Efficient Deep Learning for Massive MIMO Channel State Estimation

Mason del Rosario

University of California, Davis

Davis, CA 95616

mdelrosa@ucdavis.edu

March 16, 2021

Contents

1 Introduction **3**

1.1 MIMO Channel Overview 3

1.2 Channel Model 4

1.3 Classical CSI Estimation 5

1.4 Deep Learning 6

1.4.1 CNNs for CSI Estimation 7

2 Spherical Normalization **7**

2.1 Related Work 8

2.1.1 Notation 9

2.1.2 Spherical Normalization 10

2.2 Results 12

3 MarkovNet: A Deep Learning-based Differential Autoencoder **12**

3.1 Related Work 13

3.2 Methods 14

3.3 Numerical Results 15

4 CsiNet-Quant **16**

4.1 Trainable Quantization for CSI Feedback Compression 16

4.1.1 CsiNet-Quant: A Soft-to-Hard Vector Quantized Autoencoder for Trainable Discrete Latent Codewords 17

4.1.2 Proposed Work Subsection #2 18

5 Conclusion **19**

References **20**

1 Introduction

Section 1.1 provides an overview of the MIMO channel and relevant notation. Section 1.2 introduces the channel simulation used in this work, the COST2100 model. Section 1.4.1 provides an overview of recent works in deep learning for CSI estimation in MIMO networks.

1.1 MIMO Channel Overview

In this work, we consider a MIMO channel with a multiple antennas ($n_B \gg 1$) at the transmitter (gNodeB or gNB) servicing a single (or multiple) user equipment(s) (UE) with a single antenna. The network utilizes orthogonal frequency division multiplexing (OFDM) with N_f subcarriers, the m-th downlink and uplink channels at the receiver are given as

$$y_{d,m} = \mathbf{h}_{d,m}^H \mathbf{w}_{t,m} x_{d,m} + n_{d,m},$$
$$y_{u,m} = \mathbf{w}_{r,m}^H \mathbf{h}_{u,m} x_{u,m} + \mathbf{w}_{r,m}^H \mathbf{n}_{u,m}.$$

The resulting downlink and uplink channel state information (CSI) matrices are given as

$$\bar{\mathbf{H}}_d = \begin{bmatrix} \mathbf{h}_{d,1} & \dots & \mathbf{h}_{d,N_f} \end{bmatrix}^H \in \mathbb{C}^{N_f \times N_b},$$
$$\bar{\mathbf{H}}_u = \begin{bmatrix} \mathbf{h}_{u,1} & \dots & \mathbf{h}_{u,N_f} \end{bmatrix}^H \in \mathbb{C}^{N_f \times N_b}.$$

To achieve near-capacity transmission rates, the transmitter needs access to an appropriate estimate of $\bar{\mathbf{H}}_d$ [1]. In time division duplex (TDD), downlink CSI estimation can be performed by using pilots in uplink frames due to channel reciprocity [2–4]. In contrast, frequency domain duplex (FDD) does not admit channel reciprocity due to frequency-selective channels, and CSI estimates must be acquired using feedback.

Given their dimensionality, feeding back entire CSI matrices is impractical. Instead, we

Table 1: MIMO system variables considered in this work.

Symbol	Dimension	Description
$y_{d,m}$	$\mathbb{C}^1$	Received downlink symbol on m-th subcarrier
$\mathbf{h}_{d,m}$	$\mathbb{C}^{N_b \times 1}$	Downlink impulse response on m-th subcarrier
$\mathbf{w}_{t,m}$	$\mathbb{C}^{N_b \times 1}$	Transmitter precoding vector for m-th subcarrier
$x_{d,m}$	$\mathbb{C}^1$	Trasmitted symbol on m-th subcarrier
$n_{d,m}$	$\mathbb{C}^1$	Downlink noise on m-th subcarrier
$y_{u,m}$	$\mathbb{C}^1$	Received uplink symbol on m-th subcarrier
$\mathbf{h}_{u,m}$	$\mathbb{C}^{N_b \times 1}$	Uplink impulse response on m-th subcarrier
$\mathbf{w}_{r,m}$	$\mathbb{C}^{N_b \times 1}$	Received precoding vector for m-th subcarrier
$x_{u,m}$	$\mathbb{C}^1$	Received symbol on m-th subcarrier
$\mathbf{n}_{u,m}$	$\mathbb{C}^1$	Uplink noise on m-th subcarrier

seek a compressed representation of a sparse transformation. The sparse representation we consider is the angular-delay representation of CSI matrices [5]. Denote the unitary DFT (inverse DFT) matrix $\mathbf{F} \in \mathbb{C}^{N_f \times N_f}$ ($\mathbf{F}^H \in \mathbb{C}^{N_b \times N_b}$), and denote the spatial-frequency CSI matrix as $\bar{\mathbf{H}}$. The angular-delay domain representation $\mathbf{H}$ is given as

$$\mathbf{H} = \mathbf{F}^H \bar{\mathbf{H}} \mathbf{F}.$$

The delay spread of the resulting $\mathbf{H}$ can typically be captured with a small number of delay elements, so we restrict our attention to the first R_d elements of $\mathbf{H}$, resulting in a truncated angular-delay matrix which we denote as $\mathbf{H}_d \in \mathbb{C}^{(R_d \times N_b)}$ ($\mathbf{H}_u \in \mathbb{C}^{(R_d \times N_b)}$) for the downlink (uplink).

1.2 Channel Model

For all CSI tests, we mainly rely on the COST2100 MIMO channel model [6]. We use two datasets with a single base station (gNB) and a single user equipment (UE) in the following scenarios:

Table 2: Parameters used for COST2100 simulations for both Indoor and Outdoor datasets.

Symbol	Value	Description
N_b	32	Number of antennas at gNB
N_f	1024	Number of subcarriers for OFDM link
R_d	32	Number of delay elements kept after truncation
N	10^6	Total number of samples per dataset
T	10	Number of timeslots
δ	40ms, 80ms	Feedback delay interval between consecutive CSI timeslots

1. **Indoor** channels using a 5.3GHz downlink at 0.001 m/s UE velocity, served by a gNB at center of a 20m×20m coverage area.

2. **Outdoor** channels using a 300MHz downlink at 0.9 m/s UE velocity served by a gNB at center of a 400m×400m coverage area.

In both scenarios, we use the parameters listed in Table 2.

1.3 Classical CSI Estimation

Works in compressive feedback for CSI estimation in MIMO networks can be placed in three broad categories. The first category includes works which use direct quantization of continuous CSI elements to discrete levels. The quantized CSI are encoded and fed back to the transmitter [7, 8]. The second category includes works which use compressed sensing, a technique which applying a random measurement matrix at the transmitter and the receiver [9, 10]. Compressed sensing assumes matrices to be encoded and fed back meet certain sparsity requirements, and compressed sensing algorithms require iterative solvers [11] for decoding, resulting in undesired latency.

The last category of work in compressive CSI feedback uses deep learning (DL), neural networks with numerous layers which are trained on large datasets using backpropagation. Before describing these works, we first provide an overview of

1.4 Deep Learning

This section provides a brief overview of relevant deep learning concepts employed in this work, including convolutional neural networks (CNNs), autoencoders, and unsupervised learning. **Deep learning (DL)** is a subset of machine learning (ML), a broad class of algorithms which use data to "fit" models for prediction or classification tasks. The three predominant learning frameworks are supervised learning, unsupervised learning, and reinforcement learning. Here, we focus on *unsupervised learning*, which seeks to find a compressed representation of the data without labels (see Chapter 14 of [12] for an overview).

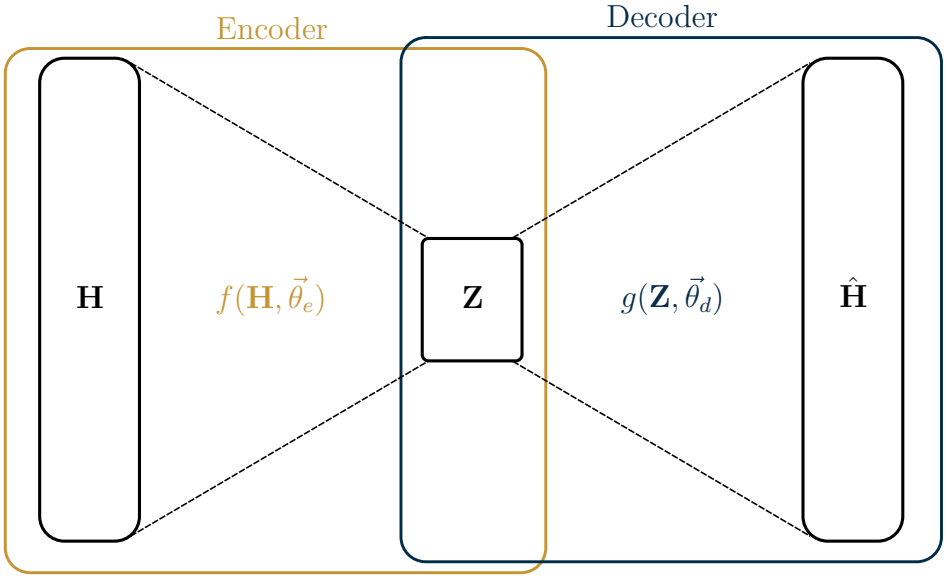


Figure 1: Abstract schematic for an autoencoder operating on CSI matrices $\mathbf{H}$. The encoder learns a latent representation, $\mathbf{Z}$, while the decoder learns to reconstruct estimates $\hat{\mathbf{H}}$.

A common architecture for deep unsupervised learning is the *autoencoder* (see Fig. 1 for a generic example). Trained end-to-end on input data, an autoencoder is comprised of an encoder and a decoder which jointly learn a compressed latent representation ($\mathbf{Z}$) and an estimate of the input ($\hat{\mathbf{H}}$). By choosing $\mathbf{Z}$ to have lower dimension than the input, the network is forced to learn a "useful" summary of the input data. The typical objective

function for such a network is the mean squared error,

$$\underset{\theta_e,\theta_d}{\operatorname{argmin}} \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{H}_i - g(f(\mathbf{H}_i,\theta_e),\theta_d)\|^2.$$

Presuming the autoencoder is a neural network, we optimize network parameters $\vec{\theta}_e, \vec{\theta}_d$ by backpropagation and a stochastic optimization algorithm (e.g., stochastic gradient descent, ADAM).

1.4.1 CNNs for CSI Estimation

Successful efforts in DL for CSI estimation have typically utilized convolutional neural networks (CNNs) in an autoencoder structure [13]. Variations on the CNN-based autoencoder have investigated different network architectures [14], variational training frameworks [15], and denoising modules [16].

2 Spherical Normalization

Most work in deep learning for CSI estimation focuses on different neural network architectures, training frameworks, or hyperparameter tuning. However, the normalization used in these works is typically the same; the extrema (i.e., the minimum and the maximum) of the dataset are used to perform minmax scaling on the entire dataset,

$$\mathbf{H}_{k,\text{minmax}}(i,j) = \frac{\mathbf{H}_k(i,j) - \mathbf{H}_{\min}}{\mathbf{H}_{\max} - \mathbf{H}_{\min}}$$

for $n \in [1,\ldots,N]$ given a dataset of N samples and i,j indexing the rows/columns of the CSI matrices. The resulting samples are cast to the range $[0,1]$.

For image data, minmax normalization results in each image's color channels scaled to the range $[0, 1]$. The resulting distribution for each color channel is typically satisfactory for image tasks, as the variance is not much smaller than the range of the normalized data (see Fig. 2).

However, for CSI matrices, minmax normalization is applied to the real and imaginary channels of each element. For typical channel models and parameters, the distribution of channel elements (see Fig. 3) tends to have much lower variance than that of ImageNet. This smaller variance can be explained by the difference in the datasets' ranges – while the channels in image data (e.g., ImageNet) assume integer values between $[0, 255]$, the channels in CSI data (e.g., COST2100) assume floating point values smaller than 10^{-3}.

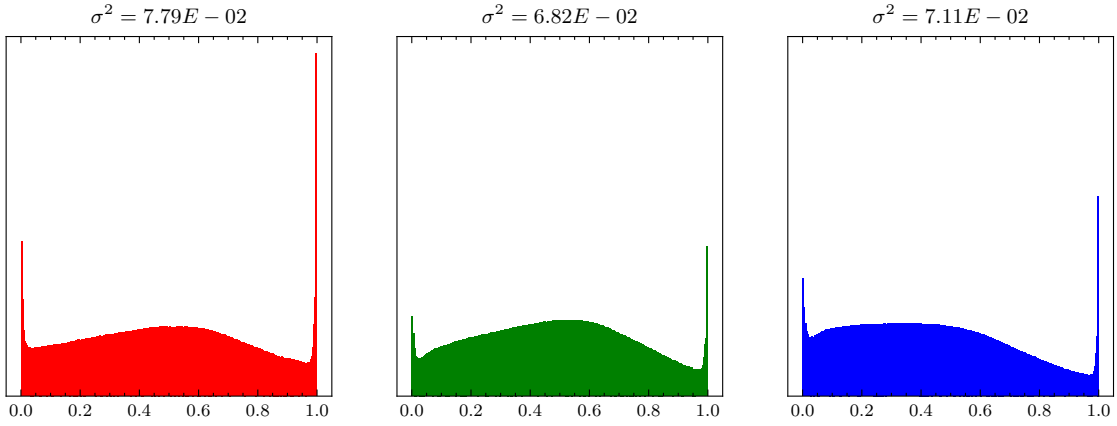


Figure 2: Distribution and variance of minmax-normalized ImageNet color channels ($N = 50000$) images.

2.1 Related Work

Several works have investigated normalization techniques for deep learning such as batch normalization [17], instance normalization [18], layer normalization [19], and group normalization [20]. These normalization techniques scale the outputs of latent layers in neural networks, which helps to solve the problem of covariate shift [17] where the mean and vari-

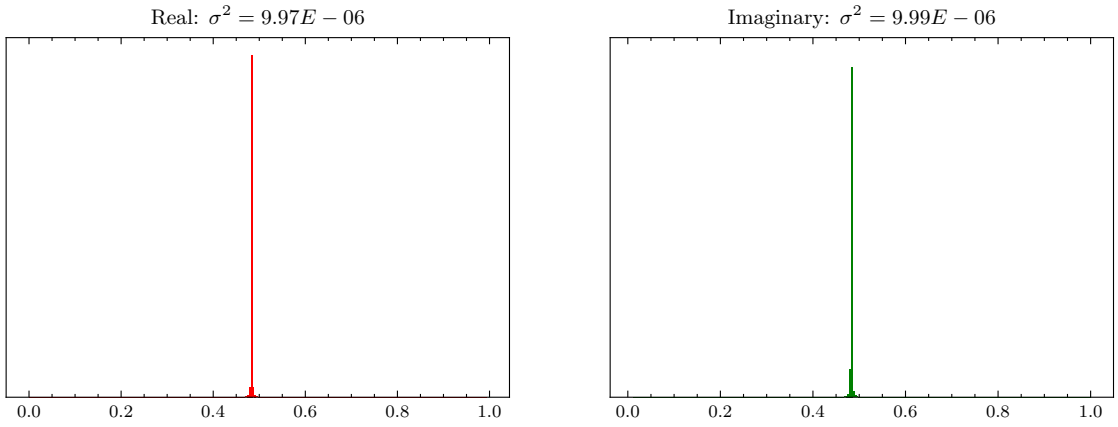


Figure 3: Distribution and variance of minmax-normalized COST2100 real/imaginary channels ($N = 99000$) images.

ance of changes between subsequent layers of the network.

Other works have studied normalization of the network's inputs. A number of works have investigated adaptive normalization techniques for time series estimation tasks [21–23]. In [24], the authors proposed a trainable input network which learns to shift, scale, and filter the unnormalized data while training the target network for a time series prediction task.

2.1.1 Notation

For an example of a table, see Table 3.

Table 3: Notations

Variable	Definition
M	Number of input hydrological variables denoted in Fig. **??**
N	Number of data samples, or days, in dataset
T	Number of days' data used for estimation
T_r	Dimension of data after pre-processing
z_n	Time series used for estimating salinity level on day n, size is $\mathbb{R}^{M\times T}$
x_n	Pre-processed time series with size $\mathbb{R}^{M\times T_r}$ for day n
f	A convolutional filter with size $\mathbb{R}^{M\times T\times T_r}$
y_n	ANN-estimated salinity level for one or more locations on day n

2.1.2 Spherical Normalization

Rather than apply minmax normalization, which is adversely impacted by outliers, we propose spherical normalization. Before describing spherical normalization in detail, consider z-score normalization. Given a random variable, x, with mean μx and standard deviation μ. The z-score normalized version of this random variable is given as

$$z = \frac{x - \mu}{\sigma^2}. \tag{1}$$

Assuming x is normally distributed, the resulting random variable, z, is a standard normal distribution such that $z \sim \mathcal{N}(0, 1)$. Inspired by z-score normalization, we seek a normalization scheme which adjusts the range of each channel sample. Under spherical normalization, each sample in the dataset is scaled by its power. Denote the k-th downlink CSI matrix of the dataset as $\mathbf{H}_d^k$. The spherically normalized version of the downlink CSI is given as

$$\check{\mathbf{H}}_d^k = \frac{\mathbf{H}_d^k}{\|\mathbf{H}_d^k\|_2}. \tag{2}$$

Observe that (2) is similar to (1) without the mean shift in the numerator[1] and with the power term of each CSI sample rather than the variance of the entire distribution. After applying (2) to each sample, minmax scaling is applied to the entire dataset. The resulting dataset under spherical normalization can exhibit a larger variance than the same dataset under minmax scaling (compare Fig. 4 with Fig. 3).

Beyond desirable properties in the input distribution, spherical normalization also results in an objective function which is better matched with the evaluation criterion. Neural

[1]Since the mean of COST2100 data is $\approx 10^{-10}$, we can safely ignore this mean shift in spherical normalization.

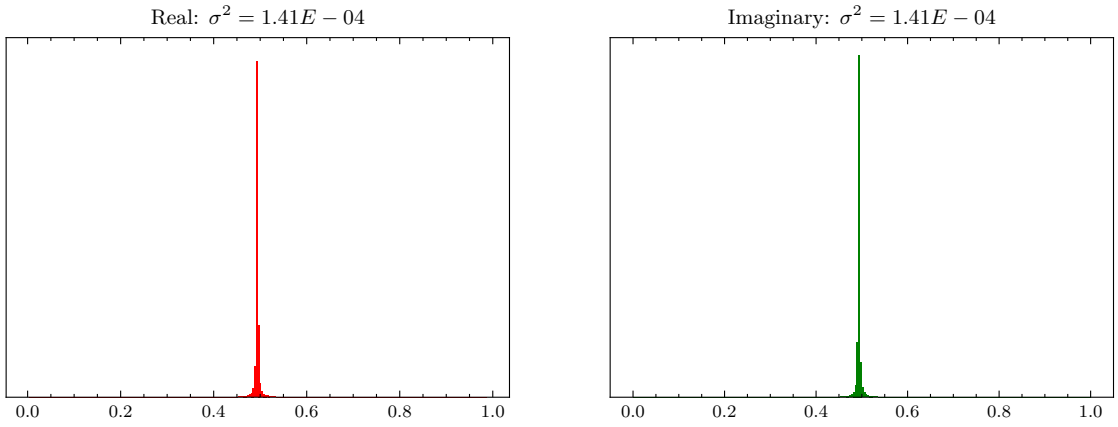


Figure 4: Distribution and variance of COST2100 real/imaginary channels under spherical normalization ($N = 99000$) images.

networks for CSI estimation are optimized using the mean-squared error loss,

$$\text{MSE} = \frac{1}{N}\sum_{k=1}^{N}\|\mathbf{H}_k - \hat{\mathbf{H}}_k\|^2, \tag{3}$$

while channel state reconstruction accuracy is measured in terms of normalized mean-squared error,

$$\text{NMSE} = \frac{1}{N}\sum_{k=1}^{N}\frac{\|\mathbf{H}_k - \hat{\mathbf{H}}_k\|^2}{\|\mathbf{H}_k\|^2}. \tag{4}$$

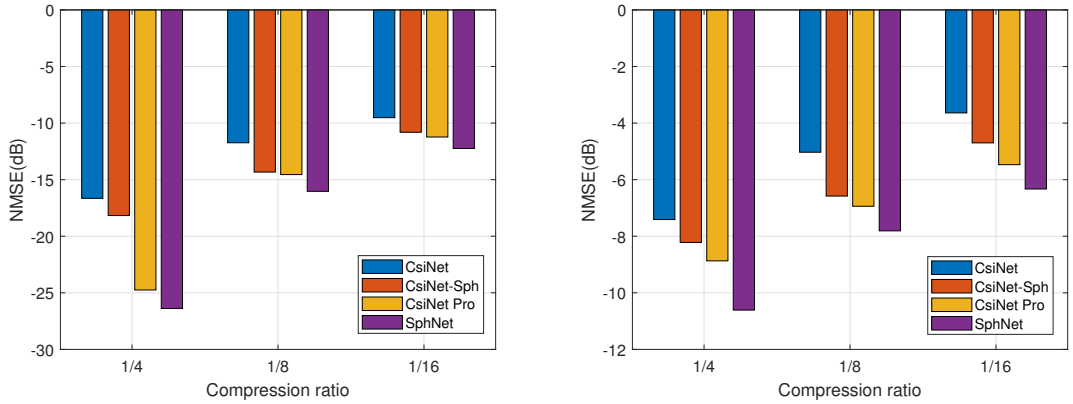
Observe that when the $\mathbf{H}_k$ ($\hat{\mathbf{H}}_k$) in (3) is replaced with $\check{\mathbf{H}}_k$ ($\hat{\check{\mathbf{H}}}_k$), we have

$$\begin{aligned}\frac{1}{N}\sum_{k=1}^{N}\|\check{\mathbf{H}}_k - \hat{\check{\mathbf{H}}}_k\|^2 &= \frac{1}{N}\sum_{k=1}^{N}\left\|\frac{\mathbf{H}_k}{\|\mathbf{H}_k\|^2} - \frac{\hat{\mathbf{H}}_k}{\|\mathbf{H}_k\|^2}\right\|^2 \\ &= \frac{1}{N}\sum_{k=1}^{N}\frac{\|\mathbf{H}_k - \hat{\mathbf{H}}_k\|^2}{\|\mathbf{H}_k\|^2},\end{aligned}$$

which is equivalent to (4). Thus, a neural network optimized with MSE as the loss function and trained using spherically normalized data is in fact being optimized with respect to NMSE of the original data.

2.2 Results

Training on spherically normalized data and optimizing with respect to NMSE can yield better accuracy. Fig. 5 demonstrates this improvement for CsiNet and CsiNet Pro on the COST2100 dataset. For both networks, the number of



(a) Indoor (b) Outdoor

Figure 5: Reconstruction error for CsiNet [13] and CsiNet Pro with and without spherical normalization. SphNet combines CsiNet Pro with spherical normalization [25].

3 MarkovNet: A Deep Learning-based Differential Autoencoder

In this section, we consider methods for exploiting temporal correlation between CSI of subsequent timeslots. Assuming the channel does not change substantially within a certain window of time, a reasonably accurate CSI estimate at some time $t-1$ can be used to estimate the CSI at time t. Generically, we can write this estimator as

$$\hat{\mathbf{H}}_t = h(\hat{\mathbf{H}}_{t-1}) \tag{5}$$

where $\mathbf{H}_t$ is the CSI matrix at time t and $\hat{\mathbf{H}}_t$ is its estimator. The estimation error under $\hat{H}_t$ is

$$\mathbf{E}_t = \mathbf{H}_t - \hat{\mathbf{H}}_t. \tag{6}$$

3.1 Related Work

Non ML-based work in temporal correlation for CSI estimation utilized state-space methods such as the Kalman filter [26–28]. Since it relies on explicit state space and noise models, the Kalman filter's predictive power in CSI estimation is limited. Furthermore, such work generally does not propose a method for feedback compression, making comparison with the following ML methods difficult.

Recent works have leveraged recurrent neural networks (RNNs) to exploit temporal correlation for CSI estimation [29–33]. RNNs include recurrent layers, such as the long short-term memory (LSTM) cell or the gated recurrent unit (GRU), which are capable of learning long-term dependencies of a given process through backpropagation [34] and can be used to predict future states of the process [35].

RNNs have been used extensively in natural language processing (NLP) for machine translation [36] and sentiment extraction [37]. For such works in NLP, authors have empirically found "stacked" or "deep" RNNs to be effective (e.g., Fig. 6), hypothesizing that having multiple recurrent layers allows the network to extract different semantic timescales [37,38]. Works in CSI estimation have taken cues from this work in NLP, proposing CSI estimation networks with stacked LSTMs after a sequence of autoencoders [33]. While such work has demonstrated the utility of RNNs, the computational cost of LSTMs can be prohibitively high. For example, the RNN portion of the network proposed in [33] accounts for 10^8 additional parameters. Since channel estimation should not place an undue computational

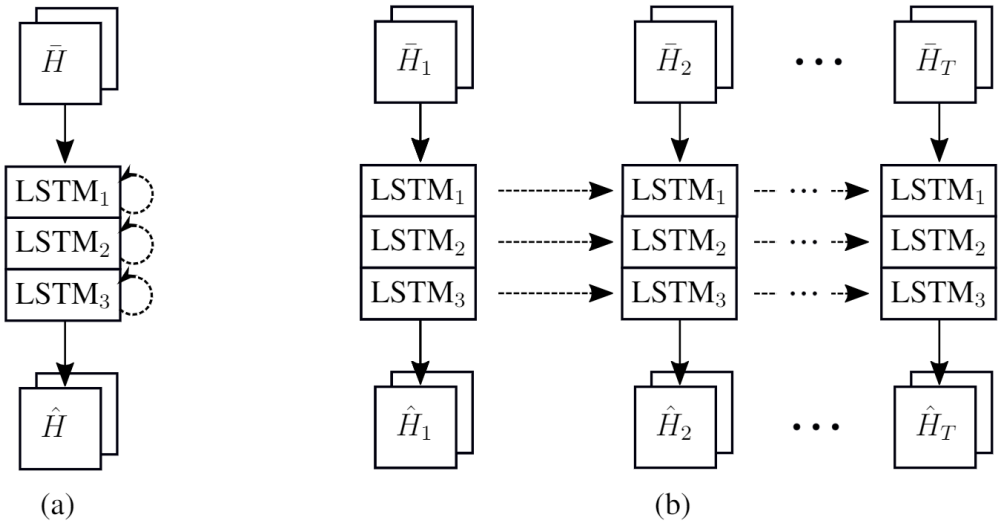


Figure 6: An example of LSTMs used for CSI estimation. (a) "Stacked" LSTM network of depth 3 shown with recurrent connections. (b) Same LSTM network "unrolled" into T timeslots

burden on the communications system, LSTMs can be problematic.

3.2 Methods

Rather than use RNNs to extract temporal dependencies in CSI data, we proposed a lightweight network based on the principle of differential encoding. We propose to train a network to estimate the error (6) when using a simple linear estimator,

$$\hat{\mathbf{H}}_t = \mathbf{W}\hat{\mathbf{H}}_{t-1}$$

where $\mathbf{W} \in \mathbb{C}^{R_b \times R_b}$ is the minimum mean squared error (MMSE) estimator.

$$\mathbf{H}_t = \mathbf{W}\mathbf{H}_{t-1} + \mathbf{E}_t$$

$$\mathbf{H}_{t-1}^H \mathbf{H}_t = \mathbf{W}\mathbf{H}_{t-1}^H \mathbf{H}_{t-1} + \mathbf{H}_{t-1}^H \mathbf{E}_t$$

Under the principle of orthogonality, the error term $\mathbf{E}_t$ is orthogonal with the observed data, and the product $\mathbf{H}_{t-1}^H \mathbf{E}_t$ becomes a zero matrix. Denoting the cross correlation matrix as $\mathbf{R}_i = \mathbb{E}\left[\mathbf{H}_{t-i}^H \mathbf{H}_t\right]$.

$$\mathbf{W} = \mathbf{R}_0^{-1} \mathbf{R}_1$$

We can further simplify this estimator to a scalar, $\gamma \in \mathbb{R}$, as

$$\gamma = \frac{\sum_i^{R_d} \sum_j^{N_b} \mathbf{R}_1(i,j)}{\sum_i^{R_d} \sum_j^{N_b} \mathbf{R}_0(i,j)},$$

where i (j) are the row (column) indices of the correlation matrices. Under the estimator γ, we proposed to encode the error, $\mathbf{E}_t$, using a convolutional autoencoder, $f(\mathbf{E}_t)$,

$$\hat{\mathbf{E}}_t = g(f(\mathbf{E}_t, \vec{\theta}_e), \vec{\theta}_d),$$

where $\mathbf{E}_t = \mathbf{H}_t - \gamma \hat{\mathbf{H}}_{t-1}$. The base station has access to the estimators γ and $\hat{\mathbf{H}}_{t-1}$, and the resulting CSI estimate at t is

$$\hat{\mathbf{H}}_t = \gamma \hat{\mathbf{H}}_{t-1} + \hat{\mathbf{E}}_t$$

3.3 Numerical Results

We compare MarkovNet with CsiNet-LSTM [33] on the indoor and outdoor COST2100 datasets (for details, see Section 1.2). For MarkovNet, we train the network at the first timeslot for 1000 epochs. In each subsequent timeslot, we initialize the network using the weights from the previous timeslot and train for 200 epochs. We use a batch size of 200. We perform a training/testing split of 75k/25k samples, and we estimate γ using the training set. To compare the estimation accuracy of each network, we report the NMSE.

Figure 7 shows a random sample from the test set, $\mathbf{H}$, and the estimates produced by CsiNet-LSTM and MarkovNet for a CR of $\frac{1}{4}$. This sample contains three "peak" magnitude regions. While both networks manage to capture the two larger samples, MarkovNet is able to recover the small peak magnitude region which CsiNet-LSTM fails to produce.

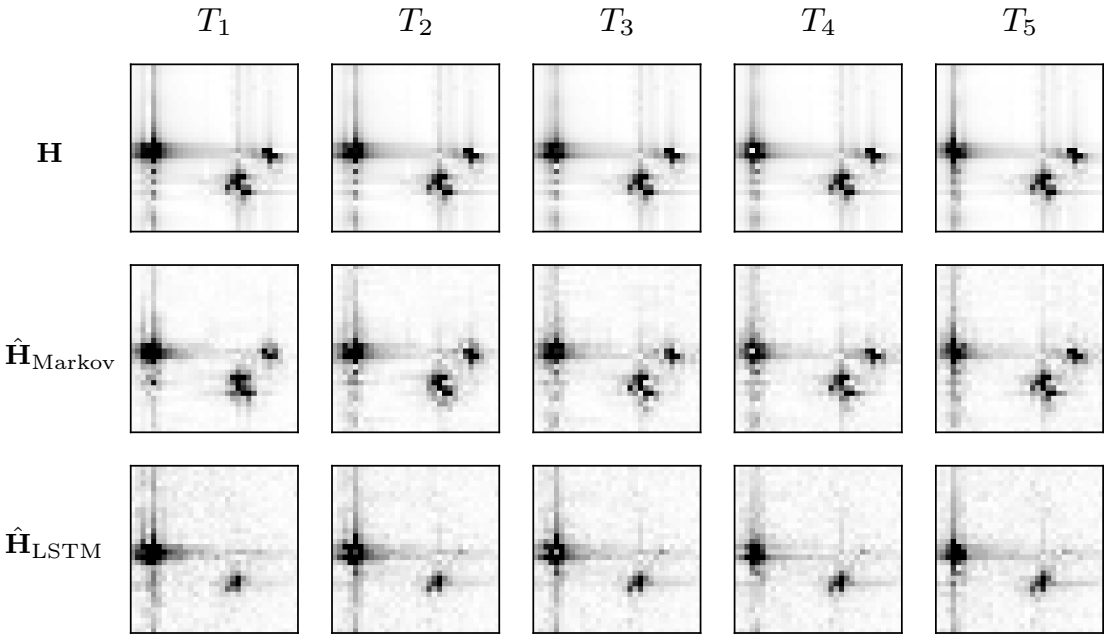


Figure 7: CSI ($\mathbf{H}$), MarkovNet estimates ($\hat{\mathbf{H}}_{\text{Markov}}$), and CsiNet-LSTM estimates ($\hat{\mathbf{H}}_{\text{LSTM}}$) across five timeslots (T_1 through T_5) on one outdoor channel sample from the test set, using CR $= \frac{1}{4}$.

4 CsiNet-Quant

4.1 Trainable Quantization for CSI Feedback Compression

While the previously discussed works investigate neural architectures for learned encoding-decoding of CSI data, such architectures rely on continuous valued latent representations. These works presume that the continuous valued representations are available at the receiver with no noise from quantization noise, but modulation protocols require quantized data to accommodate bit string representations and IQ constellations. To make neural autoencoders

compatible with common modulation schemes, trainable CSI compression techniques should incorporate quantized codewords in the learning process.

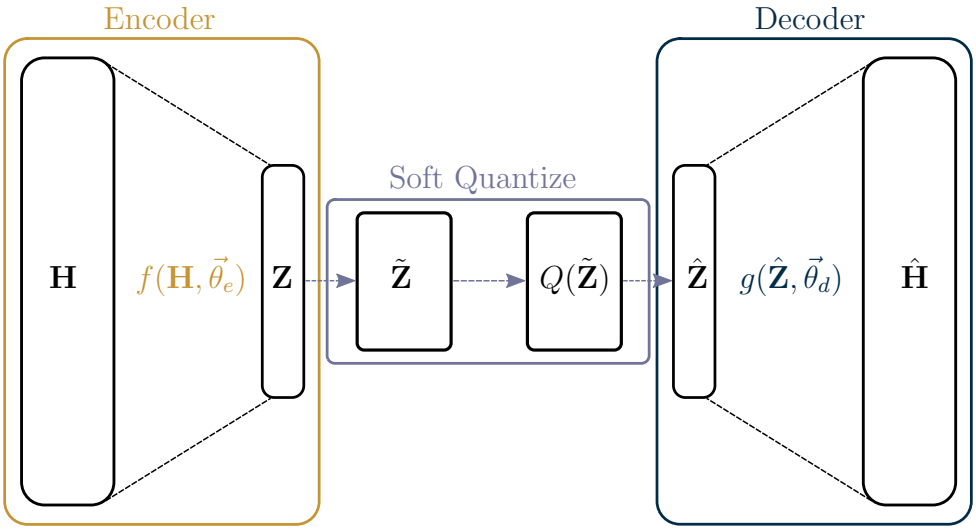


Figure 8: Abstract architecture for CsiNet-Quant. SoftQuantize layer ($Q(\tilde{\mathbf{Z}})$) is a continuous, softmax-based relaxation of a d-dimensional quantization of the latent layer $\mathbf{Z}$.

4.1.1 CsiNet-Quant: A Soft-to-Hard Vector Quantized Autoencoder for Trainable Discrete Latent Codewords

To incorporate learnable discrete latent codewords in the learning process, we propose to use soft-to-hard vector quantization framework proposed in [39]. We choose a vector dimension, m, by which to partition the latent space $\mathbf{Z} = f(\mathbf{H}, \theta_e)$, and we denote the vectorized version of $\mathbf{Z} \in \mathbb{R}^r$ as $\tilde{\mathbf{Z}} \in \mathbb{R}^{r/m \times m}$. We define the m-dimensional codebook of size L, $[L]^m$ with codewords $\mathbf{C} \in \mathbb{R}^m$. The soft assignments of the j-th latent vector $\tilde{\mathbf{z}}_j$ can be written as

$$\phi(\tilde{\mathbf{z}}_j) = \left[\frac{\exp(-\sigma \|\tilde{\mathbf{z}}_j - \mathbf{c}_\ell\|^2)}{\sum_i^L \exp(-\sigma \|\tilde{\mathbf{z}}_j - \mathbf{c}_\ell\|^2)} \right]_{i \in [1,\dots,L]} \in \mathbb{R}^L \tag{7}$$

(8) is typically referred to as the *softmax* function, which is commonly used as a differentiable alternative to the maximum function. The hyperparameter σ controls the temperature of the softmax scores, with a lower σ yielding a more uniform distribution and a higher σ yielding

a "peakier" distribution (i.e., $\sigma \rightarrow \infty \Rightarrow \phi(\tilde{z}_j) \rightarrow \max(\tilde{z}_j)$). Using the soft assignments, the latent vectors are quantized based on the codewords $\mathbf{C} \in \mathbb{R}^{L \times m}$,

$$Q(\tilde{\mathbf{z}}_j) = \phi(\tilde{\mathbf{z}}_j)\mathbf{C}. \tag{8}$$

4.1.2 Proposed Work Subsection #2

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

5 Conclusion

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

References

[1] A. Goldsmith, S. A. Jafar, N. Jindal, and S. Vishwanath, "Capacity limits of mimo channels," *IEEE Journal on selected areas in Communications*, vol. 21, no. 5, pp. 684–702, 2003.

[2] F. Kaltenberger, H. Jiang, M. Guillaud, and R. Knopp, "Relative channel reciprocity calibration in mimo/tdd systems," in *2010 Future Network Mobile Summit*, pp. 1–10, June 2010.

[3] D. Mi, M. Dianati, L. Zhang, S. Muhaidat, and R. Tafazolli, "Massive mimo performance with imperfect channel reciprocity and channel estimation error," *IEEE Transactions on Communications*, vol. 65, no. 9, pp. 3734–3749, 2017.

[4] Q. Gao, F. Qin, and S. Sun, "Utilization of channel reciprocity in advanced mimo system," in *2010 5th International ICST Conference on Communications and Networking in China*, pp. 1–5, Aug 2010.

[5] A. M. Sayeed, "Deconstructing multiantenna fading channels," *IEEE Transactions on Signal processing*, vol. 50, no. 10, pp. 2563–2579, 2002.

[6] L. Liu, C. Oestges, J. Poutanen, K. Haneda, P. Vainikainen, F. Quitin, F. Tufvesson, and P. D. Doncker, "The cost 2100 mimo channel model," *IEEE Wireless Communications*, vol. 19, pp. 92–99, December 2012.

[7] B. Makki and T. Eriksson, "On hybrid arq and quantized csi feedback schemes in quasi-static fading channels," *IEEE transactions on communications*, vol. 60, no. 4, pp. 986–997, 2012.

[8] H. Shirani-Mehr and G. Caire, "Channel state feedback schemes for multiuser mimo-ofdm downlink," *IEEE Transactions on Communications*, vol. 57, no. 9, pp. 2713–2723, 2009.

[9] X. Rao and V. K. Lau, "Distributed compressive csit estimation and feedback for fdd multi-user massive mimo systems," *IEEE Transactions on Signal Processing*, vol. 62, no. 12, pp. 3261–3271, 2014.

[10] M. E. Eltayeb, T. Y. Al-Naffouri, and H. R. Bahrami, "Compressive sensing for feedback reduction in mimo broadcast channels," *IEEE Transactions on communications*, vol. 62, no. 9, pp. 3209–3222, 2014.

[11] T. T. Do, L. Gan, N. Nguyen, and T. D. Tran, "Sparsity adaptive matching pursuit algorithm for practical compressed sensing," in *2008 42nd Asilomar conference on signals, systems and computers*, pp. 581–587, IEEE, 2008.

[12] T. Hastie, R. Tibshiran, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2016.

[13] C. Wen, W. Shih, and S. Jin, "Deep learning for massive mimo csi feedback," *IEEE Wireless Communications Letters*, vol. 7, pp. 748–751, Oct 2018.

[14] Z. Lu, J. Wang, and J. Song, "Multi-resolution csi feedback with deep learning in massive mimo system," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, pp. 1–6, 2020.

[15] M. Hussien, K. K. Nguyen, and M. Cheriet, "PRVNet: Variational autoencoders for massive MIMO CSI feedback," *arXiv*, 2020.

[16] Y. Sun, W. Xu, L. Fan, G. Y. Li, and G. K. Karagiannidis, "Ancinet: An efficient deep learning approach for feedback compression of estimated csi in massive mimo systems," *IEEE Wireless Communications Letters*, vol. 9, no. 12, pp. 2192–2196, 2020.

[17] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*, pp. 448–456, PMLR, 2015.

[18] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1501–1510, 2017.

[19] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[20] Y. Wu and K. He, "Group normalization," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 3–19, 2018.

[21] E. Ogasawara, L. C. Martinez, D. De Oliveira, G. Zimbrão, G. L. Pappa, and M. Mattoso, "Adaptive normalization: A novel data normalization approach for non-stationary time series," in *The 2010 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, IEEE, 2010.

[22] S. Nayak, B. B. Misra, and H. S. Behera, "Impact of data normalization on stock index forecasting," *International Journal of Computer Information Systems and Industrial Management Applications*, vol. 6, no. 2014, pp. 257–269, 2014.

[23] X. Shao, "Self-normalization for time series: a review of recent developments," *Journal of the American Statistical Association*, vol. 110, no. 512, pp. 1797–1817, 2015.

[24] N. Passalis, A. Tefas, J. Kanniainen, M. Gabbouj, and A. Iosifidis, "Deep adaptive input normalization for time series forecasting," *IEEE transactions on neural networks and learning systems*, vol. 31, no. 9, pp. 3760–3765, 2019.

[25] Z. Liu, M. del Rosario, X. Liang, L. Zhang, and Z. Ding, "Spherical normalization for learned compressive feedback in massive mimo csi acquisition," in *2020 IEEE International Conference on Communications Workshops (ICC Workshops)*, pp. 1–6, 2020.

[26] K. Huber and S. Haykin, "Improved bayesian mimo channel tracking for wireless communications: incorporating a dynamical model," *IEEE transactions on wireless communications*, vol. 5, no. 9, pp. 2458–2466, 2006.

[27] K. S. Ali and P. Sampath, "Time domain channel estimation for time and frequency selective millimeter wave mimo hybrid architectures: Sparse bayesian learning-based kalman filter," *Wireless Personal Communications*, pp. 1–21, 2020.

[28] H. Kim, S. Kim, H. Lee, C. Jang, Y. Choi, and J. Choi, "Massive mimo channel prediction: Kalman filtering vs. machine learning," *IEEE Transactions on Communications*, vol. 69, no. 1, pp. 518–528, 2021.

[29] C. Lu, W. Xu, H. Shen, J. Zhu, and K. Wang, "MIMO Channel Information Feedback Using Deep Recurrent Network," *IEEE Commu. Letters*, vol. 23, pp. 188–191, Jan 2019.

[30] Y. Liao, H. Yao, Y. Hua, and C. Li, "CSI Feedback Based on Deep Learning for Massive MIMO Systems," *IEEE Access*, vol. 7, pp. 86810–86820, 2019.

[31] X. Li and H. Wu, "Spatio-Temporal Representation With Deep Neural Recurrent Network in MIMO CSI Feedback," *IEEE Wireless Comm. Letters*, vol. 9, no. 5, pp. 653–657, 2020.

[32] Y. Jang, G. Kong, M. Jung, S. Choi, and I. Kim, "Deep Autoencoder Based CSI Feedback With Feedback Errors and Feedback Delay in FDD Massive MIMO Systems," *IEEE Wireless Comm. Letters*, vol. 8, no. 3, pp. 833–836, 2019.

[33] T. Wang, C. Wen, S. Jin, and G. Y. Li, "Deep Learning-Based CSI Feedback Approach for Time-Varying Massive MIMO Channels," *IEEE Wireless Comm. Letters*, vol. 8, pp. 416–419, April 2019.

[34] M. Hermans and B. Schrauwen, "Training and analysing deep recurrent neural networks," in *Advances in Neural Information Processing Systems 26* (C. J. C. Burges and et al., eds.), pp. 190–198, 2013.

[35] R. Pascanu, C. Gulcehre, K. Cho, and Y. Bengio, "How to construct deep recurrent neural networks," *2nd International Conference on Learning Representations*, no. March 2014, 2014.

[36] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," *Advances in Neural Information Processing Systems*, vol. 4, no. January, pp. 3104–3112, 2014.

[37] O. Irsoy and C. Cardie, "Opinion mining with deep recurrent neural networks," *Proc. 2014 Conference on Empirical Methods in Natural Language Processing*, pp. 720–728, 2014.

[38] Y. Bengio, "Learning deep architectures for AI," *Foundations and Trends in Machine Learning*, vol. 2, no. 1, pp. 1–27, 2009.

[39] E. Agustsson, F. Mentzer, M. Tschannen, L. Cavigelli, R. Timofte, L. Benini, and L. Van Gool, "Soft-to-hard vector quantization for end-to-end learning compressible representations,"

Advances in Neural Information Processing Systems, vol. 2017-Decem, no. Nips, pp. 1142–1152, 2017.