Received XX Month, XXXX; revised XX Month, XXXX; accepted XX Month, XXXX; Date of publication XX Month, XXXX; date of current version 11 January, 2024.

Digital Object Identifier 10.1109/OJCOMS.2024.011100

Impact of Channel and System Parameters on Performance Evaluation of Frequency Extrapolation Using Machine Learning

Michael Neuman[1] *(Member, IEEE)*, Daoud Burghal[2] *(Member, IEEE)* AND Andreas F. Molisch [1] *(Fellow, IEEE)*

[1]Ming Hsieh Department of Electrical and Computer Engineering, University of Southern California, Los Angeles, CA, USA
[2]D. Burghal was at the University of Southern California and is now at Samsung Research America, Mountain View, CA 94043, USA.

CORRESPONDING AUTHOR: Michael Neuman (e-mail: mikeneum@ usc.edu).

This work was supported by the National Science Foundation under Grant CIF-2008443 and Grant SWIFT-2229535.

ABSTRACT Channel extrapolation in the frequency domain is an important tool for reducing overhead and latency in frequency division duplex (FDD) wireless communications systems. Over the past years, various machine learning (ML) techniques have been proposed for this goal, but their effectiveness is usually evaluated only for a limited number of examples. This paper presents an extensive investigation of the impact of various parameters, both of the system and of the underlying channels, on the performance of three types of ML algorithms, namely K-nearest neighbor (KNN), convolutional multilayer perceptron (CNN/MLP), and autoencoder structures (AE). We analyze the impact of channel coherence bandwidth and coherence distance, and the number of multipath components, as well as system bandwidth, number of subcarriers, duplex distance, and signal-to-noise ratio (SNR) in uplink and downlink. We also consider both complex and magnitude normalized mean-square error (NMSE) as training and evaluation metrics. Physical interpretations of the obtained results are given. Most importantly, we find that the NMSE can vary by 10 dB or more over physically reasonable ranges of parameters but often shows saturation behavior over part of those ranges. We also find that, in particular, KNN results can be quantitatively and qualitatively different from CNN/MLP and AE. These investigations thus provide insights into meaningful parameter choices for the performance evaluation of new ML algorithms for frequency-domain channel extrapolation.

INDEX TERMS Channel Extrapolation, Coherence Bandwidth, Frequency Domain Duplexing, Machine Learning

I. INTRODUCTION

A. Background and Motivation

TRANSMISSION schemes have become considerably more adaptive as wireless communications systems have evolved to fifth-generation (5G). While, for example, the second-generation (2G) standards used fixed modulation order and scheduling, fourth-generation (4G) and 5G use orthogonal frequency-division multiplexing (OFDM) with adaptive modulation and coding, adaptive scheduling, and more [1, Chap. 30-32]. This trend is expected to continue and become even more pronounced in sixth-generation (6G) [2]. Such adaptivity requires knowledge, at the base station (BS), of the channel state information (CSI) for both uplink (UL) and downlink (DL), the latter of which constitutes CSI at the transmitter (CSIT).[1]

The most straightforward way for acquiring CSIT occurs when the DL CSI can be directly obtained from the pilot

[1]Generally, the problem can be formulated as knowledge of CSIT, which for the DL means CSI at the BS, while for the UL it means CSI at the user equipment (UE). However, in 4G and 5G, all decisions - both for UL and DL - are made by the BS. We thus henceforth exclusively consider the case of CSIT at the BS.

signals in the UL, i.e., the UL and DL channels are identical (or, at a minimum, highly correlated). The conditions for this to be fulfilled are, in the time division duplex (TDD) case, that UL and DL have to occur within less than a coherence time of the channel, while for a frequency division duplex (FDD) system, they must be spaced less than a coherence bandwidth apart. While the former condition is generally fulfilled for current systems, the latter assumption is never fulfilled in practice: typical frequency duplex distances are tens or hundreds of MHz, while coherence distances are in the low MHz or even KHz range [1, Chap. 16]. While the pros and cons of FDD or TDD for CSIT acquisition have been the subject of numerous papers, e.g., [3], the debate is to a certain extent moot: government regulators have assigned particular frequency bands for use with FDD (paired bands), and this assignment cannot be changed in the foreseeable future.[2]

When no direct inference is possible, CSIT must be acquired via feedback, i.e., the DL CSI is measured at the user equipment (UE), e.g., based on pilot signals embedded in the DL signal, and this information is fed back (possibly in a quantized and/or compressed form) to the BS. While this method is simple and easy to implement, it leads to significant overhead (when little compression is used) and/or significant performance loss when high compression is used, such as with the codebook-based approach of 3GPP [4]. The situation is worse in fast-varying channels, which not only leads to the need to spend a larger percentage of available capacity on feedback but also increases the impact of feedback latency, i.e., the CSIT is outdated.

Thus, schemes that would allow us to accurately infer, in an FDD system, the DL CSI directly from the UL pilot signals are of great interest. Such a problem, which is sketched in Fig. 1, is called the CSI extrapolation in the frequency domain (for simplicity of notation, we will simply refer to it as CSI extrapolation henceforth). Due to its practical importance, much research has been performed on this topic. Various methods ranging from least-square estimates to extrapolations based on high-resolution parameter estimation have been proposed [5], [6], [7], [8]. For example, one of these proposed approaches (see, e.g., [8]) has the advantage of direct connection to the propagation physics, relies on the extraction of the parameters of the multipath components from the UL measurements and synthesizing their interaction at the DL frequency. However, in reality, nonspecular components, generally modeled as diffuse multipath components (DMC), also play a role [9]. Other studies, e.g., [10], [11], [12], focus on second-order statistics, such as the transformation of the DL covariance matrix from the observed UL covariance matrix by interpolation, which, while valuable, does not provide the full benefits that instantaneous CSIT provides. In this paper, however, we will solely focus on machine learning (ML) based techniques, whose popularity has skyrocketed over the past years.

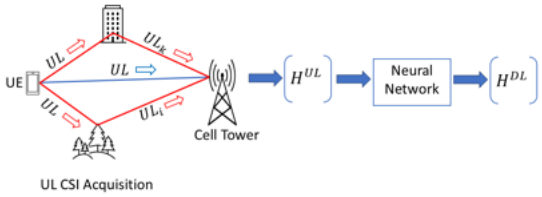


FIGURE 1: Frequency extrapolation framework.

B. Previous Work on ML Extrapolation

While the above-mentioned approaches do reduce the CSI extrapolation overhead, they all have limitations, both from a theoretical perspective (e.g., the validity of the amplitude of MPCs being the same in UL and DL) and from a practical perspective (e.g., the need for precision array calibration). Furthermore, some of these algorithms become unrealistically complex when the number of multipath components (MPCs) becomes large or when the amount of CSIT increases, such as in massive MIMO, or when the UE is moving fast, so that the frequent computations become necessary, such as estimation of the DL covariance matrix from the UL covariance matrix within the coherence time of the channel. Due to the success of ML in different challenging applications, the wireless communication community has seen a new perspective on several classical wireless problems. Therefore, much research has exploited ML to perform frequency extrapolation. Studies such as [13] have proposed ML to define a mapping that allows direct extrapolation to infer CSI information from one set of massive MIMO radiation patterns to another set of patterns, or [14] has shown how ML can determine the relationship between the UL CSI to the DL CSI in FDD systems, and [15] has suggested ML extrapolate CSI information from the sub-6 GHz to the millimeter wave band. All of which reduce the CSI extrapolation overhead. Some studies employ convolutional neural networks (CNN) [15], [16] or multi-layer perceptron networks (MLP) [13], [14], [17] to perform this mapping. More recent studies have taken advantage of improvements in ML tools to further improve the neural network used for CSI extrapolation. For example, [18] proposes a 3D complex-valued convolutional neural network, [13] proposes a complex MLP, and [19] suggests a complex domain CNN architecture named SCNet to improve the complex-valued

[2]Recently, the FCC has introduced the 3.5 GHz band to promote more efficient and flexible spectrum usage, especially for TDD-based wireless communications. However, this does not change the fact that a significant portion of the most valuable spectrum, particularly in sub-3 GHz bands, remains assigned to paired frequency allocations and will continue to be, due to the favorable propagation characteristics and established FDD infrastructure in these bands. Thus, determining CSIT for FDD is a practically relevant topic and will remain important for the foreseeable future.

CSI extrapolation. As a natural extension, combinations of CNN and MLP neural networks, convolutional multilayer perceptron (CNN/MLP) [20], [21], along with combinations of CNN and a long short-term memory network (LSTM-net) network have been suggested to perform CSI extrapolation directly [22].

As ML network architectures evolve in other fields, the wireless communication community has also explored these ML architectures' CSI extrapolation capabilities. Autoencoder (AE) and variational autoencoders (VAE) have been shown capable of performing the regression prediction tasks necessary for CSI extrapolation [23]–[25]. Recurrent neural networks (RNN) have been introduced [26], along with Bayesian neural networks (BNNs) [27] as a method to mitigate uncertainty in rapidly changing CSI during the CSI extrapolation. Modifications of generative adversarial networks (GANs) have also been explored, [28] proposes a deep generative model (DGM), and [29] developed a conditional generative adversarial network (CGAN) to perform CSI extrapolation.

While all of these studies exhibit great promise in lowering the overhead of CSI extrapolation, very few address the actual impact of varying wireless propagation environments or varying system parameters. For example, Doshi *et al*. [30] discusses the impact of pilot tone signal-to-noise ratios (SNR) while training their ML network; many others assume noiseless pilot tones. Furthermore, our search of recent articles on CSI extrapolation or CSI estimation found that few studies discuss the propagation environment, and only very few studies [19], [30] explicitly determine their results as a function of the propagation environment's parameters. Finally, several studies specified how and with what density they obtained training data (e.g., [20]), but none provided guidelines on their effect on extrapolation performance.

C. Contributions of this Paper

From the above survey, we see that a wide range of ML techniques has been brought to bear on the problem of CSI extrapolation in FDD systems. However, the performance evaluations are generally done with specific settings for the wireless systems and the propagation channels, which makes the generalizability of both the absolute performance numbers and, even potentially, the relative merits of different ML schemes difficult to assess. The main point of this paper is to analyze the impact of different system and channel parameters on the performance of generic ML systems. *We emphasize that we do not provide new ML algorithms or architectures but rather focus on insights into the interactions of the channel and system parameters with ML algorithms.* The results of this paper are thus intended to serve as guidelines for meaningful channel/system simulation settings in the performance evaluation of future (or existing) ML algorithms for frequency extrapolation.

Specifically, we provide the error performance of the frequency extrapolation of DL CSI for the following:

- *various ML algorithms:* we compare performance with generic K-nearest neighbor (KNN), CNN/MLP, and AE algorithms;
- *different error measures:* we consider the normalized mean square error for both complex CSI and its magnitude for both training and evaluation;
- *different system parameters:* we analyze (i) duplex distance, (ii) bandwidth, (iii) number of subcarriers and subcarrier density, (iv) UL and DL error metrics;
- *different channel parameters;* we consider different coherence bandwidth (B_{coh}) and coherence length (L_{coh}), as well as different numbers of MPCs in the system.

Note that all the analysis is for single-input single-output (SISO) systems; we refrain from the use of MIMO to avoid an excessive number of system and channel parameters.

D. Structure of the Paper

The remainder of the paper is organized as follows: in Section II, we discuss ML evaluation metrics and the three ML configurations. Section III describes our dataset generation, followed by a discussion of synthesizing wireless environments. Section IV is the core of this paper, elucidating how the characteristics of the wireless channel impact the ML based frequency extrapolation. Section V provides concluding remarks.

II. ML MODELS

A. Evaluation Metric

All of our three ML methods rely on supervised training, which in turn use the UL and DL CSI, more precisely, the channel's complex transfer functions $\left\{\mathbf{H}^{\mathrm{UL}}, \mathbf{H}^{\mathrm{DL}}\right\}$. The Normalized Mean Square Error (NMSE) metric is a common performance metric used to define the error of the ML's ability to estimate the DL CSI. The NMSE metric with our complex UL and DL transfer functions is defined as,

$$NMSE = \frac{\mathbb{E}_{\rho_k,\nu_l}\left|\boldsymbol{H}^{\mathrm{DL}}_{\rho_k,\nu_l} - \boldsymbol{H}^{\widehat{\mathrm{DL}}}_{\rho_k,\nu_l}\right|^2}{\mathbb{E}_{\rho_k,\nu_l}\left|\boldsymbol{H}^{\mathrm{DL}}_{\rho_k,\nu_l}\right|^2} \tag{1}$$

where: $\mathbb{E}$ is the expectation operator over both spatial location (ρ_k) and sub-carrier frequency (ν_l). The $\boldsymbol{H}^{\mathrm{DL}}_{\rho_k,\nu_l}$ and $\boldsymbol{H}^{\widehat{\mathrm{DL}}}_{\rho_k,\nu_l}$ are the actual and estimated complex values of the DL channel's transfer function respectively.

When considering the error of the magnitude of the transfer function, the NMSE is defined as

$$NMSE = \frac{\mathbb{E}_{\rho_k,\nu_l}\left||\boldsymbol{H}^{\mathrm{DL}}_{\rho_k,\nu_l}| - |\boldsymbol{H}^{\widehat{\mathrm{DL}}}_{\rho_k,\nu_l}|\right|^2}{\mathbb{E}_{\rho_k,\nu_l}\left|\boldsymbol{H}^{\mathrm{DL}}_{\rho_k,\nu_l}\right|^2} \tag{2}$$

The choice of the type of error function or distance metric shown in the numerator of (1) and (2) is an important parameter in all three ML methods. In the KNN algorithm, it is important not only in determining NMSE performance but also in determining the desired type of DL CSI, i.e.,

how we define the "nearest neighbor". Specifically, we might consider the full (complex) transfer function $\mathbf{H}^{\mathrm{UL}}$ or its (real) magnitude value. Table I is a summary of these metrics, along with their mathematical expressions and their names used in this paper.

Which of those quantities is more related to the system performance? The choice depends on the wireless application for which the estimated downlink $\mathbf{H}^{\mathrm{DL}}$ is used. For example, if it is for scheduling or power control, the magnitude of the DL signal may be all that is required. However, if it is for beam forming, the real and imaginary values of the DL estimate are required. In contrast, the AE and CNN/MLP networks treat the real and imaginary transfer function component $\left\{\mathbf{H}^{\mathrm{UL}}, \mathbf{H}^{\mathrm{DL}}\right\}$ as separate data points and are thereby treated independently during the supervised training.

Error	Distance Metric	NMSE Expression
Complex	$\left\|\boldsymbol{H}^{\mathrm{DL}}_{\rho_k,\nu_l} - \boldsymbol{H}^{\widehat{\mathrm{DL}}}_{\rho_k,\nu_l}\right\|^2$	$\frac{\mathbb{E}_{\rho_k,\nu_l}\left\|\boldsymbol{H}^{\mathrm{DL}}_{\rho_k,\nu_l} - \boldsymbol{H}^{\widehat{\mathrm{DL}}}_{\rho_k,\nu_l}\right\|^2}{\mathbb{E}_{\rho_k,\nu_l}\left\|\boldsymbol{H}^{\mathrm{DL}}_{\rho_k,\nu_l}\right\|^2}$
Magnitude	$\left\|\lvert\boldsymbol{H}^{\mathrm{DL}}_{\rho_k,\nu_l}\rvert - \lvert\boldsymbol{H}^{\widehat{\mathrm{DL}}}_{\rho_k,\nu_l}\rvert\right\|^2$	$\frac{\mathbb{E}_{\rho_k,\nu_l}\left\|\lvert\boldsymbol{H}^{\mathrm{DL}}_{\rho_k,\nu_l}\rvert - \lvert\boldsymbol{H}^{\widehat{\mathrm{DL}}}_{\rho_k,\nu_l}\rvert\right\|^2}{\mathbb{E}_{\rho_k,\nu_l}\left\|\boldsymbol{H}^{\mathrm{DL}}_{\rho_k,\nu_l}\right\|^2}$

TABLE I: Error metrics, distance metrics, and NMSE expressions.

Notation: $\mathbb{E}_{\rho_k,\nu_l}$ denotes the expectation over spatial location ρ_k and sub-carrier frequency ν_l. The terms $\boldsymbol{H}^{\mathrm{DL}}_{\rho_k,\nu_l}$ and $\boldsymbol{H}^{\widehat{\mathrm{DL}}}_{\rho_k,\nu_l}$ denote the true and estimated complex-valued downlink channel transfer functions, respectively.

B. Considered Machine Learning Algorithms

This subsection describes the three ML architectures we use in our evaluation. Overall, our goal is to choose typical, widely used ML network structures that allow us to analyze the impact of different systems and channel parameters on the performance of such systems, rather than structures optimized for particular applications.

With this in mind, we chose the KNN algorithm because it can be viewed as similar to a lookup table, relying on the similarities between data points. This simplicity makes KNN an excellent benchmark for understanding how well a model learns patterns and whether it is learning beyond mere memorization, offering insights into the impact of system and environmental parameters on performance.

Autoencoders (AE) and Variational autoencoders (VAE) have been shown to be capable of performing regression/prediction tasks [23]–[25]. Based on our survey of previous works, we chose a basic AE architecture that allows us to easily implement supervised learning from the KNN ML method.

The architecture of our CNN/MLP neural network originates, in part, from our recognition that frequency extrapolation would benefit from the regression capabilities of an MLP neural network. Also, motivated by the CNN-based network's success in image pattern recognition and the well-known fact that CNN captures local correlation better than MLP, we configure a neural network that consists of both a convolutional section and a multilayer perceptron section (CNN/MLP). Furthermore, a similar CNN/MLP was introduced in [16], [20]. Again, We stress that they are not intended to be innovative, but rather generic structures that show the fundamental behavior of these algorithms.

1) KNN Machine Learning Algorithm

Our KNN method is a supervised learning algorithm trained using the UL transfer function values distance to its K UL neighbors as a criterion to estimate the DL CSI (in our case, we choose $K = 1$, see Sec. IVA for rational). Our KNN ML methodology is shown as an algorithm flow diagram in Algorithm 1. Algorithm 1 comprises two "for loops". An outer loop (lines 2-10) that cycles through each $\boldsymbol{H}^{\mathrm{UL}}$ transfer function at their specified test locations. An inner loop (lines 3-7) that calculates the distance between each test $\boldsymbol{H}^{\mathrm{UL}}$ transfer function and the $\boldsymbol{H}^{\mathrm{UL}}$ transfer function for all training locations. The distance metrics shown in lines 4 and 5, along with the MSE metrics shown in lines 8 and 9 of Algorithm 1, are defined in Table I and are chosen according to the desired type of wireless application as discussed in Sec. IIA.

2) AE Neural Network

The overall AE configuration used for the regression task of estimating the DL CSI from the UL CSI is shown in Fig. 2(a). As shown, the AE neural network consists of an encoder and decoder with a latent or bottleneck layer interconnecting them. The encoder and decoder are identical, fully connected MLP neural networks with ReLU activation functions, while the latent layer is a linear layer with 64 neurons. The parameters of the AE are shown in the Table II.

3) CNN/MLP Neural Network

The configuration of a neural network that consists of both a convolutional section and perception layers CNN/MLP, as shown in Fig. 2(b). The input to the CNN portion of the CNN/MLP network are the real and imaginary components from the transfer functions, $\left\{\mathbf{H}^{\mathrm{UL}}, \mathbf{H}^{\mathrm{DL}}\right\}$ and represent two input channels of the CNN/MLP. The remaining details of the neural network are as follows. The CNN network consists of two 1D convolutional layers with associated pooling and dropout layers. Each of the 1D CNN layers has a kernel size

of 2 and a tanh activation function. A flattened layer provides the interface between the CNN and MLP portions of the CNN/MLP. The subsequent MLP layers are three identical, fully connected perceptron layers, each with 128 neurons and a ReLU activation function. The CNN/MLP network details are shown in Table III and closely follow those described in [20].

Layer	Params	Output Dim	Generic Dim
Input	0	2 (Re/Im)	$2F$
Dense	131,584	512	$4F$
Dense	32,832	64	$\frac{F}{2}$
Dense	33,280	512	$4F$
Dense	131,328	2 (Re/Im)	$2F$

TABLE II: Autoencoder architecture with trainable parameters, output, and generic dimensions.

Layer	Params	Output Dim	Generic Dim
Input	0	128×2 (Re/Im)	$F\times 2$
Conv1D	160	127×32	$(F-1)\times 32$
AvgPool	0	63×32	$\left(\frac{F-3}{2}+1\right)\times 32$
Conv1D	1,040	62×16	$\frac{F-3}{2}\times 16$
AvgPool	0	31×16	$\left[0.5\left(\frac{F-3}{2}-2\right)+1\right]\times 16$
Flatten	0	496	$\left(\frac{F-5}{4}+1\right)\cdot 16$
Dense	63,616	128	F
Dense	16,512	128	F
Dense	16,512	128	F
Dense	33,024	128×2 (Re/Im)	$F\times 2$

TABLE III: CNN/MLP architecture with trainable parameters, output, and generic dimensions.

C. Computational Complexity

We use the number of floating-point operations (FLOPS) as a measure of the computational complexity of the three models. Table IV presents the FLOPS for these models. We use the open-source PyTorch library to calculate the FLOPS for the two neural networks (NN). For KNN, we manually compute the number of operations required to determine the NMSE per training point.

We can use generic dimensions for the three models to further generalize these numbers. Let F represent the number of subcarriers at the input of the NN model, and let the number of spatial locations of $21,843$ be designated D (dataset size). Then, one can calculate the FLOPS for AE for the model in Table II to be (roughly) $40F^2$, which is $\mathcal{O}(F^2)$.

For the CNN/MLP shown in Table III, the complexity (roughly) equal to $16F^2 + 1264F^2 - 4356$, i.e., $\mathcal{O}(F^2)$.

For KNN, it is $6FD$, where $2F$ subtraction, $2F$ multiplication (to calculate the square), and $2F$ addition are needed; this has to be repeated D times. The KNN complexity then can be written as $\mathcal{O}(FD)$. For small D, the KNN has lower complexity, but high spatial data point density might be necessary to achieve good accuracy, causing KNN to scale poorly as D increases. While both AE and CNN/MLP are $\mathcal{O}(F^2)$, the proportionality constant for CNN/MLP scales is smaller.

We determine the training/inference time per spatial point for the three ML algorithms by timing the necessary training/inference calculations on a MSI Creator $Z16$ laptop with NVIDIA RTX 3060 GPU on our typical dataset size of $21,843$ for our nominal number of epochs of 750 and show them in Table V.

Model	Number of Parameters	FLOPs	General Complexity
AE	329k	655.4k	$\mathcal{O}(F^2)$
CNN	130.9k	427.7k	$\mathcal{O}(F^2)$
KNN	0	16,769k	$\mathcal{O}(FD)$

TABLE IV: Model complexity in terms of parameter count, floating-point operations (FLOPs), and theoretical runtime.

Time (s)	KNN	AE	MLP
Training	N/A	165.092	976.514
Inference	N/A	0.101×10^{-3}	0.117×10^{-3}
Total	5.6	165.093	976.515

TABLE V: Training and inference time for the three ML methods.

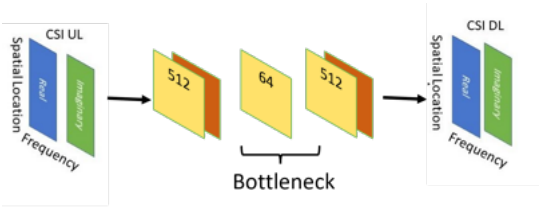
To further evaluate the three ML methods' performance, we add the NMSE performance to the computational complexity metrics of each ML method. Fig. 3(a) and Fig. 3(b) show the NMSE performance versus training and inference time, respectively, for our three ML methods. The NMSE shown in the figures is the mean NMSE across all the simulated values of B_{coh} and L_{coh} of each ML method. These plots show the trade-offs between NMSE, training requirements, and prediction speed, providing insight into implementing each ML method. Note that because the KNN method is non-supervised, the training and inference times are equal.

III. DATA-SET GENERATION

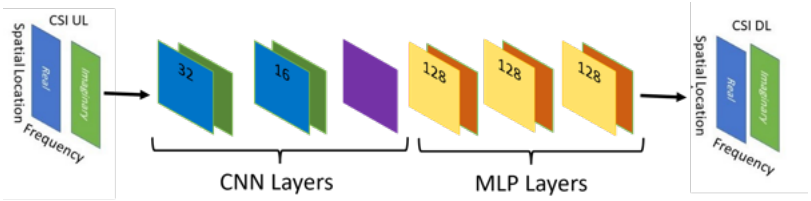
Our dataset comprises the wireless channel UL and DL transfer functions. These functions are complex-valued matrices $\boldsymbol{H}^{\text{UL}}$ and $\boldsymbol{H}^{\text{DL}}$ calculated over a specified spatial area at a specified sample density. We also synthesize these functions for a specified number of UL and DL subcarriers within a specific RF bandwidth of the wireless channel. The mathematical basis for these synthesized functions is the 3GPP model described by [31].[3] Because this model is statistical, it allows for defining multiple statistical instantiations within a specified wireless environment.

The model of [31] allows one to choose either a macrocell or microcell wireless environment. We choose the former,

[3]3GPP has developed multiple channel models over the years. We consciously selected this early (and thus simple) model, as its lower number of channel parameters allows us to more easily isolate specific physical effects.

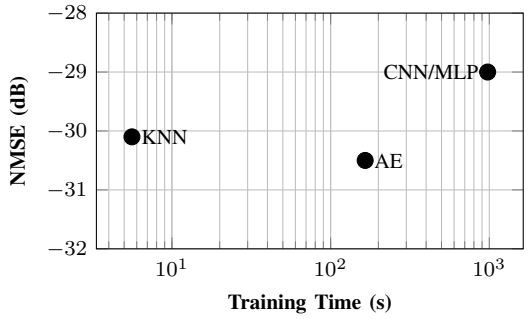


(a) Autoencoder Neural Network

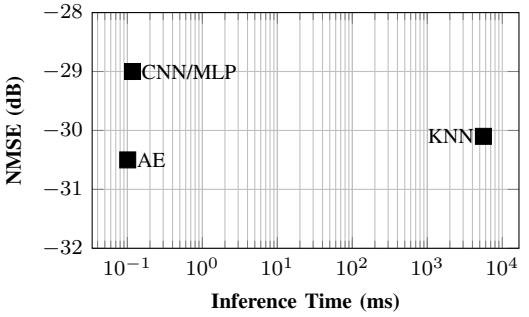


(b) Convolutional/Multilayer Perception Neural Network

FIGURE 2: Autoencoder and Convolutional/Multilayer Perception Neural Network configurations



(a) NMSE vs Training Time



(b) NMSE vs Inference Time

FIGURE 3: Comparison of NMSE against training and inference time for three ML methods.

Algorithm 1: The KNN Method Algorithm

Input : Uplink/Downlink channel transfer function:
$\left\{\mathbf{H}^{UL}_{\rho_k,\nu_l}, \mathbf{H}^{\mathrm{DL}}_{\rho_k,\nu_l}\right\}$,
$\rho_k = Spatial\ Index$; $\nu_l = Subcarrier\ Index$

Output: $\mathbf{NMSE}_{\nu_1}$

Initialization: $N(number\ of\ points) = N_{test} + N_{train}$
$N_{sc} = number\ of\ subcarriers$

for $\rho_k \leq N_{test}$ **do**

  for $\psi_j \neq \rho_k \leq N_{train}$ **do**

    $\epsilon_j = \sum_{\nu_l=1}^{N_{sc}} \left| |\mathbf{H}^{\mathrm{UL}}_{\rho_k,\nu_l}| - |\mathbf{H}^{\mathrm{UL}}_{\psi_j,\nu_l}| \right|^2$ $Magnitude$

    $\epsilon_j = \sum_{\nu_l=1}^{N_{sc}} \left| \mathbf{H}^{\mathrm{UL}}_{\rho_k,\nu_l} - \mathbf{H}^{\mathrm{UL}}_{\psi_j,\nu_l} \right|^2$ $Complex$

    $\psi_j = \psi_j + \psi_{j+1}$

    $\widehat{\psi}_j = arg\,min(\epsilon_j)$

  $\mathbf{MSE}^{\mathrm{DL}}_{\rho_k,\nu_l} = \left| |\mathbf{H}^{\mathrm{DL}}_{\rho_k,\nu_l}| - |\mathbf{H}^{\mathrm{DL}}_{\widehat{\psi}_j,\nu_l}| \right|^2$ $Magnitude$

  $\mathbf{MSE}^{\mathrm{DL}}_{\rho_k,\nu_l} = \left| \mathbf{H}^{\mathrm{DL}}_{\rho_k,\nu_l} - \mathbf{H}^{\mathrm{DL}}_{\widehat{\psi}_j,\nu_l} \right|^2$ $Complex$

  $\rho_k = \rho_{k+1}$

return $\mathbf{NMSE}_{\nu_1} = \dfrac{\mathbb{E}_{\rho_\mathbf{k}}[\mathbf{MSE}^{\mathrm{DL}}_{\rho_\mathbf{k},\nu_1}]}{\mathbb{E}_{\rho_\mathbf{k}} \left| \boldsymbol{H}^{\mathrm{DL}}_{\rho_\mathbf{k},\nu_1} \right|^2}$

which creates a set of six multipath clusters with paths (N_{path}) of distinct angles of arrival ($\delta_{n,\mathrm{AoA}}$), delays (τ_n), and path powers (P_n) within an overall power delay profile (PDP). Each of these paths is then decomposed into 20 sub-paths (N_{sub}), characterized by defined angular offsets with respect to the cluster centers. While we use this framework, we impose the following particular coherence bandwidths and coherence distances that are consistent with the model but are not specifically defined target values of the model as described in [31].

A. Creating Specific Wireless Environments

We first specify a desired $B_{\mathrm{coh}} \in [0.162\ \mathrm{MHz}, 16.2\ \mathrm{MHz}]$ and $L_{\mathrm{coh}} \in [0.5\ \lambda, 5.0\ \lambda]$. We then search for an associated set of statistical macrocell environmental parameters consis-

Algorithm 2: Environmental Parameters Algorithm

Input : Desired: $\overline{B}_{\rm coh}$ and $\overline{L}_{\rm coh}$; Iteration: Iter = $2,000$; Spatial Area = 5 m trajectory at 16 samples/λ

Output: Channel Parameters:$\{PDP, AoA, \tau\}$

1 *Initialization: σ_{DS} (5); σ_{AS} (6)*
2 **for** $i \leq Iter$ **do**
3 Calculate the delay τ path elements, τ_n, (7)
4 Calculate the PDP path elements P_n , (8)
5 Calculate the AoA path angles $\delta_{n,\rm AoA}$, (9)
6 Generate the complex-valued $\boldsymbol{H}^{\rm UL}(x,y,z)$ at each spatial area point for all subcarriers in the RF UL bandwidth (10)
7 Derive the frequency autocorrelation function $R_\nu(\tau)$
8 Derive the spatial autocorrelation function $R_\rho(\tau)$
9 Determine the $B_{\rm coh,i}$ and $L_{\rm coh,i}$ from $R_\nu(\tau)$ and $R_\rho(\tau)$
10 **if** $|\overline{B}_{\rm coh} - B_{\rm coh,i}| \leq .1\overline{B}_{\rm coh} \wedge |\overline{L}_{\rm coh} - L_{\rm coh,i}| \leq .1\overline{L}_{\rm coh}$ **then**
11 Return: Channel Parameters $\{P_n, \delta_{n,\rm AoA}, \tau_n\}$
12 $i = i + 1$

tent with the model structure of [31] (Algorithm 2: lines 1-12): path $\delta_{n,AoA}$, path P_n, and path τ_n which results in a channel that is within a specific accuracy (typically 90%) with our desired coherence bandwidth ($\overline{B}_{\rm coh}$) and desired coherence distance ($\overline{L}_{\rm coh}$). The $\boldsymbol{H}^{\rm UL}$ transfer function is then calculated as described in [31].

We verify our synthesized $\boldsymbol{H}^{\rm UL}$ transfer function solution by computing the frequency autocorrelation function ($R_\nu(\tau)$) and spatial autocorrelation function ($R_\rho(\tau)$) and ensuring that $R_\nu(\tau)$ and $R_\rho(\tau)$ are equal to 0.5 at the desired $\overline{B}_{\rm coh}$ and $\overline{L}_{\rm coh}$ (Algorithm 2: lines 7-9).
The details of the synthesis of $\boldsymbol{H}^{\rm UL}$ transfer function are outlined in the Algorithm 2 and the equations shown and referenced in Algorithm 2 are defined in Appendix A.

IV. SYSTEM STUDY AND RESULTS

This section describes the results of our simulations concerning the impact of the various system and channel parameters. To present results systematically, we will vary one parameter at a time and fix the other parameters to the default values given in Table VI, unless stated otherwise.

Each data point shown in Fig. 4 through Fig. 12 in this work is generated by taking the mean across fifteen environmental instantiations as described in Sec. III, and an NMSE error bar is generated at each data point, showing plus/minus one standard deviation (with respect to the ensemble of the fifteen instantiations). In addition, the NMSE value of each environmental instantiation is the mean across ten training simulations of the two neural networks, AE and CNN/MLP; this averages out the stochastic variations of these networks.

A. Impact of the Distance Metric

As we described in Sec. IIA, and shown in Table I, one may define the distance or error metric by assuming the transfer function at each subcarrier is represented as a complex value

TABLE VI: Nominal channel and system parameters

Channel Parameters	Value
UL carrier frequency	1.25 GHz
DL carrier frequency	1.275 GHz
RF Bandwidth	20 MHz
Spatial area	5 m by 1 m
Coherence bandwidth	0.162 MHz
Coherence distance	3.0 λ
Transmit/Receive antennas	Omni-directional
Number of paths (clusters)	6
Number of sub-paths	20
Number of subcarriers	128
UL SNR	50 dB
DL SNR	50 dB

or simply as a (real) magnitude value, depending on the application. While, in principle, any combination of distance metric and error metric is possible, it is generally desirable to be consistent in the choice of complex vs real magnitude between the distance metric used during the training and the error measure characterizing performance (note, however, that the distance metric during training is the MSE, while the performance evaluation is done in terms of the NMSE). Fig. 4 demonstrates the effect of the two distance/performance metric combinations for our three ML networks. Each subplot within Fig. 4 shows how both complex and magnitude NMSE varies, as the $L_{\rm coh}$ increases from 0.5 λ to 5.0 λ at a constant $B_{\rm coh}$ of 0.162 MHz for each of our ML networks.

The KNN network is the most sensitive to the choice of distance metric. This can be explained as follows: the algorithm does not actually know the location of the UE; it assesses the nearest neighbor purely based on the CSI. Still, generally speaking, frequency extrapolation works best if the "nearest neighbor" is at a close-by *physical* location to the true physical location. The KNN algorithm then determines the NMSE between the nearest neighbor DL and the ground truth DL. In the case of the magnitude distance metric (2), the NMSE is determined solely by the magnitude distance. However, in the case of the complex distance metric (1), the NMSE is determined by both the magnitude and phase distance between the two DL signals. Because the "nearest neighbor" generally is the closest physical location, the phase difference of the two DL signals is determined by the minimum distance between the two. Thus, the phase difference creates an additional error when the complex distance metric is used as compared to the magnitude distance metric. An additional consequence of the phase error created when the nearest neighbor increases in physical distance is the choice of how many neighbors (K) to consider. Due to the fact that the phase error increases and thus the NMSE as the physical distance increases between the two UL signals, we have chosen to keep K=1 when comparing the KNN performance to that of the CNN/MLP and AE neural networks.

We also see that the NMSE decreases as a function of the coherence length. This will be discussed in more detail in subsection C

In contrast, the AE and CNN/MLP networks treat the real and imaginary transfer function component $\{\mathbf{H}^{\mathrm{UL}}, \mathbf{H}^{\mathrm{DL}}\}$ as separate data points, as shown in Fig. 2. This allows these networks to train separately on the real and imaginary components, yielding a nearly constant 3 dB degradation in performance when using the complex error metric for performance evaluation

However, when comparing all three networks' NMSE performance trends, the magnitude distance and associated error metric demonstrate consistency across the three networks. Consequently, in the remainder of this paper, we use the $\mathbf{H}^{\mathrm{UL}}$ magnitude value when calculating the KNN nearest neighbor distance metric and the $\mathbf{H}^{\mathrm{DL}}$ magnitude value when calculating the MSE error for all NMSE calculations.

B. Training Environment: Dataset Spatial Area and Density

We next examine the effect of dataset sample density and dataset spatial area by simulating NMSE performance. The sample density is varied from 2 samples/λ to 16 samples/λ while for each of those sample densities, the spatial area is varied by changing the width of a rectangle of 5 m in length from 0 m to 3 m. Fig. 5 shows how the NMSE changes under these conditions for all three of our ML networks.

All three of our ML networks have the same qualitative behavior, i.e., performance improves in more or less equal steps as the sample density goes from 2 to 4, 8, and 16 samples per wavelength. This behavior can be expected intuitively for the KNN since the closer the nearest neighbor location is to that of the ground truth, the better the extrapolation will work. It is somewhat surprising for the AE and the CNN/MLP network since the coarsest sampling is the Nyquist rate.[4] Thus, from a classical signal processing point of view, no additional information is conveyed by the denser sampling rate. Given that neural networks can approximate any function, including interpolation functions, it is surprising that performance continues to improve with higher sample density. Of course, from a practical point of view, obtaining training data with such an extremely high sampling rate can be difficult or even prohibitive.

We next consider the impact of the rectangle's width in which training is performed (remember that the length is kept constant at 5 m). For the KNN algorithm, the width does not have a significant impact, which is expected based on the operating principle of the KNN algorithm: as the KNN algorithm is only concerned about the distance to its nearest neighbor at any specific spatial location, increasing the number of spatial locations does not improve the NMSE. For the AE and CNN/MLP algorithms, we firstly note that at the small dimension of the rectangle, the performance of AE and CNN/MLP is slightly worse than KNN, especially for low sampling density: for example, -15 dB (vs -20 dB for KNN) at 2 samples per wavelength, while for 16 samples per wavelength, performance is similar. Increasing the rectangle's width first does not change the NMSE, then leads to an improvement of around 0.125 m, and then saturates at around 0.5 m. This can be interpreted by the fact that increasing the rectangular width increases the number of available training samples, but only to a limited amount.

As a consequence of this simulation, we choose to perform all of our remaining simulations with a sample rate of 16 samples/λ and a rectangular spatial area of 5 m by 1 m.

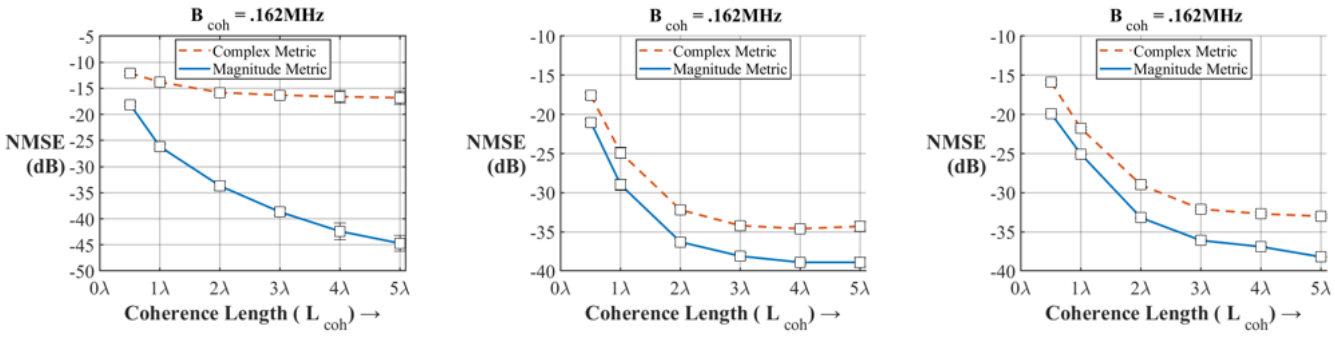
C. Channel Parameters: Coherence Distance and Coherence Bandwidth

We examine next how the NMSE performance for all three ML methods is affected by the environment's L_{coh} and B_{coh}. In this investigation, we increase the L_{coh} from 0.5 λ to 5 λ, while B_{coh} takes on 3 possible values, 0.162 MHz, 1.62 MHz, and 16.2 MHz.

These NMSE values for the three ML methods are plotted in Fig. 6. For all three ML methods, the NMSE performance improves as L_{coh} increases. The variations of the NMSE as a function of L_{coh} are greater for the KNN method than for AE and CNN/MLP networks; actually, the NMSE is larger (-15 vs -20 dB) for $L_{\mathrm{coh}} = 0.5$ λ and smaller (-45 vs -38 dB) for $L_{\mathrm{coh}} = 5$ λ. The reason for the higher NMSE at low L_{coh} is that the MPCs create distinct transfer functions in close proximity. At larger L_{coh}, due to the slower spatial variations of the transfer function, it is more likely (assuming constant sampling density) to find a training point that is "very close" (in terms of $\mathbf{H}^{\mathrm{UL}}$) to the test point. For the AE and CNN/MLP networks, the increased L_{coh} facilitates the training and makes the results less sensitive to small variations so that there is still some improvement.

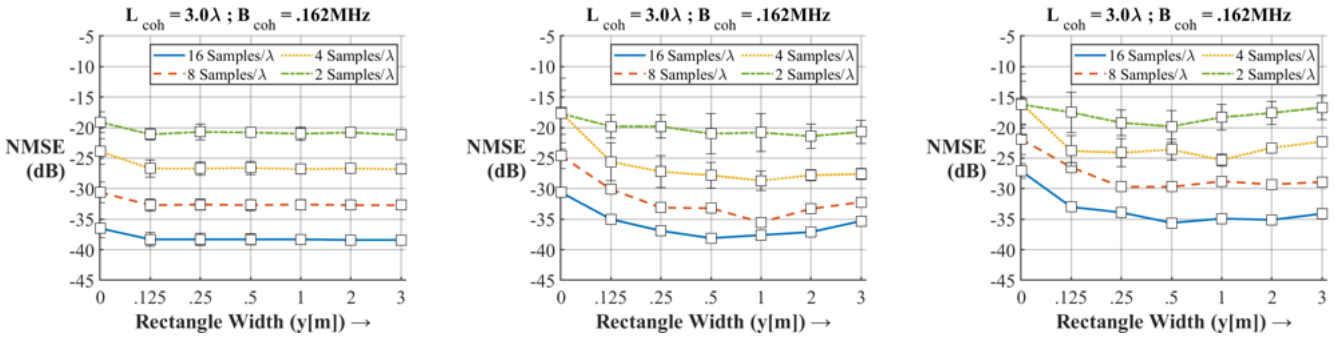
When varying the B_{coh}, Fig. 6 shows the NMSE performance to be similar for $B_{\mathrm{coh}} = 0.162$ MHz and 1.62 MHz, but exhibiting a drastic performance loss at 16.2 MHz, i.e., when approaching the RF bandwidth of 20 MHz for all three ML methods. As the B_{coh} increases to a value close to the RF bandwidth, the 128 subcarriers within that bandwidth become highly correlated. This effectively decreases the number of features (subcarriers) the three ML methods have available to improve their learning, which leads to higher NMSE. This is especially problematic for the KNN method due to its parameterless nature. In essence, the KNN can identify the UL channel only by a single number, the magnitude of the transfer function (which, due to the correlation, is essentially the same for all subcarriers); it is thus sometimes possible to find the "nearest neighbor", i.e., a training point with a similar UL magnitude, at a completely different location in the training area; of course, the associated DL channel then also takes on a completely

[4] actually, the Nyquist rate for the case that the MPCs are coming from all directions; with the limited angular spread corresponding to $L_{\mathrm{coh}} = 3\lambda$, even 2 samples per wavelength is higher than Nyquist.



(a) KNN NMSE (b) AE NMSE (c) CNN/MLP NMSE

FIGURE 4: NMSE performance comparing the complex distance metric with a magnitude distance metric



(a) KNN NMSE (b) AE NMSE (c) CNN/MLP NMSE

FIGURE 5: NMSE performance as the training sample density (samples/λ) increases as a function of spatial rectangle size

different value. To confirm this hypothesis, we extracted the cumulative distribution function of the distance error for different coherence bandwidths; as shown in Fig. 7, there is indeed a relatively high probability of very large distance errors.

A very large B_{coh} also causes the standard deviation across the 15 environmental instantiations to increase significantly for all ML methods.

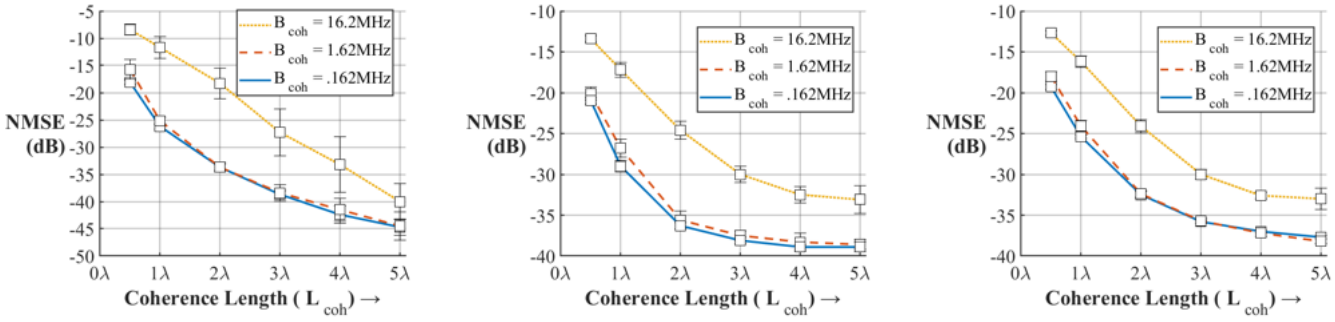
D. System Parameters: UL and DL Frequency Separation

Our first system parameter investigation consists of determining the NMSE while increasing the separation of the UL and DL center frequencies from our standard of 25 MHz up to 610 MHz.

It is worth remembering that for deterministic frequency extrapolation based on high-resolution parameter estimation, the NMSE increases as a function of this separation, saturating at very high values (around 0 dB) [8], [32]. This is because such a method extracts the parameters of the MPCs from the UL channel, computes the additional phase shift when approaching the DL frequency, and synthesizes the DL channel. Any error in the delay estimation in the UL creates a phase error of the MPC in the DL that scales with the frequency offset. As shown in Fig. 8, AE and CNN/MLP neural networks' frequency extrapolation performances behave very similarly to the deterministic frequency extrapolation. The NMSE increases as a function of this separation, saturating at high values (around -4 dB). This is due to the increasing difficulty of the neural network in extrapolating the DL parameters from the UL parameters as their frequency separation increases.

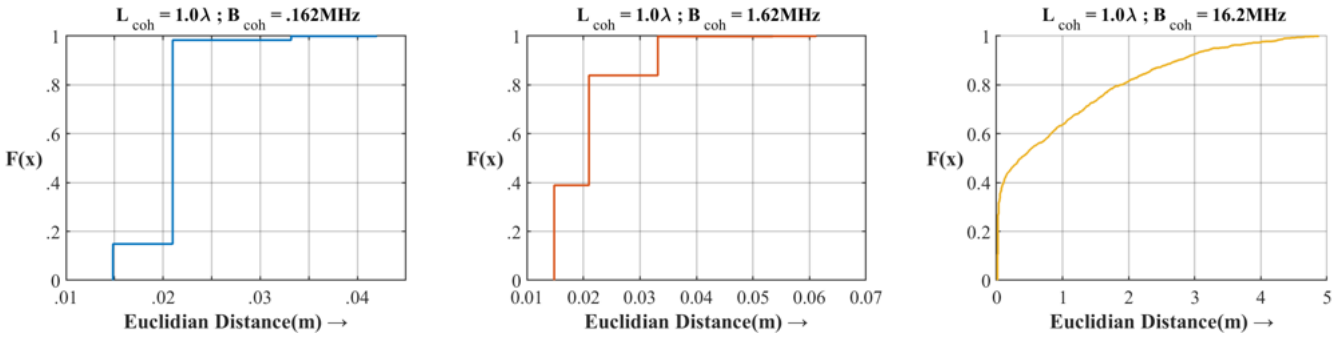
In contrast, we find that the KNN method shows an NMSE performance that is nearly identical and remains constant over the specified UL and DL separation when B_{coh} is much less than the RF bandwidth of 20 MHz. This behavior is intuitive since it simply performs a mapping between UL and DL channels based on a table of results at the training points and thus does not care about how the mapping is created, including the frequency separation.

The behavior of all the ML algorithms changes as the B_{coh} increases in value towards the RF bandwidth of 20 MHz. At $B_{\text{coh}} = 16.2$ MHz, the NMSE performance as a function of frequency separation remains similar in shape to that of the lower B_{coh} of 0.162 MHz and 1.62 MHz. However, the overall NMSE performance has degraded, and the standard deviation of the NMSE across the 15 environmental instantiations increases for all ML methods. The reason for this NMSE degradation and the increased standard deviation is the same as described in Sec. IVC.



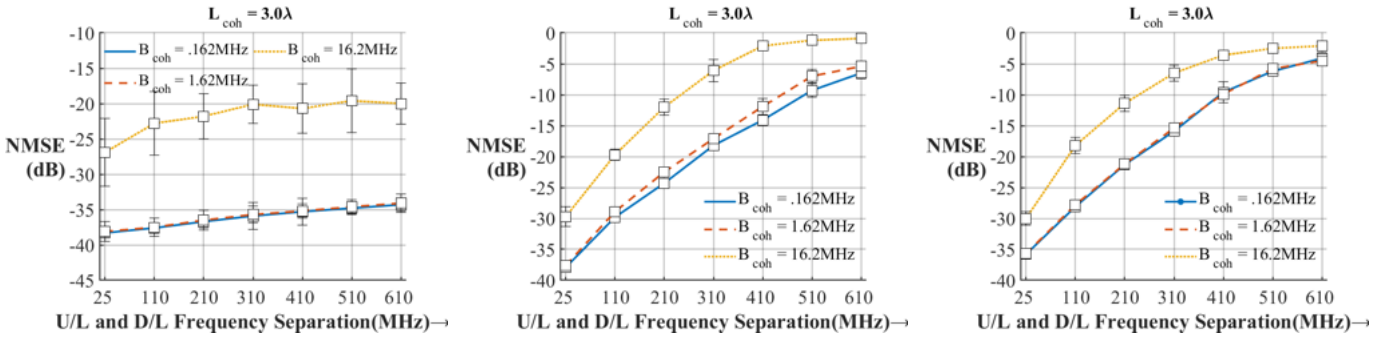
(a) KNN NMSE (b) AE NMSE (c) CNN/MLP NMSE

FIGURE 6: NMSE performance for multiple B_{coh} as a function of L_{coh}



(a) $B_{\text{coh}} = .162$ MHz (b) $B_{\text{coh}} = 1.62$ MHz (c) $B_{\text{coh}} = 16.2$ MHz

FIGURE 7: KNN cumulative distribution function for multiple B_{coh}



(a) KNN NMSE (b) AE NMSE (c) CNN/MLP NMSE

FIGURE 8: NMSE performance as a function of UL and DL frequency separation for each of the three ML methods

E. System Environment: Effect of RF Bandwidth and Subcarrier Spacing

We next explore the impact of increasing the RF bandwidth from 20 MHz to 160 MHz while maintaining the upper edge of the UL frequency band and the lower edge of the DL frequency band at a constant separation of 5 MHz. Due to this constraint, the separation between the *center frequencies* of the UL and DL increases with increasing RF bandwidth.

It is again easiest to interpret the results for KNN, as shown in Fig. 9(a). For most configurations, the NMSE is independent of the bandwidth since the KNN identifies the "pattern" of the UL transfer function as previously discussed: KNN matches it to a training transfer function (whose physical location will be close to that of the testing location) and identifies the corresponding DL channel. The NMSE is thus determined by the sampling density and the coherence length, both kept at the default values in this simulation. For the case of B_{coh} approaching the RF bandwidth, the NMSE is much larger due to the lack of a uniquely identifiable UL transfer function that can be associated with a DL transfer function, again as discussed in Sec. IVC. Increasing the RF bandwidth creates more frequency selectivity and, thus,

better identifiability. As seen in Fig. 9, an RF bandwidth of approximately 3 times the B_{coh} is sufficient to obtain an NMSE that is practically indistinguishable from that of even larger bandwidths.

Interestingly, the behavior of AE and CNN/MLP is fundamentally different. For smaller B_{coh}, the NMSE performance decreases as the RF bandwidth increases. This degradation is due to our imposed constraint of maintaining the upper edge of the UL frequency band and the lower edge of the DL frequency band at a constant separation of 5 MHz. Due to this constraint, the separation between the *center frequencies* of the UL and DL increases with increasing RF bandwidth. The amount of degradation we observe is the same amount of degradation we observed as we increase the frequency separation between the UL and DL, as shown in Fig. 8.

The above simulations varied the system bandwidth while keeping the number of subcarriers constant. To more closely investigate the impact of the subcarrier spacing, we increase the RF bandwidth and simultaneously vary the number of subcarriers from 64 to 128, 256, and 512. The subcarriers are spread uniformly across each RF bandwidth while maintaining the upper edge of the UL band and the lower edge of the DL frequency band constant, as in the previously described simulations. We perform this simulation with a constant B_{coh} of 0.162 MHz and L_{coh} at a constant of 3.0 λ.

As shown in Fig. 10, increasing the RF bandwidth, the NMSE increases for the AE and CNN/MLP neural networks, while the KNN method's NMSE remains constant due to the effects described above. In addition, all three ML methods show NMSE performance degradation when adding subcarriers to a given RF bandwidth; however, the amount of this additional error is only on the order of $1-2$ dB. We can thus conclude that the number of subcarriers and subcarrier spacing have very little impact on the overall NMSE.

F. System Environment: Number of MPCs

The $3GPP$ model described in Section III consists of 6 clusters (paths) with 20 MPCs (sub-paths) associated with each of the clusters. We examine the effect of decreasing the number of MPCs to 6 and increasing the number of MPCs per cluster to 80 for the L_{coh} of 1 λ, 3 λ, and 5 λ. Fig. 11 shows the results of this simulation for all three of our ML methods. As shown, the NMSE remains fairly constant as the number of MPCs varies from 6 to 80. The most significant NMSE variation occurs when the L_{coh} changes from 1 λ to 5 λ. This NMSE performance change is consistent with the NMSE change shown in Fig. 6. Both Fig. 6 and Fig. 11 show a NMSE increase of approximately 10 dB as the L_{coh} decreases from 1 λ to 5 λ.

G. System Environment: UL and DL Pilot Tone SNR

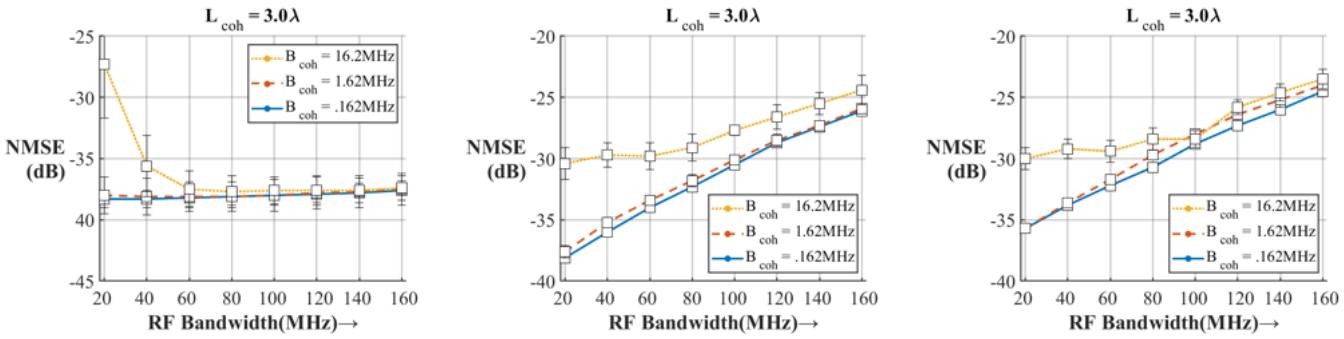
We examine the system effect of UL and DL pilot tones' SNR on NMSE performance. Since the UL SNR is determined by the wireless user terminal and the DL SNR is determined by the base station, their SNRs can differ. Furthermore, the noise realizations experienced by the respective receivers are independent. We, therefore, modify the channel model of Sec. III by adding an independent complex Gaussian noise component to the complex UL and DL transfer functions.

$$\boldsymbol{H}^{\text{DL}}_{\rho_k,\nu_l} = \sum_{n=1}^{N_{sub}} \boldsymbol{H}^{\text{DL}}_{\rho_k,\nu_l,n} + \boldsymbol{N}_{\rho_k,\nu_l} \tag{3}$$

$$\boldsymbol{H}^{\text{UL}}_{\rho_k,\nu_l} = \sum_{n=1}^{N_{sub}} \boldsymbol{H}^{\text{UL}}_{\rho_k,\nu_l,n} + \boldsymbol{N}_{\rho_k,\nu_l} \tag{4}$$

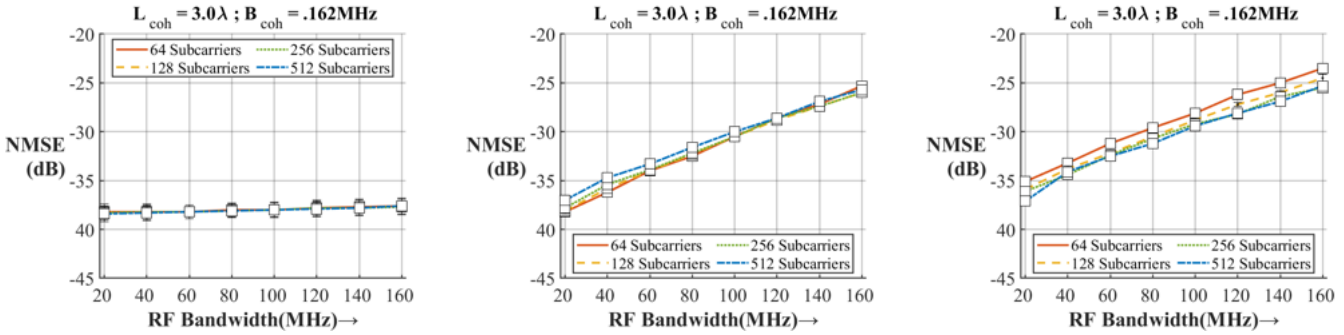
where: N_{sub} = number of sub-paths and, $\boldsymbol{N}_{\rho_k,\nu_l} \sim \mathcal{CN}(0,\sigma^2)$, with a variance σ^2 depending on the desired SNR. An independent $\boldsymbol{N}_{\rho_k,\nu_l}$ is added for each spatial position and subcarrier. It should be noted that since the DL SNR is being varied to values much below our default value of 50 dB, the normalization factor in our NMSE calculations (2) has been adjusted to be noise-free under these noisy signal conditions. As shown in Fig. 12, the UL and DL SNR affect all three ML methods' NMSE performance. In the KNN algorithm, due to its parameterless nature, the DL SNR dominates NMSE, which nearly becomes 0 dB at low DL SNR. In contrast, the AE and CNN/MLP neural networks' performance with varying UL and DL SNRs also varies as the DL SNR decreases, but not to the same extent. This is because the neural networks are being trained with UL and DL noisy signals, and the networks are adjusting their parameters to compensate for the noise. Additionally, the UL SNR has the effect of raising the NMSE as its SNR decreases for all three ML methods. This is explained by noting that the CSI extrapolation is performed with a noisy UL signal as input.

The NMSE results shown in Fig. 12 were calculated with the UL and DL SNR being equal during training and testing. This may not always be the case; the UL and DL SNRs during training and testing can differ. The NMSE performance with the DL SNR constant at 50 dB, and varying the UL training and testing SNR is shown in Table VII. Table VII shows that it is best to train and test with the same UL SNR rather than always training with the best possible UL SNR: for example, training with a 50 dB UL SNR and testing with a 10 dB UL SNR provides a *worse* NMSE (-21.8 dB) than training and testing with a 10 dB UL SNR (-25.7 dB). To mitigate this performance degradation, we consider training with a mixture of UL SNRs (i.e., train with 0 dB, 10 dB, and 50 dB UL SNR) and then testing at various UL SNRs. As shown in Table VII, such a mixture training strategy not only increases robustness but also performs better than training/testing at a specific UL SNR when the UL SNR is low (e.g., training with a mixture performs better than training with 0 dB UL SNR when testing with 0 dB UL SNR). However, note that training and testing with a high UL SNR (using the same UL SNR) still perform better than training with a mixture and testing with a high UL SNR.



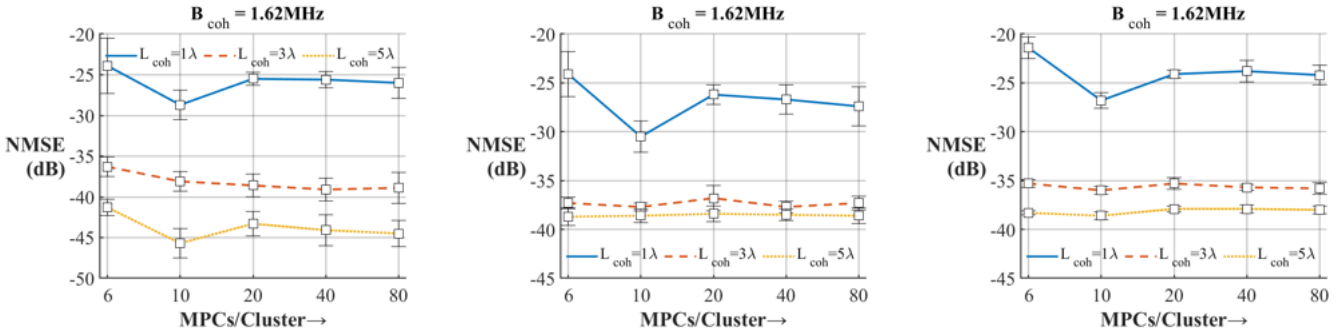
(a) KNN NMSE (b) AE NMSE (c) CNN/MLP NMSE

FIGURE 9: NMSE performance for the three ML methods as a function of RF bandwidth and B_{coh}



(a) KNN NMSE (b) AE NMSE (c) CNN/MLP NMSE

FIGURE 10: NMSE performance for the three ML methods as a function of RF bandwidth and number of subcarriers.



(a) KNN NMSE (b) AE NMSE (c) CNN/MLP NMSE

FIGURE 11: NMSE performance for the three ML methods as a function of number of MPCs.

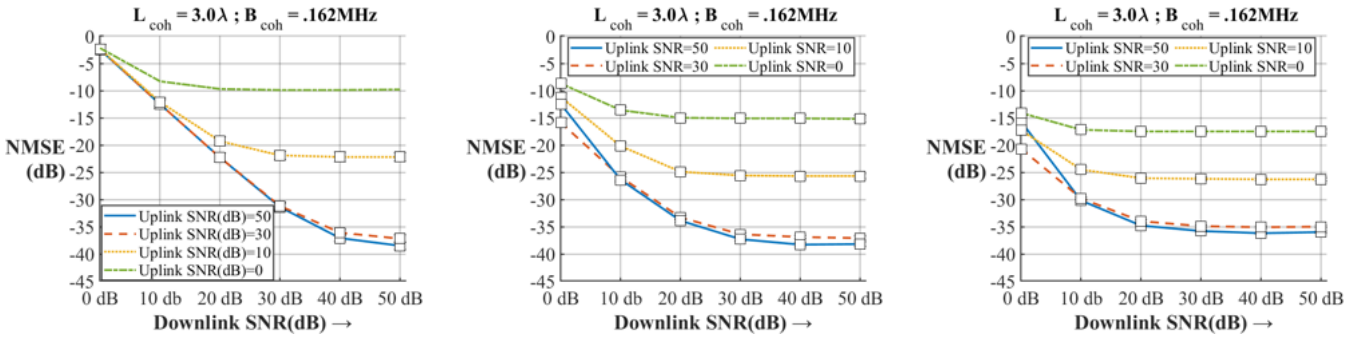
H. System Environment: Training and Evaluation Environment Mismatch

We also examine the NMSE performance when the training and evaluation wireless environments are different. Specifically, we perform simulations when the coherence bandwidth (B_{coh}) and coherence distance (L_{coh}) are mismatched. The simulation results show that the NMSE performance for all three of our ML methods is poor ($\approx -5.0\,\text{dB}$) when either the B_{coh} or L_{coh} is mismatched between the training and evaluation environments.

V. CONCLUSION

In this paper, we have shown how wireless parameters affect the NMSE when three ML methods perform frequency extrapolation in an FDD system. A summary of our conclusions is as follows:

- The AE and CNN/MLP neural networks give similar NMSE. However, the KNN methodology is fundamentally different.
- The difference in performance between the AE and CNN/MLP neural networks depends on their specific



(a) KNN NMSE (b) AE NMSE (c) CNN/MLP NMSE

FIGURE 12: NMSE performance for the three ML methods as a function of UL and DL SNR.

Train Uplink (dB)	Test Uplink (dB)	NMSE (dB)
0	0	-15.1
0	10	-18.1
0	50	-17.8
10	0	-12.4
10	10	-25.7
10	50	-29.8
50	0	-11.4
50	10	-21.8
50	50	-38.1
Mixture	0	-20.6
Mixture	10	-28.2
Mixture	50	-33.3

TABLE VII: NMSE values under varying UL SNRs during training and testing (with fixed DL SNR at 50 dB).

hyperparameter settings. For the KNN to have acceptable NMSE performance, we typically need a large quantity of training samples.

- Having an RF bandwidth comparable to the B_{coh} leads to generally bad performance as well as strong variations between different instantiations in the same environment.
- Large L_{coh} typically reduces the NMSE, but a minimum number of statistically independent training points is required.
- NMSE increases as a function of UL and DL frequency separation, similar to that of deterministic frequency extrapolation based on high-resolution parameter estimation. The number of subcarriers and their spacing have little impact on results.
- Simulations comparing different ML structures need to take these dependencies on parameters into account. Comparisons with values in the literature must ensure that all these parameters are the same as the ones we want to compare against.

APPENDIX

A. Equations Used in Algorithm 2

Approximation for the standard deviation of the delay spread [1, Chap. 6];

$$\sigma_{\text{DS}} \approx \frac{1}{2\pi B_{\text{coh}}} \tag{5}$$

Approximation for the standard deviation of the angular spread;

$$\sigma_{\text{AS}} \approx \frac{(.13)(360)}{L_{\text{coh}}} \tag{6}$$

Calculation of the τ_n;

$$\tau_n' = -r_{\text{DS}}\sigma_{\text{DS}} \ln z_n, \quad n = 1, \ldots, N_{\text{path}} \tag{7}$$

where r_{DS} is the proportionality constant defined in [31]. and z_n $(n = 1, \ldots, N_{\text{path}})$ are independent identically distributed (i.i.d) random variables with uniform distribution $U[0, 1)$.
The τ_n' are then sorted from high to low value, and their minimum is subtracted from all such that $\tau_{N_{path}} > \ldots > \tau_1 = 0$
Calculation of the power delay profile elements P_n

$$\begin{aligned} P_n' &= e^{\frac{(1-r_{\text{DS}})\tau_n}{r_{\text{DS}}\sigma_{\text{DS}}}} \cdot 10^{-0.1\xi_n}, \quad n = 1, \ldots, N_{\text{path}} \\ P_n &= \frac{P_n'}{\sum_{j=1}^{N_{\text{path}}} P_j'} \end{aligned} \tag{8}$$

where ξ_n $(n = 1, \ldots, N_{\text{path}})$ are i.i.d Gaussian random variables with 0 mean and standard deviation $\sigma_{\text{RND}} = 3$ dB.
Calculation of the path $\delta_{n,\text{AoA}}$ elements

$$\begin{aligned} \delta_{n,\text{AoA}} &\sim \mathcal{N}(0, \sigma_{n,\text{AoA}}^2) \quad n = 1, \ldots, N_{\text{path}} \\ \sigma_{n,\text{AoA}} &= 104.12^\circ \cdot (1 - \exp(0.2175 \cdot \text{P}_{n,\text{dBr}})) \end{aligned} \tag{9}$$

where $P_{n,\text{dBr}} < 0$ is the relative power of the nth path in dBr with respect to the total power.
Generation of the complex $\boldsymbol{H}^{\text{UL}}$ transfer function for path n, at a spatial point (x, y, z), and at the UL center frequency

(f_{UL}).

$$\begin{aligned} \boldsymbol{H}_n^{\mathrm{UL}}(x,y,z) &= \sqrt{\frac{P_n}{N_{\mathrm{sub}}}} \cdot \sum_{i=1}^{N_{\mathrm{sub}}} e^{j\vec{k}\cdot\vec{r}} e^{j\Phi_{n,i}} e^{j\phi_n}, \\ \boldsymbol{H}^{\mathrm{UL}}(x,y,z) &= \sum_{n=1}^{N_{\mathrm{path}}} \boldsymbol{H}_n^{\mathrm{UL}}(x,y,z) \end{aligned} \tag{10}$$

where:

$\vec{\boldsymbol{k}} = k_0(\cos(\theta_{n,i,\mathrm{AoA}})\hat{x} + \sin(\theta_{n,i,\mathrm{AoA}})\hat{y})$

$k_0 = 2\pi/\lambda, \lambda$ calculated at UL frequency

$\theta_{n,i,\mathrm{AoA}} = \zeta \cdot (\delta_{n,\mathrm{AoA}} + \Delta_{n,i,\mathrm{AoA}})$

$\Delta_{n,i,\mathrm{AoA}} =$ Table II AoA Offset at UE; [31]

$\zeta = \frac{\sigma_{\mathrm{AS}}}{68^\circ}$; Table II adjustment for our specific σ_{AS} [31]

$\Phi_{n,i} = N_{\mathrm{path}}$ by $\mathrm{N_{sub}}$ matrix with elements $\sim \mathrm{U}[0, 2\pi)$

$\phi_n = 2\pi \cdot f_{\mathrm{UL}} \cdot \tau_n$

ACKNOWLEDGMENT

The computational facilities made available by the USC Center of Advanced Research Computing and its staff are gratefully acknowledged.

REFERENCES

[1] A. F. Molisch, *Wireless communications: from fundamentals to beyond 5G*. John Wiley & Sons, 2022.

[2] H. Tataria, M. Shafi, A. F. Molisch, M. Dohler, H. Sjöland, and F. Tufvesson, "6g wireless systems: Vision, requirements, challenges, insights, and opportunities," *Proceedings of the IEEE*, vol. 109, no. 7, pp. 1166–1199, 2021.

[3] P. W. Chan, E. S. Lo, R. R. Wang, E. K. Au, V. K. Lau, R. S. Cheng, W. H. Mow, R. D. Murch, and K. B. Letaief, "The evolution path of 4g networks: Fdd or tdd?," *IEEE Communications Magazine*, vol. 44, no. 12, pp. 42–50, 2006.

[4] E. Dahlman, S. Parkvall, and J. Skold, *5G NR: The next generation wireless access technology*. Academic Press, 2020.

[5] X. Rao, V. K. Lau, and X. Kong, "Csit estimation and feedback for fdd multi-user massive mimo systems," in *2014 IEEE International conference on acoustics, speech and signal processing (ICASSP)*, pp. 3157–3161, IEEE, 2014.

[6] U. Ugurlu, R. Wichman, C. B. Ribeiro, and C. Wijting, "Multipath phase indication (mpi) feedback for csi acquisition in fdd massive mimo," in *2015 IEEE 26th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pp. 569–574, IEEE, 2015.

[7] M. D. Larsen, A. L. Swindlehurst, and T. Svantesson, "Performance bounds for mimo-ofdm channel estimation," *IEEE Transactions on Signal Processing*, vol. 57, no. 5, pp. 1901–1916, 2009.

[8] F. Rottenberg, T. Choi, P. Luo, C. J. Zhang, and A. F. Molisch, "Performance analysis of channel extrapolation in fdd massive mimo systems," *IEEE Transactions on Wireless Communications*, vol. 19, no. 4, pp. 2728–2741, 2020.

[9] A. Richter, "Estimation of radio channel parameters: Models and algorithms," ISLE Blacksburg, VA, USA, 2005.

[10] M. Barzegar Khalilsarai, S. Haghighatshoar, X. Yi, and G. Caire, "Fdd massive mimo via ul/dl channel covariance extrapolation and active channel sparsification," *IEEE Transactions on Wireless Communications*, vol. 18, no. 1, pp. 121–135, 2019.

[11] L. Miretti, R. L. Cavalcante, and S. Stanczak, "Fdd massive mimo channel spatial covariance conversion using projection methods," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3609–3613, 2018.

[12] L. Miretti, R. L. Cavalcante, and S. Stańczak, "Downlink channel spatial covariance estimation in realistic fdd massive mimo systems," in *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pp. 161–165, 2018.

[13] M. Liang and A. Li, "Deep learning-based channel extrapolation for pattern reconfigurable massive mimo," *IEEE Transactions on Vehicular Technology*, 2023.

[14] M. Alrabeiah and A. Alkhateeb, "Deep learning for tdd and fdd massive mimo: Mapping channels in space and frequency," in *2019 53rd asilomar conference on signals, systems, and computers*, pp. 1465–1470, IEEE, 2019.

[15] K. Vuckovic, M. B. Mashhadi, F. Hejazi, N. Rahnavard, and A. Alkhateeb, "Paramount: Toward generalizable deep learning for mmwave beam selection using sub-6 ghz channel measurements," *IEEE Transactions on Wireless Communications*, vol. 23, no. 5, pp. 5187–5202, 2024.

[16] M. Arnold, S. Dörner, S. Cammerer, J. Hoydis, and S. ten Brink, "Towards practical fdd massive mimo: Csi extrapolation driven by deep learning and actual channel measurements," in *2019 53rd Asilomar Conference on Signals, Systems, and Computers*, pp. 1972–1976, IEEE, 2019.

[17] 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, 2020.

[18] Y. Zhang, J. Wang, J. Sun, B. Adebisi, H. Gacanin, G. Gui, and F. Adachi, "Cv-3dcnn: Complex-valued deep learning for csi prediction in fdd massive mimo systems," *IEEE Wireless Communications Letters*, vol. 10, no. 2, pp. 266–270, 2020.

[19] Y. Yang, F. Gao, G. Y. Li, and M. Jian, "Deep learning-based downlink channel prediction for fdd massive mimo system," *IEEE Communications Letters*, vol. 23, no. 11, pp. 1994–1998, 2019.

[20] M. Arnold, S. Dörner, S. Cammerer, S. Yan, J. Hoydis, and S. t. Brink, "Enabling fdd massive mimo through deep learning-based channel prediction," *arXiv preprint arXiv:1901.03664*, 2019.

[21] L. Bai, C.-X. Wang, J. Huang, Q. Xu, Y. Yang, G. Goussetis, J. Sun, and W. Zhang, "Predicting wireless mmwave massive mimo channel characteristics using machine learning algorithms," *Wireless Communications and Mobile Computing*, vol. 2018, no. 1, p. 9783863, 2018.

[22] Y. Zhang, A. Liu, P. Li, and S. Jiang, "Deep learning (dl)-based channel prediction and hybrid beamforming for leo satellite massive mimo system," *IEEE Internet of Things Journal*, vol. 9, no. 23, pp. 23705–23715, 2022.

[23] C.-K. Wen, W.-T. Shih, and S. Jin, "Deep learning for massive mimo csi feedback," *IEEE Wireless Communications Letters*, vol. 7, no. 5, pp. 748–751, 2018.

[24] S. Lee, H. Kim, and D. Lee, "Linearization autoencoder: an autoencoder-based regression model with latent space linearization," *bioRxiv*, pp. 2022–06, 2022.

[25] Z. Liu, G. Singh, C. Xu, and D. Vasisht, "Fire: enabling reciprocity for fdd mimo systems," in *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*, MobiCom '21, (New York, NY, USA), p. 628–641, Association for Computing Machinery, 2021.

[26] J. P. Lemayian and J. M. Hamamreh, "Massive mimo channel prediction using recurrent neural networks," *RS Open Journal on Innovative Communication Technologies*, vol. 1, no. 1, 2020.

[27] Z. Tao and S. Wang, "Improved downlink rates for fdd massive mimo systems through bayesian neural networks-based channel prediction," *IEEE Transactions on Wireless Communications*, vol. 21, no. 3, pp. 2122–2134, 2021.

[28] J. Mirzaei, S. S. Panahi, R. S. Adve, and N. K. M. Gopal, "Deep generative models for downlink channel estimation in fdd massive mimo systems," *IEEE Transactions on Signal Processing*, vol. 70, pp. 2000–2014, 2022.

[29] B. Banerjee, R. C. Elliott, W. A. Krzymień, and H. Farmanbar, "Downlink channel estimation for fdd massive mimo using conditional generative adversarial networks," *IEEE Transactions on Wireless Communications*, vol. 22, no. 1, pp. 122–137, 2022.

[30] A. S. Doshi, M. Gupta, and J. G. Andrews, "Over-the-air design of gan training for mmwave mimo channel estimation," *IEEE Journal on Selected Areas in Information Theory*, vol. 3, no. 3, pp. 557–573, 2022.

[31] G. Calcev, D. Chizhik, B. Goransson, S. Howard, H. Huang, A. Kogiantis, A. F. Molisch, A. L. Moustakas, D. Reed, and H. Xu, "A wideband spatial channel model for system-wide simulations," *IEEE*

Transactions on Vehicular Technology, vol. 56, no. 2, pp. 389–403, 2007.
[32] H. Choi and J. Choi, "Downlink extrapolation for fdd multiple antenna systems through neural network using extracted uplink path gains," *IEEE Access*, vol. 8, pp. 67100–67111, 2020.

Michael E. Neuman (Member, IEEE) received the B.S. degree in electrical engineering from Ohio University, Athens, OH, USA, in 1972. He received two M.S. degrees in electrical engineering, one from the University of Michigan, Ann Arbor, MI, USA, in 1973, and another from the University of Southern California (USC), Los Angeles, CA, USA, in 2004. He also received an M.S. degree in financial engineering from USC in 2019.

From 1973 to 2018, he held several engineering positions, including Chief Engineer of satellite systems at the Boeing (formerly Hughes Aircraft) Space and Communications Division, El Segundo, CA, USA. He is currently pursuing the Ph.D. degree in electrical engineering at the University of Southern California.

His research interests are focused on the integration of machine learning into wireless communications.

Daoud Burghal received the B.S. degree in electrical engineering from The University of Jordan, Amman, Jordan, in 2007, and the M.S. and Ph.D. degrees in electrical engineering and statistics from the University of Southern California, Los Angeles, CA, USA, in 2012.

Following his Ph.D., he was a Postdoctoral Scholar with the WiDeS Laboratory. He later joined Qualcomm as a Wireless Research and Development System Engineer. In 2022, he joined Samsung Research America as a Staff AI Research Engineer.

His research interests include wireless communications, AI-assisted communication, and intelligent autonomous systems.

Andreas F. Molisch (Fellow, IEEE) received the Dipl.Ing. degree in 1990, the Ph.D. degree in 1994, and the Habilitation in 1999, all from the Technical University of Vienna, Austria. He spent ten years in industry at FTW, AT&T (Bell) Laboratories, and Mitsubishi Electric Research Labs, where he rose to Chief Wireless Standards Architect. In 2009, he joined the University of Southern California (USC), Los Angeles, CA, USA, as a Professor and founded the Wireless Devices and Systems (WiDeS) group. In 2017, he was appointed to the Solomon Golomb – Andrew and Erna Viterbi Chair.

His research interests include wireless propagation channels, wireless system design, hybrid beamforming, UWB/TOA localization, and joint communication–computation–caching. He has published five books, over 320 journal papers, and more than 400 conference papers. He holds 80 patents and has co-authored over 70 standards contributions. His work has been cited more than 77,000 times (h-index 116), and he is a Clarivate Highly Cited Researcher.

Dr. Molisch is a Fellow of the National Academy of Inventors, AAAS, IEEE, and IET, and a member of the Austrian Academy of Sciences. He has served as Editor and Chair of numerous IEEE journals and conferences. His awards include the IET Achievement Medal, IEEE VTS Avant-Garde Award, IEEE ComSoc Armstrong Award, and the IEEE Eric Sumner Technical Field Award.