Neural CSI Compression Fine-Tuning: Taming the Communication Cost of Model Updates

Mehdi Sattari, *Graduate Student Member, IEEE,* Deniz Gündüz, *Fellow, IEEE,* and Tommy Svensson, *Senior Member, IEEE*

***Abstract*—Efficient channel state information (CSI) compression is essential in frequency division duplexing (FDD) massive multiple-input multiple-output (MIMO) systems due to the substantial feedback overhead. Recently, deep learning-based compression techniques have demonstrated superior performance for CSI feedback. However, their performance often degrades under distribution shifts across wireless environments, largely due to limited generalization capability. To address this challenge, we consider a full-model fine-tuning scheme, in which both the encoder and decoder are jointly updated using a small number of recent CSI samples from the target environment. A key challenge in this setting is the transmission of updated decoder parameters to the receiver, which introduces additional communication overhead. To mitigate this bottleneck, we explicitly incorporate the bit rate of model updates into the fine-tuning objective and entropy-code the model updates jointly with the compressed CSI. Furthermore, we employ a structured prior that promotes sparse and selective parameter updates, thereby significantly reducing the model-update communication cost. Simulation results across multiple CSI datasets demonstrate that full-model fine-tuning substantially improves the rate–distortion performance of neural CSI compression, despite the additional cost of model updates. We further analyze the impact of the evaluation horizon, the quantization resolution of model updates, and the size of the target-domain dataset on the overall feedback efficiency.**

***Index Terms*—CSI compression, massive MIMO, deep learning, fine-tuning.**

I. Introduction

MASSIVE multiple-input multiple-output (MIMO) is one of the key enabling technologies for 5G and beyond. By equipping the base station (BS) with massive antenna arrays, a large number of users can be served simultaneously through high-resolution beamforming techniques, resulting in remarkable spectral efficiency. To enable massive MIMO technology, channel state information (CSI) must be available at both the BS and user sides. Various channel estimation techniques can be employed to estimate CSI by observing pilot signals. In time-division duplexing (TDD), only downlink/uplink channels need to be estimated, as the other channel can be derived thanks to reciprocity. However, in frequency-division duplexing (FDD), reciprocity does not hold, so the channels in both directions need to be estimated and fed back [1], [2].

To enable FDD transmission for massive MIMO systems, CSI compression techniques are critical to circumvent excessive feedback overhead. Various CSI compression techniques have been studied in the literature by employing compressed sensing [3], [4], vector quantization [5], [6], and more recently, deep learning tools [7]–[15]. Due to strong spatial correlation, the CSI matrix can exhibit sparsity in certain domains, and compressed sensing methodologies can be applied to efficiently compress the large CSI matrix [3], [4]. However, these algorithms often struggle to find the best basis to project the CSI matrix to lower dimensions. Furthermore, compressed sensing-based algorithms are iterative and time-consuming, making them infeasible for deployment in massive MIMO channels. Similarly, in vector quantization for CSI compression [5], [6], the overhead scales linearly with the channel dimension, rendering it impractical for massive MIMO systems.

Recent advances in data-driven compression approaches, driven by the progress in neural network (NN) architectures have gained substantial interest in all data modalities [16]. Neural data compression is a machine learning technique that performs the compression task using deep neural networks (DNNs). Current research in neural compression is largely driven by the development of deep generative models such as generative adversarial networks (GANs) [17], variational autoencoders (VAEs) [18], normalizing flows [19], and autoregressive models [20] for image and video compression. These models have proven effective in capturing complex data distributions, which is crucial for achieving high compression rates while maintaining data fidelity.

In the wireless communication domain, the first neural CSI compressor was introduced in [7], where the authors used a simple convolutional neural network (CNN) autoencoder architecture for dimension reduction of a massive MIMO channel matrix. Later, numerous network architectures and machine learning techniques have been proposed to further improve the CSI compression efficiency [8]–[11]. While most works formulated the problem in terms of the dimensions of the reduced CSI matrix, the feedback channels

M. Sattari, and T. Svensson are with the Department of Electrical Engineering, Chalmers University of Technology, Gothenburg, Sweden. E-mail: {mehdi.sattari, tommy.svensson}@chalmers.se. D. Gündüz is with the Department of Electrical and Electronic Engineering, Imperial College London, London, UK. E-mail: d.gunduz@imperial.ac.uk.

This work has been supported in part by the project SEMANTIC, funded by the EU's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 861165, and in part by the Hexa-X-II project which has received funding from the Smart Networks and Services Joint Undertaking (SNS JU) under the European Union's Horizon Europe research and innovation programme under Grant Agreement No 101095759. D. Gündüz received funding from the UKRI through ERC consolidator project AI-R under grant EP/X030806/1, and from the Horizon Europe SNS project '6G-GOALS' under grant 101139232. The computations were enabled by resources provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS), partially funded by the Swedish Research Council through grant agreement no. 2022-06725.

must ultimately convey the CSI matrix in bits, necessitating additional quantization in the latent space. In [14] and [21], the authors formulated the problem within the rate-distortion (RD) framework and further enhanced the performance of the neural CSI compressor by incorporating learned entropy encoding and decoding blocks. They employed a weighted RD loss function to jointly minimize the mean squared error (MSE) and the average bit-rate overhead, explicitly quantifying the communication cost of compression in bits per CSI dimension under a variable-rate compression scheme.

Machine learning models often perform remarkably well on test sets with a distribution similar to their training data but can fail catastrophically in deployment when the data distribution shifts. Such shifts can arise from various factors, including temporal changes, domain variations, or sampling biases, and pose significant challenges in many real-world applications [22]. Several approaches, such as transfer learning [23], data augmentation [24], and domain adaptation [25], can be employed to enhance the robustness of machine learning models against distribution shift and improve their generalization capabilities. In transfer learning, models trained on generic datasets are fine-tuned for the target domain. Data augmentation improves generalization by artificially generating diverse training samples. Domain adaptation, on the other hand, aligns source and target domain distributions through techniques such as instance re-weighting or feature alignment.

Distribution shift is a common phenomenon in wireless networks. The statistics of the underlying wireless channel change due to various factors such as user mobility, environmental changes, or variations in interference from other devices. When the channel distribution deviates from initial assumptions or models, the efficiency of algorithms for tasks such as channel estimation [26], signal detection [27], etc can degrade. Consequently, maintaining robust communication requires adaptive strategies that can adjust to these shifts. In the context of CSI compression for FDD massive MIMO systems, most models in the literature are trained and tested using data from the same environment, e.g., within a specific macro-cell coverage area. However, this would require users to either store or download [28] new models for every new cell they enter, making these approaches infeasible in practice. Therefore, it is crucial to develop techniques that can overcome the distribution shift problem in CSI compression.

Several works have studied the effect of distribution shift in the CSI compression problem [29]–[35]. In [29], a deep transfer learning method is used to handle the distribution shift and a model-agnostic meta-learning algorithm is proposed to reduce the number of CSI samples. A lightweight translation model and data augmentation method based on domain knowledge are introduced in [30]. Specifically, to adapt to new channel conditions, the authors propose an efficient scenario-adaptive CSI feedback architecture, "CSI-TransNet," and an efficient deep unfolding-based CSI compression network, "SPTM2-ISTANet+." A single-encoder-to-multiple-decoders (S-to-M) approach is presented in [31], where the authors use multi-task learning to integrate multiple independent autoencoders into a unified architecture featuring a shared encoder and several task-specific decoders.

The authors in [32] proposed online learning schemes based on a vanilla autoencoder architecture, exploring various adaptation strategies. In [33], elastic weight consolidation is incorporated to alleviate catastrophic forgetting in NNs. A federated edge learning (FEEL)-based training framework for massive MIMO CSI feedback is introduced in [34] to address privacy concerns arising from transmitting raw CSI datasets during centralized training, and a personalization strategy is further developed to enhance feedback performance. In [35], a gossip learning strategy is adopted, and two data augmentation techniques—random erasing and random phase shift—are employed to improve generalization capability.

However, we highlight that the communication overhead associated with model updates is neglected in all the aforementioned prior works. This represents a significant gap in the literature, as typical DNNs can contain millions of parameters, and transmitting model updates can result in massive feedback overhead. In [34], low-resolution quantization (e.g., 2-bit) is applied to the model updates; however, the resulting update bit-rate is not accounted for in the evaluation. To elaborate, the decoder network in [34] includes approximately 4 million trainable parameters. Even with an aggressive 2-bit quantization, the resulting communication overhead would be around 8 Mbits, substantially higher than that is required to transmit the compressed CSI matrix. More precisely, a compressed CSI matrix of size 64×64, using a modest compression ratio of 16, would only require 4 kbits and 16 kbits for 2-bit and 32-bit quantization, respectively. Comparing the communication cost for compressed CSI and model updates illustrates that transmitting model updates can introduce a prohibitive feedback overhead for FDD massive MIMO systems.

To bridge this gap, in this paper, we explicitly account for the communication overhead of model update transmission. We investigate how to efficiently transmit both the model updates and the latent compressed CSI to obtain the best overall RD trade-off. The key differences and contributions of this work are summarized as follows:

- Prior works addressing distribution shifts in neural CSI compression primarily focus on vanilla autoencoder-based schemes and overlook bit-level CSI compression as well as the communication cost of transmitting model updates to the decoder [29]–[35]. In contrast, by incorporating lossless entropy coding for both the compressed CSI latents and the model updates, we explicitly quantify the RD trade-off in neural CSI compression while accounting for the total communication cost, including both CSI feedback and model-update signaling.
- To enable efficient entropy coding of model updates, we model the update distribution using a spike-and-slab prior, formulated as a mixture of narrow (spike) and wide (slab) Gaussian components. By assigning higher probability mass to the spike component, the proposed prior promotes sparse and selective parameter updates, allowing only the most impactful weights to be modified. In addition, we incorporate the bit rate of model updates as a regularization term in the RD loss. Simulation results demonstrate that the combined use of spike-and-slab modeling and update-rate regularization is essential for

significantly reducing model-update communication cost while improving overall RD performance.
- We evaluate the proposed framework across diverse CSI datasets, including QuaDriGa [36], 3GPP clustered delay line (CDL) [37], and DeepMIMO [38], covering both site-specific and stochastic channel models and a wide range of wireless environments. We systematically investigate the impact of the amortized model-update rate over the evaluation horizon, as well as the effects of target-domain dataset size and model-update quantization. Our results reveal a fundamental trade-off between the amortized model-update rate and distortion in dynamic wireless environments, and demonstrate that robust RD performance can be achieved using a few hundred CSI samples, with finer model-update quantization further improving performance.

The remainder of this paper is organized as follows. Section II introduces the backbone neural CSI compressor for an FDD massive MIMO system and evaluates its performance in the absence of distribution shifts. Section III describes the considered CSI datasets and the proposed neural model fine-tuning framework. Section IV presents numerical results in terms of the RD trade-off, along with a complexity analysis. Finally, Section V concludes the paper.

Throughout this paper, the following notation is adopted. Bold uppercase letters denote matrices, and bold lowercase letters denote vectors. The transpose operation is represented by the superscript $(\cdot)^T$. The expectation operator is denoted by $\mathbb{E}$, and $\|\cdot\|$ represents the norm. The symbols $\lfloor\cdot\rceil$, $\lfloor\cdot\rfloor$, and $\lceil\cdot\rceil$ denote the rounding, floor, and ceiling operations, respectively.

II. FDD massive MIMO and Backbone Neural CSI Compressor

A. FDD Massive MIMO

We consider an FDD transmission scenario in which a massive MIMO BS equipped with N_t antennas serves multiple single-antenna users. Downlink transmission is performed using orthogonal frequency division multiplexing (OFDM) over N_c subcarriers. For each subcarrier $m \in \{1, \ldots, N_c\}$, let $\mathbf{h}_m \in \mathbb{C}^{N_t}$ denote the downlink channel vector between the BS and a given user, $\mathbf{w}_m \in \mathbb{C}^{N_t}$ the corresponding transmit precoding vector, $x_m \in \mathbb{C}$ the transmitted data symbol, and $\mathbf{n}_m \in \mathbb{C}$ the additive noise. The received signal at the user on subcarrier m is given by

$$\mathbf{y}_m = \mathbf{h}_m^T \mathbf{w}_m x_m + \mathbf{n}_m. \tag{1}$$

The downlink CSI across all subcarriers in the spatial–frequency domain is represented by the matrix

$$\mathbf{H} = \left[\mathbf{h}_1, \ldots, \mathbf{h}_{N_c}\right] \in \mathbb{C}^{N_t \times N_c}. \tag{2}$$

In massive MIMO systems, the channel matrix is high-dimensional, making full CSI feedback prohibitively expensive in terms of uplink bandwidth. To alleviate this bottleneck, CSI compression is commonly employed, whereby the user compresses the CSI matrix and the BS subsequently reconstructs the channel matrix. The objective of CSI compression in FDD massive MIMO systems is to minimize the reconstruction distortion while satisfying a prescribed feedback bit-rate constraint from the user to the BS.

B. Backbone Neural CSI Compressor

We consider an autoencoder-based NN architecture that explicitly targets the RD trade-off in CSI compression. Quantization, entropy encoding, and entropy decoding are incorporated into the training procedure, enabling optimization of both the feedback rate and the reconstruction distortion of the neural CSI compressor. The proposed compressor is denoted by

$$c = (f_\phi, g_\theta, \gamma_\theta), \tag{3}$$

where $f_\phi : \mathcal{C}^{N_t \times N_c} \to \mathcal{Z}$ represents the feature encoder parameterized by ϕ, $g_\theta : \mathcal{Z} \to \mathcal{C}^{N_t \times N_c}$ denotes the feature decoder, and γ_θ corresponds to the entropy coder, jointly parameterized by θ [16]. For notational simplicity, the same symbol θ is used to denote the parameter set of both the feature decoder and the entropy coder.

The *encoder* maps each CSI matrix $\mathbf{H} \in \mathbb{C}^{N_t \times N_c}$ to a latent representation $\mathbf{Z} \in \mathcal{Z}$. The latent representation is then quantized and transmitted losslessly to the receiver using a variable-length entropy code. Let $\bar{\mathbf{Z}} = Q(\mathbf{Z})$ denote the quantized latent representation, where $Q(\cdot)$ represents the quantization operation. The entropy encoder converts the quantized latent values into a bitstream using a lossless coding scheme, given by

$$b_z = \gamma_\theta(\bar{\mathbf{Z}}; p_\theta), \tag{4}$$

where p_θ denotes the prior probability model of the latent representation.

At the *decoder*, the entropy decoder is first applied to recover the quantized latent representation from the received bitstream as $\bar{\mathbf{Z}} = \gamma_\theta^{-1}(b_z; p_\theta)$. The de-quantization operation is then applied to obtain the continuous latent representation, denoted by $\hat{\mathbf{Z}} = Q^{-1}(\bar{\mathbf{Z}})$. We emphasize that $Q^{-1}(\cdot)$ does not represent a true inverse of the quantization function $Q(\cdot)$ and is generally non-bijective in lossy compression. Both the encoder and decoder employ a shared entropy coder γ_θ with prior distribution p_θ to ensure the lossless transmission of the quantized latent representation $\bar{\mathbf{Z}}$. Finally, the feature decoder reconstructs the CSI matrix from the de-quantized latent representation as

$$\hat{\mathbf{H}} = g_\theta\left(\hat{\mathbf{Z}}\right). \tag{5}$$

Let the distortion function be denoted by $\rho : \mathbf{H} \times \hat{\mathbf{H}} \to [0, \infty)$, which quantifies the reconstruction error between the ground-truth CSI matrix and its estimate. In practice, the distortion is typically measured using the squared error, i.e., $\rho(\mathbf{H}, \hat{\mathbf{H}}) \propto \|\mathbf{H} - \hat{\mathbf{H}}\|^2$. Under a lossy compression framework, the resulting distortion of a compressor c is defined as

$$D(c) = \mathbb{E}\left[\rho(\mathbf{H}, \hat{\mathbf{H}})\right]. \tag{6}$$

The expectation is taken over the random realization of the channel matrix $\mathbf{H}$, and without loss of optimality, we restrict

our attention to deterministic codes [39]. Let the bit length associated with encoding a CSI matrix $\mathbf{H}$ be defined as

$$l(\mathbf{H}) := \left|\gamma_\theta(Q(f_\phi(\mathbf{H})))\right|, \tag{7}$$

and the corresponding average bit rate be defined as

$$R(c) = \mathbb{E}[l(\mathbf{H})]\,, \tag{8}$$

which captures the expected number of bits required to encode the CSI matrix $\mathbf{H}$. In practice, this rate can be approximated by the entropy of the quantized latent representation, resulting in

$$R(c) = \mathbb{E}\big[-\log_2 p_\theta(\bar{\mathbf{Z}})\big]\,. \tag{9}$$

The objective is to jointly minimize the distortion and the rate with respect to the encoder, decoder, and latent representations. According to RD theory, the fundamental limits of lossy compression are characterized by a conditional distribution $p_{\hat{\mathbf{H}}|\mathbf{H}}$. The RD function is given by

$$R(D) = \inf_{p_{\hat{\mathbf{H}}|\mathbf{H}}:\, \mathbb{E}\left[\rho(\mathbf{H},\hat{\mathbf{H}})\right]\le D} I\left(\mathbf{H};\hat{\mathbf{H}}\right), \tag{10}$$

where $I(\mathbf{H};\hat{\mathbf{H}})$ denotes the mutual information between the CSI matrix $\mathbf{H}$ and its reconstruction $\hat{\mathbf{H}}$. In theory, this optimal rate can be achieved using vector quantization by jointly compressing many independent realizations of the channel matrix. However, such schemes become intractable for high-dimensional CSI and are incompatible with practical systems, where each CSI matrix must be compressed individually. To address this limitation, we adopt an *operational RD* formulation and denote the set of admissible codecs by $\mathcal{C}$:

$$R_{\mathrm{O}}(D) = \inf_{c\in\mathcal{C}:\, \mathbb{E}\left[\rho(\mathbf{H},\hat{\mathbf{H}})\right]\le D} \mathbb{E}[l(\mathbf{H})]\,, \tag{11}$$

which characterizes the *operational RD function* over the class of admissible codecs $\mathcal{C}$. This constrained optimization problem can be equivalently addressed via a Lagrangian formulation,

$$L(\lambda, c) = R(c) + \lambda D(c) = \mathbb{E}[l(\mathbf{H})] + \lambda\mathbb{E}\left[\rho(\mathbf{H},\hat{\mathbf{H}})\right]. \tag{12}$$

For any $\lambda \ge 0$, minimizing the Lagrangian objective over $\mathcal{C}$ yields an optimal codec c^*. The parameter λ governs the trade-off between rate and distortion. Larger values of λ prioritize distortion minimization, resulting in operation at a higher bit-rate regime, whereas smaller values favor stronger compression at the expense of reconstruction accuracy. By adopting the MSE distortion measure, the resulting RD loss is expressed as

$$\begin{aligned} L_{\mathrm{RD}}(\phi,\theta) = {} & \mathbb{E}[-\log_2 p_\theta(Q(f_\phi(\mathbf{H})))] \\ & + \lambda\,\mathbb{E}\left[\left\|g_\theta\big(Q^{-1}(Q(f_\phi(\mathbf{H})))\big) - \mathbf{H}\right\|^2\right]. \end{aligned} \tag{13}$$

Fig. 1 illustrates the encoder–decoder architecture of the proposed neural CSI compressor, where the input consists of the real and imaginary components of the spatial–frequency channel matrix. The encoder function f_ϕ is composed of a sequence of convolutional layers with nonlinear activation functions that map the high-dimensional massive MIMO CSI matrix $\mathbf{H}$ to a lower-dimensional latent representation. Specifically, the first two layers employ convolutional filters with

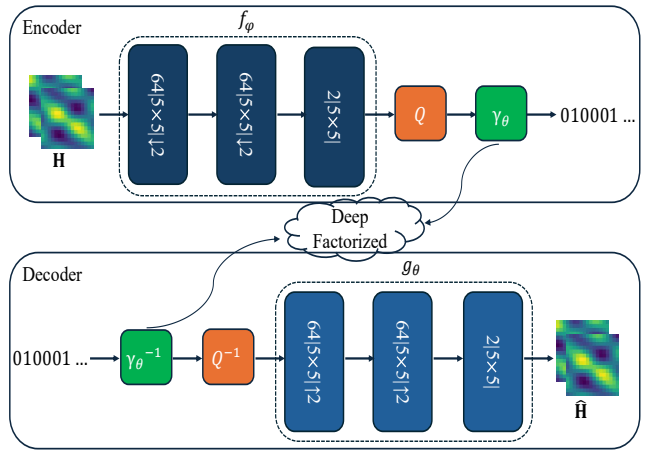

Fig. 1: The neural CSI compressor architecture.

64 kernels followed by ReLU activation functions, while the final layer uses 2 convolutional kernels with a linear activation. Max-pooling operations are applied to progressively reduce the spatial dimensions of the input CSI tensor.

After obtaining the low-dimensional latent representation, quantization and entropy coding are applied to convert the latent space into a bit sequence. Specifically, a uniform scalar quantizer with unit step size is employed to map continuous latent values to discrete symbols. Leveraging the latent probability distribution learned during training, a lossless entropy coding scheme, such as range coding or context-adaptive binary arithmetic coding (CABAC), is then used to encode the quantized latent representation, resulting in a variable-length bitstream.

Incorporating quantization into the compression framework introduces a challenge for gradient-based optimization, as the quantization operation has zero derivative almost everywhere. To address this issue, a common relaxation is adopted whereby quantization is approximated by adding independent and identically distributed uniform noise, $\Delta\mathbf{Z} \sim \mathcal{U}(0,1)$, to the latent representation, yielding $\tilde{\mathbf{Z}} = \mathbf{Z} + \Delta\mathbf{Z}$. This relaxation provides a differentiable surrogate and yields an upper bound on the expected number of bits required to encode the latent space. During inference, the true quantization operation is applied, and the rate is computed using the actual quantized latent values without additive noise.

On the decoder side, the entropy-coded bitstream is first decoded to recover the quantized latent representation, which is then mapped to a low-dimensional channel representation by the decoding network. The entropy encoder and decoder share the same latent prior distribution. In our architecture, we adopt a fully factorized prior implemented via the `DeepFactorized` model [40], which leverages deep neural networks to learn the latent probability distribution under factorization assumptions.

The decoder network, represented by the function g_θ in Fig. 1, consists of a stack of convolutional layers with a kernel size of 5×5, each followed by a ReLU activation function. To reconstruct the high-dimensional CSI matrix from the low-dimensional entropy-decoded representation, an

upsampling operation is applied to progressively restore the spatial resolution.

III. Neural CSI Compression Fine-Tuning

A fundamental challenge in learning-based methods is handling distribution shifts. When model capacity is limited or training data is insufficient, such shifts can cause significant performance degradation, thereby limiting real-world effectiveness. This issue is particularly critical in applications such as neural CSI compression for wireless communication systems, where the operating environment is inherently dynamic.

A. Evaluation of The Backbone Neural CSI Compressor

To assess the performance of the trained neural CSI compressor, we employ the DeepMIMO dataset, which constructs MIMO channel realizations using ray-tracing simulations generated by Remcom Wireless InSite [41]. Specifically, the neural CSI compressor is trained using the static outdoor scenario referred to as O1 at a carrier frequency of 28 GHz. The detailed parameter configuration for this dataset is summarized in Table I.

A total of 11,920 CSI realizations are generated and partitioned into training, validation, and test sets using a 40%, 40%, and 20% split, respectively. Training is conducted using the RD loss function defined in (13), with the objective of jointly optimizing the encoder and decoder parameters as well as the factorized probability model used for entropy coding. The backbone neural CSI compressor is implemented in Python 3.9 using the TensorFlow framework. Network parameters are optimized using the Adam optimizer with a learning rate of 10^{-3} and a batch size of 32 over 200 training epochs.

We assess the performance of the neural CSI compressor in terms of the RD trade-off achieved, where the distortion is measured as the normalized mean squared error (NMSE), defined as

$$\text{NMSE} \triangleq \mathbb{E}\left[\frac{\|\mathbf{H}-\hat{\mathbf{H}}\|^2}{\|\mathbf{H}\|^2}\right], \tag{14}$$

where $\mathbf{H}$ and $\hat{\mathbf{H}}$ denote the true and reconstructed channel matrices, respectively. The bit rate of the entropy-coded latent representation is estimated using (9) and normalized by the CSI dimensionality, i.e., 64×64.

The RD curves in Fig. 2 illustrate the performance of the trained neural CSI compressor for different values of λ, where each operating point corresponds to $\lambda \in 5\times10^4, 10^5, 5\times10^5, 10^6$. The results confirm that the proposed neural CSI compressor achieves strong RD performance when evaluated on previously unseen CSI realizations drawn from the same propagation environment.

B. Evaluation Under Distribution Shifts

To analyze the impact of distribution shifts, we evaluate the trained neural CSI compressor in propagation environments that differ significantly from the training domain. Specifically, we consider multiple distinct datasets, namely QuaDRiGa [36], 3GPP CDL [37], and DeepMIMO [38] (under a different scenario), to assess the resulting RD performance. In the following, we describe the key parameter configurations and channel characteristics of these datasets.

TABLE I: DeepMIMO dataset parameters

DeepMIMO parameters	Value
Scenario	O1
Center frequency	28 GHz
Number of paths	10
Active users	from row 1100 to 2200
Active BS number	BS 5
Bandwidth	50 MHz
Number of OFDM subcarriers	64
BS antennas	N_x = 1, N_y = 64, N_z = 1
UE antennas	N_x = 1, N_y = 1, N_z = 1

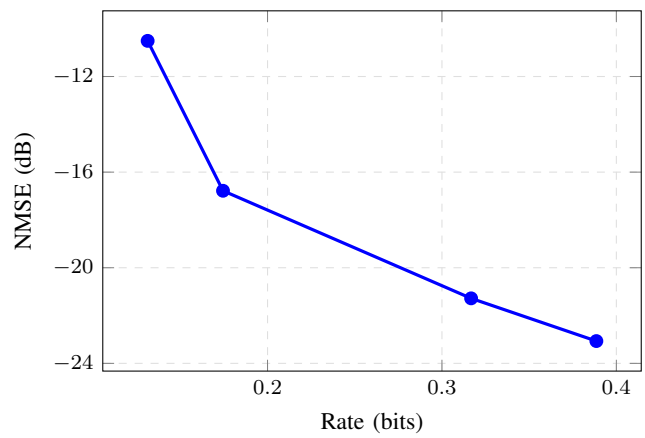

Fig. 2: RD performance of the backbone neural CSI compressor.

a) QuaDRiGa Dataset: QuaDRiGa is a geometry-based stochastic spatial channel model that captures spatial consistency and realistic multipath evolution in three-dimensional propagation environments. Using the QuaDRiGa channel generator, we simulate an urban massive MIMO deployment with a BS equipped with 64 antenna elements and operating over 64 OFDM subcarriers. The BS is placed on a rooftop, while a mobile user traverses a linear track of length 350 m. The underlying three-dimensional propagation environment is based on the Madrid grid developed within the METIS project, enabling realistic modeling of spatial channel evolution and path dynamics along the user trajectory. The simulation layout is illustrated in Fig. 3, and the corresponding dataset parameters are summarized in Table II.

b) 3GPP CDL Dataset: To further evaluate the performance of the proposed neural CSI compression framework under controlled and standardized channel conditions, we generate a dataset based on the 3GPP CDL channel model defined in TR 38.901. Specifically, we consider the CDL-B delay profile under an urban macrocell (UMa) scenario with a user velocity of 30 km/h. The CDL model characterizes time-varying multipath propagation through a finite set of clusters with fixed angular statistics, while temporal channel evolution is primarily governed by Doppler effects induced by user mobility. In our setup, the BS is equipped with 64 transmit antennas, and the user employs a single receive antenna. The

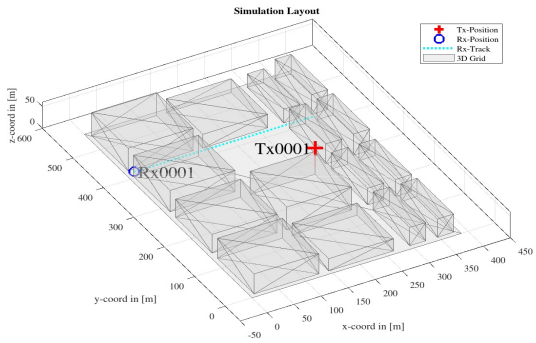

Fig. 3: Simulation layout of QuaDRiGa dataset [36].

TABLE II: Simulation parameters of the QuaDRiGa dataset.

Parameter	Value
Carrier frequency	3.7 GHz
Bandwidth	50 MHz
Number of OFDM subcarriers	64
Number of BS antennas	64
Number of UE antennas	1
Initial user position	(15, 415, 1.2)
BS position	(267, 267, 60)
BS orientation	$\pi/2$ (facing north)
Spatial sampling density	29.6 samples/m

system operates at a carrier frequency of 28 GHz using an OFDM numerology with a subcarrier spacing of 30 kHz. The detailed simulation parameters for this dataset are summarized in Table III.

c) DeepMIMO Dataset: For the DeepMIMO dataset, we consider another site-specific, ray-tracing-based channel model referred to as O2 Dynamic. In this setting, CSI realizations are generated across a sequence of discrete scenes with a sampling interval of 100 ms. The top view of the scenario, shown in Fig. 4, consists of three streets and two intersections in an urban environment. Across the 1,000 captured scenes, vehicular users change positions while the underlying propagation geometry remains fixed, resulting in quasi-stationary channel statistics within each scene. The dataset includes two BSs and approximately 115,000 candidate user locations, enabling large-scale and diverse CSI generation under realistic site-specific conditions. The detailed simulation parameters for this scenario are summarized in Table IV.

To evaluate the generalization capability of the trained neural CSI compressor in previously unseen environments, we assess its RD performance in Fig. 5. The model is tested on the aforementioned datasets without any fine-tuning. As shown in the figure, the trained compressor exhibits a severe performance degradation under distribution shifts, failing to cope with the new channel statistics. These results clearly demonstrate the limitations of direct model reuse and underscore the necessity of an effective model fine-tuning strategy for neural CSI compression in massive MIMO systems.

C. Neural Model Fine-Tuning

A trained model can be fine-tuned to new channel statistics by exposing it to data samples from the target environment.

TABLE III: Simulation parameters for the 3GPP CDL dataset.

Parameter	Value
Delay profile	CDL-B
Carrier frequency	28 GHz
User velocity	30 km/h
Delay spread	200 ns
Number of BS antennas	64
Number of UE antennas	1
OFDM subcarrier spacing	30 kHz
Number of resource blocks	25
Number of selected subcarriers	64
CSI sample stride	5 OFDM symbols
Number of CSI samples	5,000

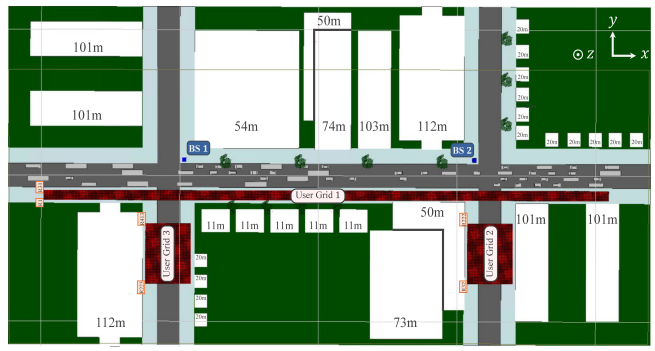

Fig. 4: The top view of the O2 Dynamic scenario [38].

In the context of neural CSI compression, model fine-tuning corresponds to updating the backbone network parameters, namely ϕ_0 and θ_0 for the encoder and decoder, respectively, which were initially trained on a generic dataset. Fine-tuning the backbone neural CSI compressor using CSI samples from a new domain yields updated encoder and decoder parameters, denoted by ϕ and θ, respectively. Since the encoder is assumed to be fixed at the transmitter, only the updated decoder parameters need to be conveyed in the bitstream.

We consider a full-model fine-tuning, where both the encoder and decoder networks are fine-tuned while explicitly accounting for the additional communication overhead associated with conveying decoder model updates from the transmitter to the receiver. Specifically, the decoder model updates, defined as $\delta \triangleq \theta - \theta_0$, are encoded jointly with the latent representation z. To encode the decoder model update vector, it is first quantized. Unlike the unit-bin quantization applied to the latent space, a higher-resolution quantization is employed for the model updates, since their values typically vary only slightly, depending on the learning rate. In particular, N equispaced bins of width t are used, and the quantization function for the model updates is defined as follows [42]:

$$\bar{\delta} = Q_t(\delta) = \text{clip}\left(\left\lfloor \frac{\delta}{t} \right\rceil t, -\frac{(N-1)t}{2}, \frac{(N-1)t}{2}\right), \quad (15)$$

where clipping and rounding operations are defined as:

$$\text{clip}(x, x_{\min}, x_{\max}) = \begin{cases} x_{\min}, & \text{if } x < x_{\min}, \\ x, & \text{if } x_{\min} \leq x \leq x_{\max}, \\ x_{\max}, & \text{if } x > x_{\max}, \end{cases} \quad (16)$$

TABLE IV: Simulation parameters of the DeepMIMO dataset (O2 Dynamic scenario).

Parameter	Value
Scenario	O2 Dynamic
Carrier frequency	3.5 GHz
Bandwidth	50 MHz
Number of OFDM subcarriers	64
Number of multipath components	10
Active user rows	1–31
Scene index	1
Active BS	BS 1
BS antenna configuration	$N_x = 1,\ N_y = 64,\ N_z = 1$
UE antenna configuration	$N_x = 1,\ N_y = 1,\ N_z = 1$

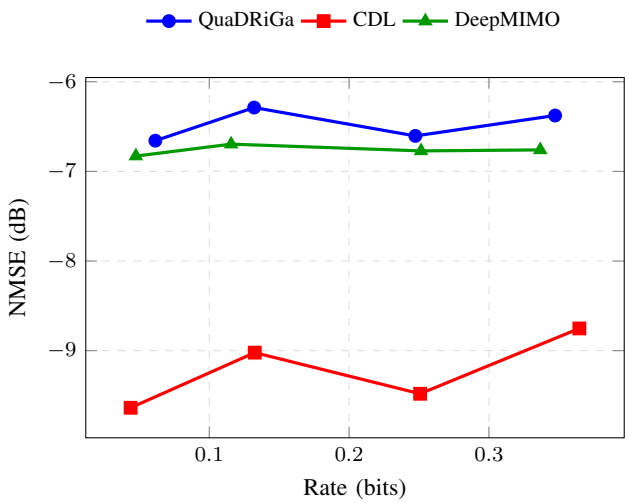

Fig. 5: Evaluation of the backbone neural CSI compressor using QuaDRiGa, CDL, and DeepMIMO datasets.

$$\lfloor x \rceil = \begin{cases} \lfloor x \rfloor, & \text{if } x - \lfloor x \rfloor < 0.5, \\ \lceil x \rceil, & \text{if } x - \lfloor x \rfloor \geq 0.5. \end{cases} \tag{17}$$

Since both rounding and clipping are non-differentiable operations, they obstruct gradient-based training of the neural CSI compressor. To address this issue, a commonly used technique known as the straight-through estimator (STE) is employed [43], where the gradient of the quantization function is approximated as $\partial Q_t(\delta)/\partial\delta = 1$. Here, the quantization bin width is controlled by t, and N denotes the number of quantization bins.

Let $\bar{\delta} = Q_t(\delta)$ denote the quantized model update. The discrete model prior $p[\bar{\delta}]$ is obtained by pushing forward the continuous prior $p(\delta)$ through the quantization function Q_t, yielding

$$\begin{aligned} p[\bar{\delta}] &= \int_{Q_t^{-1}(\bar{\delta})} p(\delta)\, d\delta \\ &= \int_{\bar{\delta}-t/2}^{\bar{\delta}+t/2} p(\delta)\, d\delta \\ &= P(\delta < \bar{\delta} + t/2) - P(\delta < \bar{\delta} - t/2). \end{aligned} \tag{18}$$

Thus, $p[\bar{\delta}]$ represents the probability mass assigned to the quantization bin centered at $\bar{\delta}$. It is computed as the difference between the cumulative distribution function (CDF) values of $p(\delta)$ evaluated at the bin boundaries.

After quantization, an appropriate model prior $p(\delta)$ must be selected to enable efficient entropy coding of the discrete model updates. Various distributions can be used to model this prior, such as a zero-mean Gaussian, i.e., $p(\delta) = \mathcal{N}(\mathbf{0}, \sigma\mathbf{I})$. However, a key limitation of a Gaussian prior is its relatively high coding cost for zero updates. As a result, even when no effective update is applied, encoding such updates can still incur a non-negligible bit rate.

To address this inefficiency, a spike-and-slab prior is adopted [44], defined as

$$p(\delta) = \frac{p_{\text{slab}}(\delta) + \alpha\, p_{\text{spike}}(\delta)}{1+\alpha}, \tag{19}$$

where

$$p_{\text{slab}}(\delta) = \mathcal{N}(\delta \mid 0, \sigma^2 \boldsymbol{I}) \tag{20}$$

denotes the slab component, and

$$p_{\text{spike}}(\delta) = \mathcal{N}(\delta \mid 0, (t/6)^2 \boldsymbol{I}) \tag{21}$$

denotes the spike component.

The resulting prior $p(\delta)$ is a mixture of Gaussian distributions, where $\alpha \in \mathbb{R}^+$ controls the relative weight of the spike component. The parameters t and σ, with $\sigma \gg t/6$, correspond to the standard deviations of the spike and slab distributions, respectively. By setting the standard deviation of the spike component to $t/6$, approximately 99.7% of its probability mass lies within the central quantization bin after quantization. Consequently, selecting a sufficiently large value of α significantly reduces the bit-rate cost associated with zero updates and encourages the model to learn only the most informative parameter updates. This prior effectively promotes sparsity in the model updates while maintaining flexibility for larger, informative deviations.

The bit-rate cost associated with the quantized model updates $\bar{\delta}$, under the discrete prior $p[\bar{\delta}]$, is given by

$$\bar{M} = -\log_2 p[\bar{\delta}]. \tag{22}$$

Following common practice, this discrete cost is approximated during training by its continuous counterpart [42],

$$M = -\log_2 p(\delta). \tag{23}$$

To regularize the bit-rate cost of model updates during full-model fine-tuning, the update cost is incorporated into the RD objective in (13) as

$$L_{\text{RDM}}(\phi, \theta) = L_{\text{RD}}(\phi, \theta) - \log_2 p(\delta), \tag{24}$$

By optimizing this objective during full-model fine-tuning, the network learns to balance the achievable RD gain against the bit-rate cost required to transmit the decoder model updates. Model fine-tuning is performed using CSI samples collected in the target environment, denoted by $\mathcal{H}_{\text{T}}$. For evaluation, the fine-tuned model is applied to CSI samples acquired after the fine-tuning phase, denoted by $\mathcal{H}_{\text{E}}$. The encoding and decoding procedures for full-model fine-tuning are summarized in Algorithms 1 and 2, respectively.

Algorithm 1 Full-Model Fine-Tuning - Encoding

Input: Global model parameters $\{\theta_0, \phi_0\}$ trained on a generic dataset, batch size B, CSI samples from a new environment $\mathcal{H} = \{\mathcal{H}_{\mathrm{T}}, \mathcal{H}_{\mathrm{E}}\}$.
Output: Compressed bitstream $b = (b_{\bar{\delta}}, b_z)$.
Initialize model parameters: $\phi = \phi_0$, and $\theta = \theta_0$
for epoch $= 1$ to num_epochs **do**
  for each batch b in $\mathcal{H}_{\mathrm{T}}$ **do**
    Load data batch $\mathbf{H} \in \{\mathcal{H}_b\}_{b=1}^{B}$ from $\mathcal{H}_{\mathrm{T}}$.
    Quantize updated decoder parameters: $\bar{\theta} = Q_t(\delta) + \theta_0$, with $\delta = \theta - \theta_0$.
    Apply feature encoder and quantization, $\mathbf{Z} = f_\phi(\mathbf{H})$ and $\tilde{\mathbf{Z}} = \mathbf{Z} + \Delta\mathbf{Z}$.
    Apply feature decoder, $\hat{\mathbf{H}} = g_\theta(\tilde{\mathbf{Z}})$.
    Compute loss $L_{\mathrm{RDM}}(\phi, \theta)$ according to equation (13).
    Backpropagate using STE for $Q_t(\delta)$, then update θ, ϕ using gradients $\partial L_{\mathrm{RDM}}(\phi, \theta)/\partial\theta$ and $\partial L_{\mathrm{RDM}}(\phi, \bar{\theta})/\partial\phi$.
  end for
end for
return Fine-tuned model parameters $\{\phi^*, \theta^*\}$.
Compress $\mathbf{H} \in \mathcal{H}_{\mathrm{E}}$ to latent representation $\bar{\mathbf{Z}} = Q(f_{\phi^*}(\mathbf{H}))$.
Compute quantized model parameters: $\bar{\theta} = \theta_0 + \bar{\delta}$, with $\bar{\delta} = Q_t(\theta^* - \theta_0)$.
Entropy encode: $b_{\bar{\delta}} = \gamma(\bar{\delta}; p[\bar{\delta}])$ and $b_z = \gamma_{\bar{\theta}}(\bar{\mathbf{Z}}; p_{\bar{\theta}})$.

Algorithm 2 Full-Model Fine-Tuning - Decoding

Input: Global model parameters θ_0 trained on a generic dataset, model prior $p[\bar{\delta}]$, bitstream $b = (b_z, b_{\bar{\delta}})$.
Output: Decompressed CSI matrix $\hat{\mathbf{H}}$.
Entropy decode: $\bar{\delta} = \gamma^{-1}(b_{\bar{\delta}}; p[\bar{\delta}])$
Compute decoder's updated parameters under model prior: $\bar{\theta} = \theta_0 + \bar{\delta}$
Entropy decode latent under fine-tuned prior: $\bar{\mathbf{Z}} = \gamma_{\bar{\theta}}^{-1}(b_z; p_{\bar{\theta}})$
Apply de-quantization and feature decoder: $\hat{\mathbf{H}} = Q^{-1}(g_{\bar{\theta}}(\bar{\mathbf{Z}}))$

IV. Performance Evaluation

In this section, we evaluate the effectiveness of different model-fine-tuning schemes applied to the backbone neural CSI compressor in target environments with shifted channel statistics, along with the associated computational cost of model fine-tuning. We consider the following neural CSI compression fine-tuning schemes:

- **Pretrained**: denotes applying the pretrained neural CSI compressor directly in the target environment without any model fine-tuning.
- **EO**: corresponds to the encoder-only fine-tuning scheme. Encoder parameters are updated using CSI samples from the target environment, while the decoder remains fixed. This enables partial specialization of the pretrained model to the target environment without requiring any additional feedback. Results are reported for models trained at different values of λ.
- **FM**: refers to the full-model fine-tuning scheme, where both encoder and decoder parameters are updated. The spike-and-slab prior parameters are set to $t = 0.005$, $N = 50$, and $\sigma = 0.05$. Only the low-bit-rate neural CSI compressor trained at $\lambda = 5 \times 10^4$ is fine-tuned. The reported rate for this scheme represents the total rate, i.e., the sum of the latent bit-rate and the amortized model-update bit-rate. Minor parameter adjustments are made to achieve the best performance.
- **FM1**: Motivated by the observation that earlier layers of DNNs capture more general features, whereas later layers encode finer, task-specific details [45], model fine-tuning is restricted to the final convolutional layer of the decoder network to reduce the bit-rate cost of model updates. Accordingly, FM1 denotes a partial full-model fine-tuning scheme in which only the final convolutional layer of the decoder is updated, while all other layers remain fixed. The same update parameters as in the FM scheme are used.
- **FM1-UP**: is identical to FM1, except that a uniform prior is used for the model updates instead of the spike-and-slab prior, as originally proposed in our initial work [46]. Since only a single decoder layer is updated, resulting in a smaller number of model update parameters compared to FM, the uniform prior imposes weaker sparsity constraints and may therefore better accommodate a larger number of effective updates.
- **FM-2bit**: represents the full-model fine-tuning scheme with coarse quantization using four quantization bins (2-bit quantization). Prior results in [34], based on a federated edge learning framework, suggest that the impact of coarse quantization noise can be mitigated in the downlink stream. Motivated by this observation, the FM scheme is evaluated under this coarse quantization setting.
- **GA**: denotes a genie-aided scheme, in which the decoder is assumed to have perfect knowledge of the model updates without any feedback overhead. This scheme provides a lower bound on the achievable performance of neural CSI compression under distribution shifts. Similar to FM, only the low-bit-rate trained neural CSI compressor is used. The approaches in [29], [32] can be interpreted as instances of this genie-aided setting, as they update a vanilla autoencoder without accounting for the communication overhead of model-update feedback.
- **TM**: corresponds to the translation-module-based scheme proposed in [30]. Inspired by image-to-image translation in computer vision, this approach maps input CSI to the target domain using a convolutional translation module at the encoder and a retranslation module at the decoder. The same network architecture and sparsity-alignment function as in [30] are employed. Similar to the encoder-only fine-tuning, models trained at different values of λ are used. To provide a lower-bound rate, the communication overhead required to convey the retranslation module parameters to the decoder is also ignored.

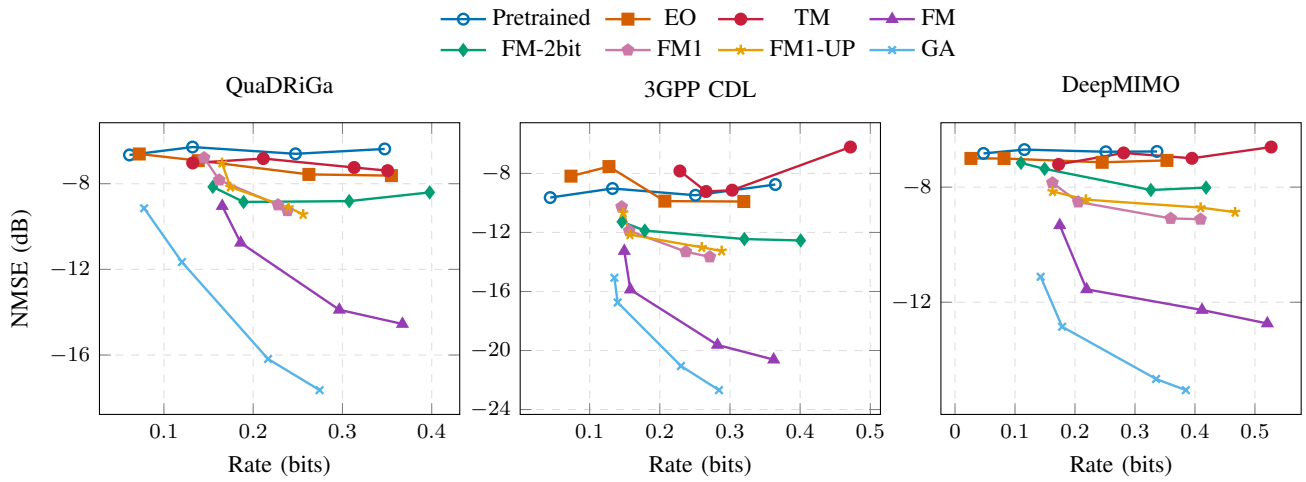

Fig. 6: RD results of different fine-tuning schemes evaluated on multiple datasets.

A. RD Numerical Results

In this subsection, we present the RD results of the different fine-tuning schemes for neural CSI compression. The schemes are evaluated using the QuaDRiGa, 3GPP CDL, and DeepMIMO datasets introduced in Section III. Fine-tuning is carried out using 100 CSI samples from the target domain with a batch size of 50 over 100 epochs.

Fig. 6 compares the RD performance of different fine-tuning schemes across the QuaDRiGa, 3GPP CDL, and DeepMIMO datasets. From the figure, it can be observed that the gains achieved by EO remain limited compared to schemes that permit decoder-side updates. This confirms that encoder-only fine-tuning can only partially compensate for distribution shifts, while the decoder remains mismatched to the target environment, thereby highlighting the importance of full-model fine-tuning. Moreover, the TM scheme, which relies on translation and retranslation modules without fine-tuning the backbone network, does not provide noticeable RD gains, and its performance remains largely comparable to that of the pretrained model. On the other hand, full-model fine-tuning yields substantial RD improvements across all datasets, demonstrating the benefit of jointly updating the encoder and decoder parameters. Notably, the reported rates for FM include both the latent bit-rate and the amortized model-update bit-rate, indicating that these improvements are achieved despite explicitly accounting for the communication overhead associated with decoder updates.

The partial full-model fine-tuning scheme (FM1 and FM1-UP), which restricts updating to the final convolutional layer of the decoder, underperforms full FM but outperforms EO and TM. This confirms that fine-tuning only high-level, task-specific layers is insufficient to fully capture environment-specific channel characteristics compared to full-model fine-tuning; nevertheless, it still provides an RD improvement relative to schemes in which the decoder remains unchanged. Furthermore, a comparison between FM1 and FM1-UP highlights that when only a single decoder layer is updated, weaker sparsity constraints on the model updates can be afforded, and the resulting RD performance is comparable to that achieved by FM1 with the spike-and-slab prior.

A comparison between FM and FM-2bits suggests that employing finer quantization for model updates improves RD performance. This observation contrasts with the results reported in [34] and can primarily be attributed to differences in the system setup. In particular, [34] considers a multi-user scenario with a federated learning framework, where quantization noise introduced by individual users can be partially compensated through aggregation, whereas such compensation is not present in our considered single-model fine-tuning setting.

Finally, the genie-aided scheme (GA) consistently provides the best RD performance and serves as a lower bound, as it assumes perfect decoder-side knowledge of model updates without any feedback overhead. The performance gap between GA and FM quantifies the cost of explicitly accounting for model-update transmission and underscores the efficiency of the proposed RDM-based full-model fine-tuning framework.

Fig. 7 illustrates the RD performance across evolving CSI samples in the QuaDRiGa, 3GPP CDL, and DeepMIMO datasets. Each operating point corresponds to a different evaluation horizon, defined as the duration over which an updated model is reused before the CSI distribution changes sufficiently to require further updates. The total feedback rate accounts for both the compressed CSI and the amortized communication cost of model updates over the evaluation horizon.

In the QuaDRiGa dataset, channel evolution is driven by user mobility along a 350 m linear track in a realistic three-dimensional urban environment, where propagation paths may appear and disappear as the user moves. This leads to pronounced structural changes in the CSI distribution over distance. Consequently, increasing the evaluation horizon results in a noticeable degradation in reconstruction quality unless model updates are transmitted more frequently. This behavior manifests as a clear trade-off between the amortized model-update rate and distortion, highlighting the limited validity period of fine-tuned models in highly nonstationary propagation environments.

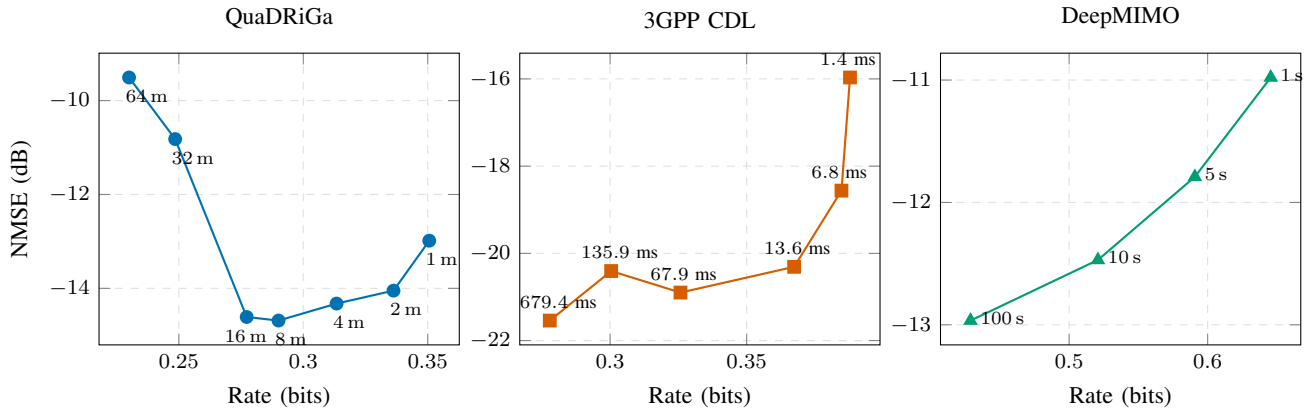

Fig. 7: RD trade-off of full-model fine-tuning across different evaluation horizons. Point annotations indicate distance (m) for QuaDRiGa, channel evolution time at 120 km/h for the CDL model, and captured scene duration for DeepMIMO.

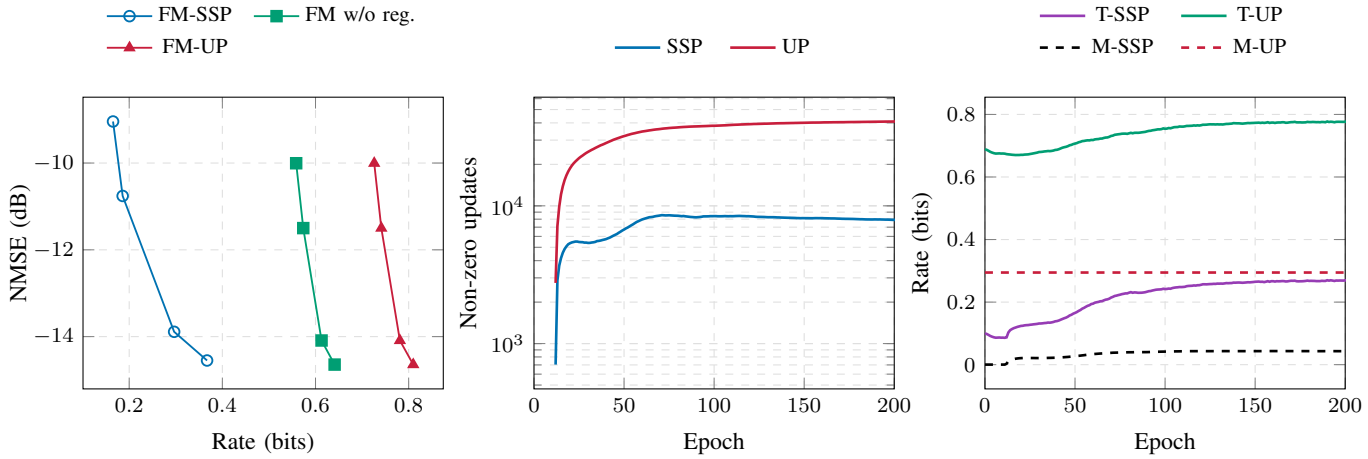

Fig. 8: Ablation results for full-model scheme, highlighting the role of the spike-and-slab prior and the model-update regularizer in controlling the model-update rate.

In contrast, the 3GPP CDL dataset exhibits smooth temporal channel evolution dominated by Doppler effects, with a fixed propagation geometry and no abrupt path changes. As a result, the updated model remains effective over longer time spans, and the RD performance improves as the amortized model-update rate decreases with longer evaluation horizons. It is worth noting that, in this experiment, the user velocity is increased to 120 km/h and a rural macro-cell (RMa) delay profile is employed in order to accelerate CSI variations and create a more challenging evaluation scenario. Similarly, in the DeepMIMO O2 Dynamic scenario, CSI realizations are collected across discrete scenes in a site-specific urban layout with fixed geometry within each scene. Although the user environment varies across scenes, the underlying propagation characteristics remain largely unchanged, leading to quasi-stationary CSI statistics. As a result, the fine-tuned model can be reused over long evaluation horizons with limited degradation in distortion performance.

Fig. 8 presents an ablation study that examines the impact of the model-update prior and the update-rate regularizer on full-model fine-tuning. The scheme denoted as FM-SSP corresponds to full-model fine-tuning with a spike-and-slab prior on the model updates, whereas FM-UP employs a uniform prior. The variant FM w/o reg. removes the model-update regularizer from the RD loss, thereby allowing unrestricted model updates. For clarity, SSP and UP refer to the spike-and-slab and uniform priors, respectively. To analyze the rate behavior in detail, the total feedback rate is decomposed into its components. The terms T-SSP and T-UP denote the total rate, including both the latent bit-rate and the amortized model-update rate, when using the spike-and-slab and uniform priors, respectively. In contrast, M-SSP and M-UP represent only the model-update rate associated with each prior.

The results in Fig. 8 lead to several conclusions regarding full-model fine-tuning. First, incorporating the model-update regularizer in the RD loss is essential for controlling the bit-rate cost of model updates. When the regularizer is removed (FM w/o reg.), the model-update rate increases rapidly, resulting in a substantially higher total rate without corresponding RD gains. This demonstrates that unconstrained full-model fine-tuning can lead to an excessive increase in model-update bit rate.

Second, the choice of prior has a pronounced impact on the sparsity and rate of model updates. The spike-and-slab prior

TABLE V: Impact of model-update quantization resolution on RD performance.

Quantization	QuaDRiGa		3GPP CDL	
	Rate	NMSE (dB)	Rate	NMSE (dB)
1 bit	0.25	-5.73	0.232	-7.88
2 bit	0.34	-8.81	0.308	-12.52
3 bit	0.33	-12.05	0.300	-16.59
4 bit	0.32	-13.70	0.287	-17.56
5 bit	0.30	-13.89	0.277	-17.59
6 bit	0.30	-13.89	0.276	-17.59
7 bit	0.30	-13.89	0.276	-17.59

TABLE VI: RD performance for different dataset size for full-model fine-tuning.

Dataset Size	QuaDRiGa		3GPP CDL	
	Rate	NMSE (dB)	Rate	NMSE (dB)
1	0.248	-7.03	0.241	-10.48
10	0.286	-10.38	0.267	-15.09
50	0.299	-11.79	0.269	-15.26
100	0.297	-13.89	0.275	-17.64
200	0.305	-14.63	0.288	-18.86
300	0.316	**-13.96**	0.302	-19.98
500	0.325	**-10.39**	0.330	-20.75

TABLE VII: Sensitivity analysis of the spike-and-slab prior hyperparameters.

α	σ	t	Rate	NMSE (dB)
t sweep (α=100, σ=0.05)				
100	0.05	0.0005	0.577	-9.60
100	0.05	0.001	0.346	-12.90
100	0.05	0.005	0.314	-13.59
100	0.05	0.01	0.275	-13.17
100	0.05	0.05	0.225	-6.25
σ sweep (α=100, t=0.005)				
100	0.005	0.005	NaN	NaN
100	0.01	0.005	inf	-13.71
100	0.05	0.005	0.314	-13.59
100	0.1	0.005	0.317	-13.68
100	0.5	0.005	0.320	-13.53
100	1	0.005	0.319	-13.58
α sweep (σ=0.005, t=0.05)				
0.1	0.005	0.05	0.506	-13.50
1	0.005	0.05	0.363	-13.79
10	0.005	0.05	0.315	-13.73
100	0.005	0.05	0.314	-13.59
1000	0.005	0.05	0.320	-13.55
10000	0.005	0.05	0.322	-13.54

consistently yields fewer nonzero updates and a lower model-update rate compared to the uniform prior, while achieving strong RD performance. This confirms that explicitly promoting sparsity in the model updates is critical for reducing the feedback overhead associated with model fine-tuning.

Table V reports the RD performance of the full-model scheme under different quantization resolutions for model updates, evaluated on the QuaDRiGa and CDL datasets. The results indicate that low-resolution quantization (below 3 bits) leads to substantial RD degradation. In contrast, increasing the quantization resolution beyond 4 bits does not yield any noticeable RD improvement, suggesting diminishing returns from high-resolution quantization. Overall, moderate-resolution quantization is sufficient to achieve good compression performance.

The results in Table VI illustrate the impact of the dataset size on the RD performance of the full-model fine-tuning scheme. The results indicate that increasing the number of CSI samples used for fine-tuning initially leads to clear improvements in RD performance. However, beyond a certain dataset size, further increasing the number of training samples yields only marginal performance gains, suggesting that a few hundred CSI samples are sufficient for effective full-model fine-tuning. In the QuaDRiGa dataset, using a larger number of CSI samples (e.g., 300–500 samples) leads to a degradation in NMSE performance. This behavior is attributed to the increased heterogeneity of the fine-tuning data, as larger training sets include more diverse channel geometries and spatial conditions. As a result, the fine-tuned model becomes less locally specialized, which is suboptimal for the fixed evaluation horizon considered in this study.

Table VII evaluates the sensitivity of the spike-and-slab prior hyperparameters (α, σ, t) on RD performance for QuaDRiGa dataset. We vary one hyperparameter at a time while keeping the others fixed, as indicated in each block. From the t-sweep, we observe that extremely small values of t lead to overly strong shrinkage toward zero, which destabilizes the update distribution and results in increased rate and degraded NMSE. In this regime, moderate-magnitude updates are excessively penalized and are inefficiently encoded through the slab component. On the other hand, excessively large t weakens the sparsity-inducing effect of the spike component, reducing the separation between near-zero and significant updates and degrading the RD performance. A broad stable region is observed for $t \in [10^{-3}, 10^{-2}]$, where the RD trade-off is well balanced.

For the σ-sweep, very small slab variances lead to numerical instability, as the slab component becomes excessively narrow and cannot accommodate moderate update magnitudes. This results in unstable entropy estimates and unreliable coding behavior. In contrast, for $\sigma \geq 0.05$, the RD performance remains stable, with only minor variations across more than one order of magnitude, indicating that the method is not sensitive to the precise choice of slab scale within this range. Finally, the α-sweep shows that α primarily governs the balance between the spike and slab components, thereby controlling the effective sparsity level of the updates. Across a wide range ($\alpha \in [1, 10^4]$), the RD trade-off exhibits only modest variation, with diminishing changes for large α.

B. Computational Complexity

We analyze the computational cost of the proposed full-model fine-tuning by explicitly accounting for the number of floating-point operations (FLOPs) required by the neural CSI compressor considered in this work. The training complexity of model fine-tuning naturally depends on the chosen backbone architecture, with simpler or more complex networks

incurring lower or higher computational costs, respectively. In this study, the backbone neural CSI compressor consists of stacked 5×5 convolutional and transposed convolutional layers operating on CSI tensors of spatial resolution 64×64.

For a 2D convolution producing an output tensor of size $H \times W \times C_{\text{out}}$ with kernel size $k \times k$ and C_{in} input channels, the number of multiply–accumulate operations (MACs) is given by $HWC_{\text{out}}(k^2 C_{\text{in}})$, where one MAC is counted as two FLOPs. Summing the contributions of all convolutional layers, a single forward pass through the encoder–decoder architecture requires approximately 0.48 GFLOPs per CSI sample. To account for backpropagation and gradient computation during fine-tuning, we adopt a standard approximation whereby the total training cost is three times the forward-pass complexity, resulting in approximately 1.43 GFLOPs per CSI sample per fine-tuning iteration.

The on-device update is performed using $N_T = 100$ CSI samples, a batch size of $B = 50$, and $E = 100$ epochs, corresponding to two optimization steps per epoch and 200 steps in total. Under these settings, the total fine-tuning cost is given by

$$\text{FLOPs}_{\text{total}} \approx 1.43 \times N_T \times E \approx 14.25 \text{ TFLOPs}. \tag{25}$$

The expected fine-tuning latency can be approximated as $T \approx \text{FLOPs}_{\text{total}}/\eta$, where η denotes the effective sustained throughput of the mobile processor in TFLOPs/s [47]. For typical mobile devices, η may range from 0.1 to 1 TFLOPs/s depending on the available hardware acceleration and numerical precision, resulting in fine-tuning latencies on the order of several tens of seconds.

The corresponding energy consumption scales linearly with the total number of operations and can be expressed as $E \approx \text{FLOPs}_{\text{total}}\epsilon_{\text{FLOP}}$, where ϵ_{FLOP} denotes the energy consumed per floating-point operation. For representative values of ϵ_{FLOP} between 10 and 100 pJ/FLOP [47], the total energy cost of fine-tuning lies in the range of 0.04–0.4 Wh, corresponding to a small fraction of a typical smartphone battery capacity. In practical deployments, the fine-tuning will be executed as a background process and invoked only occasionally, e.g., when significant distribution shifts are detected. As a result, the fine-tuning cost can be amortized over time and is unlikely to interfere with latency-critical operations.

V. Conclusion

A major limitation of neural compression approaches is their significant performance degradation when channel statistics change, which is a natural phenomenon in wireless environments. To address this distribution shift problem, we proposed a fine-tuning scheme for neural CSI compression that explicitly accounts for the communication overhead associated with transmitting model updates. Lossless entropy coding is applied to both latent representations and model updates, and a spike-and-slab prior is adopted to promote sparse and efficient parameter updates. Furthermore, the bit-rate of model updates is incorporated into the fine-tuning process through a regularized RD loss function. We conducted simulations across a wide range of wireless CSI datasets, covering both site-specific and stochastic channel models. The results demonstrate that full-model fine-tuning is essential for effectively adapting the neural CSI compressor to varying environments.

We further evaluated full-model fine-tuning under different evaluation horizons, quantization resolutions, and fine-tuning sample sizes. The results show that in environments with rapidly varying CSI statistics, applying fine-tuned models over excessively long horizons can lead to noticeable RD degradation. In addition, using highly diverse CSI samples for fine-tuning may reduce model specialization within a localized operating region. We also observe that moderate quantization resolutions, on the order of 4–5 bits, are sufficient to achieve strong RD performance. As a future direction, this work can be extended toward fully online learning of neural CSI compression. In this setting, the encoder would dynamically adapt its parameters in response to evolving CSI statistics.

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, 2010.

[2] E. G. Larsson, O. Edfors, F. Tufvesson, and T. L. Marzetta, "Massive MIMO for next generation wireless systems," *IEEE Commun. Mag.*, vol. 52, no. 2, pp. 186–195, 2014.

[3] X. Rao and V. K. N. Lau, "Distributed compressive CSIT estimation and feedback for FDD multi-user massive MIMO systems," *IEEE Trans. Signal Process.*, vol. 62, no. 12, pp. 3261–3271, 2014.

[4] X.-L. Huang, J. Wu, Y. Wen, F. Hu, Y. Wang, and T. Jiang, "Rate-adaptive feedback with bayesian compressive sensing in multiuser MIMO beamforming systems," *IEEE Trans. Wireless Commun.*, vol. 15, no. 7, pp. 4839–4851, 2016.

[5] V. Rizzello, M. Nerini, M. Joham, B. Clerckx, and W. Utschick, "User-driven adaptive CSI feedback with ordered vector quantization," *IEEE Wireless Commun. Lett.*, vol. 12, no. 11, pp. 1956–1960, 2023.

[6] R. Bhagavatula and R. W. Heath, "Predictive vector quantization for multicell cooperation with delayed limited feedback," *IEEE Trans. Wireless Commun.*, vol. 12, no. 6, pp. 2588–2597, 2013.

[7] C. Wen, W. Shih, and S. Jin, "Deep learning for massive MIMO CSI feedback," *IEEE Wireless Commun. Lett.*, vol. 7, no. 5, pp. 748–751, Mar. 2018.

[8] J. Guo, C.-K. Wen, S. Jin, and G. Y. Li, "Convolutional neural network-based multiple-rate compressive sensing for massive MIMO CSI feedback: Design, simulation, and analysis," *IEEE Trans. Wireless Commun.*, vol. 19, no. 4, pp. 2827–2840, 2020.

[9] Z. Lu, J. Wang, and J. Song, "Multi-resolution CSI feedback with deep learning in massive MIMO system," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2020, pp. 1–6.

[10] S. Tang, J. Xia, L. Fan, X. Lei, W. Xu, and A. Nallanathan, "Dilated convolution based CSI feedback compression for massive MIMO systems," *IEEE Trans. Veh. Technol.*, vol. 71, no. 10, pp. 11 216–11 221, 2022.

[11] Y. Cui, A. Guo, and C. Song, "Transnet: Full attention network for CSI feedback in FDD massive MIMO system," *IEEE Wireless Commun. Lett.*, vol. 11, no. 5, pp. 903–907, 2022.

[12] Z. Hu, G. Liu, Q. Xie, J. Xue, D. Meng, and D. Gündüz, "A learnable optimization and regularization approach to massive MIMO CSI feedback," *IEEE Trans. Wireless Commun.*, vol. 23, no. 1, pp. 104–116, 2024.

[13] H. Wu, M. Zhang, Y. Shao, K. Mikolajczyk, and D. Gündüz, "MIMO channel as a neural function: Implicit neural representations for extreme CSI compression in massive MIMO systems," 2024. [Online]. Available: https://arxiv.org/abs/2403.13615

[14] M. B. Mashhadi, Q. Yang, and D. Gündüz, "Distributed deep convolutional compression for massive MIMO CSI feedback," *IEEE Trans. Wireless Commun.*, vol. 20, no. 4, pp. 2621–2633, 2021.

[15] W. Chen, W. Wan, S. Wang, P. Sun, G. Y. Li, and B. Ai, "CSI-PPPNet: A one-sided one-for-all deep learning framework for massive MIMO CSI feedback," 2023. [Online]. Available: https://arxiv.org/abs/2211.15851

[16] Y. Yang, S. Mandt, and L. Theis, "An introduction to neural data compression," *Found. Trends. Comput. Graph. Vis.*, vol. 15, no. 2, p. 113–200, apr 2023. [Online]. Available: https://doi.org/10.1561/0600000107

[17] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Adv. Neural Inf. Process. Syst.*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, Eds., vol. 27. Curran Associates, Inc., 2014. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf

[18] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," in *2nd Int. Conf. Learn. Represent. (ICLR) 2014, Banff, AB, Canada, Apr. 14-16, 2014, Conf. Track Proc.*, 2014.

[19] H. Larochelle and I. Murray, "The neural autoregressive distribution estimator," in *Proc. 14th Int. Conf. Artif. Intell. Stat. (AISTATS)*, ser. Proc. Mach. Learn. Res., G. Gordon, D. Dunson, and M. Dudík, Eds., vol. 15. Fort Lauderdale, FL, USA: PMLR, 11–13 Apr 2011, pp. 29–37. [Online]. Available: https://proceedings.mlr.press/v15/larochelle11a.html

[20] A. van den Oord, N. Kalchbrenner, and K. Kavukcuoglu, "Pixel recurrent neural networks," in *Proc. 33rd Int. Conf. Mach. Learn. (ICML)*, ser. Proc. Mach. Learn. Res., M. F. Balcan and K. Q. Weinberger, Eds., vol. 48. New York, NY, USA: PMLR, 20–22 Jun 2016, pp. 1747–1756. [Online]. Available: https://proceedings.mlr.press/v48/oord16.html

[21] Q. Yang, M. B. Mashhadi, and D. Gündüz, "Deep convolutional compression for massive MIMO CSI feedback," in *Proc. IEEE 29th Int. Workshop Mach. Learn. Signal Process. (MLSP)*, 2019, pp. 1–6.

[22] J. Quiñonero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, *Dataset Shift in Machine Learning*. MIT Press, 2022.

[23] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," *Proc. IEEE*, vol. 109, no. 1, pp. 43–76, 2020.

[24] R. Volpi, H. Namkoong, O. Sener, J. C. Duchi, V. Murino, and S. Savarese, "Generalizing to unseen domains via adversarial data augmentation," in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31. Curran Associates, Inc., 2018. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2018/file/1d94108e907bb8311d8802b48fd54b4a-Paper.pdf

[25] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *Int. Conf. Mach. Learn. (ICML)*. PMLR, 2015, pp. 1180–1189.

[26] M. Sattari, H. Guo, D. Gündüz, A. Panahi, and T. Svensson, "Full-duplex millimeter wave MIMO channel estimation: A neural network approach," *IEEE Trans. Mach. Learn. Commun. Netw.*, vol. 2, pp. 1093–1108, 2024.

[27] M. Khani, M. Alizadeh, J. Hoydis, and P. Fleming, "Adaptive neural signal detection for massive MIMO," *IEEE Trans. Wirel. Commun.*, vol. 19, no. 8, pp. 5635–5648, May. 2020.

[28] M. Jankowski, D. Gündüz, and K. Mikolajczyk, "Airnet: Neural network transmission over the air," in *2022 IEEE Int. Symp. Inf. Theory (ISIT)*, 2022, pp. 2451–2456.

[29] J. Zeng, J. Sun, G. Gui, B. Adebisi, T. Ohtsuki, H. Gacanin, and H. Sari, "Downlink CSI feedback algorithm with deep transfer learning for FDD massive MIMO systems," *IEEE Trans. Cogn. Commun. Netw.*, vol. 7, no. 4, pp. 1253–1265, 2021.

[30] Z. Liu, L. Wang, L. Xu, and Z. Ding, "Deep learning for efficient CSI feedback in massive MIMO: Adapting to new environments and small datasets," *IEEE Trans. Wireless Commun.*, pp. 1–1, 2024.

[31] X. Li, J. Guo, C.-K. Wen, S. Jin, S. Han, and X. Wang, "Multi-task learning-based CSI feedback design in multiple scenarios," *IEEE Trans. Commun.*, vol. 71, no. 12, pp. 7039–7055, 2023.

[32] B. Zhang, H. Li, X. Liang, X. Gu, and L. Zhang, "Model transmission-based online updating approach for massive MIMO CSI feedback," *IEEE Communications Letters*, vol. 27, no. 6, pp. 1609–1613, 2023.

[33] X. Zhang, J. Wang, Z. Lu, and H. Zhang, "Continuous online learning-based CSI feedback in massive MIMO systems," *IEEE Communications Letters*, vol. 28, no. 3, pp. 557–561, 2024.

[34] Y. Cui, J. Guo, C.-K. Wen, and S. Jin, "Communication-efficient personalized federated edge learning for massive MIMO CSI feedback," *IEEE Transactions on Wireless Communications*, vol. 23, no. 7, pp. 7362–7375, 2024.

[35] J. Guo, Y. Zuo, C.-K. Wen, and S. Jin, "User-centric online gossip training for autoencoder-based CSI feedback," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 3, pp. 559–572, 2022.

[36] S. Jaeckel, L. Raschkowski, K. Börner, and L. Thiele, "Quadriga: A 3-d multi-cell channel model with time evolution for enabling virtual field trials," *IEEE Trans. Antennas Propag.*, vol. 62, no. 6, pp. 3242–3256, 2014.

[37] "Study on channel model for frequencies from 0.5 to 100 GHz," 3GPP, Tech. Rep. TR 38.901, Mar. 2022, available: https://www.3gpp.org/ftp/Specs/archive/38_series/38.901/.

[38] A. Alkhateeb, "DeepMIMO: A generic deep learning dataset for millimeter wave and massive MIMO applications," in *Proc. Inf. Theory Appl. Workshop (ITA)*, San Diego, CA, Feb 2019, pp. 1–8.

[39] L. Theis and E. Agustsson, "On the advantages of stochastic encoders," *CoRR*, vol. abs/2102.09270, 2021. [Online]. Available: https://arxiv.org/abs/2102.09270

[40] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston, "Variational image compression with a scale hyperprior," in *Int. Conf. Learn. Represent. (ICLR)*, 2018. [Online]. Available: https://openreview.net/forum?id=rkcQFMZRb

[41] Remcom, "Wireless InSite," http://www.remcom.com/wireless-insite.

[42] T. van Rozendaal, I. A. Huijben, and T. Cohen, "Overfitting for fun and profit: Instance-adaptive data compression," in *Int. Conf. Learn. Represent. (ICLR)*, 2021. [Online]. Available: https://openreview.net/forum?id=oFp8Mx_V5FL

[43] Y. Bengio, N. Léonard, and A. C. Courville, "Estimating or propagating gradients through stochastic neurons for conditional computation," *CoRR*, vol. abs/1308.3432, 2013, accessed: 2025-01-27. [Online]. Available: http://arxiv.org/abs/1308.3432

[44] V. Ročková and E. I. G. and, "The spike-and-slab lasso," *Journal of the American Statistical Association*, vol. 113, no. 521, pp. 431–444, 2018. [Online]. Available: https://doi.org/10.1080/01621459.2016.1260469

[45] O. Jourairi, M. Balcilar, A. Lambert, and F. Schnitzler, "Improving The Reconstruction Quality by Overfitted Decoder Bias in Neural Image Compression," in *2022 Picture Coding Symposium (PCS)*, 2022, pp. 61–65.

[46] M. Sattari, D. Gündüz, and T. Svensson, "Dynamically Fine-Tuned Neural Compressor for FDD Massive MIMO CSI Feedback," in *2025 IEEE 26th International Workshop on Signal Processing and Artificial Intelligence for Wireless Communications (SPAWC)*, 2025, pp. 1–5.

[47] P. Hübner, A. Hu, I. Peng, and S. Markidis, "Apple vs. Oranges: Evaluating the Apple Silicon M-Series SoCs for HPC Performance and Efficiency," in *2025 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, 2025, pp. 45–54.