

# Two-Stage Learning for Uplink Channel Estimation in One-Bit Massive MIMO

Eren Balevi and Jeffrey G. Andrews

Department of Electrical and Computer Engineering

The University of Texas at Austin, TX 78712, USA

Email: erenbalevi@utexas.edu, jandrews@ece.utexas.edu

**Abstract**—We develop a two-stage deep learning pipeline architecture to estimate the uplink massive MIMO channel with one-bit ADCs. This deep learning pipeline is composed of two separate generative deep learning models. The first one is a supervised learning model and designed to compensate for the quantization loss. The second one is an unsupervised learning model and optimized for denoising. Our results show that the proposed deep learning-based channel estimator can significantly outperform other state-of-the-art channel estimators for one-bit quantized massive MIMO systems. In particular, our design provides 5-10 dB gain in channel estimation error. Furthermore, it requires a reasonable amount of pilots, on the order of 20 per coherence time interval.

## I. INTRODUCTION

Massive multiple-input multiple-output (MIMO) is a key technology to increase data throughput by allowing many users to concurrently use the same spectrum spatially [1]. This is achieved by eliminating interference among users thanks to the substantial degrees of freedom that comes from a large number of antennas. More specifically, this inter-user interference is mitigated by aligning beams properly for each user. This beam alignment conventionally requires good enough channel state information (CSI). Hence, acquiring CSI is of great importance in massive MIMO communication systems.

Channel estimation has always been a challenge for communication systems due to the endless goal of reducing computational complexity and the number of pilot tones. This is particularly true for massive MIMO, because of the high signal dimension. Hence, massive MIMO channel estimation has to be performed with low complexity hardware and limited number of pilots, in which the former requirement is mainly to satisfy power consumption budget and the latter is for bandwidth efficiency. In this paper, these problems are handled for uplink multiuser massive MIMO. Although our main intent is for lower frequencies, it can be trivially applied for mmWave transmission.

Uplink massive MIMO nominally requires a correspondingly large number of **analog-to-digital converters** (ADCs) at the base station. This causes untenable power consumption and hardware complexity. There are many papers in the literature to alleviate these problems. These papers support employing low-resolution ADCs and analyze the impact of having a pair of one-bit quantization per each antenna, i.e., one for each real and imaginary component. More specifically, [2] shows the effects of one-bit quantization in terms of mutual information

and symbol error rate for massive MIMO. A near maximum likelihood detector for one-bit uplink massive MIMO was designed in [3]. Furthermore, [4], [5], [6] proposed some channel estimation methods against the detrimental impacts of one-bit quantization.

In this paper, we propose to use deep learning methods for channel estimation in uplink massive MIMO with one-bit quantized received signals. This is motivated by the success of deep learning while coping with significant nonlinearities [7]. There has been a growing interest to make use of deep learning in communication systems [8], [9], [10], [11] including channel estimation [12], [13], [14]. The existing deep learning-based channel estimators rely on discriminative models. Note that a discriminative model is the one that simply maps the given input data or observations to their target values without using any a prior knowledge from these data. On the other hand, a generative model makes use of the a prior information while maximizing the likelihood. This motivation has recently inspired some deep generative model-based channel estimators [15], [16], [17].

The main contribution of this paper is to estimate the uplink channel in massive MIMO under the constraint of one-bit quantization by leveraging generative models. More specifically, a deep learning pipeline architecture is proposed for this problem. Our model is composed of two types of generative deep neural networks that were previously used for single antenna one-bit quantized OFDM channel estimation [15] and single antenna unquantized OFDM channel estimation [16]. The proposed pipeline model works surprisingly well such that it can outperform state-of-the-art one-bit quantized channel estimators for uplink massive MIMO by 5-10 dB. Promisingly, this design only requires approximately 20 pilots per coherence time interval.

This paper is organized as follows. The system model and problem statement are explained in Section II. The proposed architecture is given in Section III. We provide the numerical results and computational complexity analysis in Section IV. The paper ends with concluding remarks in Section V.

## II. SYSTEM MODEL AND PROBLEM STATEMENT

In this paper,  $K$  single antenna users send orthogonal OFDM symbols at the same time and on the same frequency band to a base station that has  $M$  antennas. It is assumed that each OFDM symbol has  $N_f$  subcarriers. With these settings,

the received signal at the  $m^{th}$  antenna in response to one transmitted OFDM symbol becomes

$$\mathbf{y}[\mathbf{m}] = \sum_{k=1}^K \mathbf{H}_k[\mathbf{m}] \mathbf{F}^H \mathbf{x}_k + \mathbf{w}[\mathbf{m}], \quad (1)$$

where  $\mathbf{H}_k[\mathbf{m}]$  is an  $N_f \times N_f$  circulant matrix,  $\mathbf{F}^H$  is an  $N_f \times N_f$  inverse discrete Fourier transform (IDFT) matrix, and the OFDM symbol  $\mathbf{x}_k$  is an  $N_f \times 1$  vector. The zero-mean Gaussian noise vector is represented by  $\mathbf{w}[\mathbf{m}]$ . This received signal at each antenna port is quantized with a complex one-bit ADC. Thus, the real and imaginary part of the signal are quantized separately as

$$\mathbf{r}[\mathbf{m}] = \mathcal{Q}(\mathbf{y}[\mathbf{m}]) \quad (2)$$

where

$$\mathcal{Q}(\mathbf{y}[\mathbf{m}]) = \frac{1}{\sqrt{2}} \text{sign}(\Re\{\mathbf{y}[\mathbf{m}]\}) + \frac{j}{\sqrt{2}} \text{sign}(\Im\{\mathbf{y}[\mathbf{m}]\}) \quad (3)$$

is applied element-wise along the vector  $\mathbf{y}[\mathbf{m}]$ . Combining the received signal  $\mathbf{r}[\mathbf{m}]$  over all antennas yields an  $N_f \times M$  matrix, which is

$$\mathbf{R} = [\mathbf{r}[1] \ \mathbf{r}[2] \ \cdots \ \mathbf{r}[M]]. \quad (4)$$

In this paper, we assume that one coherence time is composed of  $N$  OFDM symbols. The channel or  $\mathbf{H}_k[\mathbf{m}]$  remains constant during a coherence time interval and it periodically changes with coherence time interval. In this setup, the channel taps between the  $k^{th}$  user and the  $m^{th}$  antenna of the base station in the frequency domain is  $\lambda_k[\mathbf{m}] = \text{diag}(\mathbf{\Lambda}_k[\mathbf{m}])$ , where

$$\mathbf{\Lambda}_k[\mathbf{m}] = \mathbf{F} \mathbf{H}_k[\mathbf{m}] \mathbf{F}^H. \quad (5)$$

Hence, the channel between the  $k^{th}$  user and the base station for all the  $M$  antennas becomes an  $N_f \times M$  matrix

$$\mathbf{\Lambda}_k = [\text{diag}(\mathbf{\Lambda}_k[1]) \ \cdots \ \text{diag}(\mathbf{\Lambda}_k[M])]. \quad (6)$$

Our problem is to estimate  $\mathbf{\Lambda}_k$  from the received signal  $\mathbf{R}$ , which is defined in (4), with some number of pilots smaller than  $N$ . These  $\mathbf{\Lambda}_k$  matrices can be found separately for each user, because each user has orthogonal pilot sequences.

### III. DEEP LEARNING PIPELINE FOR ONE-BIT MASSIVE MIMO CHANNEL ESTIMATION

One-bit quantization leads to significant information loss at the receiver front-end. This considerably complicates channel estimation. We propose a pipeline deep learning architecture to reliably estimate the one-bit massive MIMO channel. Our architecture is composed of two separate deep learning models, each of which is specialized for different purposes. To be more precise, the first deep learning model, which is a generative supervised learning (SL) model, is designed for recovering the information loss due to the quantization. The second one, which is a generative unsupervised learning (USL) model, aims to denoise the received signal for channel estimation. The overall two-stage system model is depicted in Fig. 1, in

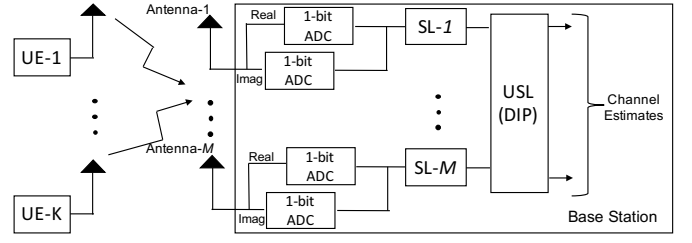


Fig. 1. One-bit quantized massive MIMO system in which  $K$  users send their pilots to the base station with  $M$  antennas. The received signal is first processed with  $M$  supervised learning (SL) blocks, and then processed with one special unsupervised learning model, which is the deep image prior (DIP).

which the tasks from SL-1 to SL- $M$  involve the first stage and the USL task is the second stage.

The main reason behind the usage of generative models both for supervised and unsupervised learning is associated with the fact that generative models take into account the a priori knowledge as opposed to discriminative models. It is worth emphasizing that these priors are quite important for our problem, because one-bit channel estimation yields an ill-posed problem. This means that there is no hope to solve it without using priors. In particular, the quality of these priors determine the channel estimation error.

#### A. Stage-1: Supervised Learning

The received signal at each antenna port, which is given in (2), is first processed with a separate supervised deep neural network. This means that  $M$  deep neural networks are maintained in parallel. It is assumed that all these  $M$  neural networks have the same architecture, which is a standard deep neural network that has two hidden layers as illustrated in Fig. 2. It is worth emphasizing that the parameters of these  $M$  deep neural network are not the same, because each one of them has its own labels and is trained individually according to their received signals and labels.

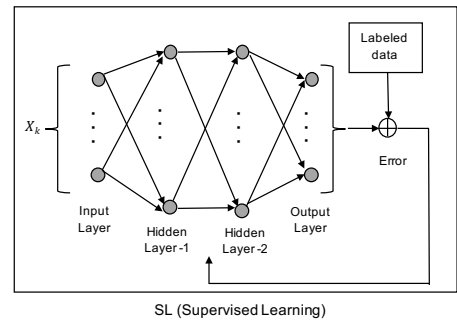


Fig. 2. The supervised deep learning architecture that is employed and trained separately for each antenna.

In supervised learning, the most critical thing is to define the labels intelligently to push the parameters of a neural network towards the desired goal. Our recent work proposed generating a labeled data set on the fly for single antenna OFDM receivers using Busgang's theorem [15]. Accordingly, the diagonals of

TABLE I  
THE SUPERVISED DNN ARCHITECTURE FOR CHANNEL ESTIMATION WITH ONE-BIT ADCS

Layer	Type	Size	Activation	Weights
Input Layer	Pilot Symbols	$2N_f$	-	-
Hidden Layer-1	Fully Connected	$4N_f$	ReLU	$\Phi_1$
Hidden Layer-2	Fully Connected	$4N_f$	ReLU	$\Phi_2$
Output	Fully Connected	$2N_f$	Linear	$\Phi_3$

the matrix  $\mathbf{F}\mathbf{r}[\mathbf{m}]\mathbf{x}_k^H$  are used as the labels. The theoretical ground for the selection of these labels is given in [15]. This labeling policy is also used in this one-bit massive MIMO channel estimation while training  $M$  supervised deep neural networks.

The layers, their types, sizes, activation functions and weights of the supervised deep neural network at each RF branch are summarized in Table I. State-of-the-art software libraries that implement neural networks do not support complex operations. Thus the real and imaginary part of the complex vectors are concatenated at each antenna to obtain a  $2N_f \times 1$  real vector. Without loss of generality, the dimension of the hidden layers is taken to be twice that of the input and output layer, giving  $32N_f^2$  trainable parameters, which increases quadratically with the number of subcarriers. Notice that a single hidden layer can give the same performance with two hidden layers if it has a sufficient number of neurons due to the universal function approximation theorem of neural networks [7]. However, this brings additional computational complexity, and hence having two hidden layers with reasonable number of neurons seems a good compromise. Rectified linear unit (ReLU) is used in the hidden layers as an activation function for fast convergence, and a linear activation function is utilized at the output layer.

We propose to use the aforementioned model as a regression task. Accordingly, the input layer takes the pilots  $\mathbf{x}_{k,p}$  and produces the corresponding output  $\mathbf{z}_p[\mathbf{m}]$  for  $p = 1, \dots, N_t$  where  $N_t$  is the total number of pilots transmitted over the channel for one coherence interval, in which  $N_t < N$ . Notice that the pilots for the users are sent orthogonally, and hence can be treated individually. This  $\mathbf{z}_p[\mathbf{m}]$  for  $m = 1, 2, \dots, M$  can be written in terms of the trainable weights or network parameters (in matrix notation) and activation functions as

$$\mathbf{z}_p[\mathbf{m}] = \sigma_3(\Phi_3\sigma_2(\Phi_2\sigma_1(\Phi_1\mathbf{x}_{k,p}))) \quad (7)$$

where  $\Phi_i$  and  $\sigma_i$  are the network parameters and the activation function, respectively. The parameters are optimized according to the following cost function

$$J_m = \min_{\Phi_1, \Phi_2, \Phi_3} \|\mathbf{z}_p[\mathbf{m}] - \text{diag}(\mathbf{F}\mathbf{r}[\mathbf{m}]\mathbf{x}_{k,p}^H)\|^2 \quad (8)$$

which are solved with gradient descent via the backpropagation algorithm.

The  $M$  deep neural networks are trained separately to minimize the MSE between the outputs and the labeled data as given in (8). After training, for each supervised neural

network we generate as many output samples as needed in response to random inputs within the same channel coherence interval, and take their average. The generated output samples for the random inputs do not cost anything other than some extra processing, because these inputs are not coming from the channel; rather they are generated randomly in the receiver. To be more precise, each trained deep neural network generates some output samples  $\mathbf{z}_i[\mathbf{m}]$  in response to the random inputs  $\mathbf{x}_{k,i}$ . In what follows, the output of each deep neural network is obtained as

$$\hat{\lambda}_k[\mathbf{m}] = \frac{1}{N_g} \sum_{i=0}^{N_g-1} \mathbf{z}_i[\mathbf{m}] \quad (9)$$

where  $N_g$  is the total number of arbitrarily generated output samples. There is no constraint to limit  $N_g$  except the processing complexity, i.e., the  $\mathbf{z}_i[\mathbf{m}]$  does not consume any bandwidth. Each time the channel changes, the model must be retrained with  $N_t$  pilots, and  $N_g$  randomly generated samples. Note that there are many different types of generative model applications other than the generative adversarial networks (GANs) [18], and hence this is one type of generative models.

### B. Stage-2: Unsupervised Learning

The main benefit of using a second deep learning model is to enhance the quality of channel estimates. In this regard, an unsupervised deep learning model is utilized. More precisely, the output of each supervised deep learning model that estimates the channel taps in the frequency domain given in (9) is replicated  $N$  times in the time domain. This means that an  $N_f \times N$  complex frequency-time grid is obtained for each user. Then, this frequency-time grid is stacked for all antennas for  $m = \{1, \dots, M\}$  in the spatial domain. This yields a 3-dimensional  $N_f \times N \times M$  signal for each user, which is denoted as  $\Lambda_{\mathbf{T}}$ .

The unsupervised deep learning model processes this 3-dimensional signal by fitting the parameters of its deep neural network. The overall unsupervised model that depicts the input, output and hidden layers is given in Fig. 3. Here, the deep neural network targets to denoise the signal by fitting the signal more and the noise less. This is possible, because the signal has a structure, whereas the noise is unstructured, i.e., the noise is independent and identically distributed (i.i.d). In particular, the hidden layers are crafted so as to exploit the structure (or correlations) of the signal. The working principle of the model in Fig. 3 is to generate  $\Lambda_{\mathbf{T}}$  by passing a randomly chosen input tensor  $Z_0$  through hidden layers. Here,  $Z_0$  is as an input filled with uniform noise and once it is randomly initialized, it is kept fixed. The weights of the hidden layers are also randomly initialized, but their weights are continuously updated via gradient descent.

The key component in the aforementioned generative unsupervised model is the hidden layers. This structure of a hidden layer is portrayed in Fig. 4. Each hidden layer is composed of four major components. These are: (i) a  $1 \times 1$  convolution, (ii) an upsampler, (iii) a ReLU activation function, and (iv) a batch normalization. A  $1 \times 1$  convolution means that each

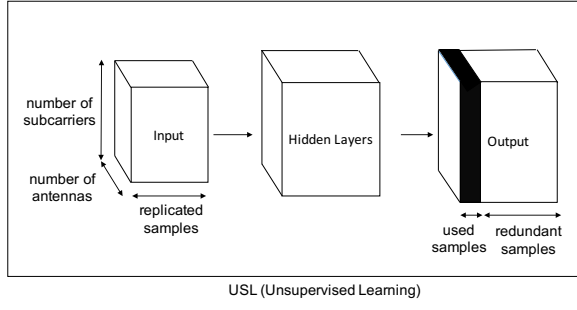


Fig. 3. The second unsupervised deep learning model whose output gives the channel estimates.

element in the time-frequency grid is processed with the same parameters through the spatial domain, which changes the dimension. There are  $N_s^{(i)}$  different kernels, which are shared for each slot in the time-frequency axes. Hence, the spatial dimension becomes  $N_s^{(i)}$ . This can be equivalently considered as each vector in the time-frequency slot being multiplied with the same (shared)  $N_s^{(i)} \times N_s^{(i-1)}$  matrix. In what follows, upsampling is performed to exploit the couplings among neighboring elements in the time and frequency grid. More precisely, the time-frequency signal is upsampled with a factor of 2 via a bilinear transformation. Next, the ReLU activation function is used to make the model more expressive for nonlinearities. The last component of a hidden layer performs batch normalization for a batch size of 1 to avoid vanishing gradients. All the hidden layers have the same structure except for the last hidden layer, which does not have an upsampler.

The mathematical representation of the aforementioned architecture is given next. Accordingly, the tensor  $\mathbf{\Lambda}_T$  is parameterized for the  $l + 1$  layer as

$$\hat{\mathbf{\Lambda}}_T = f_{\theta_l}(f_{\theta_{l-1}}(\cdots f_{\theta_0}(Z_0))) \quad (10)$$

where the input  $Z_0$  has a dimension of  $N_f^{(0)} \times N^{(0)} \times N_s^{(0)}$  in the frequency, time and spatial domain, respectively. These dimensions are determined according to the number of hidden layers and the output dimension, in which  $N_s^{(l)} = M$ ,  $N_f^{(l)} = N_f$ ,  $N^{(l)} = N$ . The layers from 0 to  $l - 1$  are counted as a hidden layer, and for  $i = 0, 1, \dots, l - 2$

$$f_{\theta_i} = \text{BatchNorm}(\text{ReLU}(\text{Upsampler}(\theta_i \otimes Z_i))) \quad (11)$$

where  $Z_i$  is the input of the  $i^{\text{th}}$  hidden layer,  $\theta_i$  are the parameters, and  $\otimes$  represents the so-called “convolution” operator, which actually refers to cross-correlation in signal processing. More precisely, a  $1 \times 1$  convolution is utilized as a cross-correlator, which means that the spatial vector for each element of the time-frequency grid is multiplied with the same shared parameter matrix to obtain the new spatial vector for the next hidden layer. The last hidden layer is

$$f_{\theta_{l-1}} = \text{BatchNorm}(\text{ReLU}(\theta_{l-1} \otimes Z_{l-1})), \quad (12)$$

and the output layer is

$$f_{\theta_l} = \theta_l \otimes Z_l. \quad (13)$$

All the parameters can be represented as

$$\Theta = (\theta_0, \theta_1, \dots, \theta_l), \quad (14)$$

which are optimized according to the square of  $l_2$ -norm

$$\Theta^* = \arg \min_{\Theta} \|\mathbf{\Lambda}_T - \hat{\mathbf{\Lambda}}_T\|_2^2. \quad (15)$$

The output of the DNN for the optimized parameters is

$$\mathbf{\Lambda}_T^* = f_{\Theta^*}(Z_0), \quad (16)$$

where the channel estimate  $\hat{\mathbf{\Lambda}}_k$  corresponds to the 2-dimensional  $N_f \times M$  signal at the first time index in (16). Notice that the spatial dimension of  $Z_0$  is a hyper-parameter and its other dimensions are determined by the output signal and the number of layers, since at each layer the time and frequency dimensions are doubled. Furthermore, it is worth emphasizing that in the architecture spatial correlations in the signal are captured by the  $1 \times 1$  convolution so as to decrease the number of parameters, and frequency and temporal correlations are exploited by upsampling.

#### IV. NUMERICAL RESULTS AND COMPLEXITY ANALYSIS

The performance of the proposed channel estimator is evaluated for the following scenario. We assume that there are 4 single antenna users and a single base station that has 16 antennas. Indeed, the number of users are not important as long as  $K \ll 16$ . Users transmit 20 orthogonal OFDM pilot symbols, each of which has 64 subcarriers, over a realistic “extended a pedestrian” channel model employed in LTE. It is also assumed that pilots are sent at the beginning of each coherence time interval. With these settings, the normalized mean square error (NMSE) of the proposed channel estimator is compared with the other methods that were proposed recently in Fig. 5. As can be seen, our deep learning-based channel estimator provides 5-10 dB gain.

One of the main concerns in deep learning models is the overall computational complexity. For our model, the computational complexity of the first stage is composed of training the  $M$  neural networks and generating random samples from the trained neural networks. The former leads to the complexity of  $M\mathcal{O}(W^2)$  where  $W = 32N_f^2$  is the total number of adaptive parameters in the neural network, which stems from the backpropagation algorithm. The latter phase has relatively less complexity, in particular its complexity comes from matrix-vector multiplication. Hence, the first model for channel estimation has a complexity of  $M\mathcal{O}(W^2)$ . The computational complexity of the second stage is much less than the first stage, because there is a convolutional deep neural network architecture that yields a small number of parameters. In particular, the number of parameters in the unsupervised learning model is on the order of hundreds or thousands [17]. Hence, the computational complexity of the proposed pipeline is dominated by the supervised learning tasks.

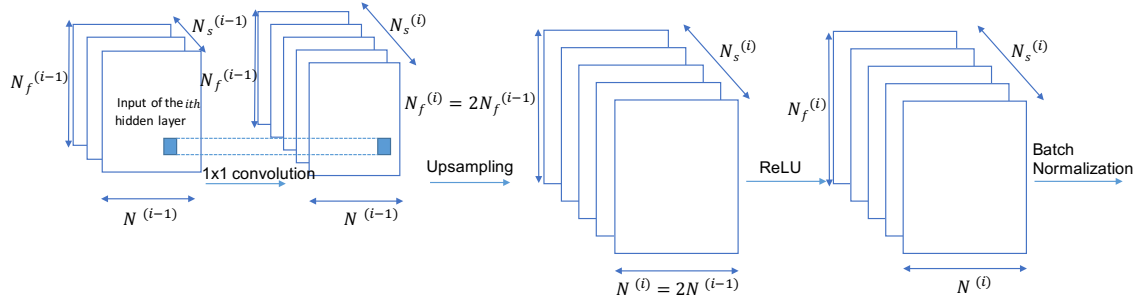


Fig. 4. The structure of the  $i^{th}$  hidden layer, whose input dimension is  $N_f^{(i-1)} \times N^{(i-1)} \times N_s^{(i-1)}$  and output dimension is  $N_f^{(i)} \times N^{(i)} \times N_s^{(i)}$ . Note that  $N_f^{(i)} = 2N_f^{(i-1)}$  and  $N^{(i)} = 2N^{(i-1)}$ . The spatial dimensions  $N_s^{(i-1)}$  and  $N_s^{(i)}$  are the hyperparameters that are used by the  $1 \times 1$  convolution operations.

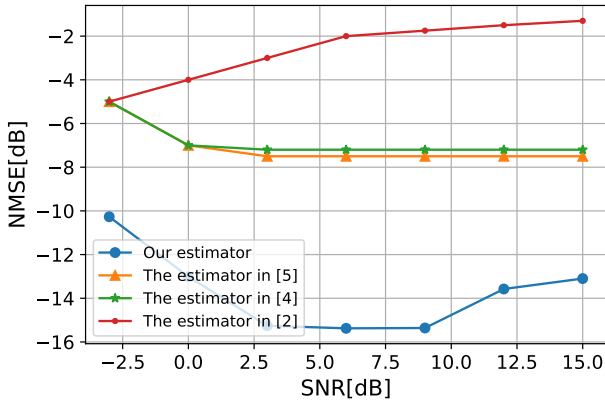


Fig. 5. The performance comparison of the proposed estimator with state-of-the-art one-bit massive MIMO channel estimators.

## V. CONCLUSIONS

In this paper we proposed a novel deep learning-based channel estimator for one-bit massive MIMO in uplink communication. Due to the significant information loss in one-bit quantization, generative models are leveraged as deep learning architectures so as to exploit some prior information in estimating the channel. Our results show that considerable progress can be made with deep generative models for challenging channel estimation problems. As a future work, it can be interesting to consider different types of deep generative models for downlink massive MIMO channel estimation. This is much more challenging than uplink, because many more pilots are required in the downlink channel estimation.

## REFERENCES

- [1] T. L. Marzetta, "Noncooperative cellular wireless with unlimited numbers of base station antennas," *IEEE Trans. Wireless Commun.*, vol. 9, no. 11, pp. 3590 - 3600, November 2010.
- [2] C. Risi, D. Persson, and E. G. Larsson, "Massive MIMO with 1-bit ADC," [Online]. Available: <https://arxiv.org/abs/1404.7736>, April 2014.
- [3] J. Choi, J. Mo, and R. W. Heath, Jr., "Near maximum-likelihood detector and channel estimator for uplink multiuser massive MIMO systems with one-bit ADCs," *IEEE Trans. Commun.*, vol. 64, no. 5, pp. 2005-2018, May 2016.
- [4] C. Mollen, J. Choi, E. G. Larsson, and R. W. Heath Jr., "Uplink performance of wideband massive MIMO with one-bit ADCs," *IEEE Transactions on Wireless Communications*, vol. 16, no. 1, pp. 87-100, January 2017.
- [5] Y. Li, C. Tao, G. Seco-Granados, A. Mezghani, A. L. Swindlehurst, and L. Liu, "Channel estimation and performance analysis of one-bit massive MIMO systems," *IEEE Trans. Signal Process.*, vol. 65, no. 15, pp. 4075-4089, August 2017.
- [6] C.-K. Wen, C.-J. Wang, S. Jin, K.-K. Wong, and P. Ting, "Bayes-optimal joint channel-and-data estimation for massive MIMO with low-precision ADCs," *IEEE Transactions on Signal Processing*, vol. 64, no. 10, pp. 2541-2556, May 2016.
- [7] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.
- [8] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Trans. on Cogn. Commun. Netw.*, vol. 3, no. 4, pp. 563-575, December 2017.
- [9] S. Dörner, S. Cammerer, J. Hoydis, and S. ten Brink, "Deep learning based communication over the air," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 1, pp. 132-143, February 2018.
- [10] C.-K. Wen, W. Shih, and S. Jin, "Deep learning for massive MIMO CSI feedback," *IEEE Wireless Commun. Lett.*, vol. 7, no. 5, pp. 1-4, October 2018.
- [11] E. Balevi and J. G. Andrews, "Online antenna tuning in heterogeneous cellular networks with deep reinforcement learning," *IEEE Transactions on Cognitive Communications and Networking*, doi:10.1109/TCCN.2019.2933420, 2019.
- [12] H. He, C. K. Wen, J. Shi, and G. Y. Li, "Deep learning-based channel estimation for beamspace mmWave massive MIMO systems," *IEEE Wireless Commun. Lett.*, vol. 7, no. 5, pp. 852-855, October 2018.
- [13] Y. Yang, F. Gao, X. Ma, and S. Zhang, "Deep learning-based channel estimation for doubly selective fading channels," *IEEE Access*, vol. 7, pp. 36579-36589, 2019.
- [14] M. Soltani, V. Pourahmadi, A. Mirzaei, and H. Sheikhzadeh, "Deep learning-based channel estimation," *ArXiv preprint arXiv:1810.05893*, February 2018.
- [15] E. Balevi and J. G. Andrews, "One-bit OFDM receivers via deep learning," *IEEE Trans. on Communications*, vol. 67, no. 6, pp. 4326-4336, June 2019.
- [16] E. Balevi and J. G. Andrews, "Deep learning-based channel estimation for high-dimensional signals," *arXiv preprint arXiv:1904.09346*, 2019.
- [17] E. Balevi, A. Doshi, and J. G. Andrews, "Massive mimo channel estimation with an untrained deep neural network," *arXiv preprint arXiv:1908.00144*, 2019.
- [18] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, pp. 2672-2680, December 2014.