Massive MIMO Channel Prediction via Meta-Learning and Deep Denoising: Is a Small Dataset Enough?

Hwanjin Kim\textsuperscript{a,} Member, IEEE, Junil Choi\textsuperscript{a} Senior Member, IEEE, and David J. Love\textsuperscript{a,b} Fellow, IEEE

Abstract—Accurate channel knowledge is critical in massive multiple-input multiple-output (MIMO), which motivates the use of channel prediction. Machine learning techniques for channel prediction hold much promise, but current schemes are limited in their ability to adapt to changes in the environment because they require large training overheads. To accurately predict wireless channels for new environments with reduced training overhead, we propose a fast adaptive channel prediction technique based on a meta-learning algorithm for massive MIMO communications. We exploit the model-agnostic meta-learning (MAML) algorithm to achieve quick adaptation with a small amount of labeled data. Also, to improve the prediction accuracy, we adopt the denoising process for the training data by using deep image prior (DIP). Numerical results show that the proposed MAML-based channel predictor can improve the prediction accuracy with only a few fine-tuning samples in various scenarios. The DIP-based denoising process gives an additional gain in channel prediction, especially in low signal-to-noise ratio regimes.

Index Terms—Channel prediction, massive MIMO, machine learning, meta-learning, denoising, deep image prior.

I. INTRODUCTION

Massive multiple-input multiple-output (MIMO) systems are expected to be critical to nearly all future broadband wireless systems because of the ever-growing demand for increased network spectral efficiency [1]. Accurate channel knowledge at the base station (BS) is often critical to maximizing massive MIMO performance. This is problematic because user equipment (UE) mobility can cause the BS’s channel state information (CSI) to become outdated [2], [3]. One possible solution is to predict the current channel with the past CSI [4], [5], [6], [7].

In 3GPP Release 18, a new study item on artificial intelligence (AI)/machine learning (ML) for the new radio (NR) air interface has been started to investigate the benefit of AI/ML on wireless communication systems [8]. Utilizing advances in AI/ML, ML-based channel predictors have recently been proposed for massive MIMO communications in [9], [10], and [11]. A convolutional neural network (CNN) combined with autoregressive (AR) predictor and a CNN-based predictor using an RNN-based input were proposed in [9]. For vehicular-to-infrastructure (V2I) networks, adaptive channel prediction, beamforming, and scheduling were proposed in [10]. A multilayer perceptron (MLP)-based channel prediction via the estimated UE mobility was developed in [11]. These predictors, however, require high training overhead to obtain accurate CSI prediction results. Furthermore, a well-trained neural network (NN) model could suffer from significant performance degradation when test environments are different from the training environment. It may be possible to mitigate these issues by exploiting more advanced ML techniques.

For adaptive ML-based techniques, meta-learning-based schemes have been developed in [12] and [13]. The basic principle of meta-learning is learning to learn since it aims to learn how to train a network. The meta-learning algorithm makes it possible to adapt to a new environment quickly without training an NN from scratch. With this meta-learning adaptation, it may be possible to predict the current wireless channel of a new environment by using only a few training samples.

So far, meta-learning algorithms have been widely used for channel estimation and prediction in [14], [15], [16], and [17]. Robust channel estimation using meta neural networks (RoemNet) was implemented to estimate the CSI of orthogonal frequency division multiplexing (OFDM) systems in [14], and an online training mechanism was developed for the long short-term memory (LSTM) optimizer based on meta-learning in [15]. Downlink channel prediction using uplink CSI for frequency-division duplexing (FDD) MIMO systems exploiting meta-learning was proposed in [16] and [17]. In this paper, we adopt the model-agnostic meta-learning (MAML) algorithm [18], which is the optimization-based meta-learning...
approach, for fast adaptive channel prediction in massive MIMO communications. In this work, we divide the UEs into two sets, i.e., already existing UEs for a training set and new UEs for a testing set. To be specific, our proposed channel predictor is first trained with the measurement data from the training set. Then, the trained predictor can adaptively predict the channels of new UEs in the testing set (which are different from the UEs used for the training) with only a small number of measurement data. The MAML algorithm is suitable to our problem of interest since it learns an initialization of model parameters, allowing quick adaptation to a new environment using a small amount of data. To the best of our knowledge, this is the first work to predict the channels of new UEs based on the MAML algorithm in massive MIMO systems.

To improve the performance of ML-based techniques, the data denoising process is crucial since the noise-corrupted data in the training phase may cause performance degradation [19]. The least square (LS) or minimum mean-squared error (MMSE) processes are typically used for denoising the training data [20], [21]. However, the LS-based denoising process has limited performance, and the MMSE-based denoising process needs prior knowledge of channel statistics, e.g., channel covariance [22]. Different from the conventional denoising process, ML-based denoising processes have been proposed in [23], [24], [25], and [26]. However, the ML-based denoising approaches have high computational complexity due to the deep neural network (DNN)-based architecture using a large number of labeled data [24], [25], [26]. Moreover, these ML-based approaches use the true channel for the training phase, which is impractical. Different from these works, we adopt the deep image prior (DIP) to resolve these problems [27]. The DIP-based denoising process neither requires the channel statistics nor the true channel data; instead, it only uses the measurement data, i.e., noise-corrupted data, for updating the model, which is well suited to wireless communication environments.

In this paper, we show that it is possible to formulate the massive MIMO channel prediction problem as an optimization problem by exploiting the previous measurements. We then propose a fast adaptive channel predictor based on meta-learning for massive MIMO. We adopt the MAML algorithm for the meta-learning since its structure can adapt to new data with different distributions. Using the temporally correlated measurement data, the channel prediction using the MAML algorithm consists of three stages: meta-training, meta-adaptation, and meta-testing stages. In the meta-training stage, the meta-learner aims to optimize global network parameters. In the meta-adaptation stage, the network parameters are refined using only a few adaptation samples from new environments where the new environments refer to the UEs not considered during the meta-training stage. With these fine-tuned network parameters, the BS predicts the channel of these new UEs in the meta-testing stage. To obtain better prediction results, especially in low signal-to-noise ratio (SNR) regimes, we also exploit the CNN architecture-based DIP to denoise the training data. The numerical results reveal that the proposed MAML-based channel predictor outperforms the conventional vector Kalman filter (VKF)-based and ML-based predictors, especially when the number of available adaptation samples is small. The DIP-based denoising process can give further improvements in channel prediction performance.

Note that even though the BS can accurately predict the channel, the UE still needs to send pilot signals for other purposes, e.g., time synchronization [28], [29], [30]. Sounding reference signal (SRS) is used for the uplink channel estimation while random access channel (RACH) is used for the uplink time synchronization [31], [32], [33]. Note that RACH cannot be used for the channel estimation since RACH is also used for the BS to properly define the timing advance (TA) of its supporting UEs such that the signals from all its supporting UEs arrive at the same time. Only after finishing these procedures, the BS can estimate the uplink channels using SRSs transmitted from the UEs. Therefore, even though RACH and other reference signals still required, channel prediction techniques can significantly lower the overall pilot overhead by reducing the SRS overhead.

The remainder of the paper is structured as follows. We describe a system model and an optimization problem for the channel prediction in Section II. We propose the MAML-based predictor in Section III and explain the DIP-based denoising process in Section IV. In Section V, we examine the computational complexity of the channel predictors and present numerical results to validate our algorithms. Finally, concluding remarks are provided in Section VI.

Notation: Upper case and lower case boldface letters indicate matrices and column vectors, respectively. The transpose, conjugate transpose, and inverse of matrix $A$ are represented by $A^T$, $A^H$, and $A^{-1}$, respectively. $O_m$ denotes the $m \times 1$ all zero vector, and $I_m$ is used for the $m \times m$ identity matrix. $CN(\mathbf{x}, \mathbf{R})$ represents the complex Gaussian distribution having mean $\mathbf{x}$ and covariance $\mathbf{R}$. The set of all $m \times n$ real matrices is represented by $\mathbb{R}^{m \times n}$. $||||$ denotes the $\ell_2$-norm of vector, and $|\cdot|$ represents the amplitude of scalar. $\lfloor\cdot\rfloor$ denotes the floor function of $x$. $O(\cdot)$ represents the Big-O notation. $\mathbb{E}[\cdot]$ denotes the expectation.

II. System Model and Problem Formulation

A. System Model and Considering Scenario

In Fig. 1, we consider a single-cell narrowband uplink massive MIMO system consisting of a BS with $M$ antennas and $K$ UEs with a single-antenna each. We assume that the BS trains the NN with the uplink pilot signals from $K$ UEs. Then, the BS aims to predict the channels of new UEs based on the pre-trained NN and a few adaptation samples from the new UEs pilot signals. Note that we define the new UEs as the UEs that are served by the BS for the first time. Since the BS predicts each UE channel separately, we consider only the $k$-th UE’s input-output expression

$$y_{n,k} = \sqrt{p}h_{n,k}x_{n,k} + w_{n,k},$$

(1)

where $p$ is the SNR, $h_{n,k}$ is the channel between the BS and the $k$-th UE, $x_{n,k}$ is the pilot signal, and the complex Gaussian noise is denoted as $w_{n,k} \sim CN(0_M, I_M)$. Also, the received
signal from the $k$-th new UE at the BS during the $n$-th time slot is expressed as
\[ y_{n,k} = \sqrt{\rho} h_{n,k} x_{n,k} + w_{n,k}, \]
where $k \in K_{\text{new}}$ is the new UE index, and $K_{\text{new}}$ is the index set of new UEs.

**Remark:** Note that all UEs are connected to the same BS in our scenario of interest. Therefore, even though the UEs may experience different channel conditions, e.g., line-of-sight (LOS) or non-line-of-sight (NLOS) paths, they would experience similar channel statistics, which will be naturally incorporated into our proposed channel predictor. The BS mobility, however, must be carefully handled before channel prediction since the mobility dictates the temporal correlation of channels. One possible way is that the BS can first classify the UEs into several groups where the UEs in each group have the same (or at least similar) mobility and then perform channel prediction for each group separately.

**B. Problem Formulation**

To predict the new UE channel $h_{n,k}$, we use the temporal correlation of the channels, i.e., based on the proper complexity order of $n_o$, the BS predicts the channel by exploiting the previous $n_o$ measurements. The optimization problem for the channel prediction is defined as
\[
\begin{align*}
\text{minimize} & \quad \| h_{n+1,k} - \hat{h}_{n+1,k} \|^2 \\
\text{subject to} & \quad h_{n+1,k} = f(y_{n-n_o+1,k}, \ldots, y_{n,k}),
\end{align*}
\]
where $\hat{h}_{n+1,k}$ is the predicted channel for $k$-th UE at the $(n + 1)$-th time slot produced by the prediction function $f(\cdot)$. Note that using the true channel $h_{n+1,k}$ as the target value for the optimization problem in (3) is impractical. Therefore, we assume that the BS only exploits realistic measurement data for the target value as
\[
\begin{align*}
\text{minimize} & \quad \| h_{n+1,k}^{LS} - \hat{h}_{n+1,k} \|^2 \\
\text{subject to} & \quad h_{n+1,k}^{LS} = f(h_{n-n_o+1,k}^{LS}, \ldots, h_{n,k}^{LS}),
\end{align*}
\]
where $h_{n,k}^{LS}$ is the least square (LS) channel estimate given by
\[
\hat{h}_{n,k}^{LS} = \frac{1}{\sqrt{\rho x_{n,k}}} y_{n,k} = h_{n,k} + w_{n,k}, \quad \forall n, \forall k,
\]
with $w_{n,k} = \frac{1}{\sqrt{\rho x_{n,k}}} w_{n,k}$. We assume that the SNR $\rho$ is a long-term statistic and can be perfectly estimated at the BS [34]. From the NN training perspective, the loss function can be defined as the sum of mean-squared error (MSE) between the LS channel estimate and predicted channel,
\[
\text{Loss} = \frac{1}{N} \sum_{n=1}^{N} \| h_{n+1,k}^{LS} - \hat{h}_{n+1,k} \|^2,
\]
where $N$ denotes the number of samples. The loss function in (6) will be used for the MAML algorithm in Section III. In the following sections, we will use the terms received signals and measurements interchangeably.

**III. MAML-Based Channel Prediction**

To obtain accurate channel prediction in (3) using conventional ML techniques for various scenarios, e.g., different UE configurations, the BS requires a large amount of training overhead for each scenario [9], [35], [36]. It is crucial to resolve this training issue for ML techniques to work in practice, and we exploit the meta-learning algorithm to address this problem. With the meta-learning algorithm, the BS can predict the channels of various UE configurations more quickly using a small number of adaptation samples.

**A. MAML Structure and Task**

Among many possible meta-learning algorithms, we adopt the MAML algorithm proposed in [18] that is used in various neural networks. The MAML algorithm has a hierarchical structure with the meta-learner and the learner, consisting of three stages: 1) meta-training stage, 2) meta-adaptation stage, and 3) meta-testing stage as in Fig. 2. We will explain the meta-learning stages in detail based on Fig. 2 in Sections III-C and III-D. Following the terminology of meta-learning, we define a meta-learning task $T$, which consists of a dataset and a loss function $T = \{ D, \text{Loss}_{D} \}$ [37], as the prediction of a target UE channel exploiting previous measurements. The meta-learning task $T$ is also composed of a source task $T_S$ for the meta-training stage and a target task $T_T$ for the meta-adaptation and meta-testing stages. We will define $D$ and $\text{Loss}_{D}$ of the task $T$ and the relation among $T_S$, $T_T$, $D$, and $\text{Loss}_{D}$ in detail in Sections III-B and III-C.

The BS first trains the meta-learner with the source task. Then, the meta-learner helps the learner adjust to a new task utilizing only a small number of adaptation samples from the target task. The meta-learner aims to learn the inductive bias while the learner adapts to a new task with this inductive bias.
Fig. 2. MAML structure: meta-training, meta-adaptation, and meta-testing stages.

Fig. 3. MAML datasets consist of the meta-training, meta-adaptation, and meta-testing datasets. After the LS estimation process of the uplink pilot signals, the BS allocates the LS channel estimates into each dataset. The meta-training datasets include the training and validation datasets. Each sample pair in the task consists of \( n_o \) input features and one label to exploit the temporal correlation of channels.

B. Definition of MAML Datasets

For each stage of the MAML algorithm, we use an independent dataset. We define the LS channel estimates from already existing \( K \) UEs in the meta-training stage as the source dataset \( \mathbb{D}_S \) in \( T_S \), and the LS channel estimates from new UEs (that are different from already existing UEs used during the meta-training stage) in the meta-adaptation and meta-testing stages as the target dataset \( \mathbb{D}_T \) in \( T_T \). We refer to the support set as \( \mathbb{D}_{Sup} \) for the training data and the query set as \( \mathbb{D}_{Que} \) for the validation data during the meta-training stage. To prevent the network model from overfitting, we split the support set \( \mathbb{D}_{Sup} \) and the query set \( \mathbb{D}_{Que} \), i.e., \( \mathbb{D}_{Sup} \cap \mathbb{D}_{Que} = \emptyset \). We define the datasets for the meta-adaptation and meta-testing stages as \( \mathbb{D}_{Ad} \) and \( \mathbb{D}_{Te} \), respectively. Also, we assume that no sample in \( \mathbb{D}_{Te} \) appears in \( \mathbb{D}_{Ad} \), i.e., \( \mathbb{D}_{Te} \cap \mathbb{D}_{Ad} = \emptyset \). Then, it is clear that \( \mathbb{D}_S = \mathbb{D}_{Sup} \cup \mathbb{D}_{Que} \) and \( \mathbb{D}_T = \mathbb{D}_{Ad} \cup \mathbb{D}_{Te} \). Note that the distribution in the target dataset \( \mathbb{D}_T \) is different from the distribution in the source dataset \( \mathbb{D}_S \). Thus, all MAML datasets are non-overlapping.

Fig. 3 reveals the MAML datasets, which consist of the meta-training, meta-adaptation, and meta-testing datasets. After the LS estimation process of the uplink pilot signals, the BS collects the uplink pilot signals from multiple UEs, then performs the LS estimation as in (5). Finally, the LS channel estimates are binned into each dataset. In the meta-training stage, the BS uses the total number of \( N_S = N_u K_s \) source tasks \( \{T_S(t)\}_{t=1}^{N_S} \), where \( N_u \) is the number of source tasks per UE, and \( K_s \) is the number of UEs for the source task. Therefore, the multiple \( K_s \) UEs are used for the source dataset. Each dataset of the \( t \)-th source task \( \mathbb{D}_{Sup}(t) \) consists of two disjoint datasets: the support set \( \mathbb{D}_{Sup}(t) \) and the query set \( \mathbb{D}_{Que}(t) \). We denote the support set of \( t \)-th source task, which includes \( N_s \) labeled data, as \( \mathbb{D}_{Sup}(t) = \{ (p_{Sup,i}^{(i)}, q_{Sup,i}^{(i)}) \}_{i=1}^{N_s} \), where \( \{ p_{Sup,i}^{(i)}, q_{Sup,i}^{(i)} \} \) is the \( i \)-th sample pair in the support set. We use \( n_o \) input features \( p_{Sup,i}^{(i)} = \{ h_{S1}^{LS}(s_{i}^{(i)} - n_o + 1, k_t), \ldots, h_{S1}^{LS}(s_{i}^{(i)}, k_t) \} \) and one label
In (8), the BS uses the LS channel estimate $\mathbf{q}_{\text{Sup}, t}^{i} = \mathbf{h}_{s_t^{i}+1, k_t}^{\text{LS}}$, where $s_t^{i}$ is the $i$-th sample index of the support set, and $k_t = \lfloor \frac{t - 1}{N_T} \rfloor + 1$ is the UE index for the $t$-th source task.

Similarly, the query set of the $t$-th source task with $N_q$ labeled data is denoted as $\mathbb{D}_{\text{Que}}(t) = \{ \{ \mathbf{p}_{\text{Que}, t}^{(i)}, \mathbf{q}_{\text{Que}, t}^{(i)} \} \}_{i=1}^{N_q}$, where $\{ \mathbf{p}_{\text{Que}, t}^{(i)}, \mathbf{q}_{\text{Que}, t}^{(i)} \}$ is the $i$-th samples pair in the query set. Also, each sample pair includes $n_o$ input features $\mathbf{p}_{\text{Que}, t}^{(i)} = \{ \mathbf{h}_{1}^{\text{LS}}(i) - n_o + 1, k_t, \ldots, \mathbf{h}_{N_o}^{\text{LS}}(i) - n_o + 1, k_t \}$ and the corresponding label $\mathbf{q}_{\text{Que}, t}^{(i)} = \mathbf{h}_{s_t^{i+1}, k_t}^{\text{LS}}$, where $s_t^{i+1}$ is the $i$-th sample index of the query set for the $t$-th source task.

In the target task $\mathcal{T}_T$, we define the meta-adaptation dataset with the number of adaptation samples $N_{\text{ad}}$ as $\mathbb{D}_{\text{Ad}} = \{ \{ \mathbf{p}_{\text{Ad}}, \mathbf{q}_{\text{Ad}} \} \}_{i=1}^{N_{\text{ad}}}$, where $\mathbf{p}_{\text{Ad}}^{(i)} = \{ \mathbf{h}_{a_t^{i}}^{\text{LS}}(i) - n_o + 1, k_t, \ldots, \mathbf{h}_{a_{N_o}^{i}}^{\text{LS}}(i) - n_o + 1, k_t \}$ and $\mathbf{q}_{\text{Ad}}^{(i)} = \mathbf{h}_{s_t^{i+1}, k_t}^{\text{LS}}$. Note that $a_t^{i}$ is the $i$-th target sample index of the meta-adaptation dataset, and $k_t$ is the target UE index of the meta-adaptation dataset, which is the same as the new UE index in (2). We also define the meta-testing dataset with the number of test samples $N_{\text{te}}$ as $\mathbb{D}_{\text{Te}} = \{ \{ \mathbf{p}_{\text{Te}}, \mathbf{q}_{\text{Te}} \} \}_{i=1}^{N_{\text{te}}}$, where $\mathbf{p}_{\text{Te}}^{(i)} = \{ \mathbf{h}_{b_t^{i}}^{\text{LS}}(i) - n_o + 1, k_t, \ldots, \mathbf{h}_{b_{N_o}^{i}}^{\text{LS}}(i) - n_o + 1, k_t \}$ and $\mathbf{q}_{\text{Te}}^{(i)} = \mathbf{h}_{s_t^{i+1}, k_t}$ with the $i$-th target sample index of the meta-testing dataset $s_t^{i}$.

### C. Meta-Training Stage

The objective of a meta-learner is to acquire the inductive bias from the entire source tasks $\{ \mathcal{T}_S(t) \}_{t=1}^{N_S}$ for fast adaptation in the meta-training stage. As in Fig. 2, the meta-training stage has three steps: 1) network parameter initialization, 2) inner-task parameter update, and 3) outer-task parameter update. The meta-learner updates inner-task parameters of the $t$-th source task $\Omega_{\text{Tr}, t}$ iteratively. Then, the global network parameter $\Omega_{\text{Ad}}$ is updated in the outer-task update. In the inner-task update, the BS trains the NN parameters of each task in the corresponding batch, where the batch is the group of source tasks for efficiently updating gradient steps. The BS groups the source tasks by the batch size of $V$ and updates the NN parameters with $V$ source tasks in each iteration. The BS uses the mini-batch stochastic gradient descent (SGD) method [38] using the batch size of $V$ to update the inner-task parameters of the $t$-th source task, $\Omega_{\text{Tr}, t}$,

$$\Omega_{\text{Tr}, t} \leftarrow \Omega_{\text{Tr}, t} - \alpha \nabla_{\Omega_{\text{Tr}, t}} \text{Loss}_{\text{Sup}}(\Omega_{\text{Tr}, t}), \quad t = 1, \ldots, V,$$

where $\alpha$ represents the inner-task learning rate and $\text{Loss}_{\text{Sup}}(t)$ denotes the loss function on $\mathbb{D}_{\text{Sup}}(t)$. We use the MSE between the target value $\mathbf{q}_{\text{Sup}, t}^{(i)}$ and the predicted value $\hat{\mathbf{q}}_{\text{Sup}, t}^{(i)}$ as the loss function

$$\text{Loss}_{\text{Sup}}(t) = \frac{1}{N_s} \sum_{i=1}^{N_s} \left\| \mathbf{q}_{\text{Sup}, t}^{(i)} - \hat{\mathbf{q}}_{\text{Sup}, t}^{(i)} \right\|^2.$$

In (8), the BS uses the LS channel estimate $\mathbf{q}_{\text{Sup}, t}^{(i)} = \mathbf{h}_{s_t^{i}+1, k_t}^{\text{LS}}$ as the target value. Note that $\mathbf{h}_{s_t^{i}+1, k_t}^{\text{LS}}$ is corrupted with the noise, and we exploit the DIP architecture to denoise the LS channel estimate in Section IV.

After the inner-task update, the outer-task update is performed to optimize the global network parameters $\Omega_{\text{Ad}}$. In the outer-task update, the BS updates the global network parameters $\Omega_{\text{Ad}}$ to minimize the sum of the loss functions of tasks on $\mathbb{D}_{\text{Que}}(t)$, i.e.,

$$\sum_{t=1}^{V} \text{Loss}_{\text{Que}}(t)(\Omega_{\text{Tr}, t}),$$

where $\text{Loss}_{\text{Que}}(t)$ is the loss function on the query set $\mathbb{D}_{\text{Que}}(t)$ as in (8). The global network parameters $\Omega_{\text{Ad}}$ is updated by the adaptive moment estimation (ADAM) optimizer [39] with the outer-task learning rate $\beta$.

The BS performs the inner-task and outer-task updates iteratively according to the number of epochs $N_{\text{epoch}}$, which indicates the total number of passes through the entire training dataset. Thus, the total number of iterations for the meta-training stage is $N_{\text{epoch}}N_S/V$ since the number of iterations in each epoch is $N_S/V$. With these definitions, now we can concretely define the source task $\mathcal{T}_S = \{ \mathbb{D}_{\text{Sup}}, \mathbb{D}_{\text{Que}}, \text{Loss}_{\text{Sup}}, \text{Loss}_{\text{Que}} \}$.

### D. Meta-Adaptation and Meta-Testing Stages

In the meta-adaptation stage, the BS updates the network parameters to adapt to a new task quickly using the adaptation dataset $\mathbb{D}_{\text{Ad}}$ based on the pre-trained global network parameters $\Omega_{\text{Ad}}$. The adaptation parameters $\Omega_{\text{Ad}}$ are updated by the

---

**Algorithm 1 MAML-Based Channel Predictor**

1. **Input**: Source task $\{ \mathcal{T}_S(t) \}_{t=1}^{N_S}$, Target task $\mathcal{T}_T$, inner-task learning rate $\alpha$, outer-task learning rate $\beta$, batch size $V$, number of epochs $N_{\text{epoch}}$
2. **Output**: Predicted channel
3. **Meta-training stage**:
   4. Randomly initialize the neural network parameters
   5. for $j = 1, \ldots, N_{\text{epoch}}N_S/V$ do
   6. Randomly sample $V$ batch of tasks from $\{ \mathcal{T}_S(t) \}_{t=1}^{N_S}$
   7. Generate datasets $\{ \mathbb{D}_{\text{Sup}}(t) \}_{t=1}^{N_S}$ and $\{ \mathbb{D}_{\text{Que}}(t) \}_{t=1}^{N_S}$
   8. for $t = 1, \ldots, V$ do
   9. Update $\Omega_{\text{Tr}, t}$ by (7) with $\mathbb{D}_{\text{Sup}}(t)$
10. end for
11. Update $\Omega_{\text{Ad}}$ to minimize (9)
12. end for
13. **Meta-adaptation stage**:
14. Generate datasets $\mathbb{D}_{\text{Ad}}$ and $\mathbb{D}_{\text{Te}}$ from $\mathcal{T}_T$
15. Load the meta-trained network parameters
16. for $j = 1, \ldots, N_{\text{gr}}$ do
17. Update $\Omega_{\text{Ad}}$ by (10) with $\mathbb{D}_{\text{Ad}}$
18. end for
19. **Meta-testing stage**:
20. Predict the channel based on $\mathbb{D}_{\text{Te}}$ and $\Omega_{\text{Ad}}$

---

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
SGD method with the number of adaptation samples $N_{\text{ad}}$ as

$$\Omega_{\text{Ad}} \leftarrow \Omega_{\text{Ad}} - \alpha \nabla \Omega_{\text{Ad}} \text{Loss}_{\text{Ad}}(\Omega_{\text{Ad}}),$$

where $\text{Loss}_{\text{Ad}}$ is the loss function of adaptation dataset $\mathcal{D}_{\text{Ad}}$. After finishing the fine-tuning with $N_{\text{gr}}$ gradient steps, the meta-testing stage gives the predicted channel using $\Omega_{\text{Ad}}$ and $\mathcal{D}_{\text{T}e}$. To summarize, the learner updates the network parameters based on the adaptation data from the target tasks (consist of the new UEs), and then the predictor provides the predicted channel from the meta-testing dataset after the data adaptation as in Fig. 2. The proposed MAML-based channel prediction algorithm is summarized in Algorithm 1. We can also rigorously define the target task $T$ as in (5) is a kind of inverse problem [40], [41]. The inverse problem is recovering the channel $h$ where $\text{Loss}$ defined as recovering the model parameter $f$ from the measurement data $g$ defined as

$$g = A(f) + e,$$  

where $A$ is the forward operator mapping the model parameter to data, and $e$ is the random observation noise. In our system, the inverse problem is recovering the channel $h_{n,k}$ from the LS estimate $h_{n,k}^{LS} = h_{n,k} + w_{n,k}$, where the forward operator $A$ is just the identity transform. With the prior of the channel statistics, we can get a clean signal by maximizing the likelihood function,

$$h_{n,k}^{LS} = \arg\max_{h_{n,k}} p(h_{n,k} | h_{n,k}^{LS})$$

subject to $h_{n,k}^{LS} = g_{\Phi}(z)$,

where $g_{\Phi}(z)$ is the NN function with the network parameters $\Phi$ and the input $z$. The principle idea of the DIP is that the NN output can represent the structured signal more than the random noise since the NN architecture imposes a strong prior. The DIP architecture itself has the ability to learn the strong prior even in the complete absence of any training process. The minimization problem in (13) does not need to have any statistical knowledge, and the solution is obtained by using the gradient descent on the NN function without any training in advance. Thus, we can obtain the denoised data with low-complexity using an untrained NN.

For the DIP-based denoising process, we stack the LS channel estimates in the time domain. Specifically, the LS channel estimates in (5) are reformulated as the 2-dimensional data $H^{LS}$,

$$H^{LS} = \{ [ H^{LS}_{1D}(m,n)]_{m=1}^{M} \}_{n=1}^{N},$$

where $H^{LS}_{1D}(m,n)$ is the LS channel estimate at the $m$-th BS antenna during the $n$-th time slot. Since the DIP architecture only supports real-valued data, the real and imaginary components of the 2-dimensional data are stacked into the BS antenna domain. This reformulated form of $H^{LS}$ is defined as $\mathcal{H}^{LS} \in \mathbb{R}^{2M \times N}$.

In Fig. 4, the DIP architecture contains an input-layer, $L_{d}$ hidden-layers, and an output-layer. Also, each hidden-layer has four components, which are the $1 \times 1$ convolutional layer, upsampling layer, rectified linear unit (ReLU) activation layer, and batch normalization layer. The $i$-th hidden layer for $1 \leq i \leq L_{d} - 1$ is given as

$$g_{\Phi_{i}} = \text{Batch} \{ \text{ReLU} \{ \text{Upsample} \{ \phi_{i} \circ \mathbf{Z}_{i} \} \} \},$$

where $\phi_{i}$ are the model parameters of $i$-th hidden layer, $\circ$ denotes the $1 \times 1$ convolution operation, and $\mathbf{Z}_{i}$ is the input of $i$-th hidden-layer. Note that the dimensions of the BS antenna domain and the time domain for the $i$-th layer are $M_{i}$ and $N_{i}$, respectively. Since we use the $1 \times 1$ convolution to capture the spatial correlation, the number of network parameters in the DIP architecture is decreased. In the upsampling layer,
the time dimension is doubled by the bilinear transformation, i.e., $N_{t+1} = 2N_t$. Since the channel is temporally correlated, the upsampling layer can leverage the correlation between the adjacent elements in the time domain. For the last hidden-layer, we set the ReLU activation layer followed by the batch normalization layer as

$$g_{\phi_{L_d}} = \text{Batch(ReLU)}(\phi_{L_d} \otimes Z_{L_d})), \quad (16)$$

to avoid the vanishing gradients problem. The final output-layer is

$$g_{\phi_{L_{d+1}}} = \phi_{L_{d+1}} \otimes Z_{L_{d+1}}. \quad (17)$$

The optimization problem for the DIP architecture is given by the $\ell_2$-norm

$$\Phi^* = \arg \min_\Phi \|\mathcal{H}^{LS} - \hat{\mathcal{H}}^{LS}\|^2, \quad (18)$$

where $\Phi = [\phi_1, \ldots, \phi_{L_{d+1}}]$, and $\hat{\mathcal{H}}^{LS} = g_{\Phi}(Z_1)$ is the estimate of $\mathcal{H}^{LS}$. Note that $Z_1$ is a random initial value with the dimension of $M_1 \times N_1$. The DIP architecture gives the solution for (18) using the ADAM optimizer with the number of iterations $N_{\text{iter}}$.

V. COMPUTATIONAL COMPLEXITY AND NUMERICAL RESULTS

The computational complexity of the proposed MAML channel prediction, the MLP channel prediction, which is our benchmark, and the denoising process based on the DIP is first analyzed in this section. Then, we evaluate the prediction performance of the MAML-based predictor compared to other techniques including the MLP channel prediction. For the complexity analysis, we exploit the floating-point operations (FLOPs) with the Big-$O$ notation [42].

A. Computational Complexity

The MAML-based channel prediction with the MLP structure has three levels of complexity, i.e., the complexity of the meta-training, meta-adaptation, and meta-testing stages. In the meta-training stage, the complexity using the number of epochs $N_{\text{epoch}}$, the total number of source tasks $N_S$, the number of meta-training sample pairs in each task $N_{mt} = N_s + N_q$, the complexity order $n_o$, and the number of hidden-layers $L$ using $n_l$ nodes is given by [43]

$$C_{\text{MAML-train}} = \mathcal{O}\left( N_{\text{epoch}} N_S N_{mt} (n_o M_1 + 1) + \sum_{l=1}^{L-1} n_l n_{l+1} + n_L M \right). \quad (19)$$

In addition, the complexity of the meta-testing stage with the number of test samples $N_{te}$ is $\mathcal{O}\left( N_{te} N_s (n_o M_2 + 1) + (L-1) M^2 \right).$ Finally, the total complexity of the MAML-based predictor becomes

$$C_{\text{MAML}} = \mathcal{O}\left( N_{\text{epoch}} N_S N_{mt} + N_S N_{ad} + N_{te} \right). \quad (20)$$

The total complexity of the MLP-based predictor can be approximated to $\mathcal{O}(N_{\text{epoch}} N_S N_{mt} M^2)$. After the meta-training stage, the complexity of the meta-adaptation stage becomes $\mathcal{O}(N_{gr} N_{ad} M^2)$, which is much lower than that of the meta-training stage since $N_S N_{mt} \gg N_S N_{ad}$. In practice, the BS can perform the meta-training stage in advance, and only the meta-adaptation stage is needed online to predict the channels of new UEs. Thus, the BS can achieve fast adaptive channel prediction using the MAML algorithm.

The complexity of the MLP channel prediction, which is our benchmark, is also analyzed. In the training stage, the MLP channel prediction uses $N_{\text{MLP}}$ training samples. Then, the complexity of the MLP channel prediction in the training stage becomes

$$C_{\text{MLP-train}} = \mathcal{O}\left( N_{\text{epoch}} N_{\text{MLP}} (n_o + (L-1) M^2) \right). \quad (21)$$

Since we use the same number for the MLP channel prediction as in the MAML prediction, the total complexity of the MLP

| Method | Stage       | Complexity                                                                 | Total complexity                                                                 |
|--------|-------------|---------------------------------------------------------------------------|---------------------------------------------------------------------------------|
| MAML   | Train       | $\mathcal{O}\left( N_{\text{epoch}} N_S N_{mt} (n_o + (L-1) M^2) \right)$   | $\mathcal{O}\left( N_{\text{epoch}} N_S N_{mt} + N_G N_{ad} + N_{te} \right)$   |
|        | Adaptation  | $\mathcal{O}\left( N_S N_{ad} (n_o + (L-1) M^2) \right)$                  |                                                                                 |
|        | Test        | $\mathcal{O}\left( N_{te} (n_o + (L-1) M^2) \right)$                      |                                                                                 |
| MLP    | Train       | $\mathcal{O}\left( N_{\text{epoch}} N_{\text{MLP}} (n_o + (L-1) M^2) \right)$ | $\mathcal{O}\left( N_{\text{epoch}} N_{\text{MLP}} + N_S N_{ad} + N_{te} \right)$ |
|        | Adaptation  | $\mathcal{O}\left( N_S N_{ad} (n_o + (L-1) M^2) \right)$                  |                                                                                 |
|        | Test        | $\mathcal{O}\left( N_{te} (n_o + (L-1) M^2) \right)$                      |                                                                                 |
| DIP    | -           | $\mathcal{O}\left( N_{\text{iter}} N_t N_f (2N_f + M) \right)$             |                                                                                 |
channel prediction is given by

\[ C_{\text{MLP}} = O\left((N_{\text{epoch}}N_{\text{MLP}} + N_{\text{gr}}N_{\text{ad}} + N_{\text{te}}) \cdot \gamma(n_o + (L - 1)\gamma + 1)M^2\right). \] (23)

The DIP-based denoising process exploits the CNN structure as in Fig 4. Since the complexity of the CNN is dominated by the convolutions, we only consider the complexity of the convolution operations for the DIP-based denoising process. In the \( i \)-th convolutional layer, a group of \( M_i \) filters of the \( 1 \times 1 \) convolution are applied to \( M_i \) feature maps of the dimension \( N_i \times 1 \). The complexity of the DIP-based denoising process with the number of iterations \( N_{\text{iter}} \) is given by [44]

\[ C_{\text{DIP}} = O\left( \sum_{i=1}^{L_d} N_i M_i^2 + N_{L_d+1} M_{L_d+1}M \right) \]

(24)

where (a) comes from \( N_i = 2^{i-1}N_1 \) for \( 1 \leq i \leq L_d \) and \( N_{L_d+1} = N_{L_d+1} \). (b) is from the assumption of using the same number of filters \( N_i \) in each layer, i.e., \( M_i = N_i \) for all \( i \), and (c) is derived by \( N_{L_d+1} = \frac{1}{2} N_i N_L + (2^{L_d-1} - 1)N_i N_L \approx 2^{L_d-1}N_i N_L \). The complexity of the DIP-based denoising process can be approximated as \( O(N_{\text{iter}}N_i M) \) for large \( M \). The computational complexity of the MAML channel prediction, the MLP channel prediction, and the DIP-based denoising process is summarized in Table I.

### B. Numerical Results

The ML algorithms are implemented by the TensorFlow 2.0 and the NVIDIA Quadro RTX 8000 GPU for the numerical simulations. We perform Monte-Carlo simulations to verify the proposed channel prediction algorithm. In this paper, we employ the normalized mean-squared error (NMSE)

\[ \text{NMSE} = \mathbb{E}\left[ \frac{\|\mathbf{h}_{n+1,k_T} - \tilde{\mathbf{h}}_{n+1,k_T}\|^2}{\|\mathbf{h}_{n+1,k_T}\|^2} \right], \] (25)

for the performance metric. Also, we use the achievable sum-rate as another performance metric. To reduce the inter-user interference, the zero-forcing (ZF) combiner is adopted

\[ \mathbf{F}_n^T = \left( \mathbf{H}_n^H \mathbf{H}_n \right)^{-1} \mathbf{H}_n^H, \] (26)

with the predicted channel matrix \( \mathbf{H}_n = \left[ \hat{\mathbf{h}}_{n,1} \cdots \hat{\mathbf{h}}_{n,K_T} \right] \). Note that \( K_T \) is the number of UEs in the target task. We obtain the unit-norm combiner \( f_{n,k_T} = \frac{\mathbf{f}_{n,k_T}}{\|\mathbf{f}_{n,k_T}\|} \), where \( \mathbf{f}_{n,k_T} \) represents the \( k_T \)-th columns of \( \mathbf{F}_n \). For the \( k_T \)-th UE, the achievable rate based on the receive combiner \( f_{n,k_T} \) can be expressed as

\[ R_{k_T} = \log_2 \left( 1 + \frac{\rho \|\mathbf{f}_{n,k_T} \mathbf{h}_{n,k_T}\|^2}{\rho \|\mathbf{f}_{n,k_T} \| ^2 + 1} \right). \] (27)

We then define the achievable sum-rate as

\[ R = \sum_{k_T=1}^{K_T} R_{k_T}. \] (28)

In (28), we do not consider any pilot overhead since all techniques to be compared exploit the same number of adaptation samples, i.e., the pilots from the new UEs.

Unless stated otherwise, we assume the spatial channel model (SCM) urban micro (UMi) scenario in [45] with carrier frequency \( f_c = 2.3 \) GHz, UE mobility \( v = 3 \) km/h, and time slot duration \( T_d = 40 \) ms. We use the MLP with \( L = 4 \) hidden layers as the NN structure for the MAML algorithm. Each hidden-layer consists of a fully-connected layer with 512 nodes. For the DIP architecture, we adopt the CNN with the number of iterations \( N_{\text{iter}} = 2000 \) and \( L_d = 4 \) hidden-layers including \( M_t = 64 \) for all \( i \). We also set the number of BS antennas \( M = 64 \), the number of source tasks per UE \( N_u = 1024 \), the complexity order \( n_o = 3 \), the number of epochs \( N_{\text{epoch}} = 20 \), and the batch size \( V = 64 \). The number of sample pairs of the support set is \( N_s = 10 \), and the number of sample pairs of the query set is \( N_q = 10 \). The inner-task learning rate is set to \( \alpha = 10^{-1} \), and the outer-task learning rate is set to \( \beta = 10^{-5} \). The system parameters are summarized in Table II.

### Table II

| Parameter                        | Value  |
|----------------------------------|--------|
| Environment                      | UMi    |
| Carrier frequency                | 2.3 GHz|
| UE mobility                      | 3 km/h |
| Time slot duration               | 40 ms  |
| Number of BS antenna             | 64     |
| Number of source tasks per UE    | 1024   |
| Complexity order                 | 3      |
| Number of epochs                 | 20     |
| Batch size                       | 64     |
| Number of sample pairs in support set | 10  |
| Number of sample pairs in query set | 10  |
| Inner-task learning rate         | 10^{-1}|
| Outer-task learning rate         | 10^{-5}|

In the simulation, we compare the following predictors:

- **Extrapolation**: extrapolation-based prediction in [11].
- **VFK**: vector Kalman filter-based channel prediction in [11]. This serves as a baseline of our predictor.
- **MLP**: first optimized with the source dataset and re-trained with \( N_{\text{ad}} \) adaptation samples from the target dataset without the denoising process.
- **MAML**: proposed MAML-based prediction without the denoising process.
- **MLP-DIP**: MLP with the denoised LS channel estimate based on the DIP.
- **MAML-DIP**: proposed MAML-based prediction with the denoised LS channel estimate based on the DIP.

The extrapolation is a simple linear filter while the VFK channel prediction is known to be optimal for many linear
systems including the channel prediction [46], [47], [48]. Note that the MLP-based and the MAML-based predictions rely on the same source and target tasks, and the main difference between them is the structure of NN. To be fair, we use the same number of training samples for the MLP and the MAML channel predictions, i.e., \(N_{\text{MLP}} = N_sN_{\text{mt}}\). The VKF channel prediction also uses the same adaptation samples \(N_{\text{ad}}\) as in the MLP and MAML-based predictions to obtain the sample covariance of channels for fair comparison.

We consider the LS channel estimates for a total number of 8 UEs with the number of UEs in the source task \(K_s = 4\) and the number of UEs in the target task \(K_t = 4\). The BS trains each network with the first 4 UE LS channel estimates from the source dataset \(\mathcal{D}_S\) and tests with the remaining 4 UE LS channel estimates from the target dataset \(\mathcal{D}_T\) to obtain the average NMSE and sum-rate with \(N_{\text{te}} = 100\).

To verify the effect of the number of source tasks per UE \(N_u\), Fig. 5 shows the NMSEs of the MLP channel prediction and the MAML channel prediction as a function of \(N_u\). In this simulation, we set the number of gradient steps \(N_{\text{gr}} = 10\), the number of adaptation samples \(N_{\text{ad}} = 20\), and SNR = 20 dB. The NMSEs of both channel predictions without the DIP decrease as the number of source tasks per UE increases, but eventually saturate. The denoising process is able to break this saturation effect on both methods while the gain of denoising is larger for the MAML channel prediction. We set the number of source tasks per UE at \(N_u = 1024\) for the following simulations.

In Fig. 6, we compare the MAML channel prediction to the MLP channel prediction in terms of NMSE according to the complexity order with \(N_{\text{gr}} = 10\), \(N_{\text{ad}} = 20\), and SNR = 20 dB. The NMSEs of both channel predictions decrease as the complexity order increases until \(n_o = 3\), but the gain becomes marginal after. Therefore, we set \(n_o = 3\) to balance the accuracy and complexity of channel predictions in the following simulations. Note that the complexity order needs to be larger to achieve the same accuracy when the UE mobility increases [11].

Fig. 7 shows the NMSEs of the MLP channel prediction and the MAML channel prediction according to the number of gradient steps with \(N_{\text{ad}} = 20\) and SNR = 20 dB. The figure shows that the proposed MAML channel prediction gives a moderate gain compared to the MLP channel prediction regardless of the number of gradient steps. In addition, the NMSE of the MAML channel prediction almost converge when the number of gradient steps reaches 10 while the MLP channel prediction requires to have more gradient steps to converge. Both channel predictors converge when the number of gradient steps becomes 40, but the gain of MAML prediction over MLP prediction still remains, i.e., the MAML-DIP has an additional 2 dB NMSE gain compared to the MLP-DIP. In the following simulations, we set \(N_{\text{gr}} = 10\).

In Figs. 8 and 9, the NMSEs of VKF, MLP, and MAML channel predictions are compared according to the number of adaptation samples with different SNR values. The figures clearly show that the NMSE of the VKF channel prediction is much worse than those of the MLP and MAML channel predictions within 100 adaptation samples. As shown in Fig. 8, the VKF channel prediction performs slightly better than the MAML channel prediction as \(N_{\text{ad}}\) increases while the
MLP-DIP and MAML-DIP channel predictions have substantial gain over the VKF channel prediction, which shows the benefit of the DIP-based denoising process when the SNR is low. On the contrary, when the SNR is large as in Fig. 9, the VKF channel prediction eventually gives better performance than the MAML-DIP channel prediction as $N_{ad}$ increases since the sample covariance matrix used in the VKF channel prediction becomes accurate. The NMSEs of the MLP and MAML channel predictions almost converge when the number of adaptation samples reaches 100, and the gain of MAML prediction over MLP prediction still exists.

In Fig. 10, the NMSEs of the VKF, MLP, and MAML channel predictions are compared according to the UE mobility with SNR = 20 dB and $N_{ad} = 20$. We also plot the extrapolation-based channel prediction in [11]. The extrapolation-based channel prediction, which is a simple average filter, uses two snapshots, so the performance degrades significantly as the UE mobility increases. Also, the VKF channel prediction performs poorly with 20 adaptation samples as discussed in Figs. 8 and 9. On the contrary, the MAML channel prediction can guarantee acceptable performance even in medium mobility cases, e.g., 50 km/h. The performance of the MLP and MAML channel predictions will eventually converge to that of VKF channel prediction as the mobility further increases though.

Fig. 11 plots the NMSEs of the VKF, MLP, and MAML channel predictions according to the SNR with different values of $N_{ad}$. Even with 200 adaptation samples, the NMSE of the VKF channel prediction is worse than that of the MAML channel prediction with 20 adaptation samples. The NMSEs of the MLP and MAML channel predictions saturate as the SNR increases but the gap between the MLP and MAML channel predictions remains. Although the effect of noise will eventually become negligible as the SNR increases, the gain of the DIP exists even for quite large SNR values.

Fig. 12 depicts the achievable sum-rates of the VKF, MLP, and MAML channel predictions according to the SNR with different values of $N_{ad}$. Even with 200 adaptation samples, the NMSE of the VKF channel prediction is worse than that of the MAML channel prediction with 20 adaptation samples. The NMSEs of the MLP and MAML channel predictions saturate as the SNR increases but the gap between the MLP and MAML channel predictions remains. Although the effect of noise will eventually become negligible as the SNR increases, the gain of the DIP exists even for quite large SNR values.
channel prediction with $N_{ad} = 20$. Because of better prediction quality, the MAML channel prediction can achieve much higher achievable sum-rate than the MLP channel prediction when the SNR is large enough. The DIP-based denoising process further boosts the achievable sum-rate, which shows the importance of data preprocessing before training the NN.

We also verify whether the proposed MAML channel prediction works when the UEs experience different mobilities. In this case, we consider only two UEs in the target dataset where the first UE has 3 km/h mobility while the other UE has 10 km/h mobility. The BS has two pre-trained NNs, one based on the 3 km/h UEs and the other based on the 10 km/h UEs. In the absence of mobility estimation, the BS predicts the channel through the NNs trained with the UEs having 3 km/h mobility. For the estimated mobility, we adopt the spatial average of temporal correlation (SATC)-based mobility estimator developed in [11] using just two snapshots of the channel. The BS then predicts the channel based on the NNs corresponding to the estimated UE mobility. Fig. 13 shows the rate per UE of the MAML channel prediction with and without mobility estimation. The figure clearly shows that our approach can effectively handle the case when UEs have different mobilities. Note that Fig. 13 shows the rate of UE with 3 km/h mobility depends on the mobility estimation even though the BS exploits the NN trained with the 3 km/h UEs for the case without mobility estimation. This is because the channel of 10 km/h UE is not accurately predicted without mobility estimation, resulting in more interference to the 3 km/h UE.

Finally, we evaluate the NMSEs of the VKF, MLP, and MAML channel predictions according to the number of adaptation samples with the rural macro (RMa) scenario in Fig. 14. The MLP and MAML channel predictions have a moderate gain over the VKF channel prediction even in this scenario. Therefore, it is clear from the figure that the proposed MAML algorithm works well in various environmental scenarios.

VI. CONCLUSION

In this paper, we proposed a fast adaptive channel predictor for massive MIMO systems using the MAML algorithm, which is the popular meta-learning technique. The proposed MAML channel prediction extracts the key characteristics of time-varying channels and exploits these features to adaptively predict the channels of new UEs. Also, the DIP-based denoising process applied to the training samples further improves the prediction performance by reducing the noise effect. Numerical results showed that the MAML channel prediction provides improvements in the complexity, accuracy, and achievable sum-rate even with only a few adaptation samples in various scenarios. These improvements make the MAML channel prediction highly practical.

A possible extension of proposed channel prediction technique is to consider wideband massive MIMO systems. Our preliminary work in [49] for wideband systems efficiently exploited the frequency correlation; however, [49] is restricted to a single UE case with a basic MLP structure. Exploiting the meta-learning algorithm for the multi-user wideband scenario would be an interesting future research topic.

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, Nov. 2010.
[2] A. K. Papazafeiropoulos, “Impact of general channel aging conditions on the downlink performance of massive MIMO,” IEEE Trans. Veh. Technol., vol. 66, no. 2, pp. 1428–1442, Feb. 2017.

[3] K. T. Truong and R. W. Heath, “Effects of channel aging in massive MIMO systems,” J. Commun. Netw., vol. 15, no. 4, pp. 338–351, Aug. 2013.

[4] J. Choi, D. J. Love, and P. Bidigare, “Downlink training techniques for FDD massive MIMO systems: Open-loop and closed-loop training with memory,” IEEE J. Sel. Topics Signal Process., vol. 8, no. 5, pp. 802–814, Oct. 2014.

[5] C. Kong, C. Zhong, A. K. Papazafeiropoulos, M. Matthaiou, and Z. Zhang, “Sum-rate and power scaling of massive MIMO systems with channel aging,” IEEE Trans. Commun., vol. 63, no. 12, pp. 4877–4893, Dec. 2015.

[6] S. G. Larew and D. J. Love, “Adaptive beam tracking with the unscented Kalman filter for millimeter-wave communication,” IEEE Signal Process. Lett., vol. 26, no. 11, pp. 1658–1662, Nov. 2019.

[7] T.-H. Chou, N. Michelusi, D. J. Love, and J. V. Krogmeier, “Fast position-aided MIMO beam training via noisy tensor completion,” IEEE J. Sel. Topics Signal Process., vol. 15, no. 3, pp. 774–788, Apr. 2021.

[8] Study on Artificial Intelligence (AI)/Machine Learning (ML) for NR Air Interface, Standard 3GPP TR 38.843 Std., Dec. 2021.

[9] J. Yuan, H. Q. Ngo, and M. Matthaiou, “Machine learning-based channel prediction in massive MIMO with channel aging,” IEEE Trans. Wireless Commun., vol. 19, no. 5, pp. 2960–2973, May 2020.

[10] T. E. Bogale, X. Wang, and L. B. Le, “Adaptive channel prediction, beamforming and scheduling design for 5G V2I network: Analytical and machine learning approaches,” IEEE Trans. Veh. Technol., vol. 69, no. 5, pp. 5055–5067, May 2020.

[11] H. Kim, S. Kim, H. Lee, C. Jang, Y. Choi, and J. Choi, “Massive MIMO channel prediction: Kalman filtering vs. machine learning,” IEEE Trans. Comms., vol. 69, no. 1, pp. 518–528, Jan. 2021.

[12] M. Andrzychowicz et al., “Learning to learn by gradient descent by gradient descent,” in Proc. Adv. Neural Inf. Process. Syst., vol. 29, 2016, pp. 3988–3996.

[13] S. Ravi and H. Larochelle, “Optimization as a model for few-shot learning,” in Proc. Int. Conf. Learn. Represent., 2017, pp. 1–11.

[14] H. Mao, H. Lu, Y. Lu, and D. Zhu, “RoomNet: Robust meta learning based channel estimation in OFDM systems,” in Proc. IEEE Int. Conf. Commun. (ICC), May 2019, pp. 1–6.

[15] J. Zhang, Y. He, Y.-W. Li, C.-K. Wen, and S. Jin, “Meta-learning based MIMO detectors: Design, simulation, and experimental test,” IEEE Trans. Wireless Commun., vol. 20, no. 2, pp. 1122–1137, Feb. 2021.

[16] Y. Yang, F. Gao, Z. Zhong, B. Ai, and A. Alkhatib, “Deep transfer learning-based downlink channel prediction for FDD massive MIMO systems,” IEEE Trans. Commun., vol. 68, no. 12, pp. 7485–7497, Dec. 2020.

[17] J. Zeng et al., “Downlink CSI feedback algorithm with deep transfer learning for FDD massive MIMO systems,” IEEE Trans. Cognit. Commun. Netw., vol. 7, no. 4, pp. 1253–1265, Dec. 2021.

[18] C. Finn, P. Abbeel, and S. Levine, “Model-agnostic meta-learning for fast adaptation of deep networks,” in Proc. 34th Int. Conf. Mach. Learn., 2017, pp. 1126–1135.

[19] H. Kim, S. Kim, H. Lee, and J. Choi, “Massive MIMO channel prediction: Machine learning versus Kalman filtering,” IEEE Trans. Microw. Theory Mater., vol. 69, no. 2, pp. 551–560, Feb. 2021.

[20] Y. Jin, J. Zhang, S. Jin, and B. Ai, “Channel estimation for cell-free massive MIMO systems,” IEEE Trans. Veh. Technol., vol. 68, no. 10, pp. 10325–10329, Oct. 2019.

[21] M. Soltani, V. Pourrahimi, A. Mirzaei, and H. Sheikhzadeh, “Deep learning-based channel estimation,” IEEE Commun. Lett., vol. 23, no. 4, pp. 652–655, Apr. 2019.

[22] E. Balevi, A. Doshi, and J. G. Andrews, “Massive MIMO channel estimation with an untrained deep neural network,” IEEE Trans. Wireless Commun., vol. 19, no. 3, pp. 2070–2090, Mar. 2020.

[23] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, “Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising,” IEEE Trans. Image Process., vol. 26, no. 7, pp. 3142–3155, Jul. 2017.

[24] H. He, C.-K. Wen, S. Jin, 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, Oct. 2018.

[25] Y. Zhang, Y. Mu, Y. Liu, T. Zhang, and Y. Qian, “Deep learning-based beamspace channel estimation in mmWave massive MIMO systems,” IEEE Wireless Commun. Lett., vol. 9, no. 12, pp. 2212–2215, Dec. 2020.

[26] H. Ye, F. Gao, J. Qian, H. Wang, and G. Y. Li, “Deep learning-based denoise network for CSI feedback in FDD massive MIMO systems,” IEEE Commun. Lett., vol. 24, no. 8, pp. 1742–1746, Aug. 2020.

[27] V. Lempitsky, A. Vedaldi, and D. Ulyanov, “Deep image prior,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 9446–9454.

[28] Physical Channels and Modulation, Standard 3GPP TS 38.211 V17.4.0 Std., Jan. 2023.

[29] Physical Layer Procedures for Control, Standard 3GPP TS 38.212 V17.4.0 Std., Jan. 2023.

[30] Physical Layer Procedures for Data, Standard 3GPP TS 38.213 V17.4.0 Std., Jan. 2023.
Hwanjin Kim (Member, IEEE) received the B.S. (Hons.) and M.S. degrees in electrical engineering from POSTECH, in 2017 and 2019, respectively, and the Ph.D. degree from the School of Electrical Engineering, KAIST, in 2022. He is currently a Post-Doctoral Researcher with KAIST. His research interests include the design and analysis of massive MIMO communication systems and machine learning for communications. Dr. Kim was a co-recipient of the 2018 JCCI Best Paper Award and the 2020 Electronic News ICT Paper Contest Best Paper Award. He was awarded the IDIS–Electronic News ICT Paper Contest Outstanding Paper Award in 2017, the KAIST EE Best Research Achievement Award (Kim Choong-Ki Award) in 2020, the IEEE Seoul Section Student Paper Contest Silver Prize in 2021, and the KAIST EE Best Ph.D. Dissertation Award in 2022. He was a recipient of the NRF Korea Research Subsidies of Ph.D. Candidates (2021–2022), the 2021 KAIST Global Leader Fellowship (Kim Younghan Fellowship), and the 2022 KAIST Global Leadership Award (Creativity). He was recognized as an Exemplary Reviewer of the IEEE COMMUNICATIONS LETTERS in 2018.

Junil Choi (Senior Member, IEEE) received the B.S. (Hons.) and M.S. degrees in electrical engineering from Seoul National University, in 2005 and 2007, respectively, and the Ph.D. degree in electrical and computer engineering from Purdue University in 2015. He is currently an (Named) Ewon Associate Professor with the School of Electrical Engineering, KAIST. From 2007 to 2011, he was a member of the Technical Staff, Samsung Advanced Institute of Technology (SAIT), Samsung Electronics Company Ltd., South Korea, where he has contributed to advanced codebook and feedback framework designs for the 3GPP LTE/LTE-advanced and IEEE 802.16m standards. Before joining KAIST, he was a Post-Doctoral Fellow with The University of Texas at Austin, from 2015 to 2016, and an Assistant Professor with POSTECH, from 2016 to 2019. His research interests include the design and analysis of massive MIMO, mmWave communication systems, distributed reception, and vehicular communication systems.

Dr. Choi was a co-recipient of the 2022 IEEE Vehicular Technology Society Best Vehicular Electronics Paper Award, the 2021 IEEE Vehicular Technology Society Neal Shepherd Memorial Best Propagation Award, the 2019 IEEE Communications Society Stephen O. Rice Prize, the 2015 IEEE Signal Processing Society Best Paper Award, and the 2013 Global Communications Conference (GLOBECOM) Signal Processing for Communications Symposium Best Paper Award. He was awarded the Michael and Katherine Birk Fellowship from Purdue University in 2011, the Korean Government Scholarship Program for Study Overseas (2011–2013), the Purdue University ECE Graduate Student Association (GSA) Outstanding Graduate Student Award in 2013, the Purdue College of Engineering Outstanding Student Research Award in 2014, the IEEE ComSoc AP Region Outstanding Young Researcher Award in 2017, the NSF Korea and Elsevier Young Researcher Award in 2018, the KICS Haedong Young Researcher Award in 2019, and the IEEE Communications Society Communication Theory Technical Committee Early Achievement Award in 2021. He is an Associate Editor of IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, IEEE TRANSACTIONS ON COMMUNICATIONS, IEEE COMMUNICATIONS LETTERS, and IEEE OPEN JOURNAL OF THE COMMUNICATIONS SOCIETY.

David J. Love (Fellow, IEEE) received the B.S. (Hons.), M.S.E., and Ph.D. degrees in electrical engineering from The University of Texas at Austin, in 2000, 2002, and 2004, respectively. Since 2004, he has been with the Elmore Family School of Electrical and Computer Engineering, Purdue University, where he is currently the Nick Trbovich Professor of electrical and computer engineering. He was a member of the Executive Committee for the National Spectrum Consortium. He holds 32 issued U.S. patents. His research interests include the design and analysis of broadband wireless communication systems, beyond-5G wireless systems, multiple-input multiple-output (MIMO) communications, millimeter wave wireless, software defined radios and wireless networks, coding theory, and MIMO array processing.

He is a fellow of the American Association for the Advancement of Science (AAAS). He was named a Thomson Reuters Highly Cited Researcher in 2014 and 2015. Along with his co-authors, he has won the Best Paper Awards from the IEEE Communications Society, such as 2016 Stephen O. Rice Prize and 2020 Fred W. Ellersick Prize; the 2015 IEEE Signal Processing Society Best Paper Award from IEEE Signal Processing Society, and the 2010 Jack Neubauer Memorial Award from IEEE Vehicular Technology Society. He served as a Senior Editor for IEEE Signal Processing Magazine, an Editor for the IEEE TRANSACTIONS ON COMMUNICATIONS, an Associate Editor for the IEEE TRANSACTIONS ON SIGNAL PROCESSING, and a Guest Editor for special issues of the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS and the EURASIP Journal on Wireless Communications and Networking.