FIRE: Enabling Reciprocity for FDD MIMO Systems

Zikun Liu[1], Gagandeep Singh[1,3], Chenren Xu[2], Deepak Vasisht[1]
[1]University of Illinois Urbana-Champaign, [2] Peking University, [3] VMware Research

ABSTRACT

Massive MIMO forms a crucial component for 5G because of its ability to improve quality of service and support multiple streams simultaneously. However, for real-world MIMO deployments, estimating the downlink wireless channel from each antenna on the base station to every client device is a critical bottleneck, especially for the widely used frequency duplexed designs that cannot utilize reciprocity. Typically, this channel estimation requires explicit feedback from client devices and is prohibitive for large antenna deployments. In this paper, we present FIRE, a system that uses an end-to-end machine learning approach to enable accurate channel estimation without requiring any feedback from client devices. FIRE is interpretable, accurate, and has low compute overhead. We show that FIRE can successfully support MIMO transmissions in a real-world testbed and achieves SNR improvement over 10 dB in MIMO transmissions compared to the current state-of-the-art.

1 INTRODUCTION

The advent of 5G promises to add new dimensions to cellular communication systems. 5G will support high bandwidth Gbps communication from smart devices and enable low power connectivity for millions of Internet-of-Things devices per square mile. These capabilities are enabled by large bandwidths and MIMO (Multiple Input Multiple Output) techniques that leverage tens to hundreds of antennas. In 5G, base stations equipped with multiple antennas will leverage advanced signal processing methods to enable a suite of new technologies like multi-user MIMO and coordinated multi-point transmissions to increase the spectral efficiency of cellular networks multifold.

To enable MIMO capabilities, base stations need to know the downlink wireless channel from each of their antennas to every client device (e.g., a smartphone). This is trivially achieved in TDD (Time Domain Duplexing) systems using reciprocity. In TDD systems, the uplink (client to base station) and downlink transmission happen on the same frequency. Therefore, the base station can measure the uplink channel using client transmissions and use reciprocity to infer the downlink channel. Due to reciprocity, the uplink and downlink channels are equal modulo hardware factors. However, in FDD (Frequency Domain Duplexing) systems, dominant in several countries including the United States, the uplink and downlink transmission happen on different frequencies, and therefore the principle of reciprocity no longer applies.

ACM MobiCom '21, January 31-February 4, 2022, New Orleans, LA, USA

ACM ISBN 978-1-4503-8342-4/22/01…$15.00
https://doi.org/10.1145/3447993.3483275

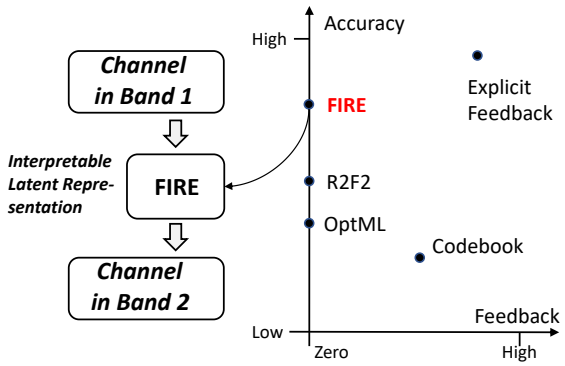


Figure 1: FIRE enables end-to-end cross-frequency channel estimation without feedback.

Today, in FDD systems, the client device measures the wireless channel using extra preamble symbols transmitted by the base station and sends it as feedback to the base station. This feedback introduces overhead that scales linearly with the number of antenna s, devices, and available bandwidth, and is prohibitive for massive MIMO systems. As we discuss in Sec. 2, the feedback overhead for 64 antenna base stations transmitting to 8 clients can be as high as 54 Mbps in mobile environments over a 10MHz channel.

This overhead has been recognized as unsustainable in the industry as well as academia [11, 21]. To solve this problem, some researchers(e.g., in R2F2 [59], OptML [9]) have observed that the uplink and downlink channel are created by the same underlying physical environment and the same paths being traveled. Therefore, they propose signal processing or machine learning models to infer the underlying paths using uplink channels measured at the base stations. Then, they use standard models to infer the downlink channel from the paths without any feedback. However, this computation has proven to be error-prone and can enable only low-accuracy primitives like beamforming but not the more advanced operations like multi-stream MIMO transmission or multi-user MIMO. Therefore, FDD cellular systems today gain limited utility out of multiple antennas due to this tradeoff between channel accuracy and feedback overhead. Accurate channel measurements can enable MIMO gains but have prohibitive overhead. In contrast, zero feedback methods fail to utilize MIMO gains.

Accurate zero-feedback MIMO: In this paper, we break the above barrier and achieve high accuracy, zero-feedback MIMO operation. We build FIRE (FDD Interpretable REciprocity) – a system that uses an end-to-end machine learning approach to infer downlink channels from uplink channels without any feedback from the client, as shown in Fig. 1. We observe that past work [9, 59] attempts to solve an unnecessarily challenging problem: to identify the accurate distance, angle, and phase of each path that the signal travels along. Given the limited bandwidth of cellular systems, this approach is bound to fail (see Sec. 4.1). However, the downlink channel inference problem is more forgiving. For example, adding

the same distance to all paths doesn't change the relative channel on the different antennas, and MIMO transmissions just care about relative channel values. This implies that using intermediate paths to infer downlink channels isn't the optimal strategy. Therefore, we use an end-to-end architecture to directly focus on the more relevant problem of predicting accurate downlink channels, rather than predicting the underlying paths.

An end-to-end architecture is advantageous for three reasons. First, the trained model can capture additional information about the environment (e.g. reflectors, buildings, base station characteristics) by combining information across multiple data points. This information is hard to capture in hand-crafted designs or signal processing approaches. Second, an end-to-end model can be easily trained by using explicit supervision. We can collect training data from some clients when a base station is set up. Such supervision for accurate distance, angle, and phase of each path of the signal is almost impossible to obtain as mentioned above (e.g., [9] uses a simulator to train their model from channels to physical paths). Finally, with 5G, moving the physical layer processing to the edge or the cloud is a new trend in the industry. An end-to-end model can be easily deployed on the cloud with automated processes for training and fine-tuning.

Generative Interpretable architecture: We model our architecture using a generative process, inspired by the physics-level intuition (and past work like [59]) that both uplink and downlink channels are generated by the same process from the underlying physical environment. Specifically, we choose a variant of the popular variational autoencoders (VAE) as our end-to-end architecture,. The VAE first (a) infers a latent low-dimensional representation of the underlying process of channel generation by observing samples of the uplink channel, and then (b) generates the downlink channels by sampling in this low-dimensional space. Given its data-driven nature, the VAE can embed real-world effects in the latent space and therefore capture the generative process more accurately. Our experiments in Sec. 6.5 validate our choice of modeling the generative process and show that our design outperforms past signal processing approaches and learning-based approaches that use discriminative models such as fully connected networks that inherently cannot capture the generative process.

A key criticism of end-to-end machine learning models, when applied to networking solutions, is that they often operate in a black-box manner lacking interpretability and therefore, are impractical in real-world scenarios like cellular networks where network operators may want to peek inside the algorithms when things go wrong. FIRE's VAE enables interpretability as it encodes the generative process in a probabilistic latent space representation which is indicative of the physical characteristics of the signal transmission (e.g. client properties, locations, reflectors in the environment) and provides potential insights for network operators.

Countering hardware randomness: Our work deals with several challenges that arise out of hardware imperfections in the real-world. Wireless channel measurements at base station and clients are not just a function of underlying signal paths. Instead, they are also strongly impacted by hardware effects like carrier frequency offset (CFO), packet detection delays, etc. In practice, CFO introduces random phase shifts to the wireless channel that is consistent across antennas and frequencies. On the other hand, packet detection delays add random phase shifts that vary across frequencies. Any machine learning model operating on such raw channel measurements will be confused by these random uncorrelated effects on the wireless signal. It will try to fit to the randomness, instead of the useful information in the wireless channel. Therefore, we design a data transformation algorithm that can standardize the input-output relationship between uplink and downlink channels.

Evaluation: We evaluate FIRE on a public large scale dataset [54] collected using 96-antenna base stations. We train FIRE using a single client device and test on multiple other client devices. We compare FIRE against state-of-the-art baselines: a feedback-free signal-processing channel inference system, a machine learning based channel prediction model, and a codebook based method. We further build a base station prototype with 4 antennas and test FIREś performance on it. Our results show that:

- **Prediction Accuracy:** FIRE can accurately predict downlink channels. Channels predicted by FIRE can achieve median MIMO SINR's of 24.9 dB as compared to the best baseline performance of 13.33 dB.
- **Data rate:** In a Multi-user MIMO setup, over 80% of FIRE's channel predictions can support the highest data rate, as opposed to nearly 10% with the best baseline.
- **Real-time Operation:** The median runtime of FIRE is 3.0 ms on CPU and 0.3 ms on GPU compared to 7 seconds (on CPU) for the baseline. FIRE's runtime can support channel estimates within coherence time intervals.
- **Cross-antenna Prediction:** FIRE's architecture can achieve reasonable performance, without optimizations, at other channel prediction tasks such as predicting downlink channel for a subset of base station antennas given uplink channels at another subset (median SNR: 11.95 dB).

To the best of our knowledge, FIRE is the first system to demonstrate high-accuracy MIMO channel measurements in a real-world testbed. We achieve this using an end-to-end Machine Learning approach. We believe this design will be crucial to next generation of cellular networks: 5G & beyond.

2 BACKGROUND AND CONTEXT

Massive MIMO will be a key component in future 5G deployments. At its best, massive MIMO uses tens to hundreds of antennas to simultaneously communicate to multiple clients and increase the net throughput of the system. Due to this promise, it is estimated that massive MIMO investments for cellular networks crossed ten billion US dollars in 2020 [21].

Wireless channels: Wireless channels are a fundamental quantity in wireless systems. For a complex valued signal x transmitted by a transmitter, the signal received by a receiver is $y = hx$, where h is the wireless channel (a complex number) and denotes the effect of the environment that the signal travelled through. Specifically, for a signal transmitted at frequency, f:

$$h \propto \sum_i \alpha_i e^{-j\frac{2\pi d_i f}{c}} \quad (1)$$

when the signal travels along multiple paths – each with attenuation, α_i, and distance, d_i. c is the speed of light.

Massive MIMO: In massive MIMO, a base station has multiple antennas (say M) and aims to talk to multiple clients (say $K < M$) simultaneously. For simplicity, we assume that all clients have one antenna each. The wireless channel can now be represented as a $K \times M$ matrix $\mathbf{H}$, where h_{km} denotes the channel from antenna m to client k. Assume, $\mathbf{x}$ is the $M \times 1$ complex vector of signals transmitted from the M antennas. Then, the signal received at the K clients, $\mathbf{y}$, is given by: $\mathbf{y} = \mathbf{Hx}$, where y_k is the signal received at client k.

Let us say the base station wants to communicate value q_k to the client k. It needs to transmit signal $\mathbf{x}$ such that y_k received at client k is just a function of q_k and does not see interference from $q_{k'}$ intended for other clients. To achieve this effect, the base station *precodes* the values q_k. The base station must identify a $M \times K$ precoding matrix $\mathbf{P}$ such that $\mathbf{x} = \mathbf{Pq}$, where $\mathbf{q}$ is the $K \times 1$ vector of q_k's. Therefore, the received signal is:

$$\mathbf{y} = \mathbf{HPq} \tag{2}$$

How do we select the pre-coding matrix $\mathbf{P}$? One standard way to do this, called zeroforcing, is to set $\mathbf{P} = \mathbf{H}^{\dagger}$, where $\mathbf{H}^{\dagger}$ is the right pseudo-inverse of $\mathbf{H}$. Therefore, $\mathbf{y} = \mathbf{HH}^{\dagger}\mathbf{q} = \mathbf{I}_K\mathbf{q}$, where I_K is the $K \times K$ identity matrix. This allows every client to receive its own signal without suffering interference from the signal intended for any other client.

Channel estimation: Note, the above procedure relies on accurate knowledge of the wireless channel, $\mathbf{H}$, at the base station. Any error in estimating $\mathbf{H}$ leads to interference for the clients and reduces their data rate. First, let us focus on TDD systems, where the uplink and downlink happen on the same frequency. Recall, from Eq. 1, the wireless channel depends on the frequency and distance. Wireless signals travel the same paths on uplink and downlink. Therefore, for a base station-client pair using TDD, the channel for the uplink and the downlink are equal modulo some hardware factors that can be calibrated for. This principle is called reciprocity. The client transmits some pilot symbols known to the base station so that the base station can estimate the uplink channel, and use reciprocity to infer the downlink channel (which is just a constant multiplication to the uplink channel). In terms of overhead, this process requires just K uplink pilots, one for each transmitting client, and is independent of the number of antennas on the base station as base station antennas can simultaneously sense the signal.

For FDD, the uplink and downlink transmission happen on different frequencies. Therefore, the downlink channel is not equal to the uplink channel anymore. For FDD base stations to leverage MIMO, the base station sends M pilot symbols one on each antenna. Each client measures the downlink wireless channel from each antenna to itself and sends the M channel values as feedback to the base station. In total, K clients send $M \times K$ channel values as feedback. This feedback incurs overhead that scales with the number of antennas and number of clients ($M \times K$). Past work [20, 39] has shown that in TDD, massive MIMO can enable theoretically infinite scaling with increasing number of antennas and clients. However, this feedback overhead caps the scaling for FDD systems, since it scales up with M and K as well, and the spectrum becomes the bottleneck. We refer the reader to [40] for a detailed discussion on massive MIMO.

Example of feedback: We use reference numbers from [40] to obtain a conservative estimate of the feedback overhead. Let us assume a downlink frequency band centered at 2 GHz with a 10 MHz width. A typical coherence bandwidth for a pedestrian in outdoor scenarios is 300 kHz and the coherence time is 50 ms. The coherence time goes down to 2.5 ms for motion at vehicular speeds (e.g. smartphones during travel). The coherence bandwidth and time indicate the frequency-time interval over which the channel doesn't change much. Conservatively, let us assume that the client just sends one value for one coherence frequency-coherence time interval. Furthermore, let us assume a standard setup with 64 base station antennas transmitting to 8 clients. In such a setup, the feedback overhead is 3 Mbps for the pedestrian scenario and 54 Mbps for the mobile scenario in cars, assuming 8 bits (4 real, 4 imaginary) for each channel value. Given that a 10 MHz channel can support between 2 to 70 Mbps depending on channel conditions, this feedback is unsustainable. Therefore, it is believed that FDD systems are not suited for massive MIMO operations or must limit themselves to coarse-grained use of multiple antennas like beamforming which does not require as accurate channel estimation. Our work aims to reduce this feedback to zero while supporting accurate downlink channel estimates.

FDD vs TDD: Today, in most parts of the world, FDD remains either the only or the heavily dominant strategy for cellular spectrum allocation [51, 61]. For instance, the leading cellular providers in the United States all use FDD. 5G NR will use a mix of existing and new spectrum. The new spectrum allocations are a mix of TDD and FDD spectrum, with FDD still dominant in the sub-6Hz bands. One might wonder why FDD spectrum is preferred despite the challenges associated with MIMO operations. It is because FDD systems provide better coverage for edge clients, require fewer base stations, and incur lower cost overall. Due to separate bands for uplink and downlink, in FDD, the client and base station do not need to coordinate transmissions to avoid interference, reducing timing synchronization overhead. The uninterrupted operation of clients and base stations also extends range and reduces the need for base stations. According to Qualcomm [50], TDD systems need up to 65% more base stations than FDD systems for similar coverage and performance. In contrast, the key disadvantage of FDD systems is the channel estimation overhead and its implication for MIMO operation. Therefore, many network vendors, like Ericson and Qualcomm [18, 52], have proposed the use of carrier aggregation across FDD and TDD to combine the best aspects of TDD (MIMO) and FDD (larger range, lower overhead, continuous coverage).

3 SYSTEM OBJECTIVES AND OVERVIEW

In this paper, we aim to reduce the burden of channel estimation and feedback for future FDD MIMO systems. We design our system with the following goals in mind:

- **Zero Feedback:** Our design must not require any feedback from the client device.
- **Accuracy:** The channel estimates must be accurate enough to support advanced MIMO techniques.

- **Interpretability:** Our method must support interpretability of the end results.
- **Robustness:** Our system must be robust to real-world variations like hardware effects, client mobility, etc.

With these objectives in mind, we design FIRE, an end-to-end design for downlink channel estimation without feedback. A base station measures the uplink channel (as it would normally do) using uplink pilot symbols. It then feeds the uplink channel estimates to FIRE which uses them to compute the corresponding downlink channel. The downlink channel can then be used to perform precoding for advanced MIMO techniques. We discuss our motivation for an end-to-end design in Sec. 4.1, describe the design in Sec. 4.2, and present the systems challenges in Sec. 4.4. Finally, we discuss our datasets in Sec. 5.1 and present a detailed evaluation in Sec. 6.

Scope: Our system is designed for operation in the traditional frequency bands used for cellular communication (<2 GHz) and for the newly allocated sub-6 bands (<6 GHz). Most 5G deployments in the near term by providers like AT&T, T-Mobile, Verizon, etc. will rely on sub-6 bands. mmWave bands (>20 GHz) offer high bandwidth capabilities and use multiple antennas, but the MIMO process and challenges in these bands are different due to high attenuation, largely line-of-sight operation, etc. These challenges are being tackled by multiple research efforts and are not the focus of FIRE.

4 END-TO-END ARCHITECTURE FOR CHANNEL PREDICTION

In this section, we describe the design of FIRE, a machine learning-based cross-band channel prediction system, equipped with a dedicated channel transformation algorithm.

4.1 Motivation

Before we delve deeper into our design, we must ask: is it at all possible to infer downlink channels from uplink channels? Prior work [6, 9, 26, 27, 59] has already shown that this is possible. To understand why this is the case, consider Eq. 1 where the channel value, h, measured at a given antenna depends on the path traveled by each signal from the transmitter to the receiver antenna. Specifically, it is a function of the distance, d_i, and attenuation, α_i. α_i denotes the attenuation due to path loss and loss incurred during reflection (including phase changes caused by reflection). So, if we can use the channel measurements at one frequency to infer the distance d_i and complex attenuation α_i, we can plug these values into Eq. 1 to infer wireless channels at a different frequency.

This process is aided by two factors: (a) for a given path, the length of the path traveled by the signal to each base station antenna is not independent, but rather a function of the angle that the signal is received at – this reduces the number of variables to estimate, (b) cellular networks use OFDMA (orthogonal frequency division multiple access) to divide the frequency bands into multiple sub-frequencies. This enables uplink channel measurements at multiple frequencies, giving the base station more measurements to identify the underlying variables. In this context, we make two observations:

Low bandwidth of cellular transmissions hinders parameter estimation: Prior works [9, 59] try to infer the distance, angle, and attenuation of each path from channel measurements. Precise estimation of these parameters is a challenging task. A small error of even 0.3 radians in channel phase (caused by 0.5 cm distance error at 3GHz frequency) will cap the SNR (signal to noise ratio) of the channel estimate at roughly 10 dB, which is insufficient to support MIMO transmissions. Low bandwidths of cellular transmissions (10-100 MHz) limit the accuracy of these distance measurements and cause errors in accurate path estimates. In fact, as we show in Sec. 6.3, past work achieves an SNR of 5 to 8 dB due to such errors. In practice, MIMO transmissions would need channel accuracy in the range of 15 to 20 dB, which is nearly 10 times more accurate (dB uses the log scale).

Path estimation is not required for MIMO: Note from Eq. 2, MIMO pre-coding does not need absolute channel values for accurate pre-coding. If the same constant is multiplied to the channel across different antennas, it can be abstracted out and does not impact the pre-coding matrix computation. Therefore, we do not need to accurately estimate all the parameters of the underlying paths to obtain the precise wireless channel. For instance, adding the same distance to all paths does not impact the relative channel. Similarly, adding the same phase shift to all reflectors does not impact the relative channel. This implies that channel inference for MIMO does not need to accurately infer all the parameters for each path. Motivated by this insight, we propose a shift in paradigm: from inferring intermediate paths to an end-to-end approach that focuses directly on the channel inference problem.

4.2 FIRE's Architecture

We leverage data-driven machine learning to model the end-to-end problem of downlink channel estimation. FIRE aims at generating downlink wireless channel by observing the uplink channel measured by the base station. The overall architecture of FIRE is shown in Fig.2. FIRE first performs a data transformation step on uplink channels to remove hardware errors (see Sec. 4.4) and then feeds them to a learned predictor based on variational autoencoder (VAE) [31].

We considered multiple architectures in our design and chose VAE for three reasons: (a) VAEs can accurately model the generative process of creating channels from underlying physical parameters. In the past, VAEs have been successfully applied for generative modeling in a variety of tasks including image extrapolation [34], text generation [25], and link prediction in graphs [32]. This type of generative modeling is not inherently possible with discriminative models such as fully-connected networks which ignore the underlying generative process and try to directly compute the downlink channel from the uplink channel. (b) VAEs are more powerful than other architectures like classic autoencoders [24] used for representation learning. Therefore, they enable higher accuracy for channel inference, as we show in Sec. 6.5. (c) The latent space representation in a VAE is usually disentangled [41] and is, therefore, a natural candidate for getting more insights into what the network is learning. This interpretability is not possible with traditional classifiers based on fully connected or LSTM architectures.

Traditionally, VAE learns the probability distribution of the training dataset $\mathbf{x}$ by first encoding $\mathbf{x}$ into a lower dimensional latent space $\mathbf{z}$ via an encoder network. The encoder learns the distribution

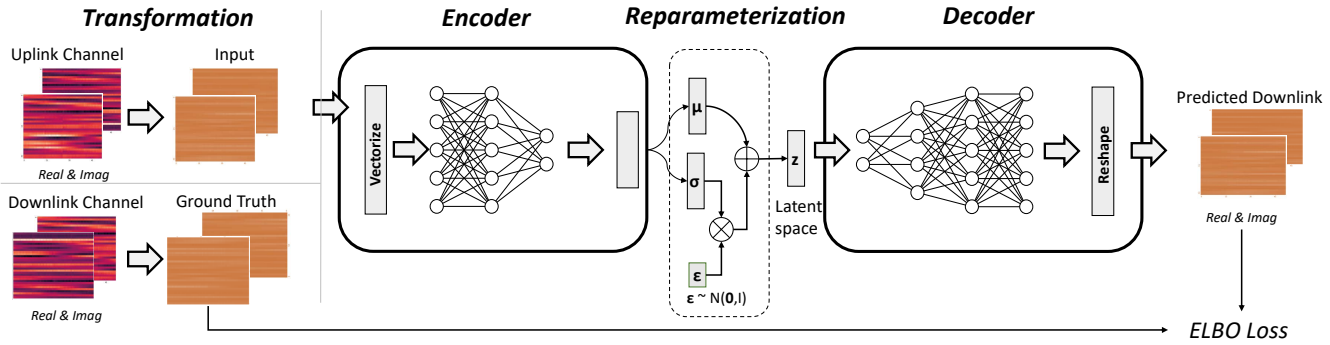


Figure 2: Our model adopts variational autoencoder which consists of an encoder network and a decoder network. The uplink channel is first processed to remove the CFO and packet detection delay. The processed channel is fed into the encoder network which outputs a probabilistic latent space representation (μ, σ). FIRE then samples the latent space vector and feeds it through the decoder to generate downlink channels.

of $\mathbf{z}$ maximizing $p(\mathbf{z}|\mathbf{x})$. Next, samples are drawn from this distribution and a decoder network decodes the samples to generate new data from the distribution of $\mathbf{x}$. Unlike traditional VAEs, our outputs and inputs are not the same. However, based on the hypothesis for our end-to-end approach in Sec. 4.1, the downlink channel can be obtained from the uplink channel. Therefore, we learn the distribution of the downlink channel given the uplink channel (instead of learning the distribution of the uplink channel as would be the case in traditional VAEs). In FIRE, we use the encoder network to learn a lower-dimensional interpretable representation $Z = \mathbb{R}^l$ for our target distribution. The decoder then uses the obtained representation to predict the downlink channel.

Let $\{\mathbf{u}^i\}_{i=1}^N$ and $\{\mathbf{d}^i\}_{i=1}^N$ be the uplink and downlink channel values consisting of N datapoints from the training set, $\mathbf{u}^i \in U = \mathbb{R}^{2\times N_a \times N_b}$, $\mathbf{d}^i \in D = \mathbb{R}^{2\times N_a \times N_b}$ where N_a and N_b are namely the number of antennas and the number of OFDMA subcarrier frequencies. The value 2 corresponds to the real and imaginary parts of the complex values channel. To avoid clutter, we use $\mathbf{du} = \mathbf{d}|\mathbf{u}$ ($\mathbf{d}$ *given* $\mathbf{u}$). The training objective in our context is to maximize the log-likelihood of the predicted downlink channel:

$$\sum_{i=1}^{N} \log p(\mathrm{du}^i) \tag{3}$$

However, computing $\log p(\mathrm{du}^i)$ exactly is an intractable problem and therefore it is approximated by the evidence lower bound (ELBO):

$$ELBO = \mathop{\mathbb{E}}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{du}^i)} \log p\left(\mathbf{du}^i \mid \mathbf{z}\right) - D_{\mathrm{KL}}\left(q\left(\mathbf{z} \mid \mathbf{du}^i\right) \| p(\mathbf{z})\right) \tag{4}$$

where the first term is the reconstruction loss and the second term corresponds to the KL divergence between the latent space distribution given the observations $q(\mathbf{z}|\mathbf{du}^i)$ and the multivariate standard normal distribution $p(\mathbf{z})$ with mean 0 and variance 1 in the latent space. $q(\mathbf{z}|\mathbf{du}^i)$ is also a multivariate Gaussian distributions, however, its mean and variance are unknown apriori and learned during training. Minimizing the KL divergence between $q(\mathbf{z}|\mathbf{du}^i)$ and the standard normal distribution ensures two properties essential for our extrapolation task. These are (a) continuity: the decoded downlink channels corresponding to samples close to each other in the latent space should not be too different, and (b) completeness: any point sampled from the latent space corresponds to a valid downlink channel.

Encoder Network: Given the transformed channel estimation matrix with the dimensions of $2 \times \mathrm{N}_a \times \mathrm{N}_b$, we first flatten it into a vector, then feed it into a three layer fully connected (FC) network with LeakyReLU [37] as the activation for the first two layers and no activation for the third one. The output of the FC network yields the mean vector μ and the variance vector σ, as shown in Fig.2. During training, the encoder network maximizes the likelihood of the latent distribution $q(\mathbf{z}|\mathbf{du}^i)$ generating $\mathbf{d}^i$ given $\mathbf{u}^i$.

Decoder Network: The decoder neural network takes as input a sample from the latent space and predicts the downlink channel value. To enable a strong prediction ability, we use four layers fully connected network (one more layer than the encoder network). The first three layers have LeakyReLU activation while the last one has Tanh activation. We reshape the output vector back to size $2\times\mathrm{N}_a\times\mathrm{N}_b$ to get the downlink channel matrix. During training, the decoder network samples $\mathbf{z}$ from $q(\mathbf{z}|\mathbf{du}^i)$ and learns the parameters maximizing $p(\mathbf{du}^i|\mathbf{z})$.

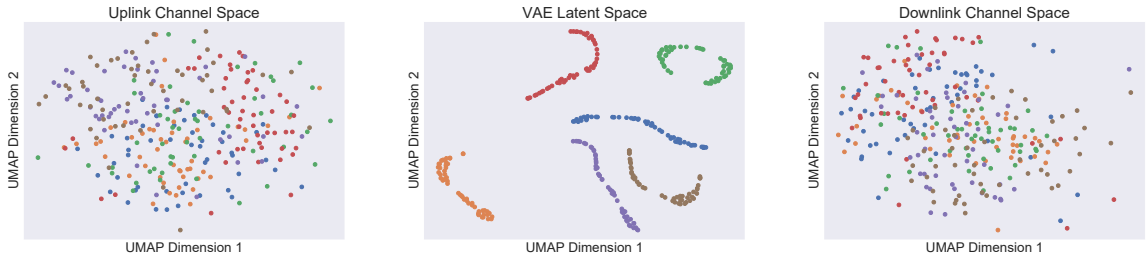
Loss Implementation Details. The loss function in Eq.4 consists of the reconstruction loss and the KL divergence. The reconstruction loss in our context is defined as the Mean Square Error (MSE) loss, $MSE\left(H_{pre}, H_{gt}\right)$, where the H_{pre} is the predicted downlink channel and H_{gt} is the ground truth downlink channel after the data transformation. We do not use the raw downlink as the ground truth, since it contains signal distortion and we only care about the relative values of the channel matrix in MIMO technologies. The KL divergence is computed in a closed form as:

$$L_{\mathrm{KL}} = \frac{1}{2}\sum_{j=1}^{l}\left(\mu_j^2\left(\mathrm{u}^i\right) + \sigma_j^2\left(\mathrm{u}^i\right) - \log\sigma_j^2\left(\mathrm{u}^i\right) - 1\right) \tag{5}$$

Overall the ELBO loss can be rewritten as:

$$ELBO = MSE\left(H_{pre}, H_{gt}\right) - \beta L_{KL} \tag{6}$$

where β is a hyperparameter that balances the contributions of the reconstruction loss and the KL divergence loss during training[23].



(a) Uplink Channel Representation (b) Latent Space (c) Downlink Channel Representation

Figure 3: We visualize the ground truth channel matrices in the uplink and downlink in 2D space by using UMAP method and the results are shown in (a) and (c) respectively. We also do the same thing to the latent space in our VAE network and the result is shown in (b). Channel matrices are collected from 6 clients at different locations which are shown with different colors. Our neural network manages to disentangle the uplink channel values into separated groups depending on different locations and reconstruct them back to the downlink.

4.3 FIRE's Interpretability

Channel values are complex numbers and as such, are hard to debug. Consider the real and imaginary matrices shown in Fig. 2 for example. A network operator cannot gain much insight out of these channel values. Therefore, interpretability is a challenge for an end-to-end design that ingests uplink channel matrices and outputs downlink channel matrices. FIRE's VAE design presents an improvement over this black-box scenario. Specifically, the VAE learns a latent representation that disentangles the uplink channel values into physically relevant information.

To highlight this point, we showcase a simple experiment. We use the uplink and downlink channel matrices collected from six clients placed at six different locations in non-line-of-sight setting (more details of our dataset are in Sec. 5.1). These clients are physically separated from each other and experience distinct physical paths. First, we consider the uplink channel measurements from these clients. To visualize the channel matrices across time from these clients, we use the Uniform Manifold Appoximation and Projection (UMAP) method [42]. UMAP is a standard method used for non-linear dimensionality reduction and embeds the channel matrix into a 2-D space. We plot the uplink channel for different measurements in Fig. 3(a). As shown, clients (denoted by different colors) are randomly scattered across the 2-D space, showing intense entanglement between uplink channels from different clients.

Next, we perform the same analysis on the latent space learnt by our VAE and plot the 2-D representation in Fig. 3(b). In the figure, the clients at different locations form different clusters indicating that the different locations get mapped to different points in our latent space. Also, it is worth noting that purple and brown-colored clients are geographically close to each other in the real world and are far from the orange-colored client. This is also indicated in the latent space representation. Moreover, note that each location doesn't map to the same point. This is because the channel changes due to reasons other than the location change like environment mobility, hardware variations over time, etc. Overall, our latent space representation provides a way for an operator to identify the most relevant factors determining the downlink channel. This insight can be leveraged by the network operators to debug the system, e.g., whether a particular location or hardware is prone to receiving bad signals.

While the encoder network shows the above ability to disentangle features, the decoder network maps the latent space back into the downlink channel values as shown in Fig.3(c) and features become entangled again showing the same pattern as for the uplink.

4.4 Countering Hardware Offsets

When estimating the uplink channels at the base station, clients send preamble symbols aimed specifically for channel estimation and packet detection. However, the channel estimated in this manner usually contains signal distortion caused by the career frequency offset (CFO) and the hardware detection delay. These offsets introduce random distortions to the channel values. If we train a neural network with these distortions, it will try to fit to the randomness and incur large errors. On the other hand, these random distortions are impossible to remove. To solve this challenge, we introduce a data transformation scheme that standardizes the effect of these distortions on the wireless channel and reduces their effect on our network. Our data transformation scheme converts the raw channel values into a channel representation that is invariant to these effects. The core insight is: MIMO techniques care about relative channel across different antennas and subfrequencies, not the absolute channel values. Based on this insight, our algorithm performs three steps defined below.

Signal Strength Scaling: The measured channel matrix could have significant differences in absolute values due to different distances of clients, environmental changes, user mobility, etc. However, for MIMO systems, we do not typically care much about the absolute scaling of the channel matrix but only the relative scaling. Therefore we normalized the channel matrix $\mathbf{H}$ as: $\mathbf{H}_{norm} = \frac{\sqrt{N}}{\|\mathbf{H}\|}\mathbf{H}$, where N is the number of antennas in the base station and $\|\mathbf{H}\|$ is the second-order Frobenius norm of $\mathbf{H}$.

Eliminating the effect of Carrier Frequency Offset: In realistic hardware settings, the CFO is usually caused by frequency misalignment in oscillators between the base station and the clients. This frequency offset, denoted by Δf, will continuously add a phase shift in the received signal $\hat{x}(t)$ with respect to the true signal $x(t)$ over time: $\hat{x}(t) = x(t)e^{j2\pi\Delta f t}$. In MIMO-OFDM systems, the antennas are often co-located as an antenna array. Hence, it is valid to

assume that only one oscillator is referenced at either the transmitter side or the receiver side. As a result, a single CFO-induced phase value is added to channel measurements across the whole antenna array at the base station. We can thus eliminate the phase rotation by dividing the channel matrix by the value at the first antenna measured at the same time. Note that key MIMO techniques only care about the relative channel values among antennas and this kind of division doesn't affect the relationship in antenna array.

Mitigating Hardware Detection Delay: There is a delay Δt between the time when the signal reaches the RF front-end and the time when the signal is actually detected by the decoding algorithm. This delay will add a further phase rotation $2\pi f_i \Delta t$ at every subcarrier i. Because the frequencies of the subcarriers increase linearly with respect to the subcarrier index, this delay adds a slope in phase at every measurement. We standardize this delay by zeroing out the slope of the phase across subcarriers for the first antenna. Specifically, we use linear regression to identify the slope of the phase across subcarriers (say m) for channel measurements on the first antenna. Then, we subtract the value mf_i from the phase of the i^{th} subcarrier on each antenna. Note that this procedure does not remove the random detection delay – in fact, it is impossible to remove. However, it standardizes our representation across different measurements. Different measurements of the same channel will now appear identical to the neural network.

We apply the above three transformations to the channel to obtain a standardized channel matrix that is suitable for neural network training. Now, we have a channel matrix of size $[N, B]$, where N is the antenna number at the base station and B is the subcarrier numbers in OFDM. However, this still cannot be used for neural network inference which only accepts real values as input. To solve this problem, we consider the complex valued channel matrix as a real valued matrix with two channels and put the real part of channel into the first channel and the imaginary part into the second. We don't use the absolute values and phase to avoid phase wrapping. We further divide the channel matrix after the fore-mentioned three transformations by the maximum absolute value in the matrix, and then scale the channel matrix so that all values lie in the interval $[-1, 1]$, to facilitate training.

5 IMPLEMENTATION

We present implementation details of FIRE below.

5.1 Dataset Selection

To satisfy the need of massive MIMO in real-world deployments, our dataset should have the following characteristics:

- **Real-world**: The dataset should be collected in the real-world instead of simulation to model all real-world effects: multipath reflections, hardware effects, noise, etc.
- **Antenna number at base stations**: Previous work [17] has shown that massive MIMO is among the most critical technology in next-generation networks which has the potential to boost the throughput by increasing the number of antennas at the base station. However, building a base station with tens to a hundred antennas from scratch faces several technical issues and needs sophisticated hardware expertise [55].

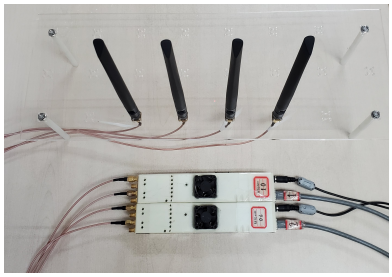


Figure 4: Hardware Platform: Our base station with a 4-antenna uniform linear array (ULA).

- **User mobility**: We envision that our trained neural network could be used not only in the static environment but also in cases when users served by the base station are moving, making the dataset consistent with real-world application demands.

There are several open-access platforms that provide programmable interfaces to conduct the Channel State Information (CSI) collection, however, they at least miss one of the above needs. To name a few, Colosseum [12] allows up to 256×256 100 MHz RF channels by using hundreds of USRP X310s but the channel environment is purely simulated and unchangeable. While providing access to a 64-antennas base station, Powder platform [14] only maintains fixed user ends. On the other hand, some ready-made CSI datasets [19, 38, 43] collected by various research teams either only contains static clients or are generated in a custom simulator.

Based on the above concerns, we chose the Argos Channel dataset [54] as our training dataset. Argos dataset has been collected on the RENEW wireless testbed [1] and contains real-world channel state information measured in diverse environments with up to 104 antennas at the base station serving 8 users. The data contains channel measurements performed in a variety of user mobility patterns in both line-of-sight (LOS) and non-line-of-sight (NLOS) settings. The dataset contains some static locations as well, but since the cellular networks mostly cater to mobile users, we focus on the mobility use case in our evaluation.

The raw traces contain the received 802.11 Long Training Symbols (LTS), thus we could extract the CSI value from them by doing the channel estimation. The trace durations in the dataset range from tens of seconds to 20 minutes with different channel sounding intervals. These traces are collected using omnidirectional monopole antennas spaced 63.5 mm apart which is half a wavelength at 2.4GHz and one wavelength at 5GHz. The bandwidth is 20 MHz with 64 OFDM subcarriers in the symbol. Similar to FDD systems, we use disjoint frequencies for uplink and downlink. We split the channel measurements in Argos such that the first half of the frequency band is the uplink and our goal is to predict the second half using the neural network. Furthermore, the uplink and downlink channels are separated by guard bands. Each data point is a matrix with size of [2, antenna number, subcarriers]. Unless otherwise specified, the uplink and downlink channels consist of 26 subcarriers each after removing the guard bands.

5.2 Hardware Implementation

We implement a four-antenna base station using the Skylarkwireless Iris-030[57] software radio platform. We use two Iris-030 devices with broadband analog front-ends and connect them in a

daisy-chain manner to synchronize them as shown in Fig. 4. We also use another Iris device as a two-antenna user device. Our software radios operate in 2.4GHz ISM band with a bandwidth of 10MHz. We operate these devices in FDD manner, i.e. the uplink and downlink operate on different frequency bands. The uplink and downlink are separated by 20 MHz. Since the Iris device allows us to set the transmit and receive frequency independently, we do not need to switch frequencies constantly to enable FDD operation.

We fix the base station and move the user end to 50 different locations in an indoor space to collect the channel measurements. During the channel measurement, we send two Long Training Sequences (LTS) for channel estimation along with five OFDM data symbols. We use the estimated channel for decoding the data symbols to verify that our channel estimation is sound. This dataset enables a rich diversity in our channel measurements. We randomly select 10 locations as our test dataset. Since we use two devices to implement the base station, hardware resets and clock resets add random phase offsets and timing delays to the channel estimates. To avoid this error, whenever we need to perform a reset (for example, to prevent overheating), we perform two measurements: one before the reset and one after the reset at the same static location. We use these two measurements to remove the random phase offsets and timing delays in software.

Finally, note that our hardware implementation is limited to a four antenna base station, therefore we use the Argos dataset described above as the default evaluation method. We use our hardware implementation to demonstrate FIRE's robustness to different environments, frequency bands, and measurement devices.

5.3 Network Structure

We did a network structure search in terms of depth and we found that when the network gets deeper, the prediction accuracy increases. But the accuracy won't increase too much after the network is deeper than 4 layers for both encoder and decoder. For the encoder network, we use 3 layers and the hidden layer sizes are 64, 64, and 100. This leads to a latent space with dimension 50 (50 values each for the means and the variances). For the decoder network, we experimentally found that a deeper network will give a better reconstruction performance. Thus, we leverage a four-layer FC network – the hidden sizes are 50, 64, 64, and $2 \times N_a \times N_b$ which depends on experiments.

We implement the neural network on Pytorch[47] and the parameters for training include batch size of 512, learning rate of 10^{-4}, β in Eq.4.2 of 0.1 and Adam optimizer for adaptive learning rate optimization. We use these hyperparameters for all of our experiments. While some tuning is useful, we found that FIRE could perform well under a wide range of hyperparameter values. Thus, we did not use existing hyperparameter tuning methods. We obtain the model after 200 epochs of training and the memory footprint of the model is 0.5MB.

The total number of points in our dataset is 100K (80K for the training and 20K for the testing). To ensure separation in data points and ensure device independence, the training set and test set are collected using different clients. We train using data on one client and test on data from seven other clients. We intend to show that FIRE, once fully trained, could generalize to unseen users under varieties of scenes. The test set contains data from multiple environments: line of sight, non-line of sight, indoors, outdoors, etc. Finally, all of our training and test data is collected in mobile scenarios, where the client is in motion. We do not use data from static scenarios available in the Argos dataset.

6 RESULTS

We present an empirical evaluation of FIRE below.

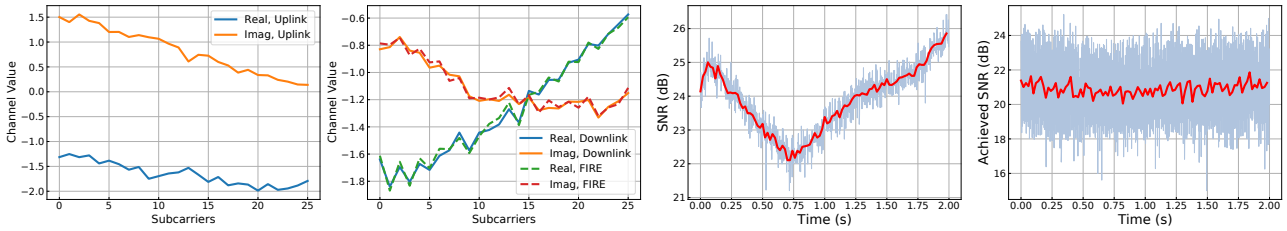
6.1 Baselines

We compare FIRE against the following baselines:

(1) **R2F2**: R2F2 [59] predicts the downlink channel at frequency band $F2$ by observing the uplink channel at frequency band $F1$. Given uplink channel values, it solves an optimization problem to identify the number of paths and the corresponding path parameters – angle of arrival, travel distance, attenuation, and reflections. It uses these parameters to estimate the downlink channel.
(2) **OptML**: OptML [9] takes an approach similar to R2F2, but uses a neural network to predict the underlying path information from uplink channels. By leveraging this information, it accomplishes the cross-band prediction in an antenna array by further using a multi-antenna optimization algorithm. We adapt the author's code for this baseline.
(3) **FNN**: There is another line of work [6, 26, 64] that use fully connected layer for cross-frequency channel estimation directly. These networks use discriminative models to covert an uplink channel to the corresponding downlink channel. This line of work has been evaluated with simulated data and therefore, cannot deal with real world issues like hardware effects. Therefore, the channel prediction does not work on real-world datasets. For fair comparison, we augment these methods with FIRE's data transformation algorithm before using the neural network. We implement this baseline via a five-layer fully connected network with batchnorm and dropout, and tune its performance with its best hyperparameters.
(4) **Codebook**: Both base station and clients maintain a codebook which consists of a large number of vectors manufactured by predefined rules [29]. Clients measure the channel locally and choose the closest vectors in the codebook, then send the index back to the base station. Note that the codebook method differs from above three baselines, for it doesn't eliminate the channel feedback but reduces it. We choose the 8-bit random codebook as used by [29], i.e., the quantization vectors are drawn from a complex Gaussian distribution. This is also the method used by many standard implementations, as recommended by the 3GPP physical channel standard [3].

6.2 Microbenchmark

We now present microbenchmarks to provide insights into the operation of our system. We trained FIRE on the Argos dataset with an 8-antenna base station and a client at different locations. Fig.5 plots the results from a representative run. Given the uplink channel value measured at 8 antennas, FIRE predicts the downlink channel value from the same 8 antennas to the client. Fig. 5(a) plots the real and imaginary parts for the uplink channel on a single antenna, and



(a) Uplink Channel (b) Downlink Channel & Prediction (c) Signal SNR during movement (d) Channel Prediction SNR

Figure 5: Microbenchmark: (a) Uplink channel values for an antenna. (b) Downlink channel ground truth and predictions for the same antenna. FIRE's predictions are accurate even though the channel varies across frequencies. (c) As a client moves, it's SNR varies. (d) FIRE can achieve high prediction accuracy throughout the motion.

the Fig. 5(b) plots the predicted downlink channel (along with the ground truth). Note that, the uplink and downlink channels look different compared to each other. Yet, FIRE can accurately predict the downlink channel.

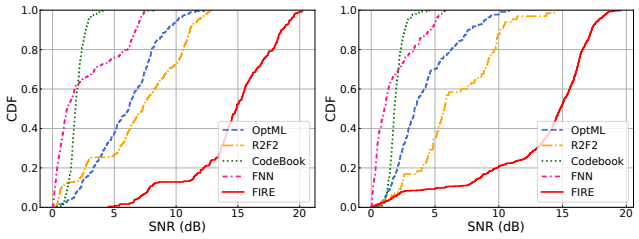
We, then, tested FIRE on a different client that is not part of the training set. The client moves with respect to the base station, leading to SNR changes.The signal SNR is plotted in Fig.5(c), the red line is the average value measured every 20ms, showing that the SNR decreases when the client moves away from the station and vice versa. We use FIRE to predict the downlink channels for each data point during motion. We calculate the SNR of the predicted channel by comparing the predicted channel H and the ground truth channel H_{gt} using:

$$SNR = -10log_{10}\left(\frac{\|H - H_{gt}\|^2}{\|H_{gt}\|^2}\right) \quad (7)$$

As shown in Fig. 5(d), the SNR of the predicted channel is consistently very high (conversely the error in predicted channel is very low), for a mobile client in continuous motion. We will compare the SNR of the predicted channel across a larger dataset on several baselines below.

6.3 Channel Prediction Accuracy

We evaluate and compare the channel quality (SNR) using Eq.7, under both LOS and NLOS settings. We plot the results in Fig. 6. Since the NLOS environment does not have a direct path and is saturated with complicated multipath effects, it increases the level of difficulty for optimization-based algorithms (e.g., R2F2 and OptML) to find the correct multipath parameters by simply looking into the uplink channel measurement. However, our method maintains high accuracy in both LOS and NLOS settings because of its end-to-end prediction methodology and our specialized data transformations. Specifically, FIRE achieves a median accuracy of 14.87dB (10th percentile: 7.89 dB, 90th percentile: 18.5dB) in LOS and 14.81 dB (10th percentile: 5.77 dB, 90th percentile: 17.29 dB) in NLOS settings. In comparison, the next best baseline is R2F2 which achieves median SNR of 7.29 dB in LOS and 5.73 dB in NLOS settings. Note that, FIRE's 10th percentile SNR outperforms R2F2's median SNR. Overall, FIRE's SNR is 7.64 dB better than R2F2, 8.61 dB better than OptML, 12.56 dB better than codebook and 11.98 dB better than FNN in LOS environment. For the NLOS environment, our accuracy is also much higher compared to baselines, and the SNR is 6.96 dB higher than R2F2, 9.16 dB over OptML, 11.32 dB over Codebook



(a) Line of Sight (b) Non Line of Sight

Figure 6: Channel prediction accuracy, measured as channel SNR, under different environments. FIRE's predictions significantly outperform all baselines.

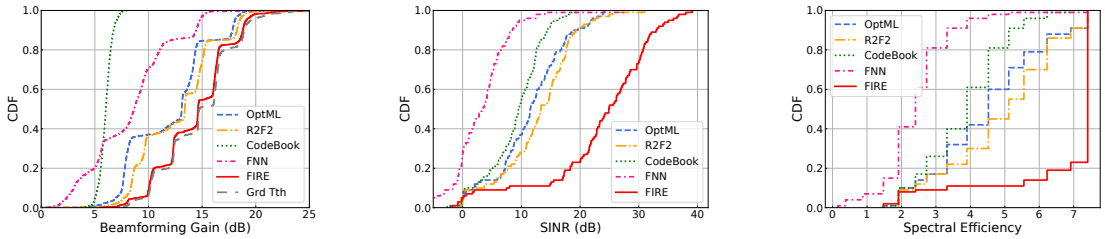
and 11.57 dB over FNN. Note, SNR is measured on log scale (10 dB corresponds to 10X gain).

We note that the errors achieved by R2F2 and OptML on our dataset are consistent with the accuracies reported in the original papers. It's also worthwhile to mention that the codebook method maintains a stable performance in both environments. This is because it quantifies the channel value using the codebook (instead of trying to estimate paths) available both in the base station and user end. Finally, our VAE design allows a latent representation that is continuous and complete, which is not ensured by FNN. Hence, our design is more suited to the extrapolation task and outperforms FNN by over 10 dB.

6.4 Beamforming Performance

Next, we analyze how the channel accuracy gains translate into massive MIMO performance gains. We first analyze beamforming gains. In beamforming, a base station uses multiple antennas to steer a signal to a specific client. This is particularly useful for low SNR clients to gain additional signal strength. Typically, past work like R2F2 and OptML do well on beamforming as the channel accuracy requirement for beamforming is low. For example, a 2 antenna setup achieves an optimal beamforming gain of 3 dB with perfect channel information. With a channel SNR of 10 dB, the gain reduces to 2.6 dB – a small sacrifice.

To compute the beamforming gain, we use maximal ratio combining [15] The ideal gain of using multiple antennas is given by: $\frac{|\mathbf{h_g}^*\mathbf{h_p}|}{M|h_{g0}|}$, where $\mathbf{h_g}$ is a $M \times 1$ vector of the ground truth channel values, h_{g0} is the channel using a single antenna, $\mathbf{h_p}$ is a vector



(a) Beamforming Gain Comparison (b) MU-MIMO SINR (c) Spectral Efficiency Comparison

Figure 7: MIMO Applications: (a) Beamforming Gain: FIRE achieves beamforming gain within 0.4 dB of the achievable gain (ground truth). (b) MU-MIMO SINR: For a 8 antenna, 2 client system, FIRE achieves >20 dB SINR for over 80% of experiments and can support the highest data rate. The spectral efficiency in this setup is plotted in (c).

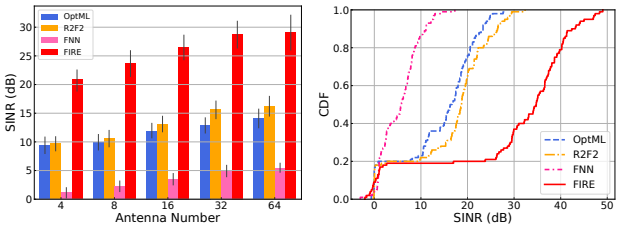
of predicted channel values, and M is the number of antennas. * denotes the conjugate operation.

We plot the beamforming gain using an 8 antenna base station for different baselines in Fig.7(a). First, note that our gains are very close to beamforming gains achieved with perfect channel estimates (ground truth). We just have a 0.37 dB loss compared to perfect channel measurements. Second, we outperform other baselines: our median gain is 8.59, 1.51, 1.23, and 5.94 dB higher than codebook, OptML, R2F2, and FNN respectively. This shows that FIRE can successfully enable accurate beamforming.

6.5 Multi-user MIMO

A more complex multi-antenna technique is multi-user MIMO (MU-MIMO). In MU-MIMO, a base station uses its multiple antennas to transmit to multiple clients simultaneously. MU-MIMO is preferred in high SNR scenarios where additional SNR to a single client doesn't provide any benefit in data rate. Therefore, it is advisable to transmit to multiple devices simultaneously. MU-MIMO has a very high bar for channel accuracy because any error in channel accuracy means the signal intended for one client will leak into signal intended for another client causing interference. The metric of interest in this case is SINR (signal to interference and noise ratio). As a ballpark estimate, with 2 base station antennas transmitting to two clients and 8-bit perfect channel measurements, MU-MIMO can achieve theoretical SINRs up to 24 dB. However, a channel SNR of 10 dB will cap the SINR at 10 dB on average. Compared to beamforming, MU-MIMO performance is hurt more by errors in channel estimates.

To evaluate MU-MIMO performance, we randomly sample two clients to transmit data to our 8-antenna base station. We repeat this experiment 300 times and report the resulting SINR. For MU-MIMO, we use the zeroforcing method to transmit to multiple antennas simultaneously (described in Sec. 2). The clients are sampled across both LOS and NLOS settings. The results of this experiment are plotted in Fig. 7(b). First of all, note that FIRE achieves a median SINR of 24.90 dB (10th percentile: 8.01 dB, 90th percentile: 33.09 dB). Prior work [48] has shown that SINRs above 20 dB can support the highest data rate. In more than 80% of our experiments, FIRE can support the highest data rate for two clients simultaneously. This is rare for other baselines – our best baseline, R2F2, achieves this



(a) Antenna Variation (b) 64-antenna MU-MIMO

Figure 8: Massive MIMO Results: FIRE's performance scales up with the number of antennas. (a) The SINR for our 2 stream transmission goes up with the number of antennas, but is limited for baselines because of errors in channel predictions. (b) CDF of MU-MIMO SINR at clients with a 64-antenna base station.

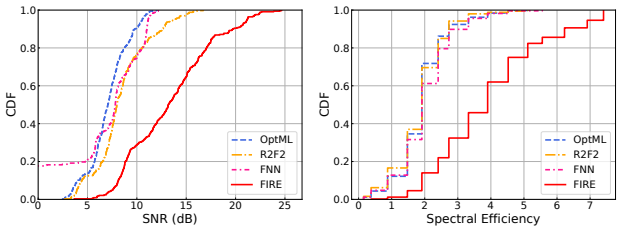
outcome in nearly 10% of our experiments. To complete the comparison, the median SINRs for R2F2, OptML, FNN, and Codebook are: 13.33 dB, 11.53 dB, 3.41 dB, and 9.52 dB respectively.

We also note that FIRE's performance is comparable to explicit 8-bit channel feedback (24 dB SINR) received per antenna per device. As we showed in Sec. 2, this feedback is unsustainable for clients today due to the spectrum overhead of transmitting this channel.

Next, we convert these results into spectral efficiency (bits per second per Hz) that shows how much data can be transmitted using our method. We convert our SINR measurements into channel quality index used in 5G standards using [48] and then use it to identify the right modulation and coding scheme using the 5G NR standard document [4]. This gives us the spectral efficiency achieved using different SINR values. We plot the spectral efficiency in Fig. 7(c). As shown in the figure, FIRE again outperforms the baseline methods significantly. The average spectral efficiency for FIRE is 6.69 bps/Hz, as compared to 4.89 bps/Hz for R2F2 – an improvement of 1.36 times.

6.6 Massive MIMO Scaling

Now, we investigate massive MIMO scaling: how does FIRE's performance scale as we increase the number of base station antennas. We still focus on the MU-MIMO application. Note that, more antennas on the base station can support (a) higher beamforming gains with narrower beams, (b) multiple clients, and (c) resilience to low channel quality from a subset of antennas. In Fig. 8(a), we compare



(a) Prediction Accuracy (b) Spectral Efficiency Comparison

Figure 9: Hardware Platform Results: (a) Channel prediction accuracy on the indoor testbed. FIRE outperforms all the feedback-free baselines significantly. (b) MU-MIMO SINR result in our testbed with 4 antennas, FIRE achieves competitive SINR in our own platform.

the SINR in the MU-MIMO case from the previous section as the number of base station antennas scale up. As expected, increasing the number of antennas from 8 to 64 increases the SINR from 20.71 dB to 28.92 dB for FIRE. This is expected because increasing the number of antennas allows the base station to cancel interference better and to focus its narrower beams on the intended client. However, the baselines cannot fully leverage this because of the errors in their channel predictions and are limited to the best value of 16.20 dB at 64 antennas. We also plot the 64-antenna MU-MIMO result in Fig. 8(b), showing that FIRE reaches the average SINR of 29.11dB which is 15 dB, 13 dB and 23 dB better than OptML, R2F2 and FNN respectively. This result demonstrates that FIRE can support base stations with large number of antennas.

6.7 FIRE Hardware Platform Results

We train and test FIRE on the dataset collected on our hardware platform for further performance validation. The objective of this analysis is twofold: (a) to demonstrate that the performance of FIRE translates to different hardware architectures, and (b) to ensure that the reciprocity assumption in our evaluation using Argos holds true. Fig. 9(a) shows the prediction accuracy comparison against other feedback-free methods. We see that FIRE achieves a median channel SNR of 13.19dB (10th percentile: 8.13 dB, 90th percentile: 19.72 dB), which is 5.21, 6.13, and 5.37dB higher than R2F2, OptML, and FNN respectively. Note that, the performance of FIRE on our hardware is similar to the performance on the ARGOS dataset shown in Fig. 6 – there is a slight decrease in median SNR (from 14.87 dB to 13.19 dB) due to the decreased number of antennas from eight to four. Smaller antenna number reduces the ability to resolve between different physical paths and hence reduces channel prediction accuracy for all schemes. This result validates FIRE's performance for a bidirectional FDD dataset collected on our hardware platform.

We also compute the spectral efficiency achieved in a 4×2 MU-MIMO transmission in Fig. 9(b). As before, we select two random client locations from our dataset as clients. As expected, FIRE significantly outperforms the other baselines. The average spectral efficiency for FIRE is 3.92 bits per second per Hz. This is 2.09 times better than R2-F2, 2.03 times better than Opt-ML, and 1.93 times better than FNN methods. This shows that FIRE can enable successful MU-MIMO operation in FDD systems.

Batch Size	1	2	4	8	16	32	64
Runtime/ms	1.30	0.80	0.51	0.38	0.33	0.30	0.40

Table 1: FIRE Average Run Time of a single CSI prediction on GPU, using different batch sizes

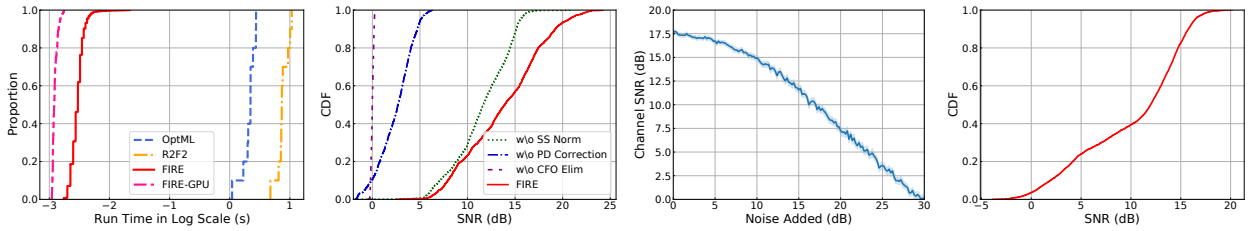
6.8 FIRE Algorithm Analysis

Runtime: For a channel prediction system to be useful, it must achieve runtime lower than channel coherence time (2.5 ms to 50 ms, see Sec. 2). We evaluate FIRE's runtime and compare it against the different baselines. We test all algorithms using the same CPU (Intel i7-9750H) and plot the results in Fig. 10(a). FIRE has an average run time of 3 ms, which is suitable for rapid cross-band channel estimation even in fast changing environment. FIRE achieves three orders of magnitude reduction compared to prior work primarily because prior work relies on numerical optimization for each prediction which involves multiple matrix inversions at each step. Our work, on the other hand, uses a neural network that performs a simple forward pass with multiplications. We also test the FIRE runtime on the RTX 2070MQ GPU, the median run time is 1.30ms for a batch size of 1. A standard base station can perform channel translation for multiple clients together and use a larger batch size. As we increase the batch size (Table. 1), the runtime decreases to sub-milliseconds (0.30 ms with batch size 32) due to the benefits of parallelization. The runtime decreases when batch size gets too large, i.e., 32 in our experiment, due to the limit on GPU memory. Finally, note that this runtime includes time for both steps: data transformation (to remove CFO and packet detection delay effect) and channel prediction.

Data Transformation: We measure the impact of our data transformation approach (Sec. 4.4) on FIRE's performance. To achieve this, we remove each of the preprocessing steps and measure the SNR of the predicted channel in Fig. 10(b). As shown, removing the signal strength normalization (SS Norm) reduces the median channel SNR from 13.90 dB to 11.66 dB, removing the CFO elimination (CFO Elim) step reduces the channel SNR to 0.01 dB, and removing the packet detection delay correction (PD Corr) reduces the channel SNR to 2.70 dB. This indicates that each of the proposed preprocessing steps is crucial for FIRE's performance.

Robustness: We analyze the robustness of FIRE by adding Gaussian noise to the uplink channel and then, test the accuracy of the predicted downlink channel. Note that our uplink channel is typically measured for signals with SNR in the range 20 to 30 dB. We add additional noise to stress-test the system. For each noise value, we compute the results on 100 data points in NLOS datasets. The results are shown in Fig.10(c). As the noise in uplink channel increases, the predicted channel SNR decreases gracefully. 20 dB additional noise decreases the performance from 17.6 dB to 7.5 dB. At around, 30 dB additional noise, the channel SNR goes down to zero. This shows that FIRE's channel prediction accuracy is limited by the overall SNR of the system, but degrades gracefully and provides meaningful predictions even in harsh channel conditions.

Cross Antenna Prediction: To evaluate FIRE further, we ask if we can use this architecture for other channel prediction tasks. Therefore, we ask if it is possible to reduce the channel measurement



(a) Runtime Evaluation (b) Data Transformation Effect (c) Robustness (d) Cross Antenna Prediction

Figure 10: (a) FIRE performs millisecond-level predictions. (b) FIRE's data transformation is crucial to it's performance. (c) FIRE is robust to additional noise. (d) FIRE can also perform cross-antenna channel prediction, although with reduced accuracy.

burden even further by predicting channel values across antennas. Specifically, we use uplink channel measurements on a set of 8 antennas to predict downlink channel measurement at a different set of 8 antennas on the same base station. We use a combination of LOS and NLOS data for training. We show the achieved SNR in Fig.10(d). FIRE achieves a favorable median SNR of 11.95 dB. While this channel SNR is insufficient for MU-MIMO but can support accurate beamforming. Note that, this task cannot be done by any other optimization-based baselines, which at least need the observed uplink at the same set of antenna. This result shows the potential of FIRE that it can complete the downlink channel matrix at the base station even if some antennas miss the uplink channel values. We leave improvements in this direction to future work.

7 RELATED WORK

Our work is broadly related to two lines of research:

Downlink channel prediction: Downlink channel estimation is challenging in FDD systems and has been recognized as such in most recent 3GPP releases [2]. Past work [9, 26, 27, 59] has tried multiple approaches to reduce this overhead. Our work is most directly related to R2F2 [59] and OptML [9]. R2F2 uses a signal processing approach to transform uplink channel measurements into the underlying paths traveled by the signal and then uses the paths to construct the downlink channels. OptML takes a machine learning approach for getting the underlying paths from the uplink channel instead. As we show in Sec. 4.1 and Sec. 6, such approaches are fundamentally error-prone and cannot support complex MIMO operations like MU-MIMO. Our work is different in its approach and its performance. We use an end-to-end interpretable ML architecture and deliver an order of magnitude higher accuracy in our channel estimates. Our model is also easier to train compared to OptML because we use end-to-end supervision, rather than training on physical path information that is hard to obtain in a real deployment.

There have been recent efforts to use machine learning for the downlink channel problem [6, 26, 53, 60]. Our work differs from these works in our approach and experiments. First, these approaches rely on discriminative models (convolutional or fully connected neural networks) as opposed to FIRE's generative approach. As discussed in Sec. 4.2, the generative process more accurately reflects the channel generation process. Empirically, we have compared to one discriminative model (FNN [6]) in Sec. 6 and shown that it doesn't achieve comparable performance on real-world datasets even after adding our data transformation scheme to the model. Finally, we evaluate FIRE using real-world channel measurements. In contrast, past work relies on channel models and simulations which do not capture real-world phenomena (e.g. diffraction) and hardware constraints (e.g. CFO and packet detection delay). This concern also holds true for past simulation-based signal processing approaches to eliminate or reduce channel feedback [30, 44, 46, 49, 64, 65].

Finally, recent work [66] has proposed directional training to reduce feedback in MIMO systems. This approach observes that the uplink channel and downlink channel share the same angles of propagation. Therefore, by estimating the channel propagation characteristics at these angles, one can reduce the overall feedback overhead when the number of dominant angles is smaller than the number of antennas. In contrast, FIRE is feedback-free and as our evaluation shows, can provide advantages even with a small number of antennas.

Machine Learning in Wireless Systems: Finally, we are inspired by the recent successes of machine learning in different tasks in wireless systems: human sensing [7, 36, 67, 68], indoor positioning [5, 8, 13, 58], modulation prediction [16, 63], MIMO systems [22, 33, 45], MAC protocol design [10, 28, 62] and also the application of VAE in wireless systems [35, 56]. We build on this trend. However, we differ from past work in the overall task as well as underlying techniques such as data transformation. Our VAE architecture is designed for a new task: an accurate downlink channel prediction system. We also provide a mechanism to standardize the hardware effects in this task.

8 CONCLUSION

In this work, we present FIRE an end-to-end channel prediction mechanism for frequency duplexed systems. At its core, FIRE leverages data-driven machine learning with powerful variational architectures and enables a new primitive: channel reciprocity without feedback when uplink and downlink transmission happen at different frequencies. While we apply this in the context of 5G systems, we believe our system is more broadly applicable. Our design sits well within the current trend of designing a more agile physical layer that runs in the edge or cloud. We envision computational tools such as FIRE will sit at the core of new radio designs and form the core of future 5G and 6G deployments.

Acknowledgments: We would like to thank Professor Martin Vechev from ETH Zurich for supporting the internship of Zikun. We also appreciate the constructive and profound comments from all the reviewers.

REFERENCES

[1] Renew: Reconfigurable eco-system for next-generation end-to-end wireless. https://renew.rice.edu/index.html, 2019.

[2] 3GPP. Evolved Universal Terrestrial Radio Access (E-UTRA); Radio Resource Control (RRC); Protocol specification. Technical specification (ts), 3rd Generation Partnership Project (3GPP), 2021.

[3] 3rd Generation Partnership Project (3GPP). 5g; nr; physical channels and modulation. *Technical Specification 38.211, ver. 15.3. 0*, 2018.

[4] 3rd Generation Partnership Project (3GPP). Physical layer procedures for data (3gpp ts 38.214 version 16.2.0 release 16). 2020.

[5] F. Adib, Z. Kabelac, D. Katabi, and R. C. Miller. 3d tracking via body radio reflections. In *11th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 14)*, pages 317–329, 2014.

[6] M. Alrabeiah and A. Alkhateeb. Deep learning for tdd and fdd massive mimo: Mapping channels in space and frequency. In *2019 53rd Asilomar Conference on Signals, Systems, and Computers*, pages 1465–1470. IEEE, 2019.

[7] M. A. Alsheikh, S. Lin, D. Niyato, and H.-P. Tan. Machine learning in wireless sensor networks: Algorithms, strategies, and applications. *IEEE Communications Surveys & Tutorials*, 16(4):1996–2018, 2014.

[8] R. Ayyalasomayajula, A. Arun, C. Wu, S. Sharma, A. R. Sethi, D. Vasisht, and D. Bharadia. Deep learning based wireless localization for indoor navigation. ACM MobiCom, 2020.

[9] A. Bakshi, Y. Mao, K. Srinivasan, and S. Parthasarathy. Fast and efficient cross band channel prediction using machine learning. In *The 25th Annual International Conference on Mobile Computing and Networking*, pages 1–16, 2019.

[10] N. Z. binti Zubir, A. F. Ramli, and H. Basarudin. Optimization of wireless sensor networks mac protocols using machine learning; a survey. In *2017 International Conference on Engineering Technology and Technopreneurship (ICE2T)*, pages 1–5. IEEE, 2017.

[11] E. Björnson, E. G. Larsson, and T. L. Marzetta. Massive mimo: Ten myths and one critical question. *IEEE Communications Magazine*, 54(2):114–123, 2016.

[12] L. Bonati, M. Polese, S. D'Oro, S. Basagni, and T. Melodia. Open, Programmable, and Virtualized 5G Networks: State-of-the-Art and the Road Ahead. *Computer Networks*, 182:1–28, December 2020.

[13] S. Bozkurt, G. Elibol, S. Gunal, and U. Yayan. A comparative study on machine learning algorithms for indoor positioning. In *2015 International Symposium on Innovations in Intelligent SysTems and Applications (INISTA)*, pages 1–8. IEEE, 2015.

[14] J. Breen, A. Buffmire, J. Duerig, K. Dutt, E. Eide, M. Hibler, D. Johnson, S. K. Kasera, E. Lewis, D. Maas, et al. Powder: Platform for open wireless data-driven experimental research. In *Proceedings of the 14th International Workshop on Wireless Network Testbeds, Experimental evaluation & Characterization*, pages 17–24, 2020.

[15] D. G. Brennan. Linear diversity combining techniques. *Proceedings of the IEEE*, 91(2):331–356, 2003.

[16] R. C. Daniels, C. M. Caramanis, and R. W. Heath. Adaptation in convolutionally coded mimo-ofdm wireless systems through supervised learning and snr ordering. *IEEE Transactions on vehicular Technology*, 59(1):114–126, 2009.

[17] N. Docomo. New sid proposal: study on new radio access technology. In *3GPP TSG RAN Meeting*, volume 71, pages 7–10, 2016.

[18] Ericsson Inc. Ericsson and qualcomm pioneer new carrier aggregation capabilities for 5g. https://www.ericsson.com/en/news/2020/8/5g-carrier-aggregation, 2021.

[19] J. Flordelis, X. Gao, G. Dahman, F. Rusek, O. Edfors, and F. Tufvesson. Spatial separation of closely-spaced users in measured massive multi-user mimo channels. In *2015 IEEE International Conference on Communications (ICC)*, pages 1441–1446. IEEE, 2015.

[20] X. Gao, O. Edfors, F. Rusek, and F. Tufvesson. Linear pre-coding performance in measured very-large mimo channels. In *2011 IEEE Vehicular Technology Conference (VTC Fall)*, pages 1–5, 2011.

[21] L. Hardesty. T-mobile exec touts massive mimo for both tdd and fdd bands. https://www.fiercewireless.com/tech/t-mobile-exec-says-massive-mimo-can-be-used-tdd-and-fdd-bands, 2020.

[22] D. He, C. Liu, T. Q. Quek, and H. Wang. Transmit antenna selection in mimo wiretap channels: A machine learning approach. *IEEE Wireless Communications Letters*, 7(4):634–637, 2018.

[23] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. 2016.

[24] G. E. Hinton and R. S. Zemel. Autoencoders, minimum description length and helmholtz free energy. In *Proc. Neural Information Processing Systems (NeurIPS)*, pages 3–10, 1993.

[25] Z. Hu, Z. Yang, X. Liang, R. Salakhutdinov, and E. P. Xing. Toward controlled generation of text. In *Proc. International Conference on Machine Learning (ICML)*, volume 70, pages 1587–1596. PMLR, 2017.

[26] C. Huang, G. C. Alexandropoulos, A. Zappone, C. Yuen, and M. Debbah. Deep learning for ul/dl channel calibration in generic massive mimo systems. In *IEEE International Conference on Communications (ICC)*, 2019.

[27] K. Hugl, K. Kalliola, J. Laurila, et al. Spatial reciprocity of uplink and downlink radio channels in fdd systems. In *Proc. COST*, volume 273, page 066. Citeseer, 2002.

[28] S. Jog, Z. Liu, A. Franques, V. Fernando, S. Abadal, J. Torrellas, and H. Hassanieh. One protocol to rule them all: Wireless network-on-chip using deep reinforcement learning. USENIX NSDI, 2021.

[29] F. Kaltenberger, D. Gesbert, R. Knopp, and M. Kountouris. Performance of multi-user mimo precoding with limited feedback over measured channels. In *IEEE GLOBECOM 2008-2008 IEEE Global Telecommunications Conference*, pages 1–5. IEEE, 2008.

[30] M. B. Khalilsarai, S. Haghighatshoar, X. Yi, and G. Caire. Fdd massive mimo via ul/dl channel covariance extrapolation and active channel sparsification. arXiv ePrint, 2018.

[31] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[32] T. N. Kipf and M. Welling. Variational graph auto-encoders. *CoRR*, abs/1611.07308, 2016.

[33] A. Klautau, P. Batista, N. González-Prelcic, Y. Wang, and R. W. Heath. 5g mimo data for machine learning: Application to beam-selection using deep learning. In *2018 Information Theory and Applications Workshop (ITA)*, pages 1–9. IEEE, 2018.

[34] T. D. Kulkarni, W. F. Whitney, P. Kohli, and J. B. Tenenbaum. Deep convolutional inverse graphics network. In *Proc. Neural Information Processing Systems (NeurIPS)*, pages 2539–2547, 2015.

[35] J. Liu, F. Chen, J. Yan, and D. Wang. Cbn-vae: A data compression model with efficient convolutional structure for wireless sensor networks. *Sensors*, 19(16):3445, 2019.

[36] X. Liu, J. Cao, S. Tang, and J. Wen. Wi-sleep: Contactless sleep monitoring via wifi signals. In *IEEE Real-Time Systems Symposium*, 2014.

[37] A. L. Maas, A. Y. Hannun, and A. Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, page 3. Citeseer, 2013.

[38] À. O. Martínez, E. De Carvalho, and J. Ø. Nielsen. Towards very large aperture massive mimo: A measurement based study. In *2014 IEEE Globecom Workshops (GC Wkshps)*, pages 281–286. IEEE, 2014.

[39] T. L. Marzetta. Noncooperative cellular wireless with unlimited numbers of base station antennas. *IEEE Transactions on Wireless Communications*, 9(11):3590–3600, 2010.

[40] T. L. Marzetta, E. G. Larsson, H. Yang, and H. Q. Ngo. *Fundamentals of Massive MIMO*. Cambridge University Press, 2016.

[41] E. Mathieu, T. Rainforth, N. Siddharth, and Y. W. Teh. Disentangling disentanglement in variational autoencoders. In *International Conference on Machine Learning (ICML)*, 2019.

[42] L. McInnes, J. Healy, and J. Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[43] C. Mehlführer, J. C. Ikuno, M. Šimko, S. Schwarz, M. Wrulich, and M. Rupp. The vienna lte simulators-enabling reproducibility in wireless communications research. *EURASIP Journal on Advances in Signal Processing*, 2011(1):1–14, 2011.

[44] C. Min, N. Chang, J. Cha, and J. Kang. Mimo-ofdm downlink channel prediction for ieee802. 16e systems using kalman filter. In *2007 IEEE Wireless Communications and Networking Conference*, pages 942–946. IEEE, 2007.

[45] T. J. O'Shea, T. Erpek, and T. C. Clancy. Deep learning based mimo communications. *arXiv preprint arXiv:1707.07980*, 2017.

[46] A. Papazafeiropoulos and T. Ratnarajah. Linear precoding for downlink massive mimo with delayed csit and channel prediction. In *2014 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 809–914. IEEE, 2014.

[47] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. 2017.

[48] P. S. Pati, S. S. Sahoo, D. Krishnaswamy, and R. Datta. A novel machine learning approach for link adaptation in 5g wireless networks. In *2020 2nd PhD Colloquium on Ethically Driven Innovation and Technology for Society (PhD EDITS)*, pages 1–2, 2020.

[49] W. Peng, W. Li, W. Wang, X. Wei, and T. Jiang. Downlink channel prediction for time-varying fdd massive mimo systems. *IEEE Journal of Selected Topics in Signal Processing*, 13(5):1090–1102, 2019.

[50] Qualcomm Inc. Fdd/tdd comparison – key messages. https://www.qualcomm.com/documents/fddtdd-comparison, 2013.

[51] Qualcomm Inc. Global Update on Spectrum for 4G & 5G. https://www.qualcomm.com/media/documents/files/spectrum-for-4g-and-5g.pdf, 2020.

[52] Qualcomm Inc. Why carrier aggregation is needed for 5g, and the latest qualcomm technologies breakthroughs making it possible. https://www.qualcomm.com/news/onq/2021/05/21/why-carrier-aggregation-needed-5g-and-latest-qualcomm-technologies-breakthroughs, 2021.

[53] M. S. Safari, V. Pourahmadi, and S. Sodagari. Deep ul2dl: Data-driven channel knowledge transfer from uplink to downlink. *IEEE Open Journal of Vehicular Technology*, 1:29–44, 2019.

[54] C. Shepard, J. Ding, R. E. Guerra, and L. Zhong. Understanding real many-antenna mu-mimo channels. In *2016 50th Asilomar Conference on Signals, Systems and Computers*, pages 461–467. IEEE, 2016.

[55] C. Shepard, H. Yu, N. Anand, E. Li, T. Marzetta, R. Yang, and L. Zhong. Argos: Practical many-antenna base stations. In *Proceedings of the 18th Annual International Conference on Mobile Computing and Networking*, Mobicom '12, page 53–64, New York, NY, USA, 2012. Association for Computing Machinery.

[56] N. Skatchkovsky, H. Jang, and O. Simeone. End-to-end learning of neuromorphic wireless systems for low-power edge artificial intelligence. *arXiv preprint arXiv:2009.01527*, 2020.

[57] Skylark Wireless. Iris-030 sdr platform. https://skylarkwireless.com/product/iris-platform/, 2021.

[58] P. Sthapit, H.-S. Gang, and J.-Y. Pyun. Bluetooth based indoor positioning using machine learning algorithms. In *2018 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia)*, pages 206–212. IEEE, 2018.

[59] D. Vasisht, S. Kumar, H. Rahul, and D. Katabi. Eliminating channel feedback in next-generation cellular networks. In *Proceedings of the 2016 ACM SIGCOMM Conference*, pages 398–411, 2016.

[60] J. Wang, Y. Ding, S. Bian, Y. Peng, M. Liu, and G. Gui. Ul-csi data driven deep learning for predicting dl-csi in cellular fdd systems. *IEEE Access*, 2019.

[61] Wikipedia contributors. 5g nr frequency bands — Wikipedia, the free encyclopedia, 2021.

[62] B. Yang, X. Cao, Z. Han, and L. Qian. A machine learning enabled mac framework for heterogeneous internet-of-things networks. *IEEE Transactions on Wireless Communications*, 18(7):3697–3712, 2019.

[63] P. Yang, Y. Xiao, M. Xiao, Y. L. Guan, S. Li, and W. Xiang. Adaptive spatial modulation mimo based on machine learning. *IEEE Journal on Selected Areas in Communications*, 37(9):2117–2131, 2019.

[64] Y. Yang, F. Gao, G. Y. Li, and M. Jian. Deep learning-based downlink channel prediction for fdd massive mimo system. *IEEE Communications Letters*, 23(11):1994–1998, 2019.

[65] Y. Yang, F. Gao, Z. Zhong, B. Ai, and A. Alkhateeb. Deep transfer learning-based downlink channel prediction for fdd massive mimo systems. *IEEE Transactions on Communications*, 68(12):7485–7497, 2020.

[66] X. Zhang, L. Zhong, and A. Sabharwal. Directional training for fdd massive mimo. *IEEE Transactions on Wireless Communications*, 2018.

[67] M. Zhao, T. Li, M. Abu Alsheikh, Y. Tian, H. Zhao, A. Torralba, and D. Katabi. Through-wall human pose estimation using radio signals. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[68] M. Zhao, Y. Tian, H. Zhao, M. A. Alsheikh, T. Li, R. Hristov, Z. Kabelac, D. Katabi, and A. Torralba. Rf-based 3d skeletons. ACM SIGCOMM, 2018.