Effect of Model Dissimilarity on Learning to Communicate in a Wireless Setting with Limited Information

Caryn Tran
Vignesh Subramanian
Kailas Vodrahalli
Anant Sahai

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2019-129
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2019/EECS-2019-129.html

August 16, 2019

Copyright © 2019, by the author(s).
All rights reserved.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission.

Acknowledgement

Thank you to Vignesh Subramanian, Kailas Vodrahalli, Josh Sanz, and Sameer Reddy for working with me on this throughout. Thank you so much to Professor Anant Sahai for guiding and showing me how research is to be done, and being so patient with me as my advisor. A thank you to Colin de Vrieze, Shane Barratt, Daniel Tsai for contributing their research from which this was built upon. And a final thank you to Professor Satish Rao for being my second reader and always being supportive.

Effect of Model Dissimilarity on Learning to Communicate in a Wireless Setting with Limited Information

by Caryn Tran

Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, in partial satisfaction of the requirements for the degree of **Master of Science, Plan II**.

Approval for the Report and Comprehensive Examination:

Committee:

Professor Anant Sahai
Research Advisor

(Date)

* * * * * * *

Professor Satish Rao
Second Reader

(Date)

Effect of Model Dissimilarity on Learning to Communicate in a Wireless Setting with Limited Information

Caryn Tran

Summer 2019

Abstract

This work engages the problem of collaborative learning in the context of wireless communication schemes. Two agents must learn modulation and demodulation schemes that enable them to communicate with each other in the presence of an AWGN channel via reinforcement learning. Proposed and examined is the echo private preamble protocol, a communication protocol enabling two agents to learn how to communicate with little shared context. Under the echo private preamble protocol, neural network based agents to learn strategies to communicate with other neural agents as well as agents that uses a fixed standardized protocol, and agents of a different model. This work also builds iteratively on top of relaxations of this protocol to show that this information restricted protocol is comparable to ones with a larger shared context. My specific contributions lie in introducing a new model (polynomials), writing the code base, and running and analyzing the baseline experiments for the echo private preamble protocol as well as the experiments to examine the effects of learning with mismatched agents whose internal models are dissimilar.

This research was and is continuing to be done in collaboration with Vignesh Subramanian, Kailas Vodrahalli, Josh Sanz, Sameer Reddy, and our advisor Professor Anant Sahai. Many thanks to everyone who has supported me especially friends, family, and students who motivate me to challenge myself whenever I can.

1 Introduction

Wireless communication research has a rich history [14], and many optimal or close-to-optimal protocols have been designed for individual pieces of the communication pipeline. However, one of the main disadvantages of these protocols is that they are ridged and must be shared beforehand to facilitate communication. Solutions that enable *learned communication* as opposed to fixed protocols have many benefits including easier usage of non-standard protocols, a more flexible interface allowing for easier development, and end-to-end learning of a full communication system. This report explores the idea of letting multiple agents learn *collaboratively*. Such work has relevancy for decentralized control schemes, adaptive communication protocols, and other areas.

This technical report considers a simplified communication model where only modulation at the transmitter and demodulation at the receiver are present. To support this simplification, there is the assumption that the channel effects consist of additive white Gaussian noise (AWGN) and also that there is perfect timing synchronization and no carrier frequency offsets. Under these environmental conditions, two agents will inter-operate to learn (1) how to modulate symbols to communicate (2) how to demodulate the received transmission to understand.

This work is a continuation of work done by [2] which introduces reinforcement learning to the modulation and demodulation tasks and establishes the echo protocol to achieve that end. To this, a further progression is made to remove the preamble which is shared across the two agents and used to learn from. Instead, the agents under the new **echo private preamble** protocol will only rely on echoed information in order to train. This new protocol is the contribution of Vignesh Subramanian and Kailas Vodrahalli. Added further is an investigation about the robustness of the changed protocol with a series of comparisons across multiple relaxations of the protocol which allow for increased information sharing. Finally, the private preamble protocol will be evaluated on its own with respect to different training and testing conditions, most notably in the case where the underlying models of the agents are very different.

My personal contributions to this research project was to introduce the polynomial model into the framework in order to investigate the effect of alienness, or model dissimilarity, on the learning behavior of the agents. Furthermore, I contributed the bulk of the currently working code and the experimental execution and results. Several threads of interest for further research that I have introduced but have yet to be completed are: the special case of implementing a double-autoencoder (titled as gradient passing round-trip in this report), and introducing a more sample efficient policy gradient algorithm, PPO, to improve the software radio simulations.

It will be shown that the echo private preamble protocol is sufficiently robust with neural net models and that it is comparable in performance to the previously established shared preamble case for round-trip bit error rates. Furthermore, when incorporating model dissmiliarity, the agents can exhibit either moderation of learning or disruption of learning, suggesting that there are effects that occur in interaction between agents that affect the system as whole, and is not just a reflection of either one agent.

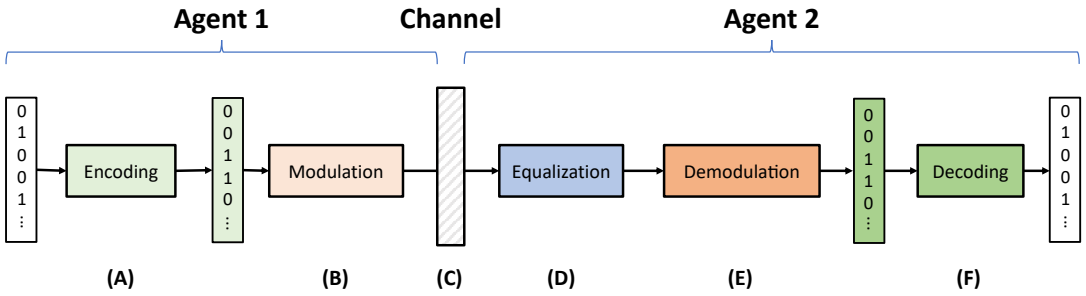


Figure 1: Visualization of a (simplified) wireless communication system. (A) Agent 1 encodes a message (a bit sequence), (b) modulates the message (maps the digital sequence to a representation more conducive to transmitting with analog equipment), and (C) sends the message over the channel. (D) Agent 2 receives the message and equalizes it, removing the effects of the channel (additive noise and intersymbol interference). (E) Agent 2 then demodulates the message and decodes it to recover the original bit sequence.

2 Background

2.1 Modulation

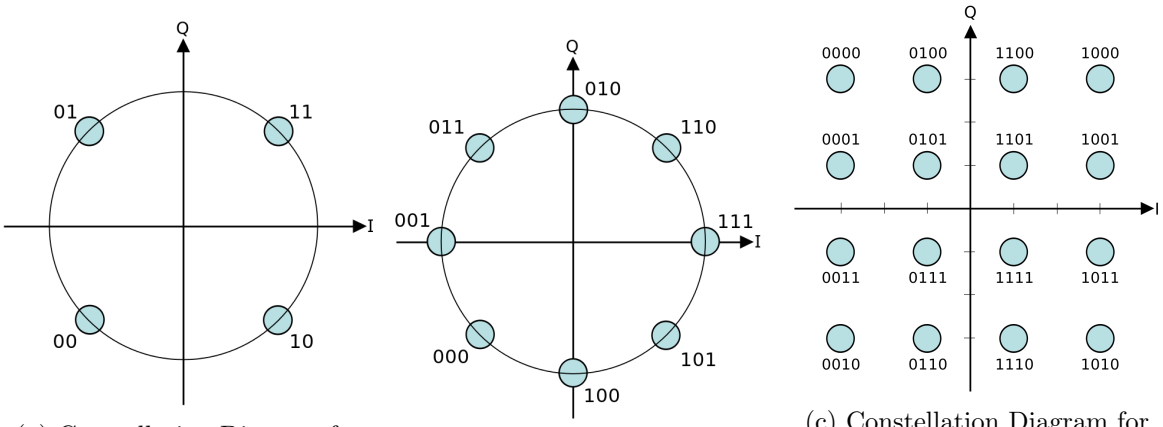
Wireless Communication Systems

The aim of wireless communication is to successfully transmit some information sequence over a channel and this process is broken down into several components (figure 1). An key part of the process is signal modulation the process of encoding information onto a carrier wave to be "carried" across to the receiver. The instantaneous state of an electromagnetic signal $s(t)$, can be synthesized from or decomposed into two components: in-phase $I(t)$ and quadrature $Q(t)$, thus providing two degrees of freedom for transmitting information. Below, $I(t) + iQ(t)$ can be interpreted as the modulated signal and $e^{i2\pi ft}$ as the carrier wave.

$$\begin{aligned} s(t) &= I(t)cos(2\pi ft)Q(t)sin(2\pi ft) \\ &= Re[I(t) + iQ(t)]e^{i2\pi ft} \end{aligned}$$

In digital modulation, information is sent as samples for uniform time intervals. Thus, consider $x(t) = I(t) + iQ(t)$ to be a discrete-time signal. With this formulation, the modulated signal can be analytically described as complex points (I/Q data) in the Cartesian form. Simply put,one can map the signal in a 2-dimensional plane. The translation of a bit sequence into I/Q data is undertaken by the **modulator**. A particular modulation/demodulation scheme operates under a limited set of complex symbols (modulation alphabet), each mapping to a unique bit grouping. The mapping can visualized by a *constellation diagram* of its modulation alphabet (figure 2). The number of symbols used for modulation is determined by the modulation order which describes the number of bits per symbol. Constellations with more points are more dense and trades off the power (to maintain the same bit error rate) for a higher bit rate. Below is the constellation diagrams for QPSK,

8PSK, and QAM16 which correspond to 2 bits per symbol, 3 bits per symbol, and 4 bits per symbol respectively.



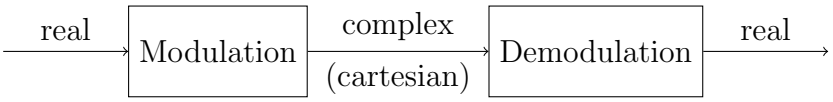
(a) Constellation Diagram for Quadrature phase-shift keying (QPSK)

(b) Constellation Diagram for 8-phase-shift keying(8PSK)

(c) Constellation Diagram for 16-quadrature amplitude-modulation (QAM16)

Figure 2: Constellation Diagrams

On the other end, the receiver will receive the signal after it has passed through the channel and been distorted. The smearing of symbols, known as intersymbol interference, is dealt with by equalization. This report focuses only on the modulation and demodulation tasks, thus we ignore any such affects throughout this report. The remaining channel effects can simplified as zero mean additive white Gaussian noise (AWGN) with variance $\sigma^2 = \frac{N0}{2}$ on both in-phase and quadrature component. And from this signal with additive white gaussian noise, the **demodulator** is tasked to recover from the intended symbol.



For this process to work, the current standards in wireless communications have transmitters and receivers using pre-defined modulation standards and protocols that are chosen based on the noise and distortion that is to be expected from the channel as well as the set by governing bodies to allocate the radio frequency spectrum for specific use. This requires overhead, is inflexible to change, and an inefficient allocation of resources. It is the aim of this research project to investigate learning approaches which do not have this overhead. Thus, the task of modulation and demodulation is reformulated in the context of two agents sending messages back and forth where modulation takes a real and discrete input turns it into a complex signal, and demodulation taking in a complex signal and turns it into discrete bits.

Because of the nature of this problem, where the two agents are supposed to be physically separated, machine learning tactics will not suffice for learning modulation (regression

task) and demodulation (classification task). Instead we look to reinforcement learning to learn without gradients and labelled data.

2.2 Reinforcement Learning

Building on the work of [2], we continue to use the policy gradient algorithm to train our system. While traditionally, reinforcement learning algorithms are mainly used in the case of episodic learning tasks in a sequential environment, the general idea behind reinforcement remains to be that in the absence of labelled data, an agent should seek to improve its actions based upon the rewards it receives. Using reward information, reinforcement learns the optimal for **Markov Decision Processes** with unknown transition dynamics and rewards functions. [11]

Many methods of reinforcement learning exist, including those which estimate the utility function, learn action-utility function, and those which directly learn a policy. Policy gradients, as the name suggests, belong to class of algorithms which directly learn the policy. Many variants and improvements on the policy gradient algorithm exist, but this project uses the vanilla **policy gradient algorithm**. The goal is to find a policy π which maximizes the expected reward. Given a parameterized policy π_θ, we seek an estimate of the gradient of the reward with respect to θ after having executed the policy in order to do gradient ascent. [11]

The standard choice for a continuous policy is a Gaussian policy parameterized by μ and σ^2 both of which can be further parameterized via some function, so that given a state s, $f(s)$ return μ and σ.

$$\pi_\theta(a|s) \sim \mathcal{N}(f(s), \sigma^2)$$

Where $f(s)$ may be a neural network, or linear regression, or polynomial regression.

Because of the non-sequential nature of modulation and demodulation as presented in this project, we do not have to consider trajectories and this simplifies our implementation of policy gradients.

Result: Policy parameters θ
initialization;
while *While not converged* **do**
  Sample actions $\vec{a}$ from $\pi_\theta(\vec{a}|\vec{s})$
  $\nabla_\theta J(\theta) = \frac{\sum_{i=1}^{n} log\pi_\theta(a_i|s_i)r(a_i,s_i)}{n}$
  $\theta \leftarrow \theta + \alpha\nabla_\theta J(\theta)$
end

Algorithm 1: Batch Policy Gradient for Non-sequential Environment

2.3 Related Works

2.3.1 Learning to Communicate

Application of end-to-end auto-encoder style machine learning approaches have been shown to work for the design of wireless [10] and fiber-optics systems [9, 7]. For the case of fiber-optics, they demonstrate results which exceed conventional methods, which are not yet optimal. These successes support the investigation of two separate agents *learning to communicate*. This is an interesting challenge that has relevancy to decentralized control schemes, adaptive communication protocols, and other areas. A decentralized approach to learning wireless communication protocol is demonstrated in [3] where demodulation schemes are learned via reinforcement learning. They achieve this via a common protocol for exchanging information and shared preamble between the agents. This work focuses on building on this approach by making preambles private and exploring 'alien' agents composed of differing function approximators.

2.4 Learning to Communicate in Humans

The core problem of communication exists across domains. It can be easily asked, how do babies come to understand sounds, words, and meaning? The development of language first begins in the development of 'categorical perception of sound' which creates discrete categories of sound perception.[13] This ability must develop robustly, for there is a lot of variation in how different individuals produce sound.

Later on, other tasks emerge such as word segmentation, which 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. [13]

This process continues and is fueled by social interaction and exchange, most often between child and caretaker. The driving factors for sucessful communication are *inter-subjectivity*, the sharing of a mutual understanding, in particular *joint attention* where the adult follows the baby's lead and comments, looks, and engages with whatever the child is looking at. [13]

There is a lot of debate whether or not humans have a innate biological module which enables language and communication in the way that it exists for humans. Does some prior exist as "shared information" across all humans? Chomsky, for example, is well known nativist in this domain, where as behaviorists, such as Skinner, would disagree. Nevertheless, the context from which a child learns and produces language is so very foreign to it, considering that children purely start off by learning to distinguish sound. [13]

These processes described in developmental social and cognitive psychology do draw a common parallel to the research done in this report. If children must start off by learning how to distinguish sound, the equivalent would be systems learning how to represent information such as we explore in the context of wireless communication. And likewise, the necessary social behaviors that humans exhibit in imitating, mirroring, and repeating

each other also supports in the general principle that providing feedback (like "echoing") is important mechanism for learning.

2.4.1 Learning to Cooperate

A more general lens to view this work is found in the theoretical works [5, 6, 8, 1, 4]. This body of work investigates the possibility for two intelligent beings to communicate when a shared context is absent or limited. In asking how two intelligent agents might understand each other without a common language, a theory of goal-oriented communication is developed in order to operationalize the idea of "understanding". The principal claim in [4] is that communication can be made robust if the goals of communication are explicit and verifiable. Our work is one specific case of the more general aims that is undertaken by this body of work. But by working in the setting of wireless communication and under the framework of reinforcement learning, we are additionally provided with rich background and practical relevance.

3 Protocols

3.1 Information Exchange

We consider the case with two agents, each being composed of a modulator and demodulator. The agents must learn modulation and demodulation schemes that allow for successful reconstruction of a message. Modulation is the regression task of taking a discrete symbol input and outputting a point in the complex field as I/Q data. Demodulation is the classification task of taking a signal represented as IQ data in the complex field. The aim is to define a protocol which is capable of achieving successful transmission in a round-trip, while minimizing the information which is shared between the two agents, beyond the transmitted and received signals.

In the communication process, messages composed of bits are assembled into symbols corresponding to the modulation order. The symbols are then modulated into a complex signal to be sent over an additive white gaussian noise (AWGN) channel. Upon receiving a complex signal, it must be demodulated back into symbols and then bits. Messages used for training are called *preambles*.

The exchange of a message from one agent to the other agent and back is considered a *round-trip*. Likewise, a *half-trip* occurs when the message only travels from one agent to the other. Success is defined as the accurate reconstruction of a message after a round-trip.

With this generic framework, we can consider different levels of information sharing and their affect on the agents' ability to learn. The following subsections describe the protocols by their increasing information sharing. The last two protocols are subclasses of the **echo protocol** which necessitates an echoing procedure in order to exchange information to be trained on. Throughout, we assume that the modulation order is fixed and common for both agents.

3.2 Gradient Passing

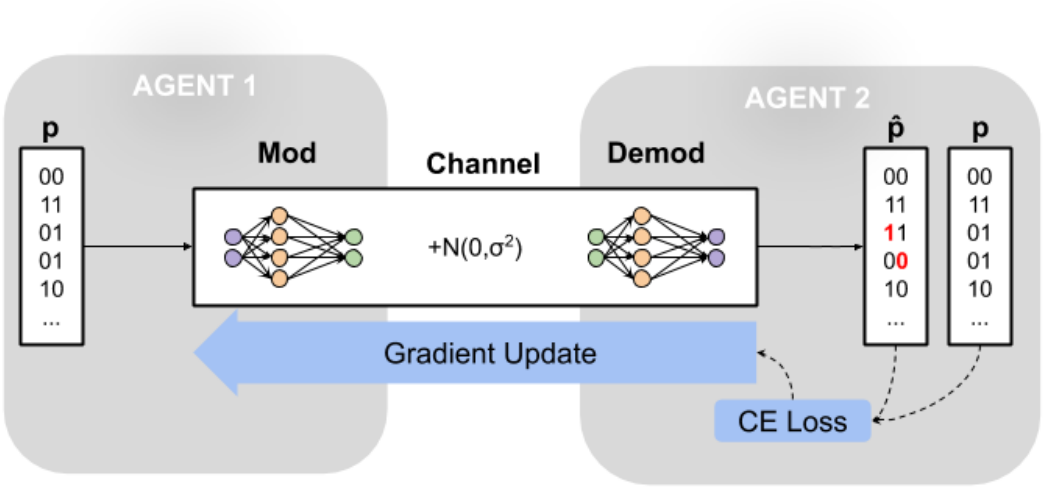


Figure 3: Gradient Passing: Half-trip
In this diagram, the preamble p is modulated and sent from Agent 1 through a connected channel to Agent 2 to be demodulated as $\hat{p}$. Using the shared preamble, both the demodulator and modulator of the two agents are updated directly via gradient passing under cross entropy loss.

The most dense information that can be shared between two agents is gradient information. If we allow for the case of a shared preamble, where both agents know the message being trained on, then this task nothing but an autocoder with random noise added to the code. The encoder in this network is one agent's modulator and the decoder is the other agent's demodulator. Using a cross entropy loss on the preambles sent and received, the two components can be trained together after a half-trip. Training in the opposite direction is omitted as it follows the identical process.

3.2.1 Round-trip Gradient Passing

If we were to consider the case of gradient passing without a shared preamble, then we would require a round-trip exchange for learning to occur. The challenge with this is allowing for gradients to pass between demodulation and modulation. After a half-trip, the message has been discretely demodulated into symbols. The gradients are traditionally lost in classification upon the dichotomous decision of the class.

To achieve round-trip gradient passing then would require some kind of alteration. The easiest alteration is to do a soft-classification and output partial bits instead of bits. This, however, changes the nature of demodulation as a classification task. In this project, there was no extensive investigation into the gradient passing through a classification decision, however, a few methods were tried:

1. Passing the max gradient through, by treating the $argmax$ decision as a max on the backwards pass.

2. Passing the gradient of an exaggerated $softmax$ through, by treating the $argmax$ decision as a $softmax$ with a very large base on the backwards pass (increasing the base as training occurs).

$$\text{As } \beta \to \infty,$$
$$softmax_i(\vec{x};\beta) = \frac{e^{\beta x_i}}{\sum_i e^{\beta x_i}} \to argmax_i(\vec{x}) = \begin{cases} 1 & x_i = max(\vec{x}) \\ 0 & else \end{cases}$$

The results from a cursory implementation of the strategies above were not promising, and this remains an interesting direction for further research.

3.3 Loss Passing

In the loss passing strategy, the agents share knowledge of the preamble and are able to directly pass the loss value. The modulator of the agent from which the preamble originated from must still update via reinforce, however, the loss information is more direct and does not have to pass through the channel. Similar to the gradient passing protocol, updates to the modulator of one agent and demodulator of the other is possible after a half-trip. The process can be repeated in the reverse direction for the same effect. Thus, training is omitted in the reverse direction.

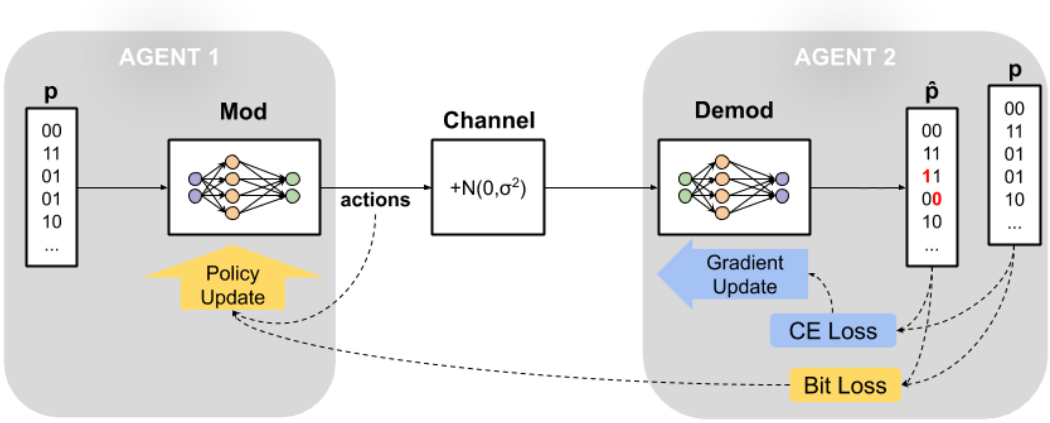


Figure 4: Loss Passing: Half-trip
In this diagram, the preamble p is modulated and sent from Agent 1 through a channel to Agent 2 which is demodulated as $\hat{p}$. Using the shared preamble, Agent 2 does a gradient update for its demodulator and also sends back a bit loss value directly to agent 1. Agent 1 then does a policy update of its modulator using the bit loss it received. Note that the loss is not passed through the channel. All implementations for the modulator currently use a Gaussian Policy with some underlying model to estimate the parameters μ and σ.

3.4 Echo: Shared Preamble

This protocol originates from [2], where the authors designed the "echo" principle in order to achieve cooperative learning for communication. In the echo shared preamble protocol, both agents now only share knowledge of the preamble and neither pass losses nor gradients to each other. The preamble must be echoed back to the originating agent, who we now will call the *origin agent*, in order for the origin agent to update its modulator. The agent who received and echoed the preamble, who we now will call the *echo agent* can also update its demodulator based on what it received and the preamble it expected.

In this protocol, a full round of training occurs after a round-trip. The two agents switch and take turns being the origin agent and echo agent.

3.5 Echo: Private Preamble Echo

Finally, in the echo private preamble protocol, the agents do not share knowledge of the preamble. Thus, learning modulation and demodulation only occurs after a round-trip exchange. In this protocol, the origin agent trains both its modulator and demodulator based on the preamble it originally sent and the preamble it received after the round-trip. The echo agent is passive. Like in the shared preamble echo protocol, the agents take turns, switching roles as the origin agent and echo agent.

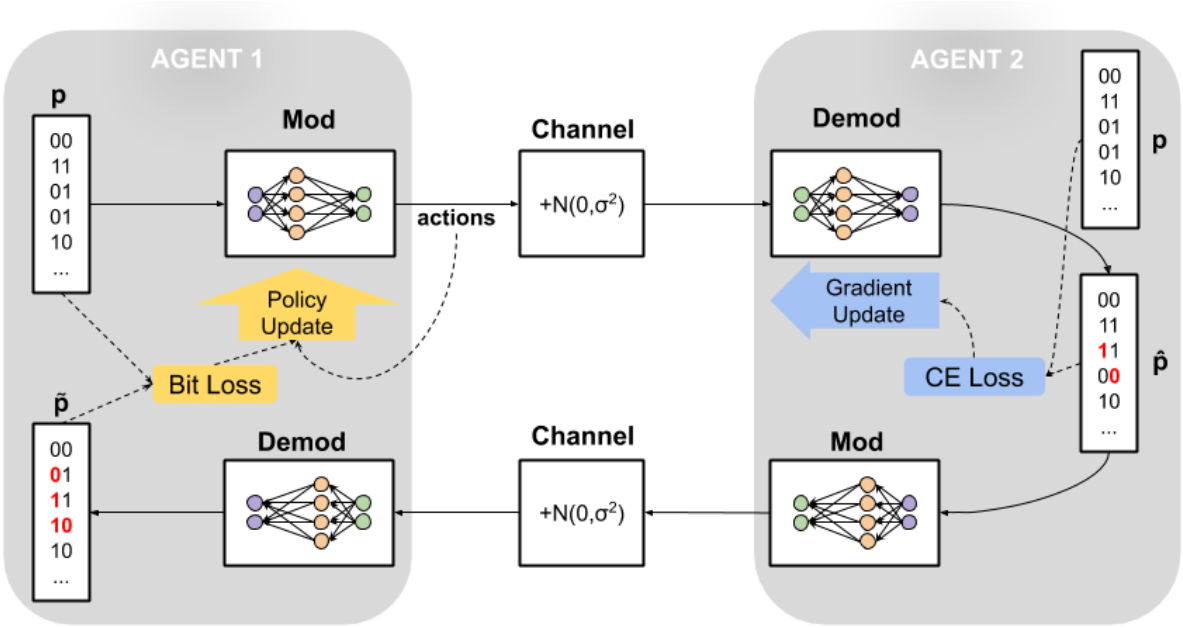


Figure 5: Shared Preamble: Round-trip
In this diagram, the preamble p is modulated and sent from Agent 1 through a channel to Agent 2 to be demodulated as $\hat{p}$. Using the shared preamble, Agent 2 does a gradient update for its demodulator and also modulates and sends back the preamble it received, $\hat{p}$, through the channel back to Agent 1. Agent 1 then demodulates the echoed preamble as $\tilde{p}$ 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. $\tilde{p}$. Agent 1 and Agent 2 then switch roles so that now Agent 2 is the originating agent and Agent 1 is the echoing agent. All implementations for the modulator currently use a Gaussian Policy with some underlying model to approximate the parameters μ and σ.

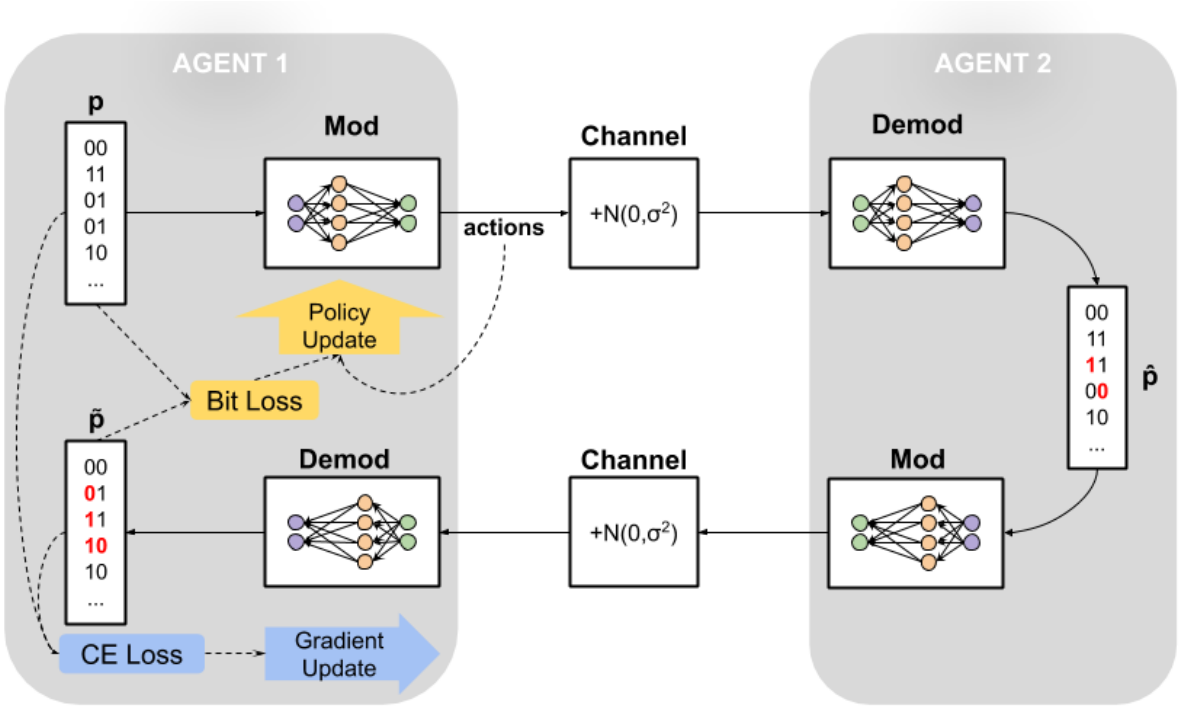


Figure 6: Private Preamble: Round-trip
In this diagram, the preamble p is modulated and sent from Agent 1 through a channel to Agent 2 to be demodulated as $\hat{p}$. Agent 2 has no information about the message it received so it cannot update. It passively modulates and echos back the message it demodulated. Agent 1 then demodulates the echoed preamble as $\tilde{p}$ 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. $\tilde{p}$, 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 originating agent and Agent 1 is the echoing agent. All implementations for the modulator currently use a Gaussian Policy with some underlying model to approximate the parameters μ and σ.

4 Experiments

4.1 Problem Definition

Recognizing that there may be mismatched encodings and decodings of the symbols for each agent when there is a lack of shared preamble, the main result we wish to show is that the private preamble echo protocol is successful in transmitting a message roundtrip. This shows that robust representation is learnable, and contributes a starting point for incorporating other features of the message to be considered.

To provide context, the gradient passing, loss passing, shared preamble, and private preamble protocols have successively built upon to show the effect of decreasing shared information. The bulk, however, of the goal is to determine the accuracy and robustness of the private preamble protocol. Concretely,

1. As a baseline, can an agent be trained against an optimal agent? Can two agents be learning simultaneously?

2. How close to optimal is the private preamble protocol? Compared to the shared preamble protocol?

3. What happens as the amount of noise in the channel increases/decreases?

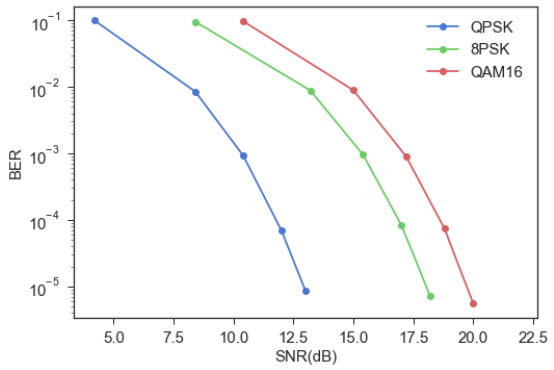
4. How does well does the protocol do for higher modulation orders?

5. Does the protocol work when the models underlying the agents are dissimilar? And what effect does increasing dissimilarity have?

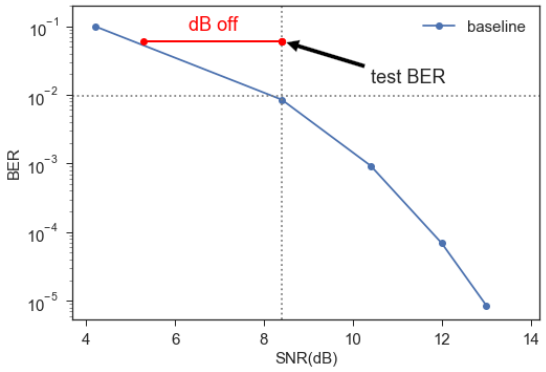
4.2 Measures

To measure accuracy, we plot the bit error rate curve for a roundtrip exchange. This curve measures the bit error rate for a round-trip in response to varying levels of noise (expressed as the SNR(dB)) being added. This can be compared to the baseline bit error rate curves under optimal modulation and demodulation.

We also look at performance across many trials and efficiency with respect to the number of preamble symbols consumed. This is measured by plotting the fraction of trials which converge with respect to the the number of preamble symbols consumed. Convergence is determined by measuring the difference between the optimal SNR(dB) for the bit error rate achieved by the system versus the SNR(dB) that the system was tested in. We consider the cases where the system is 3dB away and 5dB away. The testing SNR(dB) chosen to determine convergence with corresponds to 1% BER under optimal modulation. In otherwords, if our system achieves 1% BER at the chosen testing SNR(dB) then it is performing optimally.



(a) Baseline Bit Error Curves for QPSK, 8PSK, QAM16



(b) Example of dB off calculation used to determine convergence for round-trip exchange.

Figure 7: Illustration of Measures Used to Evaluate Experiments

4.3 Models and Agents

Recall that an agent is composed of a modulator and demodulator. Each of those components is implemented as via some model (either fixed or learning). The models used in experiments are:

1. **Classic:** A fixed, conventional modulation or demodulation scheme for the given modulation order. A classic agent performs either QPSK, 8PSK, or QAM16 for the given modulation order.

2. **Polynomial:** A learnable, polynomial regression (for modulation) or classification (for demodulation) model.

   (a) For modulation, the model takes in the bit representation of a symbol and generates polynomial features for those bits. Note that higher order functions applied bits do not result in greater representation, there is only so many terms that can be considered for a polynomial function for inputs of 0 and 1. The generated polynomial features are passed through a linear layer to generate parameter values for the Gaussian policy.

   (b) For the demodulation, the polynomial model takes in the complex signal as 2-dimensional data, performs a polynomial featurization of those points to an arbitrary degree. Those features are fed to a linear layer and a softmax to perform classication of the signal into the corresponding modulation symbols. Note that for the QPSK case, a 1 degree polynomial is sufficient for determining the optimal boundaries, however in practice, degree 2 polynomials work better on average.

3. **Neural:** A learnable, neural network regression (for modulation) or classification (for demodulation) model. Similar to the polynomial model, the neural model instead uses

a neural net to consume symbols in bit representation to output parameters for the Gaussian modulation policy or to consume complex points to output logits for the demodulation of a complex point.

Furthermore, there are three special agent types used to investigate learning with dissimilarity in agents:

1. **Model-specified:** When specified by model, agents will have both the modulator and demodulator composed of that type of model.
2. **Clone:** Clone agents, take on the exact same models and hyperparameters for modulation and demodulation of the agent they are trained together with.
3. **Selfalien:** Selfalien agents have the same models for modulation and demodulation as their counterpart, however they will have hyperparameter*s* that are different. For example, if agent 1 had a degree 2 polynomial demodulation model then a selfalien agent might have a degree 3 polynomial demodulation model.
4. **Alien:** Alien agents are agents whose modulation and demodulation models are entirely different from their counterpart.

4.4 Procedure

The following is a list of the experiments run for this project:

- **Information Sharing Experiments:**
  - Gradient passing half-trip training with permutations of neural and classic models for modulation and demodulation.
  - Loss passing half-trip training with permutations of neural and classic models for modulation and demodulation.
  - Shared preamble round-trip training for a neural agent with classic and clone agents.
  - Private preamble round-trip training for a neural agent with classic and clone agents.
- **Modulation Order Experiments:** Private preamble round-trip training for neural agent with clone for modulation order 2, 3, and 4. (*Corresponding to QPSK, 8PSK, and QAM16*)
- **Model Dissimilarity Experiments:** Private preamble round-trip training with a neural agent with classic, clone, self-alien, and alien agents. And, a training of the alien agent with its clone for measure. *The alien agent in this report is a polynomial agent.*

For each experiment, the following experiment settings were used:

- **Modulation Order:** Unless otherwise specified, all experiments were run with modulation order 2 corresponding to 2 bits per symbol and QPSK (4 unique symbols).
- **Preamble Length:** Dependent on the modulation order and the experiment.
- **Training SNR(dB):** SNRs(dB) corresponding to 10%, 1%, and 0.001% BER under optimal modulation and demodulation. For example, under 2 bits per symbol, the training SNRs would be: 13.0dB, 8.4dB, and 4.2dB. Main results are shown for training in the SNR(db) corresponding to 1% BER.
- **Number of Trials:** 50 per every setting of [agents X training SNRs].
- **Testing SNR(dB):** SNRs(dB) corresponding to 10%, 1%, 0.01%, 0.001%, and 0.0% BER under optimal modulation and demodulation.
- **Optimizer:** Adam

New random preambles were generated for every training step. All experiments were tested under with a roundtrip exchange of 100000 preamble symbols. For gradient passing and loss passing, because only 1 modulator and 1 demodulator is trained, each component is used twice to simulate having trained two full agents. For shared preamble and private preamble protocols, the testing results of the roundtrip paths between the two directions: (Agent 1 $\rightarrow$ Agent 2 $\rightarrow$ Agent 1) and (Agent 2 $\rightarrow$ Agent 1 $\rightarrow$ Agent 2) are averaged.

4.5 Implementation

4.5.1 Specifications

The code base is written in Python using PyTorch as the machine learning framework. Experiments were conducted on the BRC Cluster and Google Cloud Compute resources. The code base can be found at: https://github.com/ml4wireless/echo. In there is included all hyperparameters and setting as well as other experiments run and not included in results.

4.5.2 Code Design

- Protocols: Contain only the code for exchanging and training agents under a given protocol.
- Models: Models are individually packaged for modulation and demodulation and can be arbitrarily combined together to create an agent.
- Experiments: Experiments are specific instantiations of running a protocol under a modulation order with defined agents.

4.5.3 Details

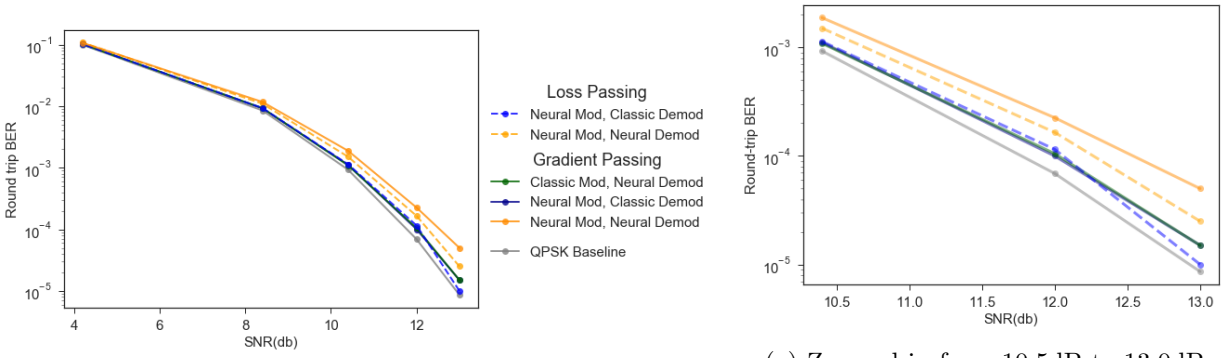
Hyperparameters are set as either a very small range to sample from or a single setting. Neural models are handtuned and Polynomial models were tuned via a Bayesian hyperparameter search. Tuning was done for experiments with uniform models (i.e. neural mod and neural demod for gradient passing) or clones (i.e. neural agent and clone for shared preamble).

After tuning, hyperparameters were reused for all usages of that model within that protocol and modulation order (i.e. all neural modulator models for shared preamble, 2 bits per symbol experiments used the same hyperparameters). A special case is that the shared and private preamble protocols are run on the same set of hyperparameters.

5 Results

5.1 Effect of Information Sharing

To build up to the private preamble protocol, first, gradient passing and loss passing variants are tested. In these simulated experiments, the neural models are coarsely hand tuned to perform well in a neural modulator and neural demodulator setting. As expected, BER is very close to optimal (figure 8) and converges quickly, with loss passing requiring more symbols to converge (figure 9). With very close inspection, neural modulator with neural demodulator achieves the best median BER both the loss passing and gradient passing experiments, which is not so surprising because the models were hand tuned for that particular setting.



(a) Zoomed in from 10.5dB to 13.0dB

Figure 8: Round-trip median bit error curves for **gradient and loss passing** protocols for QPSK at 8.4dB training SNR. Permutations of neural and classic modulator and demodulator models are tested in order to show individual learning components independently. On the left is the full curve at all test SNRs and on the right is zoomed in on the upper end of the testing SNRs.

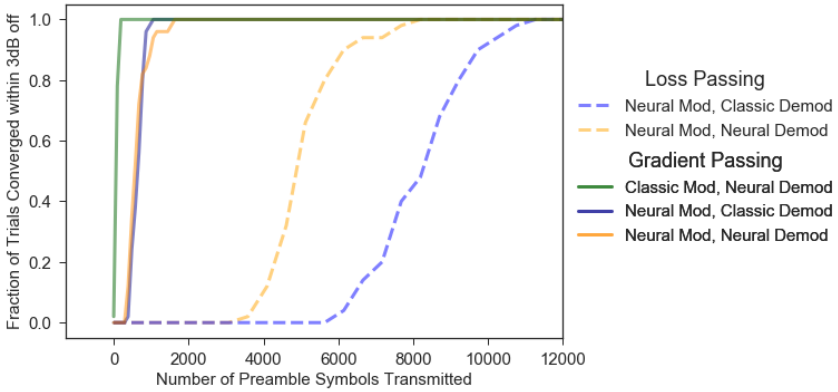


Figure 9: Convergence of 50 trials to be within 3dB (at testing SNR 8.4dB) of the QPSK baseline for **gradient and loss passing** trials at 8.4dB training SNR. Loss passing requires more symbols than gradient passing to converge.

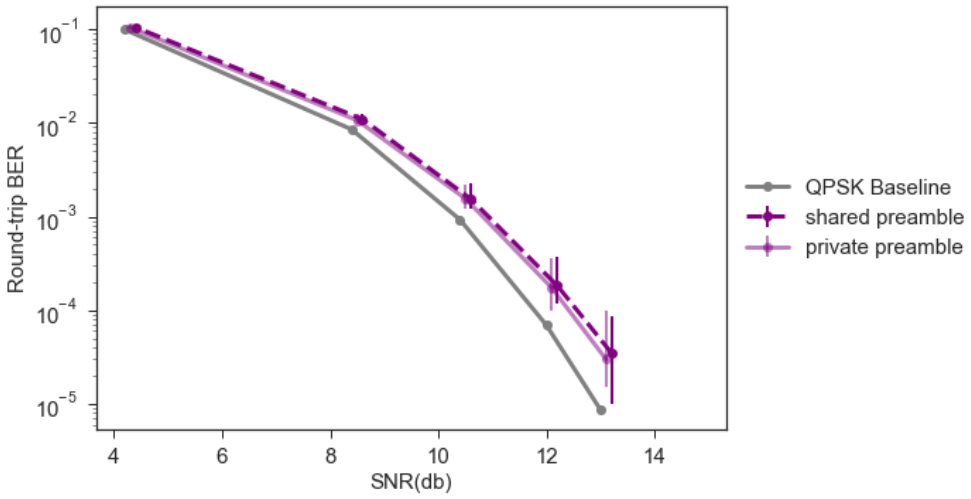


Figure 10: Round-trip median bit error curves for neural agent and clone learning QPSK comparing the **shared preamble** to the **private preamble** echo protocol at training 8.4dB training SNR. The error bars reflect the 90th to 10th percentiles across 50 trials. In terms of roundtrip accuracy, the two protocols are matched after training.

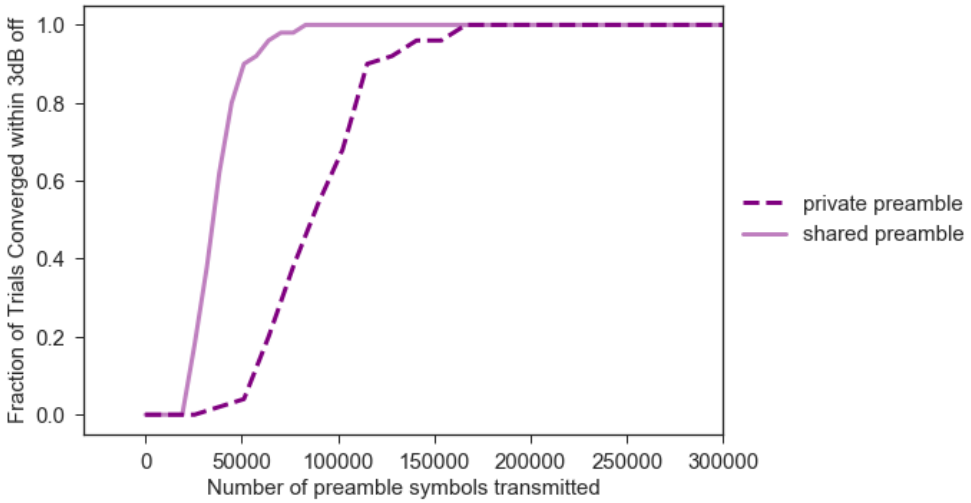


Figure 11: Convergence of 50 trials to be within 3dB (at testing SNR 8.4dB) of the QPSK baseline for **private preamble** and **shared preamble** trials with neural agent and clone at training SNR 8.4 dB. Private preamble takes more symbols to converge.

Next, we proceed to a comparison between the shared and private preamble protocols. For this, a neural agent is trained with its clone with 2 bits per symbol (QPSK). We see both protocols perform well and are nearly identical for median BER, and for the upper and lower percentiles (figure 10). **This is a main result of this report.** This show that the private preamble protocol is capable of performing as well as the shared preamble

protocol, with both being near optimal. Especially when comparing the BER curves of the echo protocols (shared and private) to the infomation passing protocols (gradient and loss), the similarity in BERs is affirming. Looking at the convergence plots in (figure 11), the private preamble protocol does come with a cost compared the shared preamble protocol in terms of number of symbols needed for all trials converge.

5.2 Private Preamble

Having established that the private preamble protocol is successful and comparable to the protocols with more information sharing, the next results examine the behavior of the protocol with respect to training noise, modulation order, and in the next section, dissimilarity in underlying agent models.

5.2.1 Training Under Increasing Noise

In figures 12 and 13, a neural agent (composed of a neural modulator and demodulator) is learning with a clone agent under the private preamble protocol for modulation order 2 (QPSK). The results are shown for different training noise settings, from least amount of noise (13.0 dB SNR) to most amount of noise (4.2 dB SNR). All settings of training noise reach convergence and are respectably close to the QPSK baseline. As can be expected, increasing noise shows increased difficulty in the training task, shown by the slower convergence in figure 13, but also introduces regularization to the training task, achieving better BERs (figure 12) once trained than compared to training at higher SNRs.

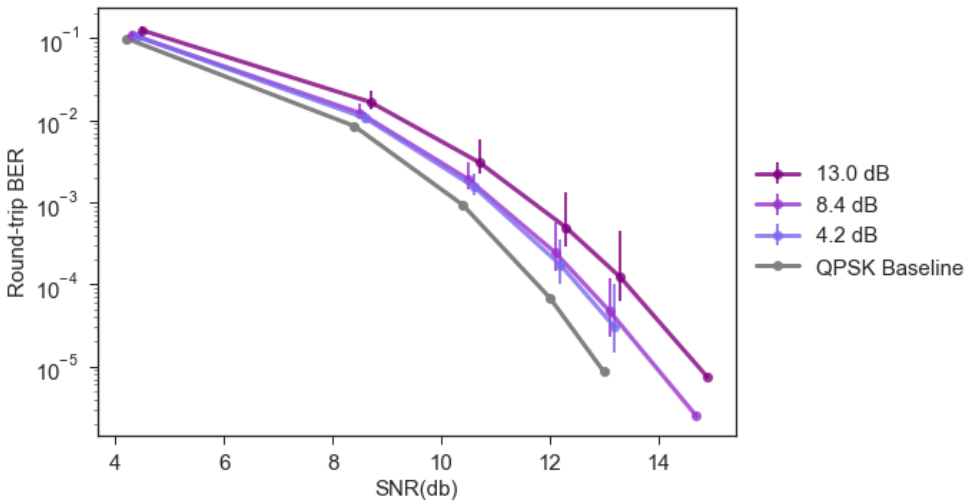


Figure 12: Round-trip median bit error curves for neural agent and clone learning QPSK under the **private preamble** protocol at training SNRs 13.0, 8.4, and 4.2 dB. The error bars reflect the 90th to 10th percentiles across 50 trials.

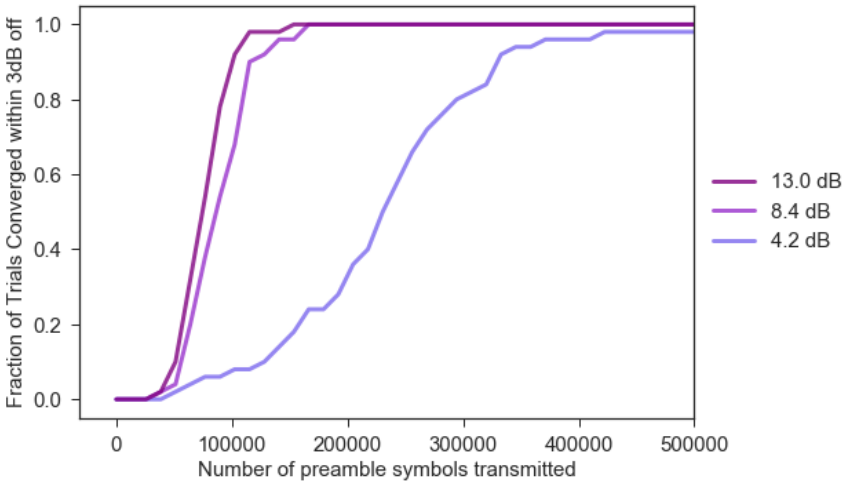
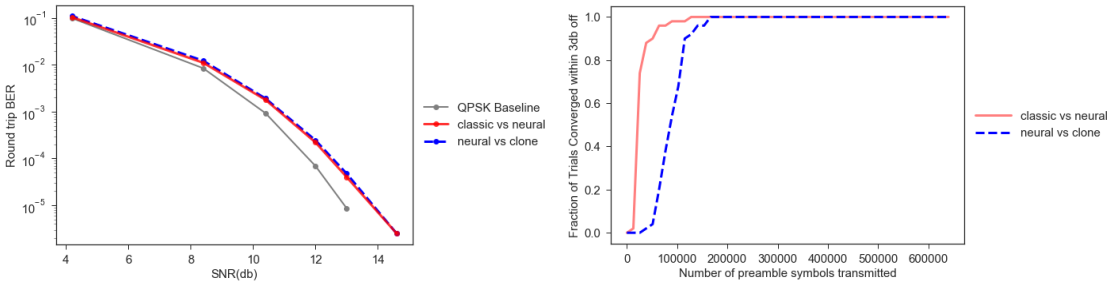


Figure 13: Convergence of 50 trials to be within 3dB (at testing SNR 8.4dB) of the QPSK baseline for **private preamble** trials with neural agent and clone at training SNRs 13.0, 8.4, and 4.2 dB.

5.2.2 Against a Fixed Agent

As a preliminary to exploring dissimilarity between agents, first the simple result of training a neural agent against a fixed one is shown and compared to the case where both agents are learning. This simple case is achievable and is similar in error and convergence to a neural agent against its clone.



(a) Round-trip median bit error curves for neural agent and clone or classic learning QPSK under the **private preamble** protocol at 8.4 dB training SNR

(b) Convergence of 50 trials to be within 3dB (at testing SNR 8.4dB) of the QPSK baseline for **private preamble** trials with neural agent and classic at training SNR 8.4dB.

Figure 14: Private Preamble: Neural Agent vs Clone

5.2.3 Effect of Modulation Order

We would like to see that the protocol works at higher modulation orders. Previously, we had only examined performance under 2 bits per symbol (QPSK). With 3 bits per symbol (8PSK) and 4 bits per symbol (QAM16), the protocol is still working albeit suffering

substantially compared to baseline at the highest modulation order (figure 15). Still, the convergence plots for getting within 3dB of the baselines show that there is full or nearly full convergence for all modulation orders (Figure 16). As expected, QAM16 with 16 constellation points is tougher than QPSK with only 4 constellation points. Higher modulation order results in more error and slower convergence, all of which is expected.

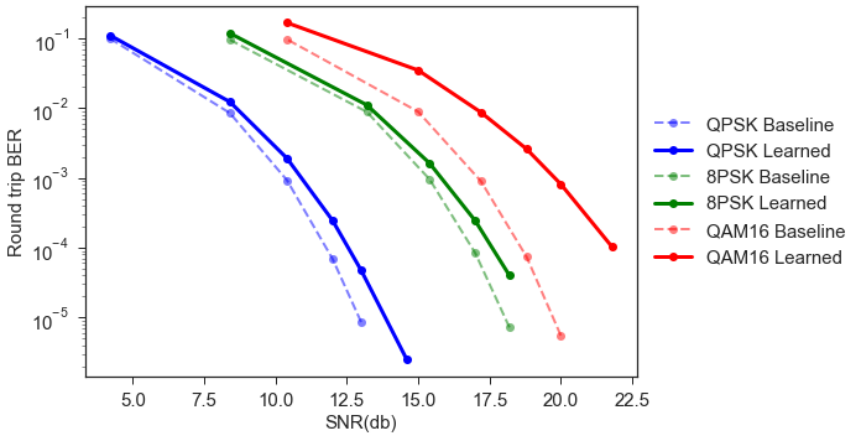


Figure 15: Round-trip median bit error curves for neural agent and clone learning QPSK, 8PSK, and QAM16 under the **private preamble** echo protocol at training SNRs corresponding to 1% BER. Alongside the bit error curves of the learned modulation and demodulation schemes is the baseline. In all cases, modulation constellations are normalized to constrain the average signal power.

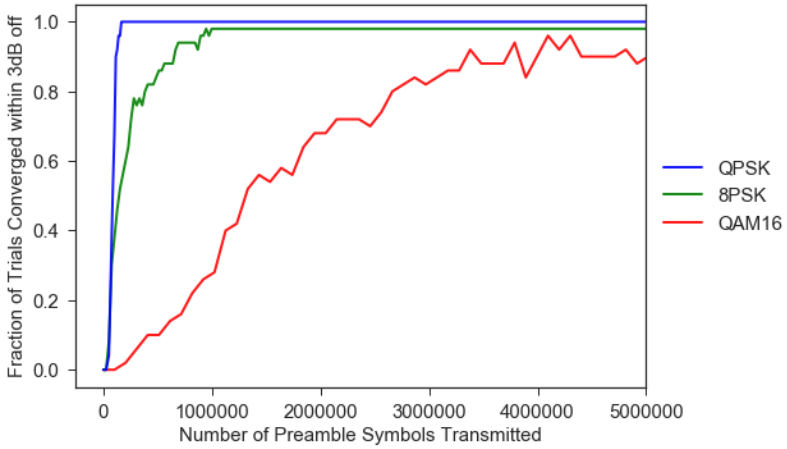
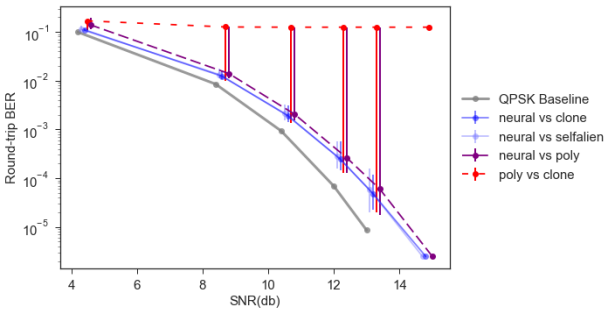
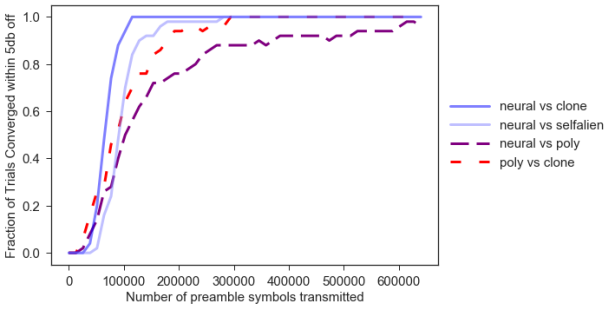


Figure 16: Convergence of 50 trials to be within 3dB (at testing SNR corresponding to 1% BER) of the corresponding baseline for **private preamble** trials of neural agent and clone at training SNR corresponding to 1% BER for increasing modulation order. QAM16, with the highest modulation order 4, takes much longer to converge than QPSK (order 2) and 8PSK (order 3).

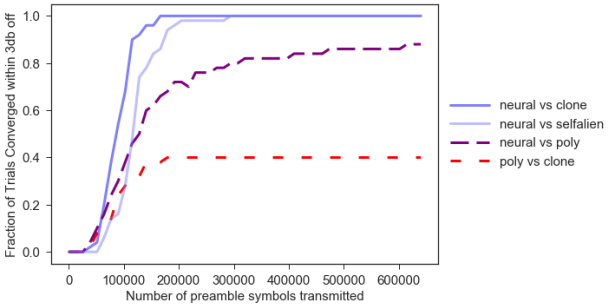
5.3 Effect of Model Dissimilarity on Echo Private Preamble



(a) **Alien (Poly1):** Round-trip median bit error curves for increased alienness to learn QPSK with the **private preamble** protocol at training SNR 8.4dB. Error bars reflect 90th and 10th percentiles.

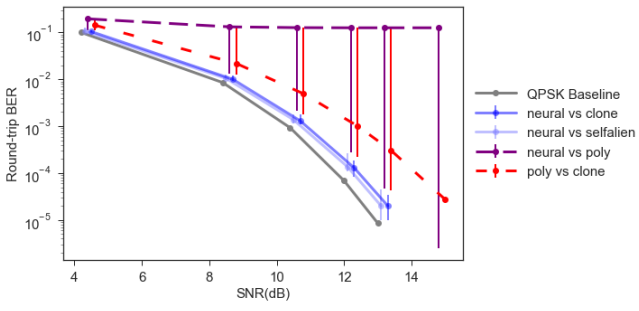


(b) **Alien (Poly1):** Convergence within 5dB for 50 trials (at testing SNR 8.4dB) of the QPSK baseline for **private preamble** trials with neural, neural-variants, and polynomial agents at training SNR 8.4 dB. The most dissimilar match up lags behind just slightly.

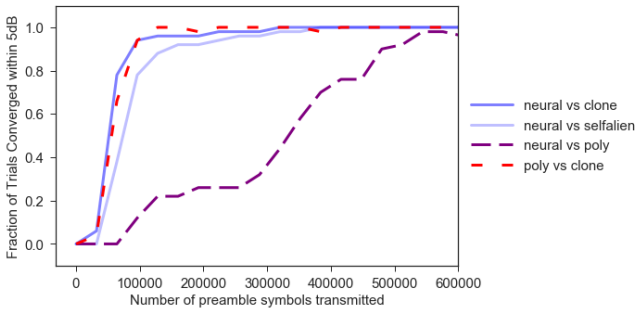


(c) **Alien (Poly1):** Convergence within 3dB for 50 trials (at testing SNR 8.4dB) of the QPSK baseline for **private preamble** trials with neural, neural-variants, and polynomial agents at training SNR 8.4 dB. The polynomial agent performs badly when matched against itself, does well trainin with neural.

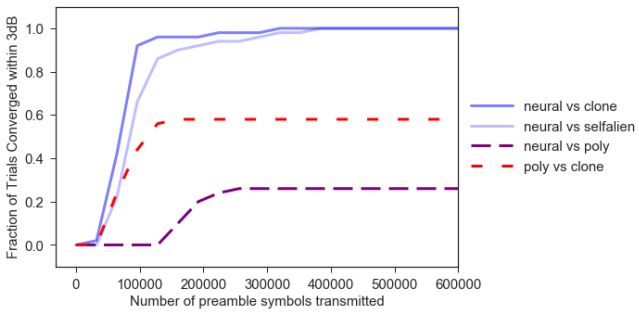
Figure 17: Dissimilarity Moderates Learning



(a) **Alien (Poly2):** Round-trip median bit error curves for increased alienness to learn QPSK under the **private preamble** protocol at training SNR 8.4dB. Error bars reflect 90th and 10th percentiles.



(b) **Alien (Poly2):** Convergence within 5dB for 50 trials (at testing SNR 8.4dB) of the QPSK baseline for **private preamble** trials with neural, neural-variants, and polynomial agents at training SNR 8.4 dB. The match up with the most dissimilarity is performing the worst.



(c) **Alien (Poly2):** Convergence within 3dB for 50 trials (at testing SNR 8.4dB) of the QPSK baseline for **private preamble** trials with neural, neural-variants, and polynomial agents at training SNR 8.4 dB. The match up with the most dissimilarity is performing the worst.

Figure 18: Dissimilarity Impedes Learning

This final section of experiment is more nuanced and not quite conclusive. Dissimilarity, also described as alienness, does not show consistent effects with regards to the echo private protocol. In the six graphs preceding the left-hand column corresponds to matching the neural agent with a polynomial agent (**poly 1**) that was hand-tuned to train with its clone. On the right column, the same neural agent is played against a different polynomial agent (**poly 2**) which had been tuned with bayesian search for training with its clone.

First, though in the six graphs preceding the behavior of polynomial models in the private preamble case is middling, it must be noted that the polynomial model does in fact work for the shared preamble case in terms of bit error and convergence within 3dB.

Finding the set of hyperparameters for a polynomial agent to do well against its clone was much harder for the private preamble protocol full convergence to 3dB was not found, but with 5dB off there is convergence. Regardless of the optimality of the polynomial model being used, there is an interesting result displayed.

On the left hand side, playing the neural agent with the polynomial agent results in moderated learning (figure 17) where the performance of the two is in between what is achieved when training either agent with its clone. The hand tuned polynomial agent (figure 17) performs quite poorly both with respect to bit error rate and convergence, however the neural agent learning together with the polynomial agent does very well.

While for another case, on the left hand side, though the polynomial model (poly 2) does better than poly 1 when played in a clone match, when pairing the second polynomial model together with the same neural model, we see that dissimilarity impedes learning (figure 18). The search tuned polynomial agent (figure 18) does better for bit error (though with high variance) and for convergence than the alien match up between neural and polynomial agent, where the results are very bad.

An explanation for this behavior was not able to be determined. Determining why one case happens versus the other would require more experimentation and research to find the distinguishing feature between the two agents which changing the learning behavior. But from just these results, we can hypothesize that the interaction of agents is not purely additive in terms of their learning behavior.

5.4 Discussion

These results are able to demonstrate that decentralized learning of modulation and demodulation is possible without the context of a shared preamble via the echo private preamble protocol and reinforcement learning. And by building up the informational assumptions, the private preamble protocol, and more generally decreasing the shared context results in requiring more training symbols in order to learn. Furthermore, it is shown that the echo protocol for learning is robust, and facilitates learning with agents that (A) use fixed protocols, (B) use similar learning models. While at the same time, we also find that under usage very different learning models, there is variance in the resulting learning behavior which suggests a future direction of research into what interactions are happening between two agents. Given these very insightful results, there is an exciting future ahead with many clear research directions to investigate for collaborative communication learning problems.

6 Conclusion

Several avenues for continued research exists within the problem space defined in this technical report. Firstly, as left off in the results where alienness of the agents resulted in varied behavior, there is a lot of interesting questions about why certain pairings fail or succeed and whether or not there exists a criteria for determining maximally robust hyperparameters for learning against multiple agents. Also as mentioned early on in the description of the different protocols, the idea of round-trip gradient passing is both a theoretical and technical challenge which may contribute to explorations of agents self-learning. Not discussed in this technical report is the radio implementation of this protocol. This line of work is already actively being pursued. A challenge which originates from the direct application to radios is how to achieve sample efficiency. The overhead of sending across the air is high and thus large batches must be sent in one pass. There already exists improved policy gradient algorithms which specifically improve sample efficiency, for example PPO [12] which can be used in place of the vanilla policy gradient algorithm.

Having dealt with the case of 2 agents, the next step would be to increase the number of agents learning to communicate. This would result in a cacophony of intermingling communication threads amongst agents. Very clear problems are how will agents manage turn-taking, how can agents avoid unlearning if they are matched up with bad agents, is there any benefit for an agent learn via eavesdropping, and will there be a global convergence of modulation and demodulation? Introducing more agents may illicit more results in investigating the interactions between differing models and surface network effects between agents.

7 Appendix

What follows is the hyperparameter settings for each of the models reported in this experiment

7.1 Preamble Lengths for each modulation order

```
'preamble_lengths' = {
       2: 32,
       3: 64,
       4: 128,
}
```

7.2 Neural QPSK Settings For Shared and Private Preamble

```
'neural_mod' : {
       'hidden_layers':          [50],
       'activation_fn_hidden':   'tanh',
       'stepsize_mu':            1e-3,
       'stepsize_sigma':         2e-4,
       'initial_std':            4e-1,
       'min_std':                1e-1,
       'max_std':                100,
},


'neural_demod' : {
       'hidden_layers':          [50],
       'activation_fn_hidden':   'tanh',
       'loss_type':              'l2',
       'stepsize_cross_entropy': 1e-3,
},
```

7.3 Neural 8PSK Settings for Private Preamble

```
'neural_mod' : {
       'hidden_layers':          [100],
       'activation_fn_hidden':   'tanh',
       'stepsize_mu':            1e-3,
       'stepsize_sigma':         1e-4,
       'initial_std':            2e-1,
```

```
        'min_std':                1e-2,
        'max_std':                100,
},

'neural_demod' : {
        'hidden_layers':          [100],
        'activation_fn_hidden':   'tanh',
        'loss_type':              'l2',
        'stepsize_cross_entropy': 1e-2,
},
```

7.4 Neural QAM16 Settings for Private Preamble

```
'neural_mod':{
        'hidden_layers':          [100],
        'activation_fn_hidden':   'tanh',
        'stepsize_mu':            1e-3,
        'stepsize_sigma':         1e-4,
        'initial_std':            1e-1,
        'min_std':                1e-2,
        'max_std':                100,
},

'neural_demod' : {
        'hidden_layers':          [200],
        'activation_fn_hidden':   'tanh',
        'loss_type':              'l2',
        'stepsize_cross_entropy': 1e-2,
}
```

7.5 Polynomial Alien 1 for QPSK Private Preamble

```
'poly_mod_qpsk' : {
        'stepsize_mu':            1e-2,
        'stepsize_sigma':         5e-5,
        'initial_std':            4e-1,
        'min_std':                1e-1,
        'max_std':                100,
},

'poly_demod_qpsk': {
        'degree_polynomial' :     3,
```

```
        'loss_type' :            'l2',
        'stepsize_cross_entropy' : 1e-2,
        'lambda_l1':             0.00,
},
```

7.6 Polynomial Alien 2 for QPSK Private Preamble

```
'poly_mod': {
        'stepsize_mu':           2.1e-2,
        'stepsize_sigma':        1.53e-5,
        'initial_std':           0.5,
        'max_std':               100.0,
        'min_std':               0.0001,
        'lambda_center':         1.26e-2,
        'lambda_l1':             3e-5,
}


'poly_demod': {
        'stepsize_cross_entropy': 1.55e-3,
        'epochs':                2,
        'degree_polynomial':     2,
        'lambda_l1':             2.52e-3,
        'lambda_l2':             0.0
}
```

7.7 Neural QPSK Settings for Gradient Passing

```
'neural_mod':{
        'hidden_layers':         [25],
        'activation_fn_hidden':  'tanh',
        'stepsize_mu':           9e-3,
},

'neural_demod':{
        'hidden_layers':         [50],
        'activation_fn_hidden':  'tanh',
        'loss_type':             'l2',
        'stepsize_cross_entropy': 1e-2,
},
```

7.8 Neural QPSK Settings for Loss Passing

```
'neural_mod':{
        'hidden_layers':          [20],
        'activation_fn_hidden':   'tanh',
        'stepsize_mu':            1e-3,
        'stepsize_sigma':         1e-4,
        'initial_std':            3e-1,
        'min_std':                1e-2,
        'max_std':                100,
},

'neural_demod' : {
        'hidden_layers':          [50],
        'activation_fn_hidden':   'tanh',
        'loss_type':              'l2',
        'stepsize_cross_entropy': 1e-2,
},
```

References

[1] Clément L. Canonne, Venkatesan Guruswami, Raghu Meka, and Madhu Sudan. Communication with imperfectly shared randomness. *CoRR*, abs/1411.3603, 2014.

[2] Colin de Vrieze, Shane Barratt, Daniel Tsai, and Anant Sahai. Cooperative multi-agent reinforcement learning for low-level wireless communication. *arXiv preprint arXiv:1801.04541*, 2018.

[3] Colin de Vrieze, Shane Barratt, Daniel Tsai, and Anant Sahai. Cooperative multi-agent reinforcement learning for low-level wireless communication. *arXiv preprint arXiv:1801.04541*, 2018.

[4] Oded Goldreich, Brendan Juba, and Madhu Sudan. A theory of goal-oriented communication. *J. ACM*, 59(2):8:1–8:65, 2012.

[5] Brendan Juba and Madhu Sudan. Universal semantic communication i. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 123–132. ACM, 2008.

[6] Brendan Juba and Madhu Sudan. Universal semantic communication ii: A theory of goal-oriented communication. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 15, 2008.

[7] Boris Karanov, Mathieu Chagnon, Félix Thouin, Tobias A Eriksson, Henning Bülow, Domaniç Lavery, Polina Bayvel, and Laurent Schmalen. End-to-end deep learning of optical fiber communications. *arXiv preprint arXiv:1804.04097*, 2018.

[8] Ilan Komargodski, Pravesh Kothari, and Madhu Sudan. Communication with contextual uncertainty. *CoRR*, abs/1504.04813, 2015.

[9] Shen Li, Christian Häger, Nil Garcia, and Henk Wymeersch. Achievable information rates for nonlinear fiber communication via end-to-end autoencoder learning. *arXiv preprint arXiv:1804.07675*, 2018.

[10] Timothy J. O'Shea and Jakob Hoydis. An introduction to machine learning communications systems. *CoRR*, abs/1702.00832, 2017.

[11] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall Press, Upper Saddle River, NJ, USA, 3rd edition, 2009.

[12] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[13] DeLoache J. Eisenberg N. Siegler, R. *How children develop.* 2003.

[14] David Tse and Pramod Viswanath. *Fundamentals of wireless communication*. Cambridge university press, 2005.