication systems," *Optics Express, Vol. 28, Issue 25, pp. 38223-38240*, vol. 28, pp. 38 223–38 240, 25 Dec. 2020, ISSN: 1094-4087. DOI: 10.1364/OE.412371.

[37] Y. Lu, P. Cheng, Z. Chen, W. H. Mow, and Y. Li, "A Learning Approach to Cooperative Communication System Design," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2020-May, pp. 5240–5244, May 2020, ISSN: 15206149. DOI: 10.1109/ICASSP40776.2020.9054093.

[38] H. Wu, Y. Shao, K. Mikolajczyk, and D. Gündüz, "Channel-Adaptive Wireless Image Transmission with OFDM," May 2022. DOI: 10.48550/arxiv.2205.02417.

[39] B. Azari, H. Cheng, N. Soltani, *et al.*, "Automated deep learning-based wide-band receiver," *Computer Networks*, vol. 218, p. 109 367, Dec. 2022, ISSN: 1389-1286. DOI: 10.1016/J.COMNET.2022.109367.

[40] C. P. Davey, I. Shakeel, R. C. Deo, and S. Salcedo-Sanz, "Channel-Agnostic Training of Transmitter and Receiver for Wireless Communications," *Sensors 2023, Vol. 23, Page 9848*, vol. 23, p. 9848, 24 Dec. 2023, ISSN: 1424-8220. DOI: 10.3390/S23249848.

[41] Y. An, S. Wang, L. Zhao, Z. Ji, and I. Ganchev, "A Learning-Based End-to-End Wireless Communication System Utilizing a Deep Neural Network Channel Module," *IEEE Access*, vol. 11, pp. 17 441–17 453, 2023, ISSN: 21693536. DOI: 10.1109/ACCESS.2023.3245330.

[42] D. Yan, M. Jia, Q. Guo, and X. Gu, "Unknown Channel End-to-End Learning of Communication System With Residual DCGAN," *IEEE International Symposium on Broadband Multimedia Systems and Broadcasting, BMSB*, vol. 2023-June, 2023, ISSN: 21555052. DOI: 10.1109/BMSB58369.2023.10211449.

[43] S. K. Jayakumar, N. M. Benoy, V. M. Mohan, A. G. Nair, and P. E. Ameenudeen, "Deep Learning Based Digital Wireless Communication System," *2023 International Conference on Control, Communication and Computing, ICCC 2023*, 2023. DOI: 10.1109/ICCC57789.2023.10165139.

[44] H. Ye, G. Y. Li, and B. H. F. Juang, "Bilinear Convolutional Auto-encoder based Pilot-free End-to-end Communication Systems," *IEEE International Conference on Communications*, vol. 2020-June, Jun. 2020, ISSN: 15503607. DOI: 10.1109/ICC40277.2020.9149030.

[45] G. Gibson, S. Siu, and C. Cowen, "Multilayer perceptron structures applied to adaptive equalisers for data communications," in *International Conference on Acoustics, Speech, and Signal Processing,*, 1989, 1183–1186 vol.2. DOI: 10.1109/ICASSP.1989.266645.

[46] G. Gibson, S. Siu, and C. Cowan, "APPLICATION OF MULTILAYER PERCEPTRONS AS ADAPTIVE CHANNEL EQUALISERS," in *Adaptive Systems in Control and Signal Processing 1989*, ser. IFAC Symposia Series, M. JOHNSON, M. GRIMBLE, D. OWENS, and T. DURRANI, Eds., Oxford: Pergamon, 1990, pp. 573–578, ISBN: 978-0-08-035727-0. DOI: 10.1016/B978-0-08-035727-0.50098-6.

[47] Y. Li, M. Chen, Y. Yang, M.-T. Zhou, and C. Wang, "Convolutional recurrent neural network-based channel equalization: An experimental study," in *2017 23rd Asia-Pacific Conference on Communications (APCC)*, 2017, pp. 1–6. DOI: 10.23919/APCC.2017.8304090.

[48] T. Raviv, S. Park, N. Shlezinger, O. Simeone, Y. C. Eldar, and J. Kang, "Meta-ViterbiNet: Online Meta-Learned Viterbi Equalization for Non-Stationary Channels," *2021 IEEE International Conference on Communications Workshops, ICC Workshops 2021 - Proceedings*, Jun. 2021. DOI: 10.1109/ICCWORKSHOPS50388.2021.9473693.

[49] K. M. Cohen, S. Park, O. Simeone, and S. Shamai (Shitz), "Towards Reliable and Efficient AI for 6G: Bayesian Active Meta-Learning for Few Pilot Demodulation and Equalization," Aug. 2021. DOI: 10.48550/arxiv.2108.00785.

[50] K. M. Cohen, S. Park, O. Simeone, and S. Shamai, "Learning to Learn to Demodulate with Uncertainty Quantification via Bayesian Meta-Learning," *WSA 2021; 25th International ITG Workshop on Smart Antennas*, Nov. 2021.

[51] P. J. Freire, J. E. Prilepsky, Y. Osadchuk, S. K. Turitsyn, and V. Aref, "Deep Neural Network-aided Soft-Demapping in Optical Coherent Systems: Regression versus Classification," Sep. 2021. DOI: 10.48550/arxiv.2109.13843.

[52] P. J. Freire, A. Napoli, B. Spinnler, N. Costa, S. K. Turitsyn, and J. E. Prilepsky, "Neural Networks-Based Equalizers for Coherent Optical Transmission: Caveats and Pitfalls," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 28, pp. 1–23, 4 May 2022, ISSN: 1077-260X. DOI: 10.1109/JSTQE.2022.3174268.

[53] A. Caciularu and D. Burshtein, “Blind Channel Equalization Using Variational Autoencoders,” in *2018 IEEE International Conference on Communications Workshops (ICC Workshops)*, IEEE, May 2018, pp. 1–6, ISBN: 978-1-5386-4328-0. DOI: 10.1109/ICCW.2018.8403666.

[54] B. Zhu, J. Wang, L. He, and J. Song, “Joint Transceiver Optimization for Wireless Communication PHY with Convolutional Neural Network,” Aug. 2018. [Online]. Available: http://arxiv.org/abs/1808.03242.

[55] Y. Li, F. Sun, W. Zu, *et al.*, “MAFENN: Multi-Agent Feedback Enabled Neural Network for Wireless Channel Equalization,” *2021 IEEE Global Communications Conference, GLOBECOM 2021 - Proceedings*, 2021. DOI: 10.1109/GLOBECOM46510.2021.9685522.

[56] W. Xu, Z. Zhong, Y. Beery, X. You, and C. Zhang, “Joint Neural Network Equalizer and Decoder,” *Proceedings of the International Symposium on Wireless Communication Systems*, vol. 2018-August, Jul. 2018, ISSN: 21540225. DOI: 10.48550/arxiv.1807.02040. [Online]. Available: https://arxiv.org/abs/1807.02040v1.

[57] A. Klautau, N. Gonzalez-Prelcic, A. Mezghani, and R. W. Heath, “Detection and Channel Equalization with Deep Learning for Low Resolution MIMO Systems,” *Conference Record - Asilomar Conference on Signals, Systems and Computers*, vol. 2018-October, pp. 1836–1840, Feb. 2019, ISSN: 10586393. DOI: 10.1109/ACSSC.2018.8645551.

[58] R. Mei, Z. Wang, and W. Hu, “Robust Blind Equalization Algorithm Using Convolutional Neural Network,” *IEEE Signal Processing Letters*, vol. 29, pp. 1569–1573, 2022, ISSN: 1070-9908. DOI: 10.1109/LSP.2022.3189319.

[59] P. K. Mohapatra, S. K. Rout, S. K. Bisoy, and M. Sain, “Training Strategy of Fuzzy-Firefly Based ANN in Non-Linear Channel Equalization,” *IEEE Access*, vol. 10, pp. 51 229–51 241, 2022, ISSN: 21693536. DOI: 10.1109/ACCESS.2022.3174369.

[60] W. Donglin and W. Dongming, “Generalized derivation of neural network constant modulus algorithm for blind equalization,” *Proceedings - 5th International Conference on Wireless Communications, Networking and Mobile Computing, WiCOM 2009*, 2009. DOI: 10.1109/WICOM.2009.5302643.

[61] C. Farhati, S. Fki, A. A. E. Bey, and F. Abdelkefi, “Blind channel equalization based on Complex-valued neural network and probability density fitting,” *2022 International Wireless Communications and Mobile Computing (IWCMC)*, pp. 773–777, May 2022. DOI: 10.1109/IWCMC55113.2022.9824100.

[62] D. F. Carrera, C. Vargas-Rosales, N. M. Yungaicela-Naula, and L. Azpilicueta, “Comparative Study of Artificial Neural Network Based Channel Equalization Methods for mmWave Communications,” *IEEE Access*, vol. 9, pp. 41 678–41 687, 2021, ISSN: 21693536. DOI: 10.1109/ACCESS.2021.3065337.

[63] H. Ye, G. Y. Li, and B. H. Juang, "Power of Deep Learning for Channel Estimation and Signal Detection in OFDM Systems," *IEEE Wireless Communications Letters*, vol. 7, pp. 114–117, 1 Feb. 2018, ISSN: 21622345. DOI: 10.1109/LWC.2017.2757490.

[64] S. Hanna, C. Dick, and D. Cabric, "Signal Processing-Based Deep Learning for Blind Symbol Decoding and Modulation Classification," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 1, pp. 82–96, 2022. DOI: 10.1109/JSAC.2021.3126088.

[65] C. Huang, G. C. Alexandropoulos, A. Zappone, C. Yuen, and M. Debbah, "Deep Learning for UL/DL Channel Calibration in Generic Massive MIMO Systems," *IEEE International Conference on Communications*, vol. 2019-May, May 2019, ISSN: 15503607. DOI: 10.1109/ICC.2019.8761962. [Online]. Available: https://arxiv.org/abs/1903.02875v2.

[66] R. T. Jones, T. A. Eriksson, M. P. Yankov, and D. Zibar, "Deep Learning of Geometric Constellation Shaping Including Fiber Nonlinearities," *European Conference on Optical Communication, ECOC*, vol. 2018-September, Nov. 2018. DOI: 10.1109/ECOC.2018.8535453.

[67] N. Shlezinger, Y. C. Eldar, N. Farsad, and A. J. Goldsmith, "ViterbiNet: Symbol Detection Using a Deep Learning Based Viterbi Algorithm," *IEEE Workshop on Signal Processing Advances in Wireless Communications, SPAWC*, vol. 2019-July, Jul. 2019. DOI: 10.1109/SPAWC.2019.8815457.

[68] N. Farsad, N. Shlezinger, A. J. Goldsmith, and Y. C. Eldar, "Data-Driven Symbol Detection Via Model-Based Machine Learning," *IEEE Workshop on Statistical Signal Processing Proceedings*, vol. 2021-July, pp. 571–575, Jul. 2021. DOI: 10.1109/SSP49050.2021.9513859.

[69] Y. Qin and J. G. Zhu, "Deep Neural Network: Data Detection Channel for Hard Disk Drives by Learning," *IEEE Transactions on Magnetics*, vol. 56, 2 Feb. 2020, ISSN: 19410069. DOI: 10.1109/TMAG.2019.2942051.

[70] N. Shlezinger, R. Fu, and Y. C. Eldar, "DeepSIC: Deep Soft Interference Cancellation for Multiuser MIMO Detection," *IEEE Transactions on Wireless Communications*, vol. 20, pp. 1349–1362, 2 Feb. 2021, ISSN: 15582248. DOI: 10.1109/TWC.2020.3032663.

[71] X. Gao, S. Jin, C. K. Wen, and G. Y. Li, "ComNet: Combination of Deep Learning and Expert Knowledge in OFDM Receivers," *IEEE Communications Letters*, vol. 22, pp. 2627–2630, 12 Dec. 2018, ISSN: 15582558. DOI: 10.1109/LCOMM.2018.2877965.

[72] Y. Jiang, H. Kim, H. Asnani, S. Kannan, S. Oh, and P. Viswanath, "Joint Channel Coding and Modulation via Deep Learning," *IEEE Workshop on Signal Processing Advances in Wireless Communications, SPAWC*, vol. 2020-May, May 2020. DOI: 10.1109/SPAWC48557.2020.9153885.

[73] C. de Vrieze, S. Barratt, D. Tsai, and A. Sahai, "Cooperative Multi-Agent Reinforcement Learning for Low-Level Wireless Communication," 2018. arXiv preprint: `1801.04541`.

[74] A. Sahai, J. Sanz, V. Subramanian, C. Tran, and K. Vodrahalli, "Blind Interactive Learning of Modulation Schemes: Multi-Agent Cooperation Without Co-Design," *IEEE Access*, vol. 8, pp. 63 790–63 820, 2020. DOI: `10.1109/ACCESS.2020.2984218`.

[75] M. Stark, F. A. Aoudia, and J. Hoydis, "Joint learning of geometric and probabilistic constellation shaping," *2019 IEEE Globecom Workshops, GC Wkshps 2019 - Proceedings*, Dec. 2019. DOI: `10.1109/GCWKSHPS45667.2019.9024567`.

[76] V. Monga, Y. Li, and Y. C. Eldar, "Algorithm Unrolling: Interpretable, Efficient Deep Learning for Signal and Image Processing," *IEEE Signal Processing Magazine*, vol. 38, pp. 18–44, 2 Mar. 2021, ISSN: 15580792. DOI: `10.1109/MSP.2020.3016905`.

[77] O. S. Kayhan and J. C. v. Gemert, "On Translation Invariance in CNNs: Convolutional Layers Can Exploit Absolute Spatial Location," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2020.

[78] E. Kauderer-Abrams, "Quantifying Translation-Invariance in Convolutional Neural Networks," *CoRR*, vol. abs/1801.01450, 2018. arXiv: `1801.01450`.

[79] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention Is All You Need," *CoRR*, vol. abs/1706.03762, 2017. arXiv: `1706.03762`.

[80] Z. Niu, G. Zhong, and H. Yu, "A review on the attention mechanism of deep learning," *Neurocomputing*, vol. 452, pp. 48–62, 2021, ISSN: 0925-2312. DOI: `10.1016/j.neucom.2021.03.091`.

[81] A. F. Molisch, "Wireless Communications: From Fundamentals to Beyond 5G," in Wiley, 2023, ch. Channel Coding and Information Theory.

[82] Y. Jiang, S. Kannan, H. Kim, S. Oh, H. Asnani, and P. Viswanath, "DEEPTURBO: Deep Turbo Decoder," *IEEE Workshop on Signal Processing Advances in Wireless Communications, SPAWC*, vol. 2019-July, Jul. 2019. DOI: `10.1109/SPAWC.2019.8815400`.

[83] Y. Jiang, H. Kim, H. Asnani, S. Kannan, S. Oh, and P. Viswanath, "LEARN Codes: Inventing Low-Latency Codes via Recurrent Neural Networks," *IEEE International Conference on Communications*, vol. 2019-May, May 2019, ISSN: 15503607. DOI: `10.1109/ICC.2019.8761286`.

[84] Y. Jiang, H. Kim, H. Asnani, and S. Kannan, "MIND: Model Independent Neural Decoder," *IEEE Workshop on Signal Processing Advances in Wireless Communications, SPAWC*, vol. 2019-July, Jul. 2019. DOI: `10.1109/SPAWC.2019.8815537`.

[85] H. Kim, Y. Jiang, S. Kannan, S. Oh, and P. Viswanath, "DeepCode: Feedback codes via deep learning," *Advances in Neural Information Processing Systems*, vol. 2018-December, pp. 9436–9446, Jul. 2018, ISSN: 10495258. DOI: `10.1109/JSAIT.2020.2986752`.

[86] E. Arikan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *CoRR*, vol. abs/0807.3917, 2008. arXiv: `0807.3917`. [Online]. Available: `http://arxiv.org/abs/0807.3917`.

[87] S. Cammerer, T. Gruber, J. Hoydis, and S. T. Brink, "Scaling Deep Learning-Based Decoding of Polar Codes via Partitioning," in *2017 IEEE Global Communications Conference, GLOBECOM 2017 - Proceedings*, vol. 2018-January, Institute of Electrical and Electronics Engineers Inc., Jul. 2017, pp. 1–6, ISBN: 9781509050192. DOI: `10.1109/GLOCOM.2017.8254811`.

[88] A. V. Makkuva, X. Liu, M. V. Jamali, H. Mahdavifar, S. Oh, and P. Viswanath, "KO codes: inventing nonlinear encoding and decoding for reliable wireless communication via deep-learning," p. 139, 2021. arXiv: `2108.12920`.

[89] M. V. Jamali, X. Liu, A. V. Makkuva, H. Mahdavifar, S. Oh, and P. Viswanath, "Reed-Muller Subcodes: Machine Learning-Aided Design of Efficient Soft Recursive Decoding," *IEEE International Symposium on Information Theory - Proceedings*, vol. 2021-July, pp. 1088–1093, Jul. 2021, ISSN: 21578095. DOI: `10.1109/ISIT45174.2021.9517885`.

[90] Y. Shao, E. Ozfatura, A. G. Perotti, B. M. Popović, and D. Gündüz, "AttentionCode: Ultra-Reliable Feedback Codes for Short-Packet Communications," *IEEE Transactions on Communications*, vol. 71, no. 8, pp. 4437–4452, Aug. 2023, ISSN: 1558-0857. DOI: `10.1109/tcomm.2023.3280563`.

[91] W. Xu, Z. Wu, Y. L. Ueng, X. You, and C. Zhang, "Improved polar decoder based on deep learning," *IEEE Workshop on Signal Processing Systems, SiPS: Design and Implementation*, vol. 2017-October, Nov. 2017, ISSN: 15206130. DOI: `10.1109/SIPS.2017.8109997`.

[92] X. Song, Z. Zhang, J. Wang, and K. Qin, "A Graph-Neural-Network Decoder with MLP-based Processing Cells for Polar Codes," *2019 11th International Conference on Wireless Communications and Signal Processing, WCSP 2019*, Oct. 2019. DOI: `10.1109/WCSP.2019.8928139`.

[93] C. F. Teng and Y. L. Chen, "Syndrome-Enabled Unsupervised Learning for Neural Network-Based Polar Decoder and Jointly Optimized Blind Equalizer," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 10, pp. 177–188, 2 Jun. 2020, ISSN: 21563365. DOI: `10.1109/JETCAS.2020.2992593`.

[94] E. Nachmani and L. Wolf, "A gated hypernet decoder for polar codes," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2020-May, pp. 5210–5214, May 2020, ISSN: 15206149. DOI: 10.1109/ICASSP40776.2020.9054607.

[95] E. Ozfatura, Y. Shao, A. Perotti, B. Popovic, and D. Gunduz, "All you need is feedback: Communication with block attention feedback codes," Jun. 2022. DOI: 10.48550/arxiv.2206.09457.

[96] J. Yang, Q. Du, and Y. Jiang, "Neural Network-Assisted Receiver Design via Learning Trellis Diagram Online," *IEEE Transactions on Communications*, pp. 1–1, Nov. 2022, ISSN: 0090-6778. DOI: 10.1109/TCOMM.2022.3219141.

[97] R. Mitra and G. Kaddoum, "Random Fourier Feature-Based Deep Learning for Wireless Communications," *IEEE Transactions on Cognitive Communications and Networking*, vol. 8, pp. 468–479, 2 Jun. 2022, ISSN: 23327731. DOI: 10.1109/TCCN.2022.3164898.

[98] A. Caciularu, N. Raviv, T. Raviv, J. Goldberger, and Y. Be'Ery, "Perm2vec: Attentive Graph Permutation Selection for Decoding of Error Correction Codes," *IEEE Journal on Selected Areas in Communications*, vol. 39, pp. 79–88, 1 Jan. 2021, ISSN: 15580008. DOI: 10.1109/JSAC.2020.3036951.

[99] T.-Y. Tung, D. B. Kurka, M. Jankowski, and D. Gündüz, "DeepJSCC-Q: Channel Input Constrained Deep Joint Source-Channel Coding," Nov. 2021. [Online]. Available: https://arxiv.org/abs/2111.13042v1.

[100] X. A. Wang and S. B. Wicker, "An artificial neural net viterbi decoder," *IEEE Transactions on Communications*, vol. 44, pp. 165–171, 2 1996, ISSN: 00906778. DOI: 10.1109/26.486609.

[101] E. Nachmani, Y. Be'Ery, and D. Burshtein, "Learning to decode linear codes using deep learning," *54th Annual Allerton Conference on Communication, Control, and Computing, Allerton 2016*, pp. 341–346, Feb. 2017. DOI: 10.1109/ALLERTON.2016.7852251.

[102] D. Tandler, S. Dorner, S. Cammerer, and S. T. Brink, "On Recurrent Neural Networks for Sequence-based Processing in Communications," *Conference Record - Asilomar Conference on Signals, Systems and Computers*, vol. 2019-November, pp. 537–543, Nov. 2019, ISSN: 10586393. DOI: 10.1109/IEEECONF44664.2019.9048728.

[103] K. Ullrich, F. Viola, and D. J. Rezende, "Neural Communication Systems with Bandwidth-limited Channel," Mar. 2020. DOI: 10.48550/arxiv.2003.13367.

[104] M. H. Sazli and C. Işik, "Neural network implementation of the BCJR algorithm," *Digital Signal Processing: A Review Journal*, vol. 17, pp. 353–359, 1 2007, ISSN: 10512004. DOI: 10.1016/J.DSP.2005.12.002.

[105] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate," *IEEE Transactions on Information Theory*, vol. 20, pp. 284–287, 2 1974, ISSN: 15579654. DOI: 10.1109/TIT.1974.1055186.

[106] T. Gruber, S. Cammerer, J. Hoydis, and S. T. Brink, "On deep learning-based channel decoding," *2017 51st Annual Conference on Information Sciences and Systems, CISS 2017*, May 2017. DOI: 10.1109/CISS.2017.7926071.

[107] A. Lapidoth, "Nearest neighbor decoding for additive non-Gaussian noise channels," *IEEE Transactions on Information Theory*, vol. 42, pp. 1520–1529, 5 1996, ISSN: 00189448. DOI: 10.1109/18.532892.

[108] C. Finn, P. Abbeel, and S. Levine, "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks," *CoRR*, vol. abs/1703.03400, 2017. arXiv: 1703.03400.

[109] L. Li, J. Xu, L. Zheng, and L. Liu, "Real-Time Machine Learning for Multi-User Massive MIMO: Symbol Detection Using Multi-Mode StructNet," *IEEE Transactions on Wireless Communications*, vol. 22, pp. 9172–9186, 12 Dec. 2023, ISSN: 15582248. DOI: 10.1109/TWC.2023.3268945.

[110] C. K. Wen, W. T. Shih, and S. Jin, "Deep learning for massive MIMO CSI feedback," *IEEE Wireless Communications Letters*, vol. 7, pp. 748–751, 5 Oct. 2018, ISSN: 21622345. DOI: 10.1109/LWC.2018.2818160.

[111] M. K. Shehzad, L. Rose, S. Wesemann, M. Assaad, and S. A. Hassan, "Design of an Efficient CSI Feedback Mechanism in Massive MIMO Systems: A Machine Learning Approach Using Empirical Data," *IEEE Transactions on Cognitive Communications and Networking*, 2024, ISSN: 23327731. DOI: 10.1109/TCCN.2024.3435915.

[112] Y. Yang, S. Zhang, F. Gao, J. Ma, and O. A. Dobre, "Graph Neural Network-Based Channel Tracking for Massive MIMO Networks," *IEEE Communications Letters*, vol. 24, pp. 1747–1751, 8 Aug. 2020, ISSN: 15582558. DOI: 10.1109/LCOMM.2020.2990487.

[113] Y. Yuan, G. Zheng, K. K. Wong, B. Ottersten, and Z. Q. Luo, "Transfer Learning and Meta Learning-Based Fast Downlink Beamforming Adaptation," *IEEE Transactions on Wireless Communications*, vol. 20, pp. 1742–1755, 3 Mar. 2021, ISSN: 15582248. DOI: 10.1109/TWC.2020.3035843.

[114] J. Gao, M. Hu, C. Zhong, G. Y. Li, and Z. Zhang, "An Attention-Aided Deep Learning Framework for Massive MIMO Channel Estimation," *IEEE Transactions on Wireless Communications*, vol. 21, pp. 1823–1835, 3 Mar. 2022, ISSN: 15582248. DOI: 10.1109/TWC.2021.3107452.

[115] M. Kokshoorn, H. Chen, P. Wang, Y. Li, and B. Vucetic, "Millimeter Wave MIMO Channel Estimation Using Overlapped Beam Patterns and Rate Adaptation," *IEEE Transactions on Signal Processing*, vol. 65, pp. 601–616, 3 Feb. 2017, ISSN: 1053587X. DOI: 10.1109/TSP.2016.2614488.

[116] T. J. O'Shea, T. Erpek, and T. C. Clancy, "Deep Learning Based MIMO Communications," Jul. 2017. DOI: `10.48550/arxiv.1707.07980`.

[117] A. M. Elbir and K. V. Mishra, "Deep learning design for joint antenna selection and hybrid beamforming in massive MIMO," *2019 IEEE International Symposium on Antennas and Propagation and USNC-URSI Radio Science Meeting, APSURSI 2019 - Proceedings*, pp. 1585–1586, Jul. 2019. DOI: `10.1109/APUSNCURSINRSM.2019.8888753`.

[118] X. Ma and Z. Gao, "Data-Driven Deep Learning to Design Pilot and Channel Estimator for Massive MIMO," *IEEE Transactions on Vehicular Technology*, vol. 69, pp. 5677–5682, 5 May 2020, ISSN: 19399359. DOI: `10.1109/TVT.2020.2980905`.

[119] F. H. C. Neto, D. C. Araújo, and T. F. Maciel, "Hybrid beamforming design based on unsupervised machine learning for millimeter wave systems," *International Journal of Communication Systems*, vol. 33, e4276, 5 Mar. 2020, ISSN: 1099-1131. DOI: `10.1002/DAC.4276`.

[120] Y. C. Lin, T. S. Lee, and Z. Ding, "Deep Learning for Partial MIMO CSI Feedback by Exploiting Channel Temporal Correlation," *Conference Record - Asilomar Conference on Signals, Systems and Computers*, vol. 2021-October, pp. 345–350, Jan. 2021, ISSN: 10586393. DOI: `10.1109/IEEECONF53345.2021.9723211`.

[121] M. Belgiovine, K. Sankhe, C. Bocanegra, D. Roy, and K. R. Chowdhury, "Deep Learning at the Edge for Channel Estimation in Beyond-5G Massive MIMO," *IEEE Wireless Communications*, vol. 28, pp. 19–25, 2 Apr. 2021, ISSN: 15580687. DOI: `10.1109/MWC.001.2000322`.

[122] *National Spectrum Strategy*. National Telecommunications and Information Administration, Nov. 2023. [Online]. Available: `https://www.ntia.gov/issues/national-spectrum-strategy`.

[123] T. Yucek and H. Arslan, "A survey of spectrum sensing algorithms for cognitive radio applications," *IEEE Communications Surveys & Tutorials*, vol. 11, no. 1, pp. 116–130, 2009. DOI: `10.1109/SURV.2009.090109`.

[124] D. Hamza, S. Aïssa, and G. Aniba, "Equal Gain Combining for Cooperative Spectrum Sensing in Cognitive Radio Networks," *IEEE Transactions on Wireless Communications*, vol. 13, no. 8, pp. 4334–4345, 2014. DOI: `10.1109/TWC.2014.2317788`.

[125] Ł. Kułacz, "Spectrum Occupancy Detection Supported by Federated Learning," in *2023 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, 2023, pp. 1–3. DOI: `10.23919/SoftCOM58365.2023.10271668`.

[126] M. A. Abdel-Moneim, W. El-Shafai, N. Abdel-Salam, E.-S. M. El-Rabaie, and F. E. Abd El-Samie, "A survey of traditional and advanced automatic modulation classification techniques, challenges, and some novel trends," *International Journal of Communication Systems*, vol. 34, no. 10, e4762, 2021. DOI: `10.1002/dac.4762`.

[127] Federal Communications Commision, *Unlicensed Use of TV Band and 600 MHz Band Spectrum*, Federal Register, 2015.

[128] Federal Communications Commision, *Shared Commercial Operations in the 3550-3650 MHz Band*, Federal Register.

[129] Federal Communications Commision, *Unlicensed Use of the 6 GHz Band; and Expanding Flexible Use in MidBand Spectrum Between 3.7 and 24 GHz*, Federal Register, 2024.

[130] S. J. Maeng, İ. Güvenç, M. L. Sichitiu, *et al.*, "National Radio Dynamic Zone Concept with Autonomous Aerial and Ground Spectrum Sensors," in *2022 IEEE International Conference on Communications Workshops (ICC Workshops)*, 2022, pp. 687–692. DOI: `10.1109/ICCWorkshops53468.2022.9814648`.

[131] D. Burghal, A. T. Ravi, V. Rao, A. A. Alghafis, and A. F. Molisch, *A Comprehensive Survey of Machine Learning Based Localization with Wireless Signals*, 2020. arXiv: `2012.11171`.

[132] A. M. Abba, J. Sanusi, O. Oshiga, and S. A. Mikail, "A Review of Localization Techniques in Wireless Sensor Networks," in *2023 2nd International Conference on Multidisciplinary Engineering and Applied Science (ICMEAS)*, vol. 1, 2023, pp. 1–5. DOI: `10.1109/ICMEAS58693.2023.10429886`.

[133] T. D. Hall, "Radiolocation using AM broadcast signals," Ph.D. dissertation, Massachusetts Institute of Technology, 2002.

[134] S. Bartoletti, A. Conti, and M. Z. Win, "Passive radar via LTE signals of opportunity," in *2014 IEEE International Conference on Communications Workshops (ICC)*, IEEE, 2014, pp. 181–185.

[135] E. Favarelli, E. Testi, L. Pucci, M. Chiani, and A. Giorgetti, "Anomaly Detection Using WiFi Signals of Opportunity," in *2019 13th International Conference on Signal Processing and Communication Systems (ICSPCS)*, 2019, pp. 1–7. DOI: `10.1109/ICSPCS47537.2019.9008700`.

[136] J. L. Garrison, J. R. Piepmeier, and R. Shah, "Signals of opportunity: enabling new science outside of protected bands," in *2018 International Conference on Electromagnetics in Advanced Applications (ICEAA)*, IEEE, 2018, pp. 501–504.

[137] J. Lackey and S. Markgraf, *Kalibrate-RTL*, version a877f9e, 2023. [Online]. Available: `https://github.com/mutability/kalibrate-rtl`.

[138] AerospaceResearch, *CalibrateSDR*, version 51d6d44, 2023. [Online]. Available: `https://github.com/aerospaceresearch/CalibrateSDR`.

[139] J. Sun, *The 1090 Megahertz Riddle: A Guide to Decoding Mode S and ADS-B Signals*, 2nd ed. TU Delft OPEN Publishing, 2021, ISBN: 978-94-6366-402-8. DOI: `10.34641/mg.11`.

[140] R. (SC-186, *Minimum Operational Performance Standards for 1090 MHz Extended Squitter: Automatic Dependent Surveillance-Broadcast (ADS-B) and Traffic Information Services-Broadcast (TIS-B)*. RTCA, 2006.

[141] M. Eichelberger, K. Luchsinger, S. Tanner, and R. Wattenhofer, "Indoor Localization with Aircraft Signals," in *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*, ser. SenSys '17, Delft, Netherlands: Association for Computing Machinery, 2017, ISBN: 9781450354592. DOI: 10.1145/3131672.3131698.

[142] A. Canals, P. Josephy, S. Tanner, and R. Wattenhofer, "Robust indoor localization with ADS-B," in *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '21, New Orleans, Louisiana: Association for Computing Machinery, 2021, pp. 505–516, ISBN: 9781450383424. DOI: 10.1145/3447993.3483257.

[143] R. B. Kershner, "Present State of Navigation by Doppler Measurement from Near Earth Satellites," Johns Hopkins University Applied Physics Laboratory, Tech. Rep., 1965.

[144] J. Khalife and Z. M. Kassas, "Assessment of Differential Carrier Phase Measurements from Orbcomm LEO Satellite Signals for Opportunistic Navigation," in *Proceedings of the 32nd International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2019)*, 2019, pp. 4053–4063. DOI: 10.33012/2019.17031.

[145] J. J. Khalife and Z. M. Kassas, "Receiver Design for Doppler Positioning with LEO Satellites," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 5506–5510. DOI: 10.1109/ICASSP.2019.8682554.

[146] J. Khalife, M. Neinavaie, and Z. M. Kassas, "Blind Doppler Tracking from OFDM Signals Transmitted by Broadband LEO Satellites," in *2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring)*, 2021, pp. 1–5. DOI: 10.1109/VTC2021-Spring51267.2021.9448678.

[147] M. L. Psiaki and B. D. Slosman, "Tracking Digital FM OFDM Signals for the Determination of Navigation Observables," *NAVIGATION: Journal of the Institute of Navigation*, vol. 69, no. 2, 2022, ISSN: 0028-1522. DOI: 10.33012/navi.521.

[148] M. L. Psiaki, "Navigation using carrier Doppler shift from a LEO constellation: TRANSIT on steroids," *NAVIGATION: Journal of the Institute of Navigation*, vol. 68, no. 3, pp. 621–641, 2021, ISSN: 0028-1522. DOI: 10.1002/navi.438.

[149] P. P. Ray, "ChatGPT: A comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope," *Internet of Things and Cyber-Physical Systems*, vol. 3, pp. 121–154, 2023, ISSN: 2667-3452. DOI: 10.1016/j.iotcps.2023.04.003.

[150] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-Resolution Image Synthesis with Latent Diffusion Models," *CoRR*, vol. abs/2112.10752, 2021. arXiv: `2112.10752`. [Online]. Available: `https://arxiv.org/abs/2112.10752`.

[151] C. Chi, Z. Xu, S. Feng, *et al.*, *Diffusion Policy: Visuomotor Policy Learning via Action Diffusion*, 2024. arXiv: `2303.04137`.

[152] R. Verkuil, O. Kabeli, Y. Du, *et al.*, "Language models generalize beyond natural proteins," *bioRxiv*, 2022. DOI: `10.1101/2022.12.21.521521`.

[153] A. Ajay, Y. Du, A. Gupta, J. Tenenbaum, T. Jaakkola, and P. Agrawal, *Is Conditional Generative Modeling all you need for Decision-Making?* 2023. arXiv: `2211.15657`.

[154] F. A. Croitoru, V. Hondru, R. T. Ionescu, and M. Shah, "Diffusion Models in Vision: A Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, pp. 10 850–10 869, 9 Sep. 2022, ISSN: 19393539. DOI: `10.1109/tpami.2023.3261988`.

[155] P. Pernias, D. Rampas, M. L. Richter, C. J. Pal, and M. Aubreville, *Wuerstchen: An Efficient Architecture for Large-Scale Text-to-Image Diffusion Models*, 2023. arXiv: `2306.00637 [cs.CV]`. [Online]. Available: `https://arxiv.org/abs/2306.00637`.

[156] Y. Jin, Z. Sun, N. Li, *et al.*, *Pyramidal Flow Matching for Efficient Video Generative Modeling*, 2024. arXiv: `2410.05954`.

[157] C. Zou, F. Yang, J. Song, and Z. Han, "Underwater Wireless Optical Communication With One-Bit Quantization: A Hybrid Autoencoder and Generative Adversarial Network Approach," *IEEE Transactions on Wireless Communications*, vol. 22, no. 10, pp. 6432–6444, 2023. DOI: `10.1109/TWC.2023.3243212`.

[158] A. Caciularu and D. Burshtein, "Unsupervised Linear and Nonlinear Channel Equalization and Decoding Using Variational Autoencoders," *IEEE Transactions on Cognitive Communications and Networking*, vol. 6, pp. 1003–1018, 3 Sep. 2020, ISSN: 23327731. DOI: `10.1109/TCCN.2020.2990773`.

[159] J. Song, V. Lauinger, Y. Wu, *et al.*, *Blind Channel Equalization Using Vector-Quantized Variational Autoencoders*, 2023. arXiv: `2302.11687`.

[160] S. Liu, T. Wang, and S. Wang, "Toward intelligent wireless communications: Deep learning - based physical layer technologies," *Digital Communications and Networks*, vol. 7, no. 4, pp. 589–597, 2021, ISSN: 2352-8648. DOI: `10.1016/j.dcan.2021.09.014`.

[161] N. Van Huynh and G. Y. Li, "Transfer Learning for Signal Detection in Wireless Networks," *IEEE Wireless Communications Letters*, vol. 11, no. 11, pp. 2325–2329, 2022. DOI: `10.1109/LWC.2022.3202117`.

[162] B. Tang, Y. Tu, Z. Zhang, and Y. Lin, "Digital Signal Modulation Classification With Data Augmentation Using Generative Adversarial Nets in Cognitive Radio Networks," *IEEE Access*, vol. 6, pp. 15 713–15 722, 2018. DOI: `10.1109/ACCESS.2018.2815741`.

[163] H. Ye, L. Liang, G. Y. Li, and B.-H. Juang, "Deep Learning-Based End-to-End Wireless Communication Systems With Conditional GANs as Unknown Channels," *IEEE Transactions on Wireless Communications*, vol. 19, no. 5, pp. 3133–3143, 2020. DOI: `10.1109/TWC.2020.2970707`.

[164] L. Sun, Y. Wang, A. L. Swindlehurst, and X. Tang, "Generative-Adversarial-Network Enabled Signal Detection for Communication Systems With Unknown Channel Models," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 1, pp. 47–60, 2021. DOI: `10.1109/JSAC.2020.3036954`.

[165] T. Zhao and F. Li, "Variational-autoencoder signal detection for MIMO-OFDM-IM," *Digital Signal Processing*, vol. 118, p. 103 230, 2021, ISSN: 1051-2004. DOI: `10.1016/j.dsp.2021.103230`.

[166] Q. Li, Z. Xiang, P. Ren, and W. Li, "Variational autoencoder based receiver for orthogonal time frequency space modulation," *Digital Signal Processing*, vol. 117, p. 103 170, 2021, ISSN: 1051-2004. DOI: `https://doi.org/10.1016/j.dsp.2021.103170`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S1051200421002098`.

[167] M. A. Alawad, M. Q. Hamdan, and K. A. Hamdi, "Innovative Variational AutoEncoder for an End-to-End Communication System," *IEEE Access*, vol. 11, pp. 86 834–86 847, 2023. DOI: `10.1109/ACCESS.2022.3224922`.

[168] U. Sengupta, C. Jao, A. Bernacchia, S. Vakili, and D.-s. Shiu, *Generative Diffusion Models for Radio Wireless Channel Modelling and Sampling*, 2023. arXiv: `2308.05583`.

[169] Y. Hu, M. Yin, W. Xia, S. Rangan, and M. Mezzavilla, *Multi-Frequency Channel Modeling for Millimeter Wave and THz Wireless Communication via Generative Adversarial Networks*, 2022. arXiv: `2212.11858`.

[170] 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, 2018. DOI: `10.1109/JSTSP.2018.2797022`.

[171] 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. DOI: `10.1109/GlobalSIP45357.2019.8969160`.

[172] K. Chowdhury, T. Melodia, F. Restuccia, and A. Sabharwal, Dec. 2024. [Online]. Available: `https://www.rfdatafactory.com/`.

[173] A. Sahai, Personal Communication, Dec. 2024.

[174] Federal Communications Commision, *Establishing the Digital Opportunity Data Collection; Modernizing the FCC Form 477 Data Program*, Federal Register.

[175] C. Zhang, Y. Xie, H. Bai, B. Yu, W. Li, and Y. Gao, "A survey on federated learning," *Knowledge-Based Systems*, vol. 216, p. 106 775, Mar. 2021, ISSN: 0950-7051. DOI: 10.1016/J.KNOSYS.2021.106775.

[176] O. Aygün, M. Kazemi, D. Gündüz, *et al.*, "Over-the-Air Federated Edge Learning with Hierarchical Clustering," vol. 1, Jul. 2022. DOI: 10.48550/arxiv.2207.09232.

[177] J. Wei, Y. Zhang, L. Y. Zhang, *et al.*, *Memorization in deep learning: A survey*, 2024. arXiv: 2406.03880.

[178] M. Abadi, A. Chu, I. Goodfellow, *et al.*, "Deep Learning with Differential Privacy," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '16, Vienna, Austria: Association for Computing Machinery, 2016, pp. 308–318, ISBN: 9781450341394. DOI: 10.1145/2976749.2978318.

[179] M. U. Hassan, M. H. Rehmani, and J. Chen, "Differential Privacy Techniques for Cyber Physical Systems: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 1, pp. 746–789, 2020. DOI: 10.1109/COMST.2019.2944748.

[180] B. Juba and M. Sudan, "Universal semantic communication I," in *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, ser. STOC '08, Victoria, British Columbia, Canada: Association for Computing Machinery, 2008, pp. 123–132, ISBN: 9781605580470. DOI: 10.1145/1374376.1374397.

[181] B. Juba and M. Sudan, "Universal semantic communication II: A theory of goal-oriented communication," in *Electronic Colloquium on Computational Complexity (ECCC)*, vol. 15, 2008. [Online]. Available: https://madhu.seas.harvard.edu/papers/2008/usc2.pdf.

[182] I. Komargodski, P. Kothari, and M. Sudan, "Communication with Contextual Uncertainty," *CoRR*, vol. abs/1504.04813, 2015. arXiv: 1504.04813.

[183] C. L. Canonne, V. Guruswami, R. Meka, and M. Sudan, "Communication with Imperfectly Shared Randomness," *CoRR*, vol. abs/1411.3603, 2014. arXiv: 1411.3603.

[184] O. Goldreich, B. Juba, and M. Sudan, "A theory of goal-oriented communication," *J. ACM*, vol. 59, no. 2, 8:1–8:65, 2012. DOI: 10.1145/2160158.2160161.

[185] C. E. Shannon, "A Mathematical Theory of Communication," *Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, 1948. DOI: 10.1002/j.1538-7305.1948.tb01338.x.

[186] 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, ISSN: 2332-7731. DOI: 10.1109/TCCN.2017.2758370.

[187] H. Ye, L. Liang, G. Y. Li, and B. F. Juang, "Deep learning based end-to-end wireless communication systems with conditional GAN as unknown channel," *CoRR*, vol. abs/1903.02551, 2019. arXiv: 1903.02551.

[188] H. Ye, G. Y. Li, B. F. Juang, and K. Sivanesan, "Channel agnostic end-to-end learning based communication systems with conditional GAN," *CoRR*, vol. abs/1807.00447, 2018. arXiv: 1807.00447. [Online]. Available: http://arxiv.org/abs/1807.00447.

[189] T. J. O'Shea, T. Roy, and N. West, "Approximating the Void: Learning Stochastic Channel Models from Observation with Variational Generative Adversarial Networks," *CoRR*, vol. abs/1805.06350, 2018. arXiv: 1805.06350.

[190] F. A. Aoudia and J. Hoydis, "End-to-End Learning of Communications Systems Without a Channel Model," *CoRR*, vol. abs/1804.02276, 2018. arXiv: 1804.02276.

[191] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine learning*, vol. 8, no. 3-4, pp. 229–256, 1992.

[192] Y. Li and Z. Ding, *Blind equalization and identification*. CRC press, 2001.

[193] E. Azzouz and A. Nandi, *Automatic Modulation Recognition of Communication Signals*. Springer US, 2013, ISBN: 9781475724707.

[194] Z. C. Lipton and J. Steinhardt, "Troubling Trends in Machine Learning Scholarship," *Queue*, vol. 17, no. 1, 80:45–80:77, Feb. 2019, ISSN: 1542-7730. DOI: 10.1145/3317287.3328534.

[195] C. Tran, K. Vodrahalli, V. Subramanian, and J. Sanz, *echo*, https://github.com/ml4wireless/echo, 2019.

[196] H. Kim, Y. Jiang, R. Rana, S. Kannan, S. Oh, and P. Viswanath, "Communication Algorithms via Deep Learning," *arXiv e-prints*, Dec. 2018. arXiv: 1805.09317.

[197] T. J. O'Shea and J. Hoydis, "An Introduction to Machine Learning Communications Systems," *CoRR*, vol. abs/1702.00832, 2017. arXiv: 1702.00832. [Online]. Available: http://arxiv.org/abs/1702.00832.

[198] T. J. O'Shea and J. Corgan, "Convolutional Radio Modulation Recognition Networks," *CoRR*, vol. abs/1602.04105, 2016. arXiv: 1602.04105.

[199] 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, Apr. 2019, ISSN: 1939-9359. DOI: 10.1109/TVT.2019.2900460.

[200] Y. Wang, J. Yang, M. Liu, and G. Gui, "LightAMC: Lightweight automatic modulation classification using deep learning and compressive sensing," *IEEE Transactions on Vehicular Technology*, pp. 1–1, 2020, ISSN: 1939-9359. DOI: 10.1109/TVT.2020.2971001.

[201] Y. Tu and Y. Lin, "Deep Neural Network Compression Technique Towards Efficient Digital Signal Modulation Recognition in Edge Device," *IEEE Access*, vol. PP, pp. 1–1, Apr. 2019. DOI: 10.1109/ACCESS.2019.2913945.

[202] G. E. Hinton and R. R. Salakhutdinov, "Reducing the Dimensionality of Data with Neural Networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006, ISSN: 0036-8075. DOI: 10.1126/science.1127647.

[203] V. Raj and S. Kalyani, "Backpropagating Through the Air: Deep Learning at Physical Layer Without Channel Models," *IEEE Communications Letters*, vol. 22, no. 11, pp. 2278–2281, Nov. 2018, ISSN: 1089-7798. DOI: 10.1109/LCOMM.2018.2868103.

[204] J. Schmitz, C. von Lengerke, N. Airee, A. Behboodi, and R. Mathar, "A Deep Learning Wireless Transceiver with Fully Learned Modulation and Synchronization," Dec. 2019. arXiv: 1905.10468.

[205] S. Dörner, S. Cammerer, J. Hoydis, and S. t. Brink, "Deep Learning Based Communication Over the Air," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 132–143, Feb. 2018, ISSN: 1932-4553. DOI: 10.1109/JSTSP.2017.2784180.

[206] T. J. O'Shea, L. Pemula, D. Batra, and T. C. Clancy, "Radio Transformer Networks: Attention Models for Learning to Synchronize in Wireless Systems," *CoRR*, vol. abs/1605.00716, 2016. arXiv: 1605.00716.

[207] A. Caciularu and D. Burshtein, "Blind Channel Equalization using Variational Autoencoders," *arXiv e-prints*, Mar. 2018. arXiv: 1803.01526.

[208] L. Brink, A. Sahai, and J. Wawrzynek, "Deep Networks for Equalization in Communications," M.S. thesis, University of California, Berkeley, 2018. [Online]. Available: https://www2.eecs.berkeley.edu/Pubs/TechRpts/2018/EECS-2018-177.pdf.

[209] V. Corlay, J. J. Boutros, P. Ciblat, and L. Brunel, "Neural Lattice Decoders," *CoRR*, vol. abs/1807.00592, 2018. arXiv: 1807.00592.

[210] K. Choi, K. Tatwawadi, T. Weissman, and S. Ermon, "NECST: Neural Joint Source-Channel Coding," *CoRR*, vol. abs/1811.07557, 2018. arXiv: 1811.07557.

[211] E. Bourtsoulatze, D. B. Kurka, and D. Gündüz, "Deep Joint Source-Channel Coding for Wireless Image Transmission," *CoRR*, vol. abs/1809.01733, 2018. arXiv: 1809.01733.

[212] B. Zhu, J. Wang, L. He, and J. Song, "Joint transceiver optimization for wireless communication PHY with convolutional neural network," *arXiv e-prints*, arXiv:1808.03242, arXiv:1808.03242, Aug. 2018. arXiv: 1808.03242 [eess.SP].

[213] D. J. Ji, J. Park, and D. Cho, "ConvAE: A new channel autoencoder based on convolutional layers and residual connections," *IEEE Communications Letters*, pp. 1–1, 2019. DOI: 10.1109/LCOMM.2019.2930287.

[214] N. Wu, X. Wang, B. Lin, and K. Zhang, "A CNN-based end-to-end learning framework toward intelligent communication systems," *IEEE Access*, vol. 7, pp. 110 197–110 204, 2019. DOI: 10.1109/ACCESS.2019.2926843.

[215] T. Mu, X. Chen, L. Chen, H. Yin, and W. Wang, "An End-to-End Block Autoencoder For Physical Layer Based On Neural Networks," *arXiv e-prints*, arXiv:1906.06563, arXiv:1906.06563, Jun. 2019. arXiv: `1906.06563 [cs.IT]`.

[216] A. Felix, S. Cammerer, S. Dörner, J. Hoydis, and S. ten Brink, "OFDM-autoencoder for end-to-end learning of communications systems," *CoRR*, 2018. arXiv: `1803.05815`.

[217] S. Li, C. Häger, N. Garcia, and H. Wymeersch, "Achievable Information Rates for Nonlinear Fiber Communication via End-to-end Autoencoder Learning," 2018. arXiv: `1804.07675`.

[218] B. Karanov, M. Chagnon, F. Thouin, *et al.*, "End-to-end Deep Learning of Optical Fiber Communications," 2018. arXiv: `1804.04097`.

[219] C. Lee, H. B. Yilmaz, C. Chae, N. Farsad, and A. Goldsmith, "Machine learning based channel modeling for molecular MIMO communications," in *2017 IEEE 18th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, Jul. 2017, pp. 1–5. DOI: `10.1109/SPAWC.2017.8227765`.

[220] S. Mohamed, J. Dong, A. R. Junejo, and D. C. Zuo, "Model-Based: End-to-End Molecular Communication System Through Deep Reinforcement Learning Auto Encoder," *IEEE Access*, vol. 7, pp. 70 279–70 286, 2019, ISSN: 2169-3536. DOI: `10.1109/ACCESS.2019.2916701`.

[221] R. Raileanu, E. Denton, A. Szlam, and R. Fergus, "Modeling Others using Oneself in Multi-Agent Reinforcement Learning," *CoRR*, 2018. arXiv: `1802.09640`.

[222] M. Lanctot, V. Zambaldi, A. Gruslys, *et al.*, "A Unified Game-Theoretic Approach to Multiagent Reinforcement Learning," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, *et al.*, Eds., vol. 30, Curran Associates, Inc., 2017.

[223] K. Zhang, Z. Yang, H. Liu, T. Zhang, and T. Basar, "Fully Decentralized Multi-Agent Reinforcement Learning with Networked Agents," *CoRR*, 2018. arXiv: `1802.08757`.

[224] J. Foerster, I. A. Assael, N. de Freitas, and S. Whiteson, "Learning to Communicate with Deep Multi-Agent Reinforcement Learning," in *Advances in Neural Information Processing Systems 29*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds., Curran Associates, Inc., 2016, pp. 2137–2145.

[225] R. Siegler, J. DeLoache, and N. Eisenberg, *How Children Develop*, 5th ed. Worth Publishers, 2017, pp. 249–272, ISBN: 9781319014230.

[226] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks," *arXiv e-prints*, arXiv:1703.10593, arXiv:1703.10593, Mar. 2017. arXiv: `1703.10593 [cs.CV]`.

[227] V. Mnih, K. Kavukcuoglu, D. Silver, *et al.*, "Playing Atari with deep reinforcement learning," Dec. 2013. arXiv: `1312.5602`.

[228] R. Novak, Y. Bahri, D. A. Abolafia, J. Pennington, and J. Sohl-Dickstein, "Sensitivity and Generalization in Neural Networks: an Empirical Study," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018. [Online]. Available: `https://openreview.net/forum?id=HJC2SzZCW`.

[229] GNU Radio Website, 2019. [Online]. Available: `http://www.gnuradio.org`.

[230] S. van der Walt, S. C. Colbert, and G. Varoquaux, "The NumPy array: A structure for efficient numerical computation," *Computing in Science & Engineering*, vol. 13, pp. 22–30, 2011. DOI: `10.1109/MCSE.2011.37`.

[231] C. R. Barrett, "Adaptive Thresholding and Automatic Detection," in *Principles of Modern Radar*, J. L. Eaves and E. K. Reedy, Eds., Boston, MA: Springer US, 1987, pp. 368–393, ISBN: 978-1-4613-1971-9. DOI: `10.1007/978-1-4613-1971-9_12`.

[232] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*, 2019. arXiv: `1810.04805`.

[233] L. Ericsson, H. Gouk, C. C. Loy, and T. M. Hospedales, "Self-Supervised Representation Learning: Introduction, advances, and challenges," *IEEE Signal Processing Magazine*, vol. 39, no. 3, pp. 42–62, 2022. DOI: `10.1109/MSP.2021.3134634`.

[234] A. Nichol, J. Achiam, and J. Schulman, "On First-Order Meta-Learning Algorithms," *CoRR*, vol. abs/1803.02999, 2018. arXiv: `1803.02999`.

[235] S. Park and O. Simeone, "Predicting Flat-Fading Channels via Meta-Learned Closed-Form Linear Filters and Equilibrium Propagation," Oct. 2021. DOI: `10.48550/arxiv.2110.00414`.

[236] *Realizing the Full Potential of Government-held Spectrum to Spur Economic Growth.* President's Council of Advisors on Science and Technology, Jul. 2012. [Online]. Available: `https://obamawhitehouse.archives.gov/sites/default/files/microsites/ostp/pcast_spectrum_report_final_july_20_2012.pdf`.

[237] H. Holma, A. Toskala, and T. Nakamura, *5G technology : 3GPP new radio, year = 2020.* John Wiley & Sons, Inc., p. 506, ISBN: 9781119236313.

[238] T. Taher, R. Attard, A. Riaz, *et al.*, "Global spectrum observatory network setup and initial findings," in *2014 9th International Conference on Cognitive Radio Oriented Wireless Networks and Communications (CROWNCOM)*, 2014, pp. 79–88. DOI: `10.4108/icst.crowncom.2014.255402`.

[239] K. Chintalapudi, V. Navda, R. Ramjee, V. N. Padmanabhan, V. Navda, and C. R. Murthy, "SpecNet: Spectrum sensing sans Frontières," in *8th USENIX Symposium on Networked Systems Design and Implementation (NSDI 11)*, Boston, MA: USENIX Association, Mar. 2011. [Online]. Available: `https://www.usenix.org/conference/nsdi11/specnet-spectrum-sensing-sans-fronti%7B%5C%60e%7Dres`.

[240] S. Rajendran, R. Calvo-Palomino, M. Fuchs, *et al.*, "Electrosense: Open and Big Spectrum Data," *IEEE Communications Magazine*, vol. 56, no. 1, pp. 210–217, 2018. DOI: `10.1109/MCOM.2017.1700200`.

[241] A. Abedi, J. Sanz, and A. Sahai, "Automatic Calibration in Crowd-Sourced Network of Spectrum Sensors," in *Proceedings of the 22nd ACM Workshop on Hot Topics in Networks*, ser. HotNets '23, Cambridge, MA, USA: Association for Computing Machinery, 2023, pp. 157–164, ISBN: 9798400704154. DOI: `10.1145/3626111.3628187`.

[242] J. Sanz, A. Abedi, and A. Sahai, "Automatic Indoor-Outdoor Detection Using Signals of Opportunity," in *2024 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, May 2024. DOI: `10.1109/DySPAN60163.2024.10632851`.

[243] A. Abedi, J. Sanz, and A. Sahai, "Using Signals of Opportunity to Establish Trust in Distributed Spectrum Monitoring Systems," in *2024 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, 2024, pp. 33–38. DOI: `10.1109/DySPAN60163.2024.10632745`.

[244] A. Canals, P. Josephy, S. Tanner, and R. Wattenhofer, "Robust Indoor Localization with ADS-B," in *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '21, 2021, pp. 505–516.

[245] A. Stove, "Calibration of active arrays using signals of opportunity," in *IEE Seminar on Calibration of Active Phased Array Antennas*, IET, 2005, pp. 1–2.

[246] L. Balzano and R. Nowak, "Blind Calibration of Sensor Networks," in *2007 6th International Symposium on Information Processing in Sensor Networks*, 2007, pp. 79–88. DOI: `10.1109/IPSN.2007.4379667`.

[247] A. Ramamurthy, V. Sathya, M. I. Rochman, and M. Ghosh, "ML-Based Classification of Device Environment Using Wi-Fi and Cellular Signal Measurements," *IEEE Access*, vol. 10, pp. 29 461–29 472, 2022. DOI: `10.1109/ACCESS.2022.3158056`.

[248] A. Ramamurthy, V. Sathya, M. I. Rochman, and M. Ghosh, "ML-based classification of device environment using Wi-Fi and cellular signal measurements," *IEEE Access*, vol. 10, pp. 29 461–29 472, 2022.

[249] SC-239, "Assessment of C-Band Mobile Telecommunications Interference Impact on Low Range Radar Altimeter Operations," RTCA, Inc, Tech. Rep., 2020.

[250] N. A. of Sciences Engineering and Medicine, *Analysis of Potential Interference Issues Related to FCC Order 20-48*. Washington, DC: The National Academies Press, 2023, ISBN: 978-0-309-69007-2. DOI: `10.17226/26611`.

[251] I. F. Akyildiz, B. F. Lo, and R. Balakrishnan, "Cooperative spectrum sensing in cognitive radio networks: A survey," *Physical Communication*, vol. 4, no. 1, pp. 40–62, 2011, ISSN: 1874-4907. DOI: `10.1016/j.phycom.2010.12.003`.

[252] M. Zhang, L. Wang, and Y. Feng, "Distributed cooperative spectrum sensing based on reinforcement learning in cognitive radio networks," *AEU - International Journal of Electronics and Communications*, vol. 94, pp. 359–366, 2018, ISSN: 1434-8411. DOI: `10.1016/j.aeue.2018.07.029`.

[253] Y. Hu and R. Zhang, "A Spatiotemporal Approach for Secure Crowdsourced Radio Environment Map Construction," *IEEE/ACM Transactions on Networking*, vol. 28, no. 4, pp. 1790–1803, 2020. DOI: `10.1109/TNET.2020.2992939`.

[254] D. Herbert, V. Sundaram, Y.-H. Lu, S. Bagchi, and Z. Li, "Adaptive correctness monitoring for wireless sensor networks using hierarchical distributed run-time invariant checking," *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. 2, no. 3, 8–es, 2007.

[255] P. Eugster, V. Sundaram, and X. Zhang, "Debugging the Internet of Things: The case of wireless sensor networks," *IEEE Software*, vol. 32, no. 1, pp. 38–49, 2014.

[256] Ł. Kułacz, P. Kryszkiewicz, A. Kliks, *et al.*, "Coordinated Spectrum Allocation and Coexistence Management in CBRS-SAS Wireless Networks," *IEEE Access*, vol. 7, pp. 139 294–139 316, 2019. DOI: `10.1109/ACCESS.2019.2940448`.

[257] O. Fatemieh, R. Chandra, and C. A. Gunter, "Secure Collaborative Sensing for Crowd Sourcing Spectrum Data in White Space Networks," in *2010 IEEE Symposium on New Frontiers in Dynamic Spectrum (DySPAN)*, 2010, pp. 1–12. DOI: `10.1109/DYSPAN.2010.5457893`.

[258] K. Chen, D. G. Dobhakhshari, V. Gupta, and Y.-F. Huang, "An Incentive Scheme for Sensor Fusion With Strategic Sensors," *IEEE Transactions on Signal Processing*, vol. 67, no. 24, pp. 6342–6351, 2019. DOI: `10.1109/TSP.2019.2954974`.

[259] S. Roy, K. Shin, A. Ashok, *et al.*, "CityScape: A Metro-Area Spectrum Observatory," in *2017 26th International Conference on Computer Communication and Networks (ICCCN)*, 2017, pp. 1–9. DOI: `10.1109/ICCCN.2017.8038427`.

[260] M. Z. Zheleva, R. Chandra, A. Chowdhery, *et al.*, "Enabling a Nationwide Radio Frequency Inventory Using the Spectrum Observatory," *IEEE Transactions on Mobile Computing*, vol. 17, no. 2, pp. 362–375, 2018. DOI: `10.1109/TMC.2017.2716936`.

[261] N. Kleber, A. Termos, G. Martinez, *et al.*, "RadioHound: A pervasive sensing platform for sub-6 GHz dynamic spectrum monitoring," in *2017 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, 2017, pp. 1–2. DOI: `10.1109/DySPAN.2017.7920764`.

[262] R. Morales-Ferre, E. S. Lohan, G. Falco, and E. Falletti, "GDOP-based analysis of suitability of LEO constellations for future satellite-based positioning," in *2020 IEEE International Conference on Wireless for Space and Extreme Environments (WiSEE)*, 2020, pp. 147–152. DOI: `10.1109/WiSEE44079.2020.9262624`.

[263] S. Kobayakawa, M. Tsutsui, and Y. Tanaka, "A blind calibration method for an adaptive array antenna in DS-CDMA systems using an MMSE algorithm," in *VTC2000-Spring. 2000 IEEE 51st Vehicular Technology Conference Proceedings (Cat. No.00CH37026)*, vol. 1, 2000, 21–25 vol.1. DOI: `10.1109/VETECS.2000.851410`.

[264] M. Asai and H. Arai, "Blind Phase Calibration by Near Field Null Detection for Linear Array Antenna, year = 2019," pp. 1–2.

[265] A. Yang, P. Wang, and H. Yang, "Blind Drift Calibration of Sensor Networks Using Multi-Output Gaussian Process," *Proceedings of IEEE Sensors*, vol. 2018-October, Dec. 2018, ISSN: 21689229. DOI: `10.1109/ICSENS.2018.8589548`.

[266] B. W. Parkinson, "Introduction And Heritage Of Navstar, The Global Positioning System," in *Global Positioning System: Theory and Applications, Volume I*, pp. 3–28. DOI: `10.2514/5.9781600866388.0003.0028`.

[267] R. Raja, S. M. M. Roomi, D. Dharmalakshmi, and S. Rohini, "Classification of indoor/outdoor scene," in *2013 IEEE International Conference on Computational Intelligence and Computing Research*, 2013, pp. 1–4. DOI: `10.1109/ICCIC.2013.6724252`.

[268] B. Kumar, H. Gupta, S. P. Ingale, and O. P. Vyas, "Classification of Indoor–Outdoor Scene Using Deep Learning Techniques," in *Machine Learning, Image Processing, Network Security and Data Sciences*, R. Doriya, B. Soni, A. Shukla, and X.-Z. Gao, Eds., Singapore: Springer Nature Singapore, 2023, pp. 517–535, ISBN: 978-981-19-5868-7.

[269] A. Matei, A. Glavan, and E. Talavera, "Deep Learning for Scene Recognition from Visual Data: A Survey," in *Hybrid Artificial Intelligent Systems*, E. A. de la Cal, J. R. Villar Flecha, H. Quintián, and E. Corchado, Eds., Cham: Springer International Publishing, 2020, pp. 763–773, ISBN: 978-3-030-61705-9.

[270] A. Alameer, P. Degenaar, and K. Nazarpour, "Context-Based Object Recognition: Indoor Versus Outdoor Environments," in *Advances in Computer Vision*, K. Arai and S. Kapoor, Eds., Cham: Springer International Publishing, 2020, pp. 473–490, ISBN: 978-3-030-17798-0.

[271] V. Bui, N. T. Le, T. L. Vu, V. H. Nguyen, and Y. M. Jang, "GPS-Based Indoor/Outdoor Detection Scheme Using Machine Learning Techniques," *Applied Sciences*, vol. 10, no. 2, 2020, ISSN: 2076-3417. DOI: `10.3390/app10020500`.

[272] B. Lee, C. Lim, and K. Lee, "Classification of indoor-outdoor location using combined global positioning system (GPS) and temperature data for personal exposure assessment," *Environmental health and preventive medicine*, vol. 22, 2017. DOI: `10.1186/s12199-017-0637-4`.

[273] J. Wu, C. Jiang, D. Houston, D. Baker, and R. Delfino, "Automated time activity classification based on global positioning system (GPS) tracking data," *Environmental health*, vol. 10, 2011. DOI: `10.1186/1476-069X-10-101`.

[274] S. M. Mishra, R. Tandra, and A. Sahai, "The case for multiband sensing," in *Proceedings of the 45th Annual Allerton Conference on Control, Communications, and Computation*, 2007. [Online]. Available: `https://api.semanticscholar.org/CorpusID:8640796`.

[275] M. Erel-Özçevik and B. Canberk, "Road to 5G reduced-latency: A software defined handover model for eMBB services," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 8, pp. 8133–8144, 2019. DOI: `10.1109/TVT.2019.2925393`.

[276] V. Sathya, A. Ramamurthy, and B. R. Tamma, "On placement and dynamic power control of femtocells in LTE HetNets," in *2014 IEEE Global Communications Conference*, 2014, pp. 4394–4399. DOI: `10.1109/GLOCOM.2014.7037499`.

[277] L. Zhang, Q. Ni, M. Zhai, J. Moreno, and C. Briso, "An ensemble learning scheme for indoor-outdoor classification based on KPIs of LTE network," *IEEE Access*, vol. 7, pp. 63 057–63 065, 2019. DOI: `10.1109/ACCESS.2019.2914451`.

[278] T. Ahmad, R. Chandra, A. Kapoor, M. Daum, and E. Horvitz, "Wi-Fly: Widespread Opportunistic Connectivity via Commercial Air Transport," in *Proceedings of the 16th ACM Workshop on Hot Topics in Networks*, ser. HotNets-XVI, Palo Alto, CA, USA: Association for Computing Machinery, 2017, pp. 43–49, ISBN: 9781450355698. DOI: `10.1145/3152434.3152458`.

[279] F. Sailhan, V. Issarny, and O. Tavares-Nascimiento, "Opportunistic Multiparty Calibration for Robust Participatory Sensing," in *2017 IEEE 14th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, 2017, pp. 435–443. DOI: `10.1109/MASS.2017.56`.

[280] J.-F. Markert, M. Budde, G. Schindler, M. Klug, and M. Beigl, "Private Rendezvous-Based Calibration of Low-Cost Sensors for Participatory Environmental Sensing," in *Proceedings of the Second International Conference on IoT in Urban Space*, ser. Urb-IoT '16, Tokyo, Japan: Association for Computing Machinery, 2016, pp. 82–85, ISBN: 9781450342049. DOI: `10.1145/2962735.2962754`.

[281] K. Wiesner, F. Dorfmeister, and C. Linnhoff-Popien, "Privacy-Preserving Calibration for Participatory Sensing," in *Mobile and Ubiquitous Systems: Computing, Networking, and Services*, I. Stojmenovic, Z. Cheng, and S. Guo, Eds., Cham: Springer International Publishing, 2014, pp. 276–288.

[282] H. Mousa, S. B. Mokhtar, O. Hasan, O. Younes, M. Hadhoud, and L. Brunie, "Trust management and reputation systems in mobile participatory sensing applications: A survey," *Computer Networks*, vol. 90, pp. 49–73, 2015, Crowdsourcing, ISSN: 1389-1286. DOI: `10.1016/j.comnet.2015.07.011`.

[283] W. Feng, Z. Yan, H. Zhang, K. Zeng, Y. Xiao, and Y. T. Hou, "A Survey on Security, Privacy, and Trust in Mobile Crowdsourcing," *IEEE Internet of Things Journal*, vol. 5, no. 4, pp. 2971–2992, 2018. DOI: `10.1109/JIOT.2017.2765699`.

[284] *ADS-B Airspace and Coverage Map*, https://www.faa.gov/air_traffic/technology/equipadsb/research/airspace, 2023.

[285] FlightAware, *dump1090*, https://github.com/flightaware/dump1090, version v8.2, Jun. 29, 2023.

[286] N. Salleh, S. S. Yuhaniz, N. F. M. Azmi, and S. F. Sabri, "Enhancing Simplified General Perturbations-4 Model for Orbit Propagation Using Deep Learning: A Review," in *Proceedings of the 2019 8th International Conference on Software and Computer Applications*, ser. ICSCA '19, 2019, pp. 27–32.

[287] Brandon Rhodes, *pyEphem*, https://rhodesmill.org/pyephem/, version 4.1.4, Sep. 1, 2023.

[288] M. A. Fischler and R. C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, Jun. 1981, ISSN: 0001-0782. DOI: 10.1145/358669.358692.

[289] srsRAN Project, *srsUE*, https://github.com/srsran/srsRAN_4G, version 23.04.1, Jun. 29, 2023.

[290] GNU Radio project, *GNU Radio*, https://gnuradio.org, version 3.10.6.0, Jun. 27, 2023.

[291] Purple Air, Inc. "Air Quality Sensors." (2023), [Online]. Available: https://www2.purpleair.com/products/list (visited on 09/21/2023).

[292] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2017.

[293] F. Pedregosa, G. Varoquaux, A. Gramfort, *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[294] C. R. Steger, B. Steger, and C. Schär, "HORAYZON v1.2: an efficient and flexible ray-tracing algorithm to compute horizon and sky view factor," *Geoscientific Model Development*, vol. 15, no. 17, pp. 6817–6840, 2022. DOI: 10.5194/gmd-15-6817-2022.

[295] M. Christ, N. Braun, J. Neuffer, and A. W. Kempa-Liehr, "Time Series FeatuRe Extraction on basis of Scalable Hypothesis tests (tsfresh – A Python package)," *Neurocomputing*, vol. 307, pp. 72–77, 2018, ISSN: 0925-2312. DOI: 10.1016/j.neucom.2018.03.067.

[296] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Machine Learning*, vol. 63, pp. 3–42, 2006. [Online]. Available: https://api.semanticscholar.org/CorpusID:15137276.

[297] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001, ISSN: 0885-6125. DOI: 10.1023/A:1010933404324.

[298] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training, year = 2018,"

[299] J. Kaplan, S. McCandlish, T. Henighan, *et al.*, "Scaling Laws for Neural Language Models," *CoRR*, vol. abs/2001.08361, 2020. arXiv: `2001.08361`.

[300] K. Mei, Z. Tu, M. Delbracio, H. Talebi, V. M. Patel, and P. Milanfar, *Bigger is not Always Better: Scaling Properties of Latent Diffusion Models*, 2024. arXiv: `2404.01367 [cs.CV]`.

[301] Z. Liang, H. He, C. Yang, and B. Dai, *Scaling Laws For Diffusion Transformers*, 2024. arXiv: `2410.08184 [cs.CV]`.

[302] A. Aghajanyan, L. Yu, A. Conneau, *et al.*, *Scaling Laws for Generative Mixed-Modal Language Models*, 2023. arXiv: `2301.03728 [cs.CL]`.

[303] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language Models are Unsupervised Multitask Learners," 2019.

[304] H. Touvron, T. Lavril, G. Izacard, *et al.*, *LLaMA: Open and Efficient Foundation Language Models*, 2023. arXiv: `2302.13971 [cs.CL]`.

[305] L. Gao, S. Biderman, S. Black, *et al.*, *The Pile: An 800GB Dataset of Diverse Text for Language Modeling*, 2020. arXiv: `2101.00027 [cs.CL]`.

[306] C. Schuhmann, R. Vencu, R. Beaumont, *et al.*, *LAION-400M: Open Dataset of CLIP-Filtered 400 Million Image-Text Pairs*, 2021. arXiv: `2111.02114 [cs.CV]`.

[307] K. Desai, G. Kaul, Z. Aysola, and J. Johnson, *RedCaps: web-curated image-text data created by the people, for the people*, 2021. arXiv: `2111.11431 [cs.CV]`.

[308] A. Radford, J. W. Kim, C. Hallacy, *et al.*, *Learning Transferable Visual Models From Natural Language Supervision*, 2021. arXiv: `2103.00020 [cs.CV]`.

[309] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised Visual Representation Learning by Context Prediction," *CoRR*, vol. abs/1505.05192, 2015. arXiv: `1505.05192`.

[310] A. van den Oord, Y. Li, and O. Vinyals, *Representation Learning with Contrastive Predictive Coding*, 2019. arXiv: `1807.03748 [cs.LG]`.

[311] Q. Xie, E. H. Hovy, M. Luong, and Q. V. Le, "Self-training with Noisy Student improves ImageNet classification," *CoRR*, 2019. arXiv: `1911.04252`.

[312] M. Chen, A. Radford, R. Child, *et al.*, "Generative Pretraining From Pixels," in *Proceedings of the 37th International Conference on Machine Learning*, H. D. III and A. Singh, Eds., ser. Proceedings of Machine Learning Research, vol. 119, PMLR, Jul. 2020, pp. 1691–1703. [Online]. Available: `https://proceedings.mlr.press/v119/chen20s.html`.

[313] T.-M. Hung, B.-Y. Chen, Y.-T. Yeh, and Y.-H. Yang, *A Benchmarking Initiative for Audio-Domain Music Generation Using the Freesound Loop Dataset*, 2022. arXiv: `2108.01576 [cs.SD]`.

[314] S. Ji, J. Luo, and X. Yang, *A Comprehensive Survey on Deep Music Generation: Multi-level Representations, Algorithms, Evaluations, and Future Directions*, 2020. arXiv: `2011.06801 [cs.SD]`.

[315] A. van den Oord, S. Dieleman, H. Zen, *et al.*, *WaveNet: A Generative Model for Raw Audio*, 2016. arXiv: `1609.03499 [cs.SD]`.

[316] R. Ardila, M. Branson, K. Davis, *et al.*, "Common Voice: A Massively-Multilingual Speech Corpus," in *Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020)*, 2020, pp. 4211–4215.

[317] M. Defferrard, K. Benzi, P. Vandergheynst, and X. Bresson, "FMA: A dataset for music analysis," in *18th International Society for Music Information Retrieval Conference (ISMIR)*, 2017. arXiv: `1612.01840`.

[318] K. Drossos, S. Lipping, and T. Virtanen, *Clotho dataset*, version 2.1, Zenodo, May 2021. DOI: `10.5281/zenodo.4783391`.

[319] A. L. Hodgkin and A. F. Huxley, "Currents carried by sodium and potassium ions through the membrane of the giant axon of Loligo," *The Journal of Physiology*, vol. 116, no. 4, pp. 449–472, 1952. DOI: `10.1113/jphysiol.1952.sp004717`.

[320] J. W. Pillow, J. Shlens, L. Paninski, *et al.*, "Spatio-temporal correlations and visual signalling in a complete neuronal population," *Nature 2008 454:7207*, vol. 454, pp. 995–999, 7207 Jul. 2008, ISSN: 1476-4687. DOI: `10.1038/NATURE07140`.

[321] R. S. Somerville and R. Davé, "Physical Models of Galaxy Formation in a Cosmological Framework," *Annual Review of Astronomy and Astrophysics*, vol. 53, no. 1, pp. 51–113, Aug. 2015, ISSN: 1545-4282. DOI: `10.1146/annurev-astro-082812-140951`.

[322] G. Pombo, R. Gray, M. J. Cardoso, *et al.*, "Equitable modelling of brain imaging by counterfactual augmentation with morphologically constrained 3D deep generative models," *Medical Image Analysis*, vol. 84, p. 102 723, 2023, ISSN: 1361-8415. DOI: `10.1016/j.media.2022.102723`.

[323] F. Urbina, C. T. Lowden, J. C. Culberson, and S. Ekins, "MegaSyn: Integrating Generative Molecular Design, Automated Analog Designer, and Synthetic Viability Prediction," *ACS Omega*, vol. 7, no. 22, pp. 18 699–18 713, 2022. DOI: `10.1021/acsomega.2c01404`.

[324] C. Rommel, J. Paillard, T. Moreau, and A. Gramfort, "Data augmentation for learning predictive models on EEG: a systematic comparison," *Journal of Neural Engineering*, vol. 19, no. 6, p. 066 020, Nov. 2022, ISSN: 1741-2552. DOI: `10.1088/1741-2552/aca220`.

[325] J. Vetter, J. H. Macke, and R. Gao, "Generating realistic neurophysiological time series with denoising diffusion probabilistic models," *Patterns*, vol. 5, no. 9, p. 101 047, 2024, ISSN: 2666-3899. DOI: `10.1016/j.patter.2024.101047`.

[326] A. Gaulton, A. Hersey, M. Nowotka, *et al.*, "The ChEMBL database in 2017," *Nucleic acids research*, vol. 45, no. D1, pp. D945–D954, 2017.

[327] T. M. Inc., *Communications Toolbox*, Natick, Massachusetts, United States, 2024. [Online]. Available: `https://www.mathworks.com`.

[328] J. Hoydis, S. Cammerer, F. A. Aoudia, *et al.*, *Sionna: An Open-Source Library for Next-Generation Physical Layer Research*, 2023. arXiv: `2203.11854 [cs.IT]`.

[329] H. Jiang, M. Cui, D. W. K. Ng, and L. Dai, "Accurate Channel Prediction Based on Transformer: Making Mobility Negligible," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 9, pp. 2717–2732, 2022. DOI: `10.1109/JSAC.2022.3191334`.

[330] Q. Zhang, A. Ferdowsi, and W. Saad, "Distributed Generative Adversarial Networks for mmWave Channel Modeling in Wireless UAV Networks," in *ICC 2021 - IEEE International Conference on Communications*, 2021, pp. 1–6. DOI: `10.1109/ICC42927.2021.9501056`.

[331] D. Uvaydov, S. D'Oro, F. Restuccia, and T. Melodia, "Deepsense: Fast wideband spectrum sensing through real-time in-the-loop deep learning," in *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*, IEEE, 2021, pp. 1–10.

[332] 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.

[333] D. Roy, V. Chaudhury, C. Tassie, C. Spooner, and K. Chowdhury, "ICARUS: Learning on IQ and Cycle Frequencies for Detecting Anomalous RF Underlay Signals," in *IEEE International Conference on Computer Communications (INFOCOM)*, IEEE, 2020.

[334] C. Tassie, A. Gaber, V. Chaudhary, *et al.*, "Detection of Co- Existing RF Signals in CBRS Using ML: Dataset and API-Based Collection Testbed," *IEEE Communications Magazine*, vol. 61, no. 9, pp. 82–88, 2023. DOI: `10.1109/MCOM.002.2200682`.

[335] G. Reus-Muns, D. Jaisinghani, K. Sankhe, and K. Chowdhury, "Trust in 5G Open RANs through Machine Learning: RF Fingerprinting on the POWDER PAWR Platform," in *IEEE Globecom 2020-IEEE Global Communications Conference*, IEEE, 2020.

[336] M. Polese, F. Restuccia, and T. Melodia, "DeepBeam: Deep Waveform Learning for Coordination-Free Beam Management in mmWave Networks," *Proc. of ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2021.

[337] A. Al-Shawabka, P. Pietraski, S. B. Pattar, F. Restuccia, and T. Melodia, "DeepLoRa: Fingerprinting LoRa Devices at Scale Through Deep Learning and Data Augmentation," in *Proceedings of the Twenty-Second International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing*, ser. MobiHoc '21, Shanghai, China: Association for Computing Machinery, 2021, pp. 251–260, ISBN: 9781450385589. DOI: `10.1145/3466772.3467054`. [Online]. Available: `https://doi.org/10.1145/3466772.3467054`.

[338] M. Piva, G. Maselli, and F. Restuccia, "The tags are alright: Robust large-scale RFID clone detection through federated data-augmented radio fingerprinting," *CoRR*, vol. abs/2105.03671, 2021. arXiv: `2105.03671`. [Online]. Available: `https://arxiv.org/abs/2105.03671`.

[339] N. Soltani, G. Reus-Muns, B. Salehi, D. Jennifer, S. Ioannidis, and K. Chowdhury, "RF Fingerprinting Unmanned Aerial Vehicles with Non-standard Transmitter Waveforms," 2020.

[340] S. Mohanti, N. Soltani, K. Sankhe, D. Jaisinghani, M. Di Felice, and K. Chowdhury, "AirID: Injecting a Custom RF Fingerprint for enhanced UAV Identification using Deep Learning," in *IEEE GLOBECOM 2020-IEEE Global Communications Conference*, IEEE, 2020, pp. 370–378.

[341] J. Miraglia, "Signal Discovery with Convolutional Neural Nets," The University of Utah, Tech. Rep., 2022.

[342] X. Du and A. Sabharwal, "Massive MIMO Channels with Inter-User Angle Correlation: Open-Access Dataset, Analysis and Measurement-Based Validation," *IEEE Transactions on Vehicular Technology*, pp. 1–1, 2021. DOI: `10.1109/TVT.2021.3131606`.

[343] X. Zhang, L. Zhong, and A. Sabharwal, "Directional Training for FDD Massive MIMO," *IEEE Transactions on Wireless Communications*, vol. 17, no. 8, pp. 5183–5197, 2018. DOI: `10.1109/TWC.2018.2838600`.

[344] R. Doost-Mohammady, O. Bejarano, L. Zhong, *et al.*, "RENEW: Programmable and Observable Massive MIMO Networks," in *2018 52nd Asilomar Conference on Signals, Systems, and Computers*, 2018, pp. 1654–1658. DOI: `10.1109/ACSSC.2018.8645391`.

[345] Z. Chen, C. N. Barati, J. Veihl, C. Shepard, and A. Sabharwal, "LensFD: Using Lenses for Improved Sub-6 GHz Massive MIMO Full-Duplex," *IEEE Transactions on Vehicular Technology*, pp. 1–13, 2023. DOI: `10.1109/TVT.2023.3240558`.

[346] Q. An, S. Segarra, C. Dick, A. Sabharwal, and R. Doost-Mohammady, "A Deep Reinforcement Learning-Based Resource Scheduler for Massive MIMO Networks," *IEEE Transactions on Machine Learning in Communications and Networking*, vol. 1, pp. 242–257, 2023. DOI: `10.1109/TMLCN.2023.3313988`.

[347] T. Rice, "Experimental evaluation of AoA estimation for UAV to massive MIMO," Rice University, Tech. Rep., 2022.

[348] C. Shepard, J. Ding, R. E. Guerra, and L. Zhong, "Understanding real many-antenna MU-MIMO channels," pp. 461–467, 2016.

[349] L. Bonati, M. Polese, S. D'Oro, S. Basagni, and T. Melodia, "OpenRAN Gym: AI/ML Development, Data Collection, and Testing for O-RAN on PAWR Platforms," *Computer Networks*, vol. 220, pp. 1–11, Jan. 2023.

[350] J. Breen, A. Buffmire, J. Duerig, *et al.*, "Powder: Platform for Open Wireless Data-driven Experimental Research," *Computer Networks*, vol. 197, p. 108 281, 2021, ISSN: 1389-1286. DOI: 10.1016/j.comnet.2021.108281.

[351] LAION e.V., *Releasing Re-LAION 5B: Transparent Iteration on LAION-5B with Additional Safety Fixes*, Aug. 2024. [Online]. Available: https://laion.ai/blog/relaion-5b/.

[352] C. Schuhmann, R. Beaumont, R. Vencu, *et al.*, *LAION-5B: An open large-scale dataset for training next generation image-text models*, 2022. arXiv: 2210.08402 [cs.CV].

[353] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, *High-Resolution Image Synthesis with Latent Diffusion Models*, 2022. arXiv: 2112.10752 [cs.CV].

[354] *KYLLO V. UNITED STATES*. Jun. 2001. [Online]. Available: https://www.law.cornell.edu/supct/html/99-8508.ZO.html.

[355] M. Smolacks, "Google pays out $13m to make Wi-Spy scandal go away: Bung goes to peeps and privacy orgs," *The Register*, Jul. 23, 2019. [Online]. Available: https://www.theregister.com/2019/07/23/google_wispy_payout/.

[356] Y. Zhao, J. T. Du, and J. Chen, "Scenario-based Adaptations of Differential Privacy: A Technical Survey," *ACM Comput. Surv.*, vol. 56, no. 8, Apr. 2024, ISSN: 0360-0300. DOI: 10.1145/3651153.

[357] Y. Mao, Q. Ye, Q. Wang, and H. Hu, "Differential Privacy for Time Series: A Survey," *Data Engineering*, p. 68, 2023.

[358] J. Fontaine, A. Shahid, and E. D. Poorter, *Towards a Wireless Physical-Layer Foundation Model: Challenges and Strategies*, 2024. arXiv: 2403.12065 [cs.NI].

[359] S. Gunasekar, Y. Zhang, J. Aneja, *et al.*, *Textbooks Are All You Need*, 2023. arXiv: 2306.11644 [cs.CL].

[360] G. Hinton, O. Vinyals, and J. Dean, *Distilling the Knowledge in a Neural Network*, 2015. arXiv: 1503.02531 [stat.ML].

[361] J. Kaddour and Q. Liu, *Synthetic Data Generation in Low-Resource Settings via Fine-Tuning of Large Language Models*, 2024. arXiv: 2310.01119 [cs.CL].

[362] J. Bourne, *Scrambled text: training Language Models to correct OCR errors using synthetic data*, 2024. arXiv: 2409.19735 [cs.CL].

[363] D. Cifarelli, L. Boiardi, A. Puppo, and L. Jovanovic, *Safurai-Csharp: Harnessing Synthetic Data to improve language-specific Code LLM*, 2023. arXiv: 2311.03243 [cs.CL].

[364] A. Mumuni, F. Mumuni, and N. K. Gerrar, "A Survey of Synthetic Data Augmentation Methods in Machine Vision," *Machine Intelligence Research*, vol. 21, no. 5, pp. 831–869, Mar. 2024, ISSN: 2731-5398. DOI: 10.1007/s11633-022-1411-7.

[365] I. Shumailov, Z. Shumaylov, Y. Zhao, Y. Gal, N. Papernot, and R. Anderson, *The Curse of Recursion: Training on Generated Data Makes Models Forget*, 2024. arXiv: `2305.17493 [cs.LG]`.

[366] M. Briesch, D. Sobania, and F. Rothlauf, *Large Language Models Suffer From Their Own Output: An Analysis of the Self-Consuming Training Loop*, 2024. arXiv: `2311.16822 [cs.LG]`.

[367] E. Dohmatob, Y. Feng, P. Yang, F. Charton, and J. Kempe, *A Tale of Tails: Model Collapse as a Change of Scaling Laws*, 2024. arXiv: `2402.07043 [cs.LG]`.

[368] M. E. A. Seddik, S.-W. Chen, S. Hayou, P. Youssef, and M. Debbah, *How Bad is Training on Synthetic Data? A Statistical Analysis of Language Model Collapse*, 2024. arXiv: `2404.05090 [cs.LG]`.

[369] S. Forsgren and H. Martiros, "Riffusion - Stable diffusion for real-time music generation," 2022. [Online]. Available: `https://riffusion.com/about`.

[370] G. J. J. van den Burg and C. K. I. Williams, *On Memorization in Probabilistic Deep Generative Models*, 2021. arXiv: `2106.03216 [cs.LG]`.

[371] J. Sanz, A. Bastani, and R. Kaveh, "Generative Modeling for Dataset Anonymization."

[372] B. Kaur, D. Singh, and P. P. Roy, "Age and gender classification using brain–computer interface," *Neural Computing and Applications*, vol. 31, no. 10, pp. 5887–5900, Mar. 2018, ISSN: 1433-3058. DOI: `10.1007/s00521-018-3397-1`.

[373] P. Kaushik, A. Gupta, P. P. Roy, and D. P. Dogra, "EEG-Based Age and Gender Prediction Using Deep BLSTM-LSTM Network Model," *IEEE Sensors Journal*, vol. 19, no. 7, pp. 2634–2641, 2019. DOI: `10.1109/JSEN.2018.2885582`.

[374] S. L. Oh, Y. Hagiwara, U. Raghavendra, *et al.*, "A deep learning approach for Parkinson's disease diagnosis from EEG signals," *Neural Computing and Applications*, vol. 32, pp. 10 927–10 933, 2020.

[375] A. M. Maitin, J. P. Romero Muñoz, and Á. J. García-Tejedor, "Survey of Machine Learning Techniques in the Analysis of EEG Signals for Parkinson's Disease: A Systematic Review," *Applied Sciences*, vol. 12, no. 14, 2022, ISSN: 2076-3417. DOI: `10.3390/app12146967`.

[376] Z. Chen, D. Geng, and A. Owens, "Images that Sound: Composing Images and Sounds on a Single Canvas," *arXiv preprint*, 2024. eprint: `2405.12221`.

[377] N. Perraudin, P. Balazs, and P. L. Søndergaard, "A fast Griffin-Lim algorithm," in *2013 IEEE workshop on applications of signal processing to audio and acoustics*, IEEE, 2013, pp. 1–4.

[378] A. Gonzales, G. Guruswamy, and S. R. Smith, "Synthetic data in health care: A narrative review," *PLOS Digital Health*, vol. 2, no. 1, e0000082, 2023.

[379] M. D. Malik and D. Humair, *Evaluating the feasibility of using Generative Models to generate Chest X-Ray Data*, 2023. arXiv: `2305.18927 [eess.IV]`.

[380] B. Segal, D. M. Rubin, G. Rubin, and A. Pantanowitz, "Evaluating the Clinical Realism of Synthetic Chest X-Rays Generated Using Progressively Growing GANs," *SN Computer Science*, vol. 2, no. 4, Jun. 2021, ISSN: 2661-8907. DOI: `10.1007/s42979-021-00720-7`.

[381] T. Weber, M. Ingrisch, B. Bischl, and D. Rügamer, *Cascaded Latent Diffusion Models for High-Resolution Chest X-ray Synthesis*, 2023. arXiv: `2303.11224 [eess.IV]`.

[382] M. Tajmirriahi, R. Kafieh, Z. Amini, and V. Lakshminarayanan, "A Dual-Discriminator Fourier Acquisitive GAN for Generating Retinal Optical Coherence Tomography Images," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–8, 2022. DOI: `10.1109/TIM.2022.3189735`.

[383] A. Segato, V. Corbetta, M. D. Marzo, L. Pozzi, and E. De Momi, "Data Augmentation of 3D Brain Environment Using Deep Convolutional Refined Auto-Encoding Alpha GAN," *IEEE Transactions on Medical Robotics and Bionics*, vol. 3, no. 1, pp. 269–272, 2021. DOI: `10.1109/TMRB.2020.3045230`.

[384] Z. Zheng and J. Zhang, *FD-Vision Mamba for Endoscopic Exposure Correction*, 2024. arXiv: `2402.06378 [cs.CV]`.

[385] A. Gu and T. Dao, *Mamba: Linear-Time Sequence Modeling with Selective State Spaces*, 2024. arXiv: `2312.00752 [cs.LG]`.

[386] J. Vetter, J. H. Macke, and R. Gao, "Generating realistic neurophysiological time series with denoising diffusion probabilistic models," *bioRxiv*, 2023. DOI: `10.1101/2023.08.23.554148`.

[387] A. Lugmayr, M. Danelljan, A. Romero, F. Yu, R. Timofte, and L. V. Gool, *RePaint: Inpainting using Denoising Diffusion Probabilistic Models*, 2022. arXiv: `2201.09865 [cs.CV]`.

[388] G. Tosato, C. M. Dalbagno, and F. Fumagalli, *EEG Synthetic Data Generation Using Probabilistic Diffusion Models*, 2023. arXiv: `2303.06068 [eess.SP]`.

[389] S. Bischoff, A. Darcher, M. Deistler, *et al.*, *A Practical Guide to Statistical Distances for Evaluating Generative Models in Science*, 2024. arXiv: `2403.12636 [cs.LG]`.

[390] M. J. Chong and D. Forsyth, *Effectively Unbiased FID and Inception Score and where to find them*, 2020. arXiv: `1911.07023 [cs.CV]`.

[391] M. Oquab, T. Darcet, T. Moutakanni, *et al.*, *DINOv2: Learning Robust Visual Features without Supervision*, 2024. arXiv: `2304.07193 [cs.CV]`.

[392] F. Lotte, L. Bougrain, A. Cichocki, *et al.*, "A review of classification algorithms for EEG-based brain–computer interfaces: a 10 year update," *Journal of Neural Engineering*, vol. 15, no. 3, p. 031 005, Apr. 2018, ISSN: 1741-2552. DOI: `10.1088/1741-2552/aab2f2`.

[393] A. Défossez, C. Caucheteux, J. Rapin, O. Kabeli, and J.-R. King, "Decoding speech perception from non-invasive brain recordings," *Nature Machine Intelligence*, vol. 5, no. 10, pp. 1097–1107, Oct. 2023, ISSN: 2522-5839. DOI: `10.1038/s42256-023-00714-5`.

[394] L. S. Vidyaratne and K. M. Iftekharuddin, "Real-Time Epileptic Seizure Detection Using EEG," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 25, no. 11, pp. 2146–2156, 2017. DOI: `10.1109/TNSRE.2017.2697920`.

[395] W. Liu, J.-L. Qiu, W.-L. Zheng, and B.-L. Lu, "Comparing Recognition Performance and Robustness of Multimodal Deep Learning Models for Multimodal Emotion Recognition," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 14, no. 2, pp. 715–729, 2022. DOI: `10.1109/TCDS.2021.3071170`.

[396] T. Zhu, W. Luo, and F. Yu, "Convolution-and attention-based neural network for automated sleep stage classification," en, *Int. J. Environ. Res. Public Health*, vol. 17, no. 11, p. 4152, Jun. 2020.

[397] G. Siddhad, A. Gupta, D. P. Dogra, and P. P. Roy, "Efficacy of transformer networks for classification of EEG data," *Biomedical Signal Processing and Control*, vol. 87, p. 105 488, Jan. 2024, ISSN: 1746-8094. DOI: `10.1016/j.bspc.2023.105488`.

[398] P. Nguyen, D. Tran, T. Vo, X. Huang, W. Ma, and D. Phung, "EEG-Based Age and Gender Recognition Using Tensor Decomposition and Speech Features," in *Neural Information Processing*, M. Lee, A. Hirose, Z.-G. Hou, and R. M. Kil, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 632–639, ISBN: 978-3-642-42042-9.

[399] W. Liu, J.-L. Qiu, W.-L. Zheng, and B.-L. Lu, "Comparing Recognition Performance and Robustness of Multimodal Deep Learning Models for Multimodal Emotion Recognition," *IEEE Transactions on Cognitive and Developmental Systems*, 2021.

[400] A. Singh, R. Cole, A. Espinoza, J. Cavanagh, and N. Narayanan, *"Rest eyes open"*, OpenNeuro, 2023. DOI: `doi:10.18112/openneuro.ds004584.v1.0.0`.

[401] A. A. Abdulsatar, V. V. Davydov, V. V. Yushkova, A. P. Glinushkin, and V. Y. Rud, "Age and gender recognition from speech signals," *Journal of Physics: Conference Series*, vol. 1410, no. 1, p. 012 073, Dec. 2019. DOI: `10.1088/1742-6596/1410/1/012073`.

[402] M. Tröbs and G. Heinzel, "Improved spectrum estimation from digitized time series on a logarithmic frequency axis," *Measurement*, vol. 39, no. 2, pp. 120–129, 2006, ISSN: 0263-2241. DOI: `10.1016/j.measurement.2005.10.010`.

[403] S. A. Fulop and K. Fitz, "A spectrogram for the twenty-first century," *Acoustics Today*, vol. 2, no. 3, pp. 26–33, 2006.

[404] Z. Yang, Z. Wu, M. Luo, *et al.*, “SkyPilot: An intercloud broker for sky computing,” in *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*, Boston, MA: USENIX Association, Apr. 2023, pp. 437–455, ISBN: 978-1-939133-33-5. [Online]. Available: `https://www.usenix.org/conference/nsdi23/presentation/yang-zongheng`.

[405] J. Song, C. Meng, and S. Ermon, *Denoising Diffusion Implicit Models*, 2022. arXiv: `2010.02502 [cs.LG]`.

[406] D. Morales-Brotons, T. Vogels, and H. Hendrikx, “Exponential Moving Average of Weights in Deep Learning: Dynamics and Benefits,” *Transactions on Machine Learning Research*, 2024, ISSN: 2835-8856. arXiv: `2411.18704`.

[407] Z. Fei, M. Fan, C. Yu, and J. Huang, *Scalable Diffusion Models with State Space Backbone*, 2024. arXiv: `2402.05608 [cs.CV]`.

[408] V. T. Hu, S. A. Baumann, M. Gui, *et al.*, “ZigMa: A DiT-style Zigzag Mamba Diffusion Model,” in *ECCV*, 2024.

[409] Y. Liu, Y. Tian, Y. Zhao, *et al.*, *VMamba: Visual State Space Model*, 2024. arXiv: `2401.10166 [cs.CV]`.

[410] Z. Wang, J.-Q. Zheng, Y. Zhang, G. Cui, and L. Li, *Mamba-UNet: UNet-Like Pure Visual Mamba for Medical Image Segmentation*, 2024. arXiv: `2402.05079 [eess.IV]`.

[411] J. Liu, H. Yang, H.-Y. Zhou, *et al.*, *Swin-UMamba: Mamba-based UNet with ImageNet-based pretraining*, 2024. arXiv: `2402.03302 [eess.IV]`.

[412] J. Ho, A. Jain, and P. Abbeel, “Denoising Diffusion Probabilistic Models,” 2020. arXiv: `2006.11239`.

[413] J. Ho and T. Salimans, *Classifier-Free Diffusion Guidance*, 2022. arXiv: `2207.12598 [cs.LG]`.

[414] A. Obukhov, M. Seitzer, P.-W. Wu, S. Zhydenko, J. Kyl, and E. Y.-J. Lin, *High-fidelity performance metrics for generative models in PyTorch*, version v0.3.0, Version: 0.3.0, DOI: 10.5281/zenodo.4957738, 2020. DOI: `10.5281/zenodo.4957738`. [Online]. Available: `https://github.com/toshas/torch-fidelity`.

[415] D. Hendrycks and K. Gimpel, *Gaussian Error Linear Units (GELUs)*, 2023. arXiv: `1606.08415 [cs.LG]`.

[416] R. Müller, S. Kornblith, and G. Hinton, *When Does Label Smoothing Help?* 2020. arXiv: `1906.02629 [cs.LG]`.

[417] D. Donoho, *Data Science at the Singularity*, 2023. arXiv: `2310.00865 [stat.OT]`.

[418] D. Hong, B. Zhang, X. Li, *et al.*, “SpectralGPT: Spectral Remote Sensing Foundation Model,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 8, pp. 5227–5244, Aug. 2024, ISSN: 1939-3539. DOI: `10.1109/tpami.2024.3362475`.

[419] L. Pang, X. Cao, D. Tang, *et al.*, *HSIGene: A Foundation Model For Hyperspectral Image Generation*, 2024. arXiv: `2409.12470 [cs.CV]`.

[420] N. Chen, J. Yue, L. Fang, and S. Xia, "SpectralDiff: A Generative Framework for Hyperspectral Image Classification With Diffusion Models," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 61, pp. 1–16, 2023, ISSN: 1558-0644. DOI: `10.1109/tgrs.2023.3310023`.

[421] N. A. A. Braham, C. M. Albrecht, J. Mairal, J. Chanussot, Y. Wang, and X. X. Zhu, *SpectralEarth: Training Hyperspectral Foundation Models at Scale*, 2024. arXiv: `2408.08447 [cs.CV]`.

[422] L. Zhang, A. Rao, and M. Agrawala, *Adding Conditional Control to Text-to-Image Diffusion Models*, 2023. arXiv: `2302.05543 [cs.CV]`.

[423] A. Bansal, H.-M. Chu, A. Schwarzschild, *et al.*, *Universal Guidance for Diffusion Models*, 2023. arXiv: `2302.07121 [cs.CV]`.

[424] P. Dhariwal and A. Nichol, *Diffusion Models Beat GANs on Image Synthesis*, 2021. arXiv: `2105.05233 [cs.LG]`.

[425] D. Tang, X. Cao, X. Hou, Z. Jiang, J. Liu, and D. Meng, *CRS-Diff: Controllable Remote Sensing Image Generation with Diffusion Model*, 2024. arXiv: `2403.11614 [cs.CV]`.

[426] A. Grattafiori, A. Dubey, A. Jauhri, *et al.*, *The Llama 3 Herd of Models*, 2024. arXiv: `2407.21783 [cs.AI]`.

[427] R. Rafailov, A. Sharma, E. Mitchell, S. Ermon, C. D. Manning, and C. Finn, *Direct Preference Optimization: Your Language Model is Secretly a Reward Model*, 2024. arXiv: `2305.18290 [cs.LG]`.

[428] A. Gholami, S. Kim, Z. Dong, Z. Yao, M. W. Mahoney, and K. Keutzer, *A Survey of Quantization Methods for Efficient Neural Network Inference*, 2021. arXiv: `2103.13630 [cs.CV]`.

[429] H. Cheng, M. Zhang, and J. Q. Shi, "A Survey on Deep Neural Network Pruning: Taxonomy, Comparison, Analysis, and Recommendations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 12, pp. 10 558–10 578, 2024. DOI: `10.1109/TPAMI.2024.3447085`.

[430] M. Tan and Q. V. Le, *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks*, 2020. arXiv: `1905.11946 [cs.LG]`.

[431] T. Arora, "Accelerating Machine Learning Compute for the IoT and Embedded Market," Arm, Tech. Rep., 2020.

[432] Google, *A platform from Google for local AI*, 2024. [Online]. Available: `https://coral.ai/about-coral/`.

[433] J. Butts, *What Is the Apple Neural Engine and What Does It Do?* 2024. [Online]. Available: `https://www.macobserver.com/tips/what-is-apple-neural-engine/` (visited on 12/19/2024).

[434] "Unlocking on-device generative AI with an NPU and heterogeneous computing," Qualcomm, Tech. Rep., Feb. 2024.

[435] C. Su, "NVIDIA Unveils Its Most Affordable Generative AI Supercomputer," Dec. 2024. [Online]. Available: `https://blogs.nvidia.com/blog/jetson-generative-ai-supercomputer/`.

[436] G. A. E. Team, "TensorFlow Lite is now LiteRT," Sep. 2024. [Online]. Available: `https://developers.googleblog.com/en/tensorflow-lite-is-now-litert/`.

[437] H. Cai, C. Gan, L. Zhu, and S. Han, "TinyTL: Reduce Memory, Not Parameters for Efficient On-Device Learning," in *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[438] J. Lin, W. Chen, Y. Lin, J. Cohn, C. Gan, and S. Han, "MCUNet: Tiny Deep Learning on IoT Devices," *CoRR*, 2020. arXiv: `2007.10319`.

[439] H. B. Pasandi and T. Nadeem, "Autonomous On-Device Protocols: Empowering Wireless with Self-Driven Capabilities," in *2024 IEEE Wireless Communications and Networking Conference (WCNC)*, 2024, pp. 1–6. DOI: `10.1109/WCNC57260.2024.10571037`.

[440] H. Barahouei Pasandi, F. Pasandi, F. Parastar, A. Moradbeikie, and T. Nadeem, *Echoing the Future: On-Device Machine Learning in Next-Generation Networks -A Comprehensive Survey*, May 2023.

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

Appendix A

Code for Echo Protocol

Code for the Echo protocol, simulation environment, and experiment runs can be found at `https://github.com/ml4wireless/echo` in the *ieee-paper* branch. Code for the GNU Radio implementation of the Echo protocol can be found at `https://github.com/ml4wireless/gr-echo` [195].

Appendix B

Detailed Agent Descriptions

B.1 Classic

Modulator – The modulator uses a fixed strategy known to be optimal for AWGN channels (e.g., Gray coded QPSK for 2 bits per symbol, 8PSK for 4 bits per symbol, and 16QAM for 4 bits per symbol) [6].

Demodulator – The demodulator uses the 1 nearest-neighbor method to return the closest neighbor from the constellation of the corresponding optimal modulator. Essentially, the demodulator partitions the complex plane into different regions and demodulates based on which region the input to the demodulator lies in. When using classic demodulation schemes for the GP protocol, we require that the output be differentiable, and here we output probabilities for each symbol by taking a softmax of the squared distance of the point to each symbol from the optimal constellation.

B.2 Neural

First, we describe parameter settings that are common to both modulators and demodulators:

Network architecture: We use one layer networks with fully connected layers with the 'tanh' activation. Input and output sizes differ for the modulator and demodulator, as described below.

Initialization: The weights for each layer are initialized by sampling from the distribution $U[\frac{-1}{\sqrt{n}}, \frac{1}{\sqrt{n}}]$, where n is the number of input units to the layer; the biases are initialized as 0.01.

Optimizer: We use the Adam optimizer [441].

Next, we describe the modulator and demodulator-specific parameters and provide details about their update methods. For the rest of the section, let b denote bits per symbol (equivalently, the modulation order).

Modulator

Input width: b. We take in input in bit format (but treat these 0-1 values as floats).

Output width: 2. The output width is fixed since it represents a complex number to be sent over the channel.

Parameters: In addition to the network weights and biases, θ, we also include a separate learned parameter σ, a scalar denoting the standard deviation of the Gaussian distribution we sample from for our policy.

Modulation procedure: The neural net outputs μ. Here, μ is the output of the neural network and is the mean of the Gaussian distribution that we sample from. Note that if the input is of size $[N, b]$, μ will have size $[N, 2]$ (first dimension corresponding to the real part of a complex number and the other corresponding to the imaginary part of the complex number). While training, the modulator outputs symbols s sampled from a Gaussian distribution with mean μ and standard deviation σ (σ is bounded by minimum and maximum values.), i.e., $s \sim \mathcal{N}(\mu, \sigma^2 I)$.

Update procedure: Suppose for our given actions s we receive the reward r, the negative of the number of incorrect bits (comparing the original bit sequence to the received echo). The log probability for each action is given by,

$$\log p_i = -C \frac{(s_i - \mu_i)^2}{2\sigma^2}, \tag{B.1}$$

for some constant C. The loss function we minimize is given by,

$$l_{pg} = -\left(\sum_i \log p_i * r_i \right), \tag{B.2}$$

In some settings we modify the reward r to include penalty terms, such as one for the distance of average output from the origin as detailed in Appendix D. We update our parameters as,

$$\theta \leftarrow \theta + \text{Adam update}(\theta, \eta_\mu, l_{pg}) \tag{B.3}$$

$$\sigma \leftarrow \sigma + \text{Adam update}(\sigma, \eta_\sigma, l_{pg}), \tag{B.4}$$

where η_μ and η_σ denote the separate learning rate parameters for the network parameters and the standard deviation σ.

Demodulator

Input width: 2

Output width: 2^b. The demodulator is a classifier that outputs logits for each class that, on application of the softmax layer, correspond to the probabilities of the classes. The classes are the set of possible bit sequences for the modulation order.

Parameters: The network weights and biases denoted as ϕ.

Demodulation procedure: Given input of size $[N, 2]$, the neural net outputs logits (logits) of shape $[N, 2^b]$. On applying the softmax operation these correspond to a probability distribution over classes. The demodulated symbols, $\hat{p}$, are computed by choosing the class with the highest probability,

$$\hat{p} = \arg\max(\mathsf{softmax}(\mathsf{logits}))). \tag{B.5}$$

Update procedure: Suppose after applying the softmax layer, we have probability $q_{i,c}$ corresponding to the true class label of symbol i, $i = 1 \dots N$. We compute the cross-entropy loss as

$$l_{CE} = -\sum_i \log\left(\frac{\exp q_{i,c}}{\sum_{j=1}^{2^b} \exp q_{i,j}}\right) \tag{B.6}$$

We update our parameters as

$$\phi \leftarrow \phi - \text{Adam update}(\phi, \eta_\phi, l_{CE}) \tag{B.7}$$

where η_ϕ is the learning rate parameter for the demodulator updates.

B.3 Polynomial

First, we describe parameter settings that are common to both modulators and demodulators:

Network architecture: The inputs to the network are used to form a polynomial of degree d. We use a single fully connected linear layer to connect the polynomial terms to the output. Input and output sizes are different for the modulator and demodulator, as described below.

Initialization: The weights for each layer are initialized by sampling from the distribution $U[\frac{-1}{\sqrt{n}}, \frac{1}{\sqrt{n}}]$, where n is the number of input units to the layer; we do not use biases for polynomial agents.

Optimizer: We use the Adam optimizer [441].

Next, we describe the modulator and demodulator-specific parameters and provide details about their update methods. For the rest of the section, let b denote bits per symbol and d the degree of the polynomial.

Modulator

Input width: b. We take in input in bit format (but treat these 0-1 values as floats).

Output width: 2. The output width is fixed since it represents a complex number to be sent over the channel.

Parameters: Internally, the input bits are used to calculate all unique polynomial terms of order d. Since the bits b_i are in $\{0, 1\}$, terms including $b_i^2, b_i^3, \ldots$ are redundant and omitted from our calculations, thus allowing us to determine a unique maximum-degree polynomial. The polynomial terms are fed into the single fully connected layer with parameters θ. We also include a separate parameter σ, a scalar denoting the standard deviation of the Gaussian distribution we sample from for our policy.

Modulation procedure: The polynomial network outputs μ. Here, μ is the mean of the Gaussian distribution that we sample from. Note that if the input is of size $[N, b]$, μ will have size $[N, 2]$ (first dimension corresponding to the real part of a complex number and the other corresponding to the imaginary part of the complex number). While training, the modulator outputs symbols s sampled from a Gaussian distribution with mean μ and standard deviation σ, i.e. $s \sim \mathcal{N}(\mu, \sigma^2 I)$.

Update procedure: The update procedure for polynomial modulators is identical to the procedure for neural modulators.

Demodulator

Input width: 2

Output width: 2^b. The demodulator is a classifier that outputs logits for each class that, on application of the softmax layer, correspond to the probabilities of the classes. The classes are the set of possible bit sequences for the modulation order.

Parameters: Internally, the input symbols are used to calculate all unique polynomial terms of order d containing the real and imaginary parts of the symbol. For example,

$$P(s, 2) = \big[\mathrm{Re}\{s\}, \mathrm{Re}\{s\}^2, \mathrm{Im}\{s\}, \tag{B.8}$$

$$\mathrm{Im}\{s\}^2, \mathrm{Re}\{s\}\,\mathrm{Im}\{s\}\big]^\top \tag{B.9}$$

The polynomial terms are fed into the single fully connected layer with parameters ϕ.

Demodulation procedure: The demodulation procedure is the same as the neural agent.

Update procedure: The update procedure is the same as the neural agent, except for an $L1$ penalty added to the demodulator's loss term.

Appendix C

Additional Results

This appendix contains additional experimental results which, although not required to support our primary conclusions, we believe are of interest to anyone who wants to replicate or build upon our work. Appendix C.1 shows the effects of modulation order and training SNR on the performance of the ESP and EPP protocols with clone agents. Since any learning communications system in the wild will be exposed to multiple SNR conditions and desired signaling rates, understanding performance variation across SNR and modulation order will be crucial. Our results indicate that moderately high training SNR leads to the best performance, confirming observations by others ([23], [204]). Appendix C.2 presents experiments with Poly clone and self-alien agents demonstrating similar behavior to Neural clone and self-alien agents.

C.1 Effect of Modulation Order and Training Signal to Noise Ratio

In the experiments detailed in Sec. 2.7, we learned to modulate with 2 bits per symbol. Here, we explore whether the learning protocols continue to work for higher modulation orders, i.e., more bits per symbol. We conduct experiments using the EPP protocol for a Neural agent learning to communicate with a clone for 3 and 4 bits per symbol. We compare these cases and the 2 bits per symbol case in Figure C.1. From Figure C.1a, we observe that, at higher modulation orders, there is a larger gap between the BER curves of the learned agents and the corresponding baselines. Although some agents continue to approach the baseline BERs, as evidenced by the error bars, the median agent no longer achieves near-optimal performance at high SNRs. Figure C.1b shows that, for higher modulation orders, fewer trials learn a good modulation scheme, and it takes longer to learn good schemes. From Table C.1, we see that the increase in convergence times is exponential, with EPP requiring 2$\times$ and 24$\times$ more symbols for convergence for 8PSK and 16QAM, respectively. ESP requires 1.5$\times$ and 6$\times$ more symbols for convergence. Still, even for the highest modulation order examined (16QAM), 96% of trials eventually converge to a good scheme. This phenomenon of performance

Training SNR (BER)	**QPSK** (2 BPS)	**8PSK** (3 BPS)	**16QAM** (4 BPS)
ESP			
8.4 dB (1%)	25600	39936	152064
EPP			
4.2 dB (10%)	901120*	-	-
8.4 dB (1%)	115200 (404480*)	238080	2734592
13.0 dB (0.001%)	309760*	-	-

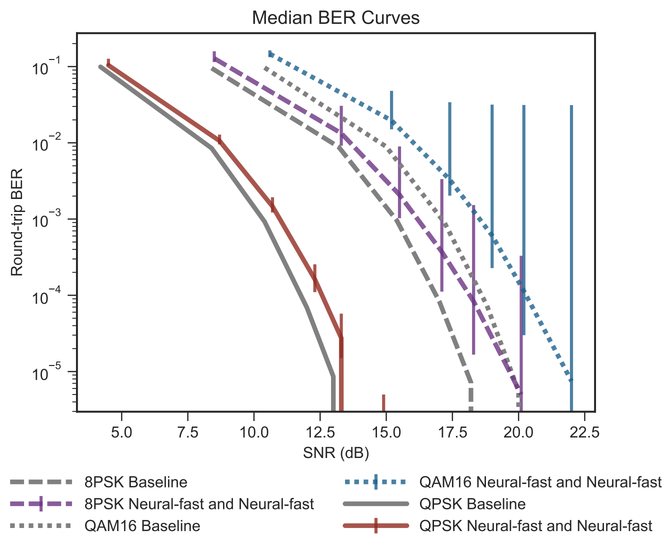
Table C.1: Number of symbols exchanged before $\geq 90\%$ of trials reached 3 dB off of optimal BER for the ESP and EPP protocols with Neural agents and varying bits per symbol (BPS) and SNR.
The results show the increased difficulty of learning at higher modulation orders. The EPP protocol is impacted much more than ESP by high modulation order, requiring 24$\times$ more symbols for 16QAM than QPSK compared to 6$\times$ for ESP. *The experiments comparing performance with training SNR were conducted using a different set of hyperparameters, found in Table G.8 in Appendix G. Lower training SNRs require longer to converge.

degradation with increasing modulation order is expected since the modulation functions for higher-order modulation schemes are more complex.

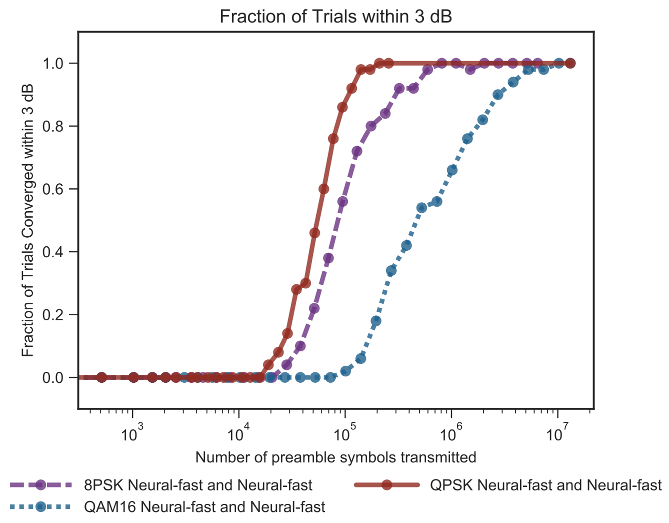
Next, we investigate the effect of training SNR. In all other experiments, we trained our agents at the SNR corresponding to 1% BER for the baseline scheme of the given modulation order. Is this the optimal SNR to train at? Does the learning protocol work at lower SNRs? We explore answers to these questions by conducting experiments using the EPP protocol for the setting where a Neural agent learns to communicate with a clone at various training SNRs corresponding to 0.001%, 0.1%, and 10% BERs for the baseline modulation scheme. From our results in Figure C.2, we observe that in all 3 settings, we achieve a BER close to the QPSK baseline. It takes longer to learn a modulation scheme at lower SNRs. However, not every trial converges when trained at high SNR. This can be explained by the regularization role that noise seems to have on our learning task. At very low SNRs, some trials fail to converge but those that do converge achieve similar BERs to agents trained at higher SNRs. It is possible that the agents are taking gradient steps that are too large and being forced into local minima by steps in poor directions caused by noisy feedback. This suggests the question of whether there is a "speed limit" to how fast agents can reliably (i.e., having all trials converge to within 3 dB-off-optimal) learn at a given SNR. We hope to answer this question in future work.

C.2 Polynomial Agent Experiments

Here, we include results using Poly agents, which demonstrate the same behaviors as the Neural agents in Section 2.7. Figures C.3 to C.5 demonstrate the effects of information sharing and that the EPP protocol works with fixed Classic and self-alien agents using polynomial function approximators. As with Neural agents, more information sharing leads to faster training. Similarly, Classic agents speed up training, and two self-alien agents converge at a rate in between their individual convergence speeds. Figure C.6 shows the performance of several combinations of Classic, clone, and self-alien agents using the ESP protocol. The relative ordering of performance is the same as when using EPP, even though each combination trains faster with ESP.



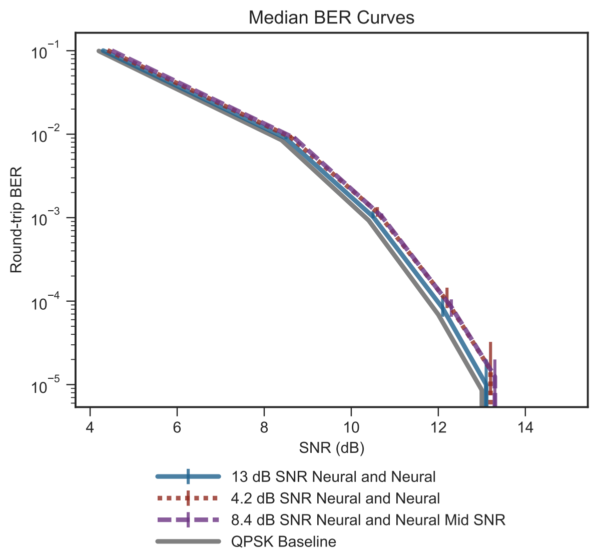
(a) Round-trip median BER curves for Neural agent learning with clone with mod orders (bits per symbol) 2, 3, 4 using the EPP protocol at training SNRs corresponding to 1% BER. Alongside the BER curves of the learned modulation schemes are the baseline QPSK (order 2), 8PSK(order 3), and 16QAM(order 4) curves. In all cases, modulation constellations are normalized to constrain the average signal power.



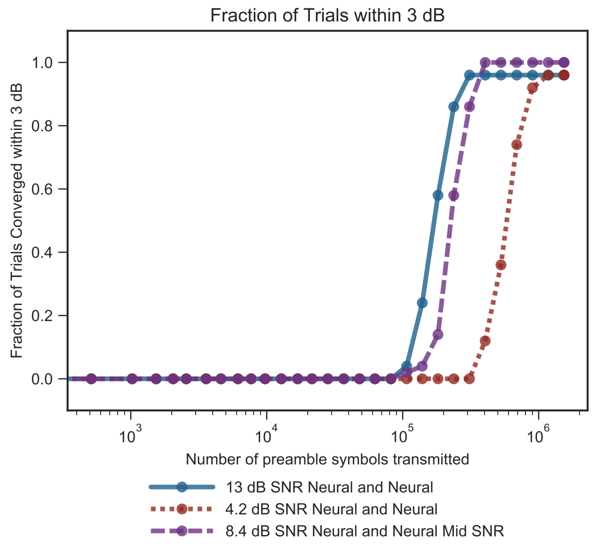
(b) Convergence of 50 trials to be within 3 dB (at testing SNR corresponding to 1% BER) of the corresponding baseline for EPP trials of at training SNR corresponding to 1% BER for increasing modulation order. 16QAM, with the highest modulation order 4, takes much longer to converge than QPSK (order 2) and 8PSK (order 3).

Figure C.1: Neural agents learning with a clone using the EPP protocol for different modulation orders.

The BER plot (a) shows that the gap between the median BER of the learned scheme and the corresponding baseline increases for higher mod orders. From the convergence plot (b), we observe that higher mod orders take exponentially longer to converge to a good strategy.

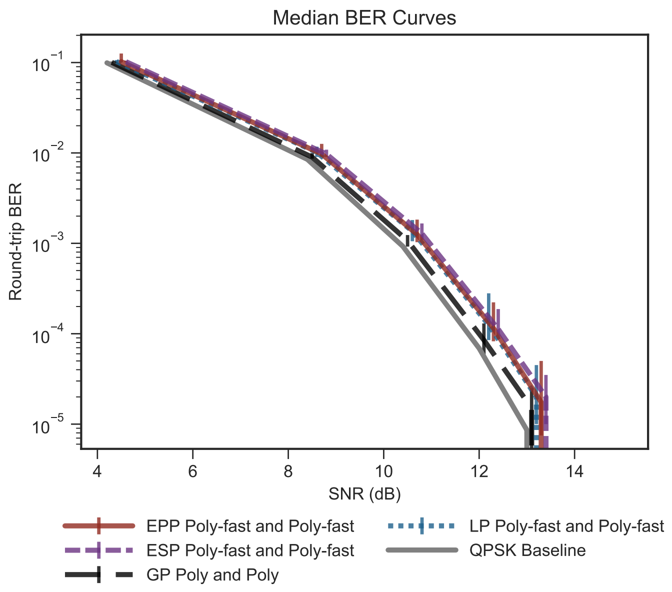


(a) Round-trip median BER curves for a Neural agent learning with a clone using the EPP protocol at training SNRs 13.0, 8.4, and 4.2 dB corresponding to 0.001%, 0.1%, and 10% BERs for the baseline. The error bars reflect the 10th to 90th percentiles across 50 trials. All agents are evaluated at the same SNR, but error bars have been dithered for readability.

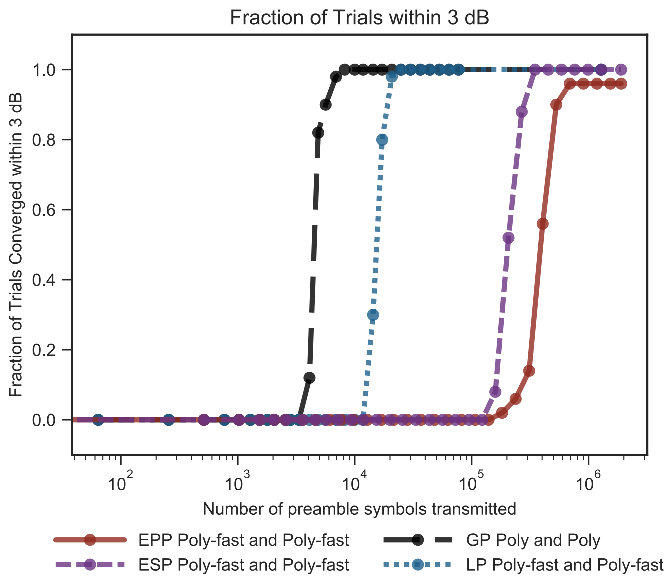


(b) Convergence of 50 trials to be within 3 dB at testing SNR 8.4 dB at training SNRs 13.0, 8.4, and 4.2 dB. Training at higher SNR reduces the number of symbols required for most trials to converge.

Figure C.2: Neural agents learning with a clone using EPP for different training signal to noise ratios.
The BER plot (a) shows that training at higher SNR leads to lower BERs across all SNRs. From the convergence plot (b), we observe that limited noise plays a regularizing effect, helping more trials to converge. Too much noise, however, has a detrimental effect and slows down convergence. Training at higher SNRs helps agents to converge more quickly, although not every trial converges.

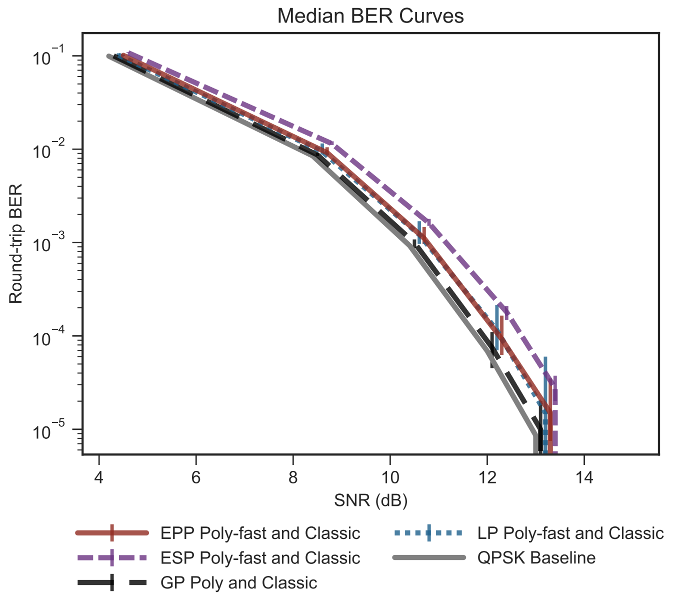


(a) Round-trip median BER. The error bars reflect the 10th to 90th percentiles across 50 trials. All agents are evaluated at the same SNR, but error bars have been dithered for readability.

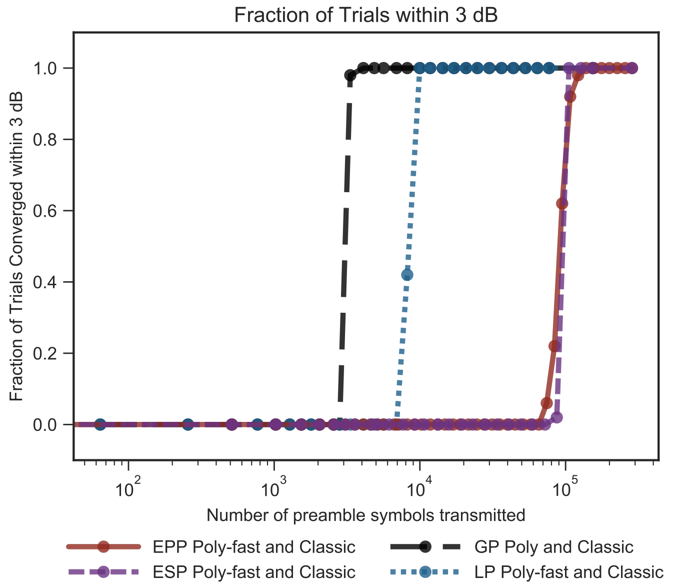


(b) Convergence of 50 trials to be within dB-off-optimalat 8.4 dB training SNR.

Figure C.3: Effect of Information Sharing while learning to communicate with a clone, Poly-fast-and-Poly-fast.
The BER plot (a) shows that all protocols achieve BER close to that of QPSK baseline. From the convergence plot (b), we observe that EPP is much slower than ESP, which is an order of magnitude slower than LP and GP. GP converges the fastest of all the protocols.

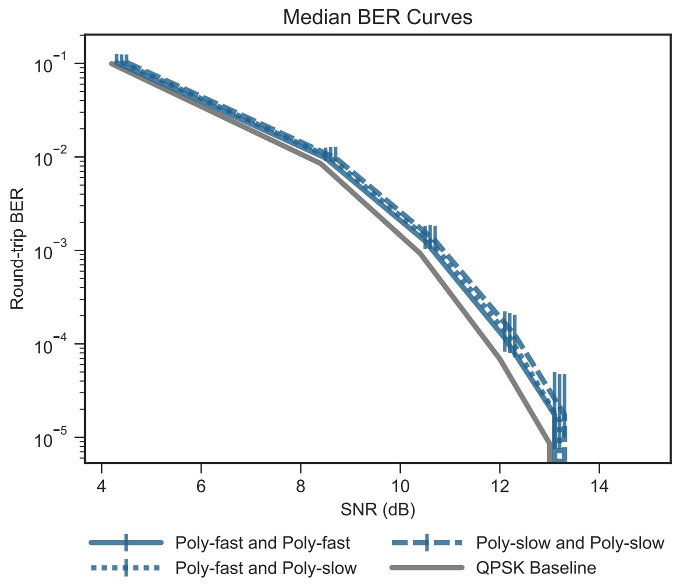


(a) Round-trip median BER. The error bars reflect the 10th to 90th percentiles across 50 trials. All agents are evaluated at the same SNR, but error bars have been dithered for readability.

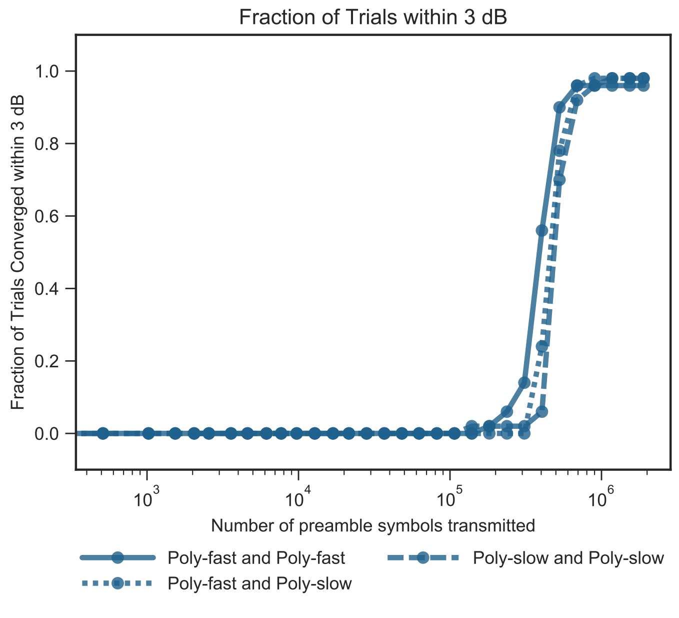


(b) Convergence of 50 trials to be within 3 dB at testing SNR 8.4 dB.

Figure C.4: Learning to communicate with a fixed agent, Poly-fast-and-Classic. The BER plot(a) shows that all protocols achieve BER close to the QPSK baseline. From the convergence plot (b), we observe that EPP and ESP have similar convergence behavior and are an order of magnitude slower than LP and GP. GP converges the fastest of all the protocols. Across all protocols, convergence is faster than learning with a clone (Figure: C.3).

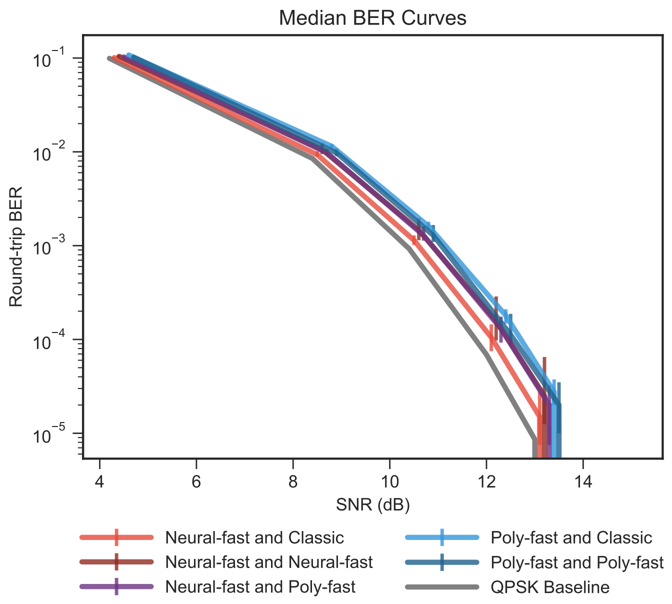


(a) Round-trip median BER. The error bars reflect the 10th to 90th percentiles across 50 trials. All agents are evaluated at the same SNR, but error bars have been dithered for readability.

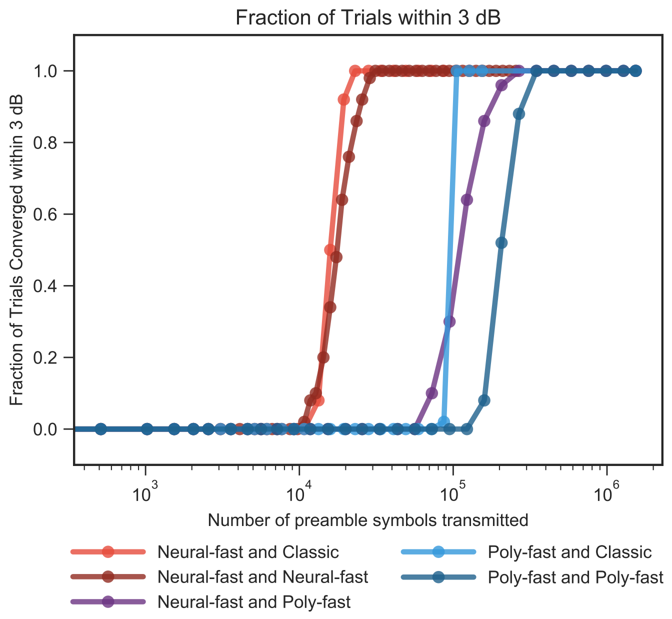


(b) Convergence of 50 trials to be within 3 dB at 8.4 dB training SNR.

Figure C.5: Learning with clones (Poly-fast-and-Poly-fast, Poly-slow-and-Poly-slow) compared to learning with self-alien (Poly-fast-and-Poly-slow) under the EPP protocol.
The BER plot (a) shows that the round-trip BER in all cases is almost identical and close to the QPSK baseline. From the convergence plot (b), we observe that Poly-fast-and-Poly-fast converges about twice as fast as Poly-slow-and-Poly-slow. The convergence time for the self-alien pairing, Poly-fast-and-Poly-slow, is in between the clone pairings; the Poly-fast agent helps the Poly-slow agent learn faster when paired together.



(a) Round-trip median BER. The error bars reflect the 10th to 90th percentiles across 50 trials. All agents are evaluated at the same SNR, but error bars have been dithered for readability.



(b) Convergence of 50 trials to be within 3 dB at testing SNR 8.4 dB.

Figure C.6: Learning under ESP protocol.

The BER plot (a) shows that all protocols achieve BER close to the QPSK baseline. From the convergence plot (b), we observe that learning with fixed agents is faster than learning with clones for both Neural and Poly agents. The alien pairing, Neural-fast and Poly-fast, has convergence times in between those of the individual clone pairings. The Neural agent helps the Poly agent learn faster when paired together.

Appendix D

Unsuccessful Constellation Centering Methods

We investigated several methods for forcing modulator constellations to be centered to avoid the training problems caused by DC offset correction in the USRP radios. Because not all of them worked, it is important to report the results for scientific integrity. The first method we investigated was to add a function, implemented in PyTorch, which calculated the center of the means output by the Gaussian policy and subtracted that value from the modulated symbols.

$$f_{\text{center}}(\boldsymbol{s}) = \boldsymbol{s} - \frac{1}{2^b} \sum_{i=1}^{2^b} \mu_i,$$

where μ_i are the means of each possible constellation point and $\boldsymbol{s}$ is the set of complex symbols modulated by the current policy.

The rate of successfully trained trials improved while using this centering method, but we discovered that QPSK constellations were often unable to split out from a pseudo-BPSK constellation, where two pairs of constellation points existed in nearly the same location. Figure D.1 shows an example pseudo-BPSK constellation reached during one training run. We hypothesize that this hard centering required two constellation points to split out in tandem, which is difficult using noisy feedback.

As an alternative to the hard centering method, we applied 'soft' centering by adding a term for the constellation center's distance from the origin to the loss function. With this soft centering we were able to achieve successful training rates similar to the baseline simulation results. We verified that setting the weight of the constellation center location loss term to infinity reproduced the behavior seen in the hard forcing method above, namely that modulators reach a pseudo-BPSK constellation but were unable to split into a true QPSK constellation. Similarly, reducing the weight of the loss term to zero produced results seen in the baseline method, where DC offset correction caused the modulators to be unable to train.

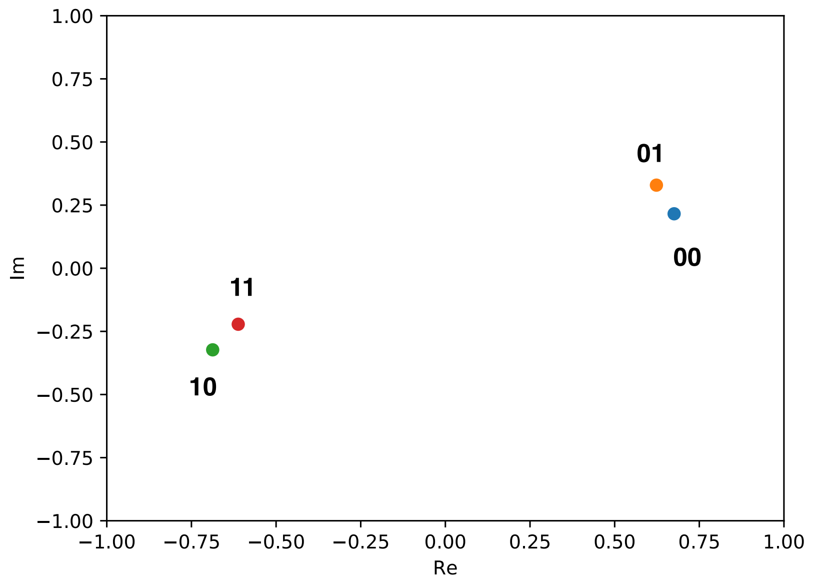
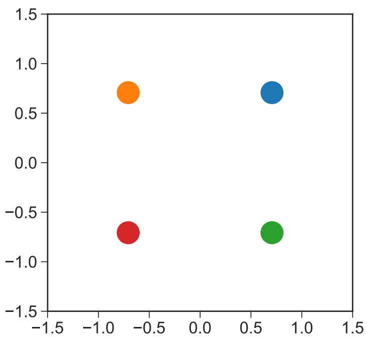


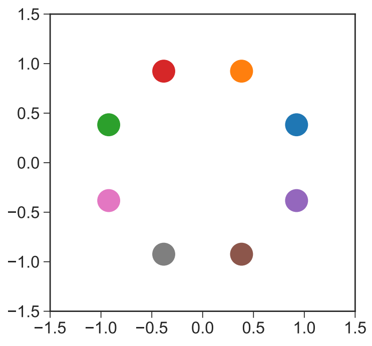
Figure D.1: An example of a pseudo-BPSK constellation reached during one training run with the GNU Radio EPP implementation. Two pairs of constellation points exist antipodally, just like a BPSK constellation. There is not enough separation between constellation points within the pairs to reliably demodulate the correct bit sequences.

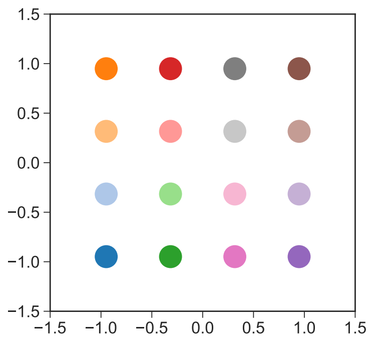
Appendix E

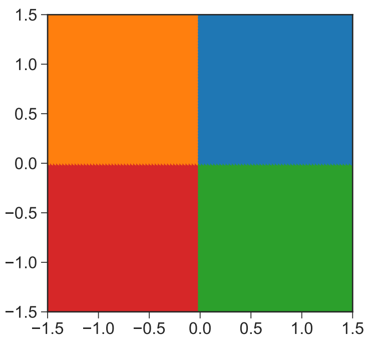
Classic Modulation Schemes

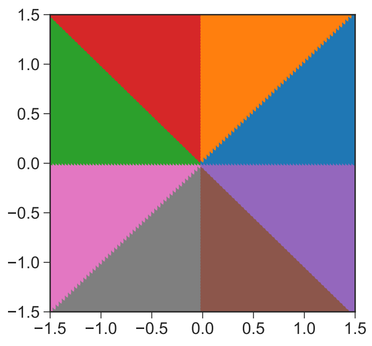
Figure E.1 illustrates the fixed modulation schemes used by Classic agents. These schemes are known to be optimal for AWGN channels [6].


(a) QPSK Classic Modulator


(b) 8PSK Classic Modulator


(c) 16QAM Classic Modulator


(d) QPSK Classic Demodulator


(e) 8PSK Classic Demodulator

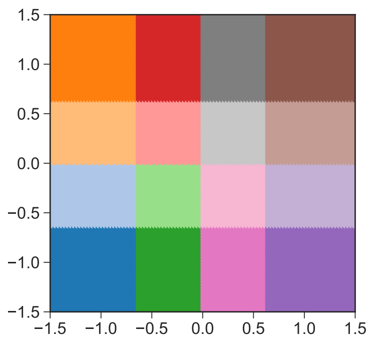

(f) 16QAM Classic Demodulator

Figure E.1: Figures (a) through (c) show the fixed, optimal modulation used for Classic models; (d) through (f) show the corresponding demodulation boundaries.

Appendix F

Simulation Settings

Unless specified otherwise, the training SNR values default to those in Table 2.1. Testing SNR values default to those corresponding to 0.001%, 0.01%, 0.1%, 1%, and 10% BER from Table 2.1. Table F.1 describes other simulation settings, such as the number of iterations, preamble length, and testing frequency.

	preamble length (symbols)	# iterations	testing intervals (spacing, # tests)
gradient passing			
QPSK	256	100	log, 30
loss passing			
QPSK	256	600	log, 30
echo, shared			
QPSK	256	2500	log, 30
8PSK	256	6000	log, 30
16QAM	256	8000	log, 30
echo, private			
QPSK	256	3000*	log, 30
8PSK	256	10000	log, 30
16QAM	256	20000	log, 30

Table F.1: Experiment settings for the different protocols and modulation orders. (* Because Neural-fast-and-Neural-fast converged so fast, we only trained for 500 iterations to adequately sample the convergence curve.)

Appendix G

Simulation Training Hyperparameters

The hyperparameter names and values in this appendix match the exact arguments used for running experiments with the code found at `https://github.com/ml4wireless/echo`. Table G.1 describes the purpose of each hyperparameter.

Hyperparameter	**Description**
hidden_layers	List of hidden layer widths
bits_per_symbol	Number of bits per symbol
restrict_energy	Energy constraint for modulator output
activation_fn_hidden	Hidden layer activation function
optimizer	Optimizer used
lambda_prob	Used for numerical stability in loss function
stepsize_cross_entropy	Learning rate for demodulator network weights
cross_entropy_weight	Weighting for cross-entropy loss terms
stepsize_mu	Learning rate for modulator network weights
stepsize_sigma	Learning rate for Gaussian policy stdev
initial_std	Starting exploration stdev
max_std	Maximum stdev for the Gaussian policy
min_std	Minimum stdev for the Gaussian policy
max_amplitude	Maximum average power of constellation
lambda_center	Weight for constellation center offset loss
lambda_l1	Weight for L1 penalty on weights
degree_polynomial	Degree of the polynomial features

Table G.1: Descriptions of hyperparameters used in experiments.

G.1 GP Training Hyperparameters

Hyperparameter	**Neural Modulator**	**Neural Demodulator**
hidden_layers	[50]	[50]
bits_per_symbol	2	2
restrict_energy	1	-
activation_fn_hidden	'tanh'	'tanh'
optimizer	'Adam'	'Adam'
stepsize_mu	3e-2	-
max_amplitude	1.0	-
stepsize_cross_entropy	-	3e-2
cross_entropy_weight	-	1.0

Table G.2: Neural hyperparameters used for GP QPSK Simulation experiments. There are no exploration-related parameters because GP does not use a Gaussian policy.

Hyperparameter	**Poly Modulator**	**Poly Demodulator**
degree_polynomial	*default	1
bits_per_symbol	2	2
restrict_energy	1	-
optimizer	'Adam'	'Adam'
stepsize_mu	1e-1	-
max_amplitude	1.0	-
stepsize_cross_entropy	-	1e-1
cross_entropy_weight	-	1.0
lambda_l1	-	1e-3

Table G.3: Poly hyperparameters used for GP QPSK Simulation experiments. There are no exploration-related parameters because GP does not use a Gaussian policy. (* Due to the fact that inputs to modulators are bits, a unique max degree polynomial can be used.)

G.2 LP, ESP, and EPP Neural Training Hyperparameters

Hyperparameter	**Neural Modulator**	**Neural Demodulator**
hidden_layers	[50]	[50]
bits_per_symbol	2	2
max_std	1	-
min_std	1e-1	-
initial_std	3e-1	-
restrict_energy	1	-
activation_fn_hidden	‘tanh’	‘tanh’
optimizer	‘Adam’	‘Adam’
lambda_prob	1e-10	-
stepsize_mu	8e-3	-
stepsize_sigma	1e-4	-
max_amplitude	1.0	-
stepsize_cross_entropy	-	5e-3
cross_entropy_weight	-	1.0

Table G.4: **Neural-fast** hyperparameters used for LP, ESP, and EPP QPSK Simulation experiments.

Hyperparameter	**Neural Modulator**	**Neural Demodulator**
hidden_layers	[50]	[50]
bits_per_symbol	2	2
max_std	1	-
min_std	1e-1	-
initial_std	3e-1	-
restrict_energy	1	-
activation_fn_hidden	'tanh'	'tanh'
optimizer	'Adam'	'Adam'
lambda_prob	1e-10	-
stepsize_mu	6e-4	-
stepsize_sigma	1e-4	-
max_amplitude	1.0	-
stepsize_cross_entropy	-	1e-3
cross_entropy_weight	-	1.0

Table G.5: **Neural-slow** hyperparameters used for ESP and EPP QPSK Simulation experiments.

Hyperparameter	**Neural Modulator**	**Neural Demodulator**
hidden_layers	[100]	[100]
bits_per_symbol	3	3
max_std	1	-
min_std	1e-2	-
initial_std	2e-1	-
restrict_energy	1	-
activation_fn_hidden	'tanh'	'tanh'
optimizer	'Adam'	'Adam'
lambda_prob	1e-10	-
stepsize_mu	8e-3	-
stepsize_sigma	4e-3	-
max_amplitude	1.0	-
stepsize_cross_entropy	-	1e-2
cross_entropy_weight	-	1.0

Table G.6: Neural hyperparameters used for ESP and EPP 8PSK Simulation experiments

Hyperparameter	**Neural Modulator**	**Neural Demodulator**
hidden_layers	[200]	[200]
bits_per_symbol	4	4
max_std	1	-
min_std	1e-2	-
initial_std	1e-1	-
restrict_energy	1	-
activation_fn_hidden	‘tanh’	‘tanh’
optimizer	‘Adam’	‘Adam’
lambda_prob	1e-10	-
stepsize_mu	7e-4	-
stepsize_sigma	5e-4	-
max_amplitude	1.0	-
stepsize_cross_entropy	-	1e-3
cross_entropy_weight	-	1.0

Table G.7: Neural hyperparameters used for ESP and EPP 16QAM Simulation experiments

Hyperparameter	**Neural Modulator**	**Neural Demodulator**
hidden_layers	[50]	[50]
bits_per_symbol	2	2
max_std	1	-
min_std	1e-1	-
initial_std	3e-1	-
restrict_energy	1	-
activation_fn_hidden	‘tanh’	‘tanh’
optimizer	‘Adam’	‘Adam’
lambda_prob	1e-10	-
stepsize_mu	1e-3	-
stepsize_sigma	1e-4	-
max_amplitude	1.0	-
stepsize_cross_entropy	-	1e-3
cross_entropy_weight	-	1.0

Table G.8: Neural hyperparameters used for EPP QPSK Training SNR Simulation experiments.

G.3 LP, ESP, and EPP Polynomial Training Hyperparameters

Hyperparameter	**Poly Modulator**	**Poly Demodulator**
degree_polynomial	*default	1
bits_per_symbol	2	2
max_std	2	-
min_std	2e-1	-
initial_std	1.0	-
restrict_energy	1	-
optimizer	'Adam'	'Adam'
stepsize_mu	4e-2	-
stepsize_sigma	4e-3	-
max_amplitude	1.0	-
stepsize_cross_entropy	-	1e-2
lambda_l1	-	1e-3

Table G.9: **Poly-fast** hyperparameters used for LP, ESP, and EPP QPSK Simulation experiments. (* Due to the fact that inputs to modulators are bits, a unique max degree polynomial can be used.)

Hyperparameter	**Polynomial Modulator**	**Polynomial Demodulator**
degree_polynomial	*default	1
bits_per_symbol	2	2
max_std	2	-
min_std	2e-1	-
initial_std	1.0	-
restrict_energy	1	-
optimizer	‘Adam’	‘Adam’
stepsize_mu	3e-2	-
stepsize_sigma	3e-3	-
max_amplitude	1.0	-
stepsize_cross_entropy	-	1e-2
lambda_l1	-	1e-3

Table G.10: **Poly-slow** hyperparameters used for EPP QPSK Alien Simulation experiments. (* Due to the fact that inputs to modulators are bits, a unique max degree polynomial can be used.)

Appendix H

GNU Radio Training Hyperparameters

The hyperparameter names and values in Table H.1 match the exact arguments used for running experiments with the code found at `https://github.com/ml4wireless/gr-echo`. Because of the extra lambda_center term in the loss function, the hyperparameters were tuned on the radio rather than reusing the hyperparameters from the rest of the simulations. These hyperparameters were used for both trials on the radio and simulations used for comparison with radio results.

Hyperparameter	**Neural Modulator**	**Neural Demodulator**
hidden_layers	[50]	[50]
bits_per_symbol	2	2
max_std	100	-
min_std	1e-1	-
initial_std	2e-1	-
restrict_energy	1	-
activation_fn_hidden	'tanh'	'tanh'
optimizer	'Adam'	'Adam'
lambda_prob	1e-10	-
stepsize_mu	1e-3	-
stepsize_sigma	1e-4	-
max_amplitude	0.5	-
lambda_center	125	-
stepsize_cross_entropy	-	1e-2

Table H.1: Neural hyperparameters used for GNU Radio experiments

Appendix I

Common EEG Channels

Each EEG dataset contains a relatively unique set of measured channels due to the recording device used and purpose of the study. When selecting datasets, we chose two that contained a common subset of 20 near-equivalent channels that were either identically placed or very close by. During preprocessing, we reordered the channels to match. Table I.1 lists the channels used from each dataset. We originally included more datasets which had more disparate channels measured, resulting in more near-neighbor channels.

SEED-V		Rest eyes open	
Channel	Original Index	Channel	Original Index
Fp1	0	Fp1	0
Fp2	30	Fp2	2
F3	2	F3	7
F4	28	F4	11
Fz	1	Fz	9
F7	3	F7	5
F8	29	F8	13
C3	7	C3	25
C4	23	C4	29
Cz	22	Cz	27
P3	12	P3	45
P4	17	P4	49
Pz	51	Pz	47
O1	14	O1	60
O2	16	O2	62
T7	8	T7	23
T8	24	T8	31
TP7	40	TP7	33
TP8	53	TP8	41
TP10	19	P8	51

Table I.1: Common channels and original indexes from datasets.

Appendix J

Spectrogram Model Hyperparameters

The following tables contain the hyperparameter choices for training models in Chapter 4.

Hyperparameter	Value
seed	19281
optimizer	AdamW
adam_beta1	0.9
adam_beta2	0.999
adam_epsilon	1e-8
adam_weight_decay	1e-6
lr_scheduler	constant
learning_rate	1e-5
max_grad_norm	1
lr_warmup_steps	100
max_train_steps	2000
train_batch_size	4
gradient_accumulation_steps	32

Table J.1: Training hyperparameters for the SDv1 diffusion model.

Hyperparameter	Value
lr	1e-4
model	DiS-M/2
epochs	200
min_lr	1e-6
vae	stabilityai/sdxl-vae
precision	fp16
task_type	class-cond
accum_iter	8
global_seed	3141
num_classes	4
latent_space	true
warmup_epochs	5
global_batch_size	8
scheduler_restart_epochs	10

Table J.2: Training hyperparameters for the DiS diffusion model.

Hyperparameter	Value
pooling	mean
batch_size	32
epochs	20
lr	0.01
optimizer	AdamW
lr_scheduler	None
label_smoothing	0.3
weight_decay	0.1
dropout	0.2
clip_grad_norm	1.0
ema_decay	0.999
transform	trivial_augment_wide

Table J.3: Training hyperparameters for the gender classification model.