Design of an Efficient CSI Feedback Mechanism in Massive MIMO Systems: A Machine Learning Approach Using Empirical Data

M. Karam Shehzad®, Member, IEEE, Luca Rose, Member, IEEE, Stefan Wesemann®, Member, IEEE, Mohamad Assaad®, Senior Member, IEEE, and Syed Ali Hassan®, Senior Member, IEEE

Abstract—Massive MIMO regime reaps the benefits of spatial diversity and multiplexing gains, subject to precise channel state information (CSI) acquisition. In the current communication architecture, the downlink CSI is estimated by the user equipment (UE) via dedicated pilots and then fed back to the gNodeB (gNB). The feedback information is compressed to reduce overhead. This compression increases the inaccuracy of acquired CSI; thus, degrading precoding quality. Though various autoencoder-based CSI feedback mechanisms have been proposed in recent studies, their computational complexity is enormous. Motivated by these issues, this paper proposes a machine learning-based CSI feedback prediction network, CsiFB-PNet, which exploits twin channel predictors. The feedback at the UE is evaluated with respect to the predicted channel. Further, to enhance the performance, the UE trains the model and reports it to the gNB. CsiFB-PNet can work for both time-division and frequency-division duplex systems, reducing feedback overhead and effectively recovering the compressed CSI. We demonstrate the performance of CsiFB-PNet in a multi-carrier system using the empirical data recorded at the Nokia campus. Furthermore, we use a clustered delay line channel model of the 3GPP 5G new radio standard protocol to make a fair comparison with the benchmark scheme. Numerical results and the complexity analysis verify that the CsiFB-PNet outperforms existing autoencoder-based technique. In particular, CsiFB-PNet precisely recovers the compressed CSI at a reduced overhead cost.

Index Terms—Autoencoder, channel prediction, CSI feedback, massive MIMO, machine learning, 3GPP.
achieves the objective. In the following, we address the literature before highlighting the contributions of our work.

A. Related Work

1) Non-ML-Based Schemes: The challenge of CSI feedback in mMIMO systems has been tackled by a few researchers [15], [16], [17], [18]. The authors of [15] and [16] use spatial and temporal correlation of CSI to reduce feedback overhead, inspiring the establishment of protocols for CSI feedback [15], [16]. Multi-dimensional compressed sensing (CS) theory, which is based on spatial and frequency correlation of the channel, as well as Tucker decomposition [19], is used in [17] to reduce feedback overhead. In addition, an antenna grouping-based feedback reduction method is proposed in [18], where multiple correlated antenna elements are mapped to a single value by exploiting presdesigned patterns. Similarly, many algorithms have been proposed in CS [20], [21], which cannot fully recover compressed CSI due to the use of simple sparsity prior while the channel matrix is used is not perfectly but is approximately sparse. None of these algorithms recover accurate CSI, and massive data processing is required, making them infeasible.

The alternative of codebook-based CSI acquisition has been explored in [22], [23], [24], [25], [26]. This technique is at the base of various commercial systems, such as long-term evolution (LTE)/LTE-advanced and fifth-generation new-radio (5G-NR) that reduces over-the-air (OTA) feedback. Broadly, the channel representation or precoding matrix is chosen from a shared and known set of matrices, known as codebook. By considering line-of-sight (LoS) and non-LoS (NLoS) components between the gNB and the UE, codebooks are designed by [22] and [23], respectively. Using CS theory, a codebook is designed to compress CSI in [24]. Similarly, in [25], an angle-of-departure (AoD)-based codebook design is proposed, compressing CSI more accurately; nevertheless, overhead increases linearly with the AoDs. However, the effectiveness of the codebook depends on the configured environment.

2) ML-Based Schemes: ML has widely been considered in various aspects of wireless communications [6], [7], [10], [27], [28], [29]. Similarly, deep learning (DL), a class of ML having multiple hidden layers, has been utilized for the design of CSI feedback mechanism [9], [30], [31], [32], [33], [34], [35], [36], [37], [38], [39], [40], [41], [42], [43], [44]. The literature borrows the idea of an autoencoder framework used in image compression for CSI feedback. For example, in [9], the CSI matrix is considered an image, and then a DL-based encoder and decoder, named CsiNet, is designed to reduce feedback overhead. The encoder is used at the UE side to convert the CSI matrix into a codeword decoded at the gNB using a decoder. A more practical approach, i.e., compression of noisy (estimation noise) CSI, is studied in [38], where noisy features are first removed, and then compression is performed. Similarly, [44] exploits the structural features of CSI images to enhance the performance of an autoencoder-based CSI compression. More specifically, a self-information model is designed, which dynamically measures the amount of information contained in a patch of a CSI image. This model is then applied to propose a model-and-data-driven network named IdasNet for CSI feedback. In a nutshell, the literature treats the CSI matrix as an image, and then the concept of encoders and decoders is used, where the computation cost of network entities is ignored. Further, to train the deep networks, a massive amount of synthetic data-set is considered [43].

Different from the existing DL-based strategies, i.e., designing encoders and decoders [9], [33], [34], [35], [36], [43], [44], [45], and accordingly transforming CSI matrix into an image, we introduce the concept of using channel prediction for CSI feedback, named CsiFB-PNet, which performs CSI compression and recovery. Particularly, we exploit twin channel predictors at the gNB and UE for CSI feedback. We call these predictors twins because they use the same weights (trained) and test data-set for channel prediction. In summary, the proposed approach trains the ML model at UE and reports the trained model to the gNB for identical channel prediction. The feedback from UE is evaluated with respect to the predicted channel. The purpose of a channel predictor is to forecast CSI realizations. For channel prediction, two statistical algorithms, i.e., autoregressive (AR) and parametric models, have been addressed in [47] and [48], respectively. Both models are based on the statistical modeling of a wireless channel. Broadly, a parametric model estimates Doppler shift, the number of scattering resources, angles of arrival and departure, and amplitude. The assumption is that a wireless channel is a superposition of a finite number of complex sinusoids. Finally, CSI is predicted based on estimated parameters. However, estimated parameters can expire quickly in a highly dynamic environment; therefore, iterative re-estimation of parameters is required, increasing computation cost [48]. In contrast, the AR model approximates the wireless channel as an AR process [49], where the future CSI is extrapolated using a weighted linear combination of current and past CSI. The downside of AR is its vulnerability to impairments, e.g., additive noise, making the AR model infeasible. Therefore, seeing the promising benefits of ML in various domains, we utilize ML for CSI prediction [51], [52], [53], [54].

B. Our Contributions

This paper provides a pragmatic solution for CSI compression and its recovery by exploiting ML-based channel prediction. The major contributions of this paper are:

- By considering a realistic scenario, we propose the idea of using twin channel predictors, called as CsiFB-PNet, for CSI feedback. The proposed idea is completely different from the traditional concept, i.e., using encoders and decoders, for CSI feedback. More specifically, the proposed approach is hybrid, i.e., it is a combination of

---

2 A relevant approach has been studied in [46], where training of channel predictors is done on both sides using compressed CSI. Initial results are presented using 2 × 1 MIMO configuration with MATLAB generated data-set. The downsides of such methodology are the synchronization of two models, reporting of training data to gNB, and exploiting compressed CSI for identical outcomes on both sides.

3 CsiFB-PNet can also work without ML assuming the underlying channel model is evolving using an AR process [50].
conventional CSI feedback and ML-based twin predictors. CsiFB-PNet can help to eliminate the feedback if the prediction is aligned with the estimation; thus, avoiding feedback overhead. Alternatively, it reaps the benefits of reducing feedback overhead and CSI compression errors.

- CsiFB-PNet can train the ML model on two sides and one side, where in the later selected entity for training reports the trained model to the other. The training of CsiFB-PNet is based on true CSI rather than using compressed CSI. Further, CsiFB-PNet can help to support the possible standardization in third generation partnership project (3GPP). Therefore, we also cover the possible standardization aspects of CsiFB-PNet.

- To corroborate the validity of CsiFB-PNet, we use experimental data, which is a huge value-adding over the existing work. The experimental data-set is based on a measurement campaign performed at Nokia Bell-Labs, Stuttgart, Germany, and provides a real-life setting for evaluating and providing interesting insights on our work.

To the best of our knowledge, this is the first work that considers the use of measurement data in the domain of CSI feedback, both for conventional CSI feedback and validation of CsiFB-PNet.

C. Paper Organization and Notations

The rest of the paper is organized as follows. The system model is presented in Section II. Conventional CSI feedback mechanism is summarized in Section III. In Section IV, CsiFB-PNet is explained. Numerical results and measurement campaign are presented in Section V. Finally, in Section VI, conclusions and future outlooks are drawn.

Throughout this paper, the matrices and vectors are represented by boldface upper and lower-case, respectively. Also, scalars are denoted by normal lower-case. The estimated, quantized estimated, predicted, and true channel matrix are denoted by $\hat{H}$, $\hat{H}^Q$, $\hat{H}$, and $H$, respectively, and their vectorized forms are respectively denoted by $\hat{h}$, $\hat{h}^Q$, $\hat{h}$, and $h$. In addition, $Q_{\cdot}(\cdot)$ and $Q_{\cdot}^Q(\cdot)$ represent the quantization functions for the conventional and CsiFB-PNet, respectively. The superscripts $[\cdot]^T$ and $(\cdot)^*$ denote the transpose and conjugate transpose of a matrix/vector, respectively. Furthermore, $E\{\cdot\}$ denotes expectation operator, $\|\cdot\|_F^2$ is squared Frobenius norm and $\|\cdot\|_2^2$ is the squared Euclidean norm.

II. SYSTEM MODEL

Consider the downlink of a mMIMO system, as depicted in Fig. 1, where a gNB is serving a UE within a cell. The gNB and UE are equipped with $N_t$ and $N_r$ transmit and receive antennas, respectively. Without loss of generality, the received signal at the UE can be represented as

$$y(t) = \hat{H}(t)p(t) + n(t),$$

where $y(t) = [y_1(t), y_2(t), \ldots, y_{N_t}(t)]^T$ having dimension $N_r \times 1$, and $t$ is the time index. And,

$$\hat{H}(t) = \begin{bmatrix} h_{11}(t) & \cdots & h_{1N_t}(t) \\ h_{21}(t) & \cdots & h_{2N_t}(t) \\ \vdots & \ddots & \vdots \\ h_{N_r1}(t) & \cdots & h_{N_rN_t}(t) \end{bmatrix}$$

is the channel matrix, which has dimension $N_r \times N_t$. In the above matrix, $h_{n_rn_t} \in \mathbb{C}^{1 \times 1}$ is the channel gain between the $n_r^{th}$ transmit and $n_t^{th}$ receive antennas, where $1 \leq n_r \leq N_t$ and $1 \leq n_t \leq N_r$. Further, $p(t) = [p_1(t), p_2(t), \ldots, p_{N_t}(t)]^T$ is an $N_t \times 1$ symbol vector transmitted by the $N_t$ antennas, and $n(t) = [n_1(t), n_2(t), \ldots, n_{N_r}(t)]^T$ is the additive noise vector, which has dimension $N_r \times 1$.

In the following, we summarize CSI feedback process followed in the conventional approach. Then, we will explain CsiFB-PNet, which is the combination of conventional approach and ML-based channel predictors.

III. CONVENTIONAL APPROACH

Conventionally, CSI at the gNB is acquired by transmitting CSI reference symbols (CSI-RS). Consequently, the UE estimates the channel, denoted by $\tilde{H}_{UE}$, and then feedback a compressed estimated channel, expressed as

$$\tilde{H}_{UE}^Q(t) = Q_{\cdot}(\tilde{H}_{UE}(t)),$$

where $Q_{\cdot}(\cdot)$ is the quantization function using $B_Q$ quantization bits to compress the channel. Here, it is important to highlight that we aim at reducing $B_Q$ as higher value

Fig. 1. Single-cell mMIMO communication environment, where two network entities are depicted, i.e., a gNB and a UE. To acquire the channel at gNB, gNB is sending CSI-RS; consequently, compressed estimated channel is fed back via dedicated feedback link.
will result in higher overhead. $\hat{H}_c^Q$ is degraded by two major factors: estimation and compression or say quantization. Throughout the paper, we use these two terms interchangeably. In this work, we assume channel estimation is perfect, and we focus on CSI feedback scheme. Motivated by the errors caused by compression, we explain CsiFB-PNet below.

IV. CsiFB-PNet

CsiFB-PNet considers using twin channel predictors at both ends of the communication system, i.e., gNB and UE. Specifically, UE trains the ML-based channel predictor by exploiting the estimated channel, $\hat{H}_{UE}$, and reports the trained model to gNB. Consequently, a twin-channel predictor model is deployed at both ends. The UE will evaluate the feedback with respect to prediction. As a toy example, if there is no difference between the predicted and the estimated channel at the UE, then feedback is not required. The rationale behind exploiting the uncompressed CSI is to have a more precise model for the channel prediction. Besides, we assume perfect channel estimation, i.e., $H_{UE} = H$. Later in Section V, we will show that CsiFB-PNet outperforms compared to when exploiting the estimated channel, i.e., $H_{UE} = \hat{H}_{UE}$.

1. mMIMO Channel Prediction Using ML

With the rapid advancements in ML algorithms, their applications have been considered in almost every field. For example, long short-term memory (LSTM)-based artificial neural network (ANN) is used for time-series predictions [55]. LSTM network is different from standard NNs since their training and prediction are based on current and past information; hence called as recurrent neural network (RNN). Such capability leads to process not only single data point but a sequence of data. Motivated by the fact that channel realizations show a large correlation in the time domain, we utilize LSTM network for mMIMO channel prediction.

LSTM has both long-term and short-term memory, influencing the output of NN. LSTM is composed of a cell, and three gates: input, output, and forget. The cell memorizes the information over arbitrary time intervals whereas gates regulate the flow of information into and out of the cell. By exploiting the previous short-term memory state and current external input, forget gate discards unnecessary information. Similarly, input gates store useful information in the current state. At the end, output gate decides which information to pass to the next cell.

In the context of mMIMO channel prediction, the objective of LSTM-based channel predictor is to forecast multi-step ahead values of CSI, denoted by $H(t + D)$, which are as close as possible to true values, i.e., $H(t + D)$, where $D$ is prediction horizon. Without loss of generality, the input $H_{UE}(t)$ of channel predictor is estimated channel at UE, $\hat{H}_{UE}(t) \in \mathbb{C}^{N_t \times N_t}$, along with its $\tau$-step delayed versions, denoted by $[\hat{H}_{UE}(t+\tau), \hat{H}_{UE}(t+2\tau), \ldots, \hat{H}_{UE}(t+\tau)]$. The total external input is given as

$$\hat{H}_{UE} = [\hat{H}_{UE}(t), \hat{H}_{UE}(t-1), \hat{H}_{UE}(t-2), \ldots, \hat{H}_{UE}(t-\tau)].$$

Due to the fact that ML models do not process matrices as input, let us rewrite inputs of channel predictor, i.e., (4), into vector form as $X = [x_1, x_2, \ldots, \ldots, x_t, \ldots]$, where $X \in \mathbb{R}$ as we utilize real-valued LSTM network. As alluded to earlier, LSTM network has recurrent connections in the hidden layer’s neurons. At each time step, LSTM cell generates output $y$ with respect to corresponding input $x$ and the hidden states sent from the previous time step. In Fig. 2, an LSTM network composed of one input layer, two hidden layers, and an output layer is shown. Specifically, a data vector $x$ goes through the input feedforward layer to get $d^{(1)}$, representing the activation for memory cells in the first hidden layer. With reference to Fig. 2, at time step $t - 1$, the memory cell at the $t^{th}$ hidden layer, where $l = 1, \ldots, L$, generates two internal states $s_{l-1}$ and $c_{l-1}$ called as long-term and short-term states, respectively. $c_{l-1}$ retains historical information after throwing out unnecessary information through forget gate, and aggregates new information selected by input gates. The integrated information is passed as long-term state to next LSTM cell for time step $t + 1$. Through this way, prediction at each time step is influenced by the past information and current external input. The input $d^{(l)}$ along with $s_{l-1}$ are fed into fully connected (FC) layers to generate activation vector for each gate as

$$f^{(l)} = \sigma_s \left(W_f^{(l)} d^{(l)} + W_i^{(l)} s_{l-1} + b_f^{(l)} \right),$$

$$i^{(l)} = \sigma_s \left(W_i^{(l)} d^{(l)} + W_i^{(l)} s_{l-1} + b_i^{(l)} \right),$$

$$o^{(l)} = \sigma_s \left(W_o^{(l)} d^{(l)} + W_o^{(l)} s_{l-1} + b_o^{(l)} \right),$$

where $\sigma_s = \frac{1}{1 + e^{-x}}$ is sigmoid activation function, $W, V$ and $b$ are the weights and bias for the FC layers. The subscripts $j, i, o$ are showing the representation for forget, input, and output gates, respectively. The output for the memory cell is obtained as

$$s^{(l)} = d^{(l+1)} = o^{(l)} \otimes \sigma_h \left(c^{(l)} \right),$$

where $\sigma_h = \frac{e^{2x} - 1}{e^{2x} + 1}$ is the hyperbolic tangent function.

$$c^{(l)} = f^{(l)} \otimes c^{(l-1)} + i^{(l)} \otimes g^{(l)},$$

where

$$g^{(l)} = \sigma_h \left(W_g^{(l)} d^{(l)} + V_g^{(l)} s^{(l)} + b_g^{(l)} \right).$$

Similarly, operating through each hidden layer, final output $y_t$ can be predicted. By applying the post-processing steps,

---

7CsiFB-PNet can work in TDD assuming channel reciprocity.

8In our analysis, we have also used compressed channel, i.e., $H^Q$, to train the channel predictor, as followed in [46].
i.e., combining real and imaginary parts, and converting the predicted vector to matrix representation, $y_t$ can be written into a predicted mMIMO channel matrix as

$$
\tilde{H}_{\text{UE}}(t) = \begin{bmatrix}
\tilde{h}_{11}(t) & \cdots & \tilde{h}_{1N_{r}}(t) \\
\tilde{h}_{21}(t) & \cdots & \tilde{h}_{2N_{r}}(t) \\
\vdots & \ddots & \vdots \\
\tilde{h}_{N_{t}1}(t) & \cdots & \tilde{h}_{N_{t}N_{r}}(t)
\end{bmatrix},
$$

(9)

where $\tilde{h}_{n_{t}n_{r}}(t) \in \mathbb{C}^{1 \times 1}$ denotes the predicted channel for the $n_{t}^{th}$ transmit and $n_{r}^{th}$ receive antennas at time $t$.

The prediction of the LSTM network is based on the weights $W, V$, and bias $b$, and the applied activation functions to the linear equation (combination of input and weight values). In summary, $W, V,$ and $b$ are optimized for the connection between the output of a neuron in the predecessor layer and the input of a neuron in the successor layer. The output of a neuron is calculated by applying the activation function (introduces non-linearity) to the linear equation.

To train the LSTM network, first, the LSTM architecture is selected, i.e., the number of input, output, and hidden layers, and the weights are initialized. Then, the hyperparameters, e.g., the learning rate and optimization algorithm, are chosen. Finally, the channel values, $[\tilde{H}_{\text{UE}}(t-1), \tilde{H}_{\text{UE}}(t-2), \ldots, \tilde{H}_{\text{UE}}(t-\tau)]$, given in (4), are fed at input of LSTM network, along with corresponding output label, $\hat{H}_{\text{UE}}(t)$. Consequently, the LSTM network processes a batch of samples depending on the batch size, $\beta_{\text{size}}$, and compares the predicted value, $\hat{H}_{\text{UE}}(t)$, with the true label, $\tilde{H}_{\text{UE}}(t)$. The weights are updated through backpropagation, which is repeated until the cost function is minimized. We use mean-squared-error (MSE) as a cost function, calculated as

$$
\text{MSE} = \min_{W, V, b} \frac{1}{T} \sum_{t=1}^{T} \left\| \tilde{H}_{\text{UE}}(t) - \hat{H}_{\text{UE}}(t) \right\|^2.
$$

(10)

Once the LSTM network is trained, it can predict mMIMO channel, represented in (9). In the following, we explain CsiFB-PNet by exploiting trained channel predictor at the UE.

**B. CSI Feedback**

CsiFB-PNet consists of the following phases: assessment, initialization, prediction, estimation, compression, feedback and recovery of compressed CSI. Below, we describe these phases in more detail.

1) **Assessment Phase:** To utilize CsiFB-PNet, the two network entities exchange a few messages: 1) gNB and UE perform a handshake to assess available capabilities, for instance, support of ML algorithm 2) both agree to use channel prediction 3) UE agrees to aggregate past CSI realizations to train the channel predictor 4) the length of the data-set (to be acquired in the initialization phase) is decided for the training of channel predictor at UE. The assessment phase is pictorially represented in Fig. 3. Once such messages are exchanged, then the gNB and UE perform the initialization phase, which is summarized below.

2) **Initialization Phase:** The UE aggregates the CSI realizations by exploiting CSI-RS sent by the gNB. Consequently, a set of past CSI realizations is established at the UE. Let us denote the set of past CSI realizations as $S = \{\tilde{H}(t), \tilde{H}(t-1), \ldots, \tilde{H}(t-\tau)\}$.
The feedback and recovery of the compressed channel is categorized into two scenarios. In the following, we explain each scenario.

- In the first scenario, we consider the case when the UE has to feedback. In short, when $\hat{H}_{\text{UE}}(t) \neq \tilde{H}_{\text{UE}}(t)$. Therefore, the UE feedback the compressed update, i.e., $\tilde{H}_{p}(t)$. Correspondingly, the acquired channel (or the recovered channel) at the gNB can be calculated as

$$\tilde{H}_{gNB}(t) = \tilde{H}_{gNB}(t) - Q_p(\tilde{H}_{\text{UE}}(t) - \hat{H}_{\text{UE}}(t)),$$

which is simply the difference between the predicted channel at the gNB and feedback from the UE. As the channel prediction is employed at both ends; hence, error may add at both ends if the prediction is bad. To overcome this problem, we propose hybrid framework, i.e., the conventional approach and the proposed approach are used jointly. Therefore, in CsiFB-PNet, before reporting $\tilde{H}_{p}(t)$, the UE computes the error, $\Lambda$, and cosine similarity, $\rho$, between what will be acquired at gNB, i.e., $\tilde{H}_{gNB}(t)$, and the estimated channel at UE, $\hat{H}_{\text{UE}}(t)$. Importantly, in the conventional approach, $\tilde{H}_{gNB}(t) = \tilde{H}_{c}(t)$. In the case when error in the proposed approach, $\Lambda_p$, is less than the error in the conventional approach, $\Lambda_c$, as well as cosine similarity in the proposed approach, $\rho_p$, is higher than the cosine similarity in the conventional approach, $\rho_c$, then the UE will feedback $\tilde{H}_{p}(t)$. Otherwise, UE reports $\tilde{H}_{c}(t)$ with a flag indicating the use of conventional approach. Besides, it is made sure that the gNB and UE operate on the same data to make identical predictions. For instance, in the beginning, both rely on (3). Then (16) in the case of feedback, otherwise both use $\tilde{H}$.

The pseudo-code of CsiFB-PNet is given in Algorithm 1. Briefly, in the first step of Algorithm 1, the training data is accumulated at the UE by exploiting CSI-RS sent by the gNB, and channel predictor is trained. At step-2, trained model is reported to gNB, and channel is predicted on both sides. Later, the channel is acquired at the gNB, which is summarized in step-3 and step-4. CsiFB-PNet needs to be supported by the standard as the invention requires the establishment of a protocol between the gNB and UE, which is currently not part of the 3GPP specification. To this end, different prediction algorithms can be tabulated and standardized. Furthermore, the possible standardization aspects include algorithm(s), memory, message exchanges, etc.
Algorithm 1: CsiFB-PNet

Input: $\hat{H}$
Output: $\hat{H}_{gNB}(t)$

// Step-1: Aggregate training data at UE, create a set $S$, and reverse its order w.r.t. time, i.e., most fresh entry will go at the end.
1 for $s = 1$ to $S - 1$ do
2 \quad Train the channel predictor using the process explained in Section IV-A.
3 // Step-2: Report the trained model to gNB. Also, predict the channel at time $t$ at gNB and UE by exploiting past channel realizations $\hat{H}^Q$.
4 // Step-3: Estimate the channel, $\hat{H}_{UE}(t)$, at UE at time $t$.
5 // Step-4: Calculate $\Lambda_p, \Lambda_c, \rho_p$, and $\rho_c$ and acquire $\hat{H}_{gNB}(t)$.
6 if $\Lambda_p < \Lambda_c$ & $\rho_p > \rho_c$ then
7 \quad Obtain $\hat{H}_{gNB}(t)$ using Equation (16).
8 else
9 \quad $\hat{H}_{gNB}(t) = \hat{H}_{c}^Q(t)$

C. Remarks on CsiFB-PNet

There are threefold advantages of using CsiFB-PNet over the conventional approach. First, the errors caused by compression can be smaller in $\hat{H}_{gNB}(t)$ than in $\hat{H}_{c}^Q(t)$ due to having low amplitude entries in $\hat{H}_{gNB}(t)$, thanks to prediction. Therefore, a more accurate version of the true channel can be acquired at the gNB. The second benefit can be attained in terms of quantization bits, $B_Q$, required to send feedback from the UE. The better the prediction at the UE, the fewer bits would be needed to send the feedback. Thus, CsiFB-PNet can significantly reduce feedback overhead. Lastly, if the prediction at the UE is perfect, then feedback is not necessary; hence, feedback-related overhead is completely eliminated. This benefit can be attained in the environment where the UEs are static (e.g., in a stadium, or moving at a predictable speed). This is because when the UEs are stationary, the variations in the channel tend to be very small; hence, prediction can be more accurate.

If the quantization functions, $Q_p(\cdot)$ and $Q_c(\cdot)$, are using an infinite amount of bits to send the feedback, i.e., $B_Q = \infty$, then there is no advantage of using CsiFB-PNet. In short, quantization followed in the conventional approach and in CsiFB-PNet will be same, i.e., $Q_c(x) = Q_p(x) = x$, where $x$ is the input data for quantization. In other words, we can say that there is no compression while sending feedback. Hence, the real benefits can be acquired when the channel is highly compressed, which is true in practical wireless environment, i.e., type-I and type-II CSI feedback used in 3GPP are highly compressed [12].

V. NUMERICAL RESULTS

In this section, we verify the effectiveness of CsiFB-PNet. To this end, first communication environment is presented. Then we address distribution of the data-set for CsiFB-PNet. We demonstrate that CsiFB-PNet outperforms the existing feedback methods. The performance is validated using various evaluation parameters including normalized MSE (NMSE), $\Omega_{nMSE}$, precoding gain, $\Gamma$, and cosine similarity, $\rho$.

A. Communication Environment and Data-Set

We consider a mMIMO communication environment, where a UE is moving on different tracks, as indicated in Fig. 4. The network entities and one of the tracks followed during the measurement campaign are given in Fig. 5. The measurement campaign was performed at Nokia Bell-Labs campus. The UE had been moved on the tracks with a speed of 3 to 5 kmph. The gNB is equipped with 64 transmit antennas, and the UE has 1 receive antenna. The transmission is done using time-frequency orthogonal pilots and using OFDM waveforms (10 MHz long-term evolution (LTE) numerology, i.e., 600 subcarriers with 15 kHz spacing). The rest of the simulation parameters are summarized in Table I, and further details of the measurement campaign are available in [51].

For the training of channel predictor at the UE, we extracted the data-set of track-1 comprised of 58 seconds. The 80% data-set of track-1 is used for training, 10% for the validation
and remaining for the test. Similarly, 10% data-set of track-2 and track-6 is taken randomly for the test purpose. In the pre-processing steps before training the channel predictor, we extracted the relevant information related to a single subcarrier, and we learned that the channel is highly correlated when compared with different sub-carriers. Therefore, we trained and evaluated the model using a single subcarrier. Unless stated otherwise, the performance of CsiFB-PNet is evaluated on track-2. Numerically, we had a total data-set of track-1, comprised of consecutive 116k data blocks, i.e., \( H(t) | 1 \leq t \leq 116k \), where each row of \( H \) indicates the channel samples captured periodically and columns represent the sample-value of each transmit antenna.

**B. Performance Evaluation Metrics for CsiFB-PNet**

In this subsection, we compare CsiFB-PNet with the benchmark schemes by using various evaluation parameters, which are given below.

1) **NMSE**: To observe the effectiveness of CsiFB-PNet, we use NMSE, denoted by \( \Omega_{\text{nmse}} \), which is the difference between the true channel and the acquired channel at the gNB. Mathematically,

\[
\Omega_{\text{nmse}} = \mathbb{E} \left\{ \frac{\|H - \hat{H}_{\text{gNB}}\|_{\text{FRO}}^2}{\|H\|_{\text{FRO}}^2} \right\},
\]

where \( \hat{H}_{\text{gNB}} \) denotes the acquired channel at the gNB. Importantly, in the case of CsiFB-PNet, acquired channel at the gNB is the one given in Equation (16), whereas in the case of conventional approach, acquired channel at the gNB is compressed estimated channel, i.e., Equation (3). In some results, NMSE is expressed in decibels (dB), calculated as \( 10\log_{10}(\Omega_{\text{nmse}}) \).

2) **Precoding Gain**: As the feedback CSI is used for precoding, to observe precoding gain, we use a simple matched filter precoder. Let us denote the acquired or reconstructed complex channel vector at the gNB as \( \hat{h}_{\text{gNB}} \) and true complex channel vector as \( h \). By using this information, equivalent channel can be represented as

\[
h_{\text{eq}} = \left( \frac{\hat{h}_{\text{gNB}}}{\|\hat{h}_{\text{gNB}}\|_2} \right) \times \left( \frac{h}{\|h\|_2} \right).
\]

By using (18), precoding gain is calculated as

\[
\Gamma = \mathbb{E}\left\{ (h_{\text{eq}})^* \times h_{\text{eq}} \right\}.
\]

3) **Cosine Similarity**: To measure the quality of precoding vector, we consider the cosine similarity of the two vectors, which is calculated as

\[
\rho = \cos \theta = \mathbb{E}\left\{ \frac{(\hat{h}_{\text{gNB}})^* \cdot h}{\|\hat{h}_{\text{gNB}}\|_2 \|h\|_2} \right\}.
\]

**C. Comparison of CsiFB-PNet With Conventional Approach**

Fig. 6 shows the gain of CsiFB-PNet over the conventional approach, where \( \Omega_{\text{nmse}} \) is plotted against the different number
of overhead bits (or say compression levels) used to feedback CSI. The multi-fold advantages of CsiFB-PNet can be seen. The first advantage is when the compression is high, i.e., $B_Q = 2$. It can be seen that CsiFB-PNet provides a gain of approximately 1.00. However, a smaller gain is observed when the feedback overhead is increased. In short, using a massive number of feedback bits brings no advantage, which verifies the remark that $Q_c(\bar{x}) = Q_p(\bar{x}) = \bar{x}$. The feedback followed in 3GPP is highly compressed; therefore, CsiFB-PNet is beneficial. Conversely, when $B_Q = 5$, i.e., low compression, the gain is nearly zero. Nevertheless, CsiFB-PNet brings the advantage of eliminating feedback, which is because the same CSI is already available at the gNB with the aid of a channel predictor.

Besides, it can be seen in Fig. 6 that $\Omega_{\text{nmse}} \approx 0.02$ when $B_Q = 2$. Thus, it is up to the standards to define if the error of this threshold must be feedback or not. For example, if it is decided that $\gamma = 0.02$ is the maximum threshold to not feedback CSI, then it will be considered a perfect prediction in our case. Thus, the UE does not need to provide feedback, as it is already available at the gNB, thanks to the CSI prediction. It is important to highlight that the results plotted in our studies are averaged over both the scenarios, described in Section IV-B6, i.e., perfect and imperfect prediction. In a nutshell, we have concluded from Fig. 6 that the CsiFB-PNet reduces feedback overhead and minimizes compression errors under high compression, and it eliminates feedback when compression is low. Besides, the visual illustration of the reconstructed samples for each antenna is shown in Fig. 7. The observation shows that the reconstruction done by CsiFB-PNet resembles to the true channel.

To verify the robustness of CsiFB-PNet, Fig. 8 reveals the performance using NMSE and precoding gain on track-6. The trend of $\Omega_{\text{nmse}}$, plotted in blue color on the left y-axis, shows that CsiFB-PNet outperforms MLaBE. This is because CsiFB-PNet training on the uncompressed CSI, which is done at the UE. Thus, a more accurate CSI prediction model is obtained at the UE, improving performance. Numerically, we observed a performance gain of approximately 3.25 dB using CsiFB-PNet when the feedback CSI is highly compressed, i.e., $B_Q = 2$.

D. Comparison of One-Sided vs. Two-Sided ML Model

Fig. 9 reveals the performance of CsiFB-PNet against using the compressed CSI on both ends to train the channel predictor, i.e., two-sided ML model, referred as MLaBE [46]. The results show that CsiFB-PNet outperforms MLaBE. This is because of CsiFB-PNet training on the uncompressed CSI, which is done at the UE. Thus, a more accurate CSI prediction model is obtained at the UE, improving performance. Numerically, we observed a performance gain of approximately 3.25 dB using CsiFB-PNet when the feedback CSI is highly compressed, i.e., $B_Q = 2$.

E. Comparison of CsiFB-PNet With Autoencoder

To compare CsiFB-PNet with state-of-the-art, i.e., autoencoder-based CSI feedback, called as CsiNet in [9], Fig. 10 shows the comparison in terms of NMSE and cosine

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
Fig. 9. A comparison between CsiFB-PNet and MLaBE.

Fig. 10. Comparison of CsiFB-PNet with CsiNet by using 3GPP channel model [9], [56].

We used a pre-trained CsiNet [56] to have a fair comparison. Accordingly, we have trained CsiFB-PNet on the same data-set utilized in [56]. The data-set is generated using MATLAB 5G toolbox function `nrCDLChannel`, composed of 15k samples. To get further details of the data-set and pre-trained CsiNet, please refer to [56]. We compare the performance of CsiNet and CsiFB-PNet by using different overhead bits used to compress the CSI. In Fig. 10, we plot NMSE and cosine similarity between the recovered and the true channel. NMSE is plotted on the left y-axis in blue color, and cosine similarity is plotted on the right y-axis in red color. It is evident from the results that CsiFB-PNet recovers the CSI using only 2 overhead bits, and CsiNet requires 6 bits. The rationale is that CsiFB-PNet compresses the update function (see Equation (14)); hence, less quantization noise. A negligible NMSE shows that feedback from the UE can be completely eliminated as the prediction is aligned with the estimated channel at UE. On the other hand, CsiNet compresses the generated codewords; therefore, the decoder network cannot recover the CSI. During the simulations, it was observed that CsiNet performs similarly to CsiFB-PNet without compressing the codewords. Numerically, cosine similarity of CsiNet and CsiFB-PNet is observed same, i.e., 0.99, and NMSE is −44 dB and −32 dB, respectively. The results corroborate that CsiFB-PNet outperforms CsiNet.

In Table II, a comprehensive comparison is given by using the data-set from the measurement campaign (see Section V-A). The best results are presented in bold font. CsiNet is retrained using the data-set of track-1, where training parameters follow their default setting in [56] and transmitted codewords are not compressed. It can be observed that CsiFB-PNet outperforms benchmark schemes. For instance, CsiFB-PNet obtains the lowest NMSE values and beats benchmark schemes at all compression levels. Compared to [46], CsiFB-PNet also provides significant gains due to utilizing uncompressed CSI at the UE for training of channel predictor. When the compression is reduced, benchmark schemes give similar results to CsiFB-PNet, showing that they require more overhead. Further, CsiNet performs better than the conventional method when a low amount of overhead is used, and conventional performs better in high overhead bits. The reason for the worst performance of CsiNet in high overhead bits is that it is not well trained on the measurement campaign, and perhaps more effort is required to train CsiNet; nevertheless, even well-trained CsiNet has lower performance in comparison to CsiFB-PNet (see Fig. 10). In a nutshell, we have learned from the presented results that CsiFB-PNet gives the best performance with a low overhead cost.

Finally, we report some results in terms of the size of data-set used to train the ML models. Specifically, the performance is observed by using Root-MSE (RMSE) as an evaluation parameter, which is calculated between $\hat{H}$ and $\tilde{H}$. The observations showed that CsiFB-PNet can learn future CSI realizations by using minimum amount of data. For example, the observed RMSE of the channel predictor was approximately 0.02 by using only $D_{\text{length}} = 2500$. In comparison to CsiNet, CsiFB-PNet resulted in reducing $\Omega_{\text{rmse}}$ by 88% in the case of

$^{9}$In CsiNet, generated codewords by the encoder are compressed. Whereas in CsiFB-PNet, an update function, i.e., the difference between prediction and estimation, is compressed.

$^{10}$Here the results are written with respect to $B_Q = 2$ for CsiFB-PNet.
$D_{\text{length}} = 2500$. Further, it is also observed in the simulations that by using $D_{\text{length}} = 15000$, CsiFB-PNet gives $\Omega_{\text{mrmse}} = 2 \times 10^{-5}$ whereas CsiNet has approximately $12 \times 10^{-4}$, where the transmitted codewords by the CsiNet are not compressed.

F. Computational Complexity of CsiFB-PNet vs. Autoencoder

To measure the complexity of CsiFB-PNet, we calculate the number of complex multiplications required for the training and prediction of channel predictor. In the case of simple RNN, i.e., without LSTM memory cells, to get a single-step prediction of mMIMO channel, the hidden layer has to perform $I \times J$ complex multiplications, where $I$ and $J$ are the total number of neurons of the input and the hidden layer, respectively. We used two hidden layers; thereby, both hidden layers have to perform $J \times J$ complex multiplications. The complex multiplications at the output layer are $J \times K$, where $K$ are the total output neurons. Hence, the total complex multiplications are

$$P_{\text{rnn}} = J \times (I + J + K).$$  \hfill (21)

The total number of input neurons, $I$, depends on the number of mMIMO sub-channels, recurrent components, and delayed channel realizations. Hence, $I$ is calculated as

$$I = N_F \times N_t + \tau \times (N_F \times N_t) + N_F \times N_t \quad \text{Recurrent}$$

$$= (\tau + 2) \times N_F \times N_t. \quad \hfill (22)$$

Conversely, the output layer’s neurons are equivalent to mMIMO sub-channels; therefore, $K = N_F \times N_t$. By substituting Equation (22) and $K = N_F \times N_t$ into Equation (21), $P_{\text{rnn}}$ can be written in the simplified form as

$$P_{\text{rnn}} = J \times \left[ (\tau + 2) \times N_F \times N_t + J + N_F \times N_t \right]$$

$$= J \times \left[ N_F \times N_t \left( \tau + 3 + \frac{J}{N_F \times N_t} \right) \right]. \quad \hfill (23)$$

With reference to Equations (5)-(7), the number of parameters for a shallow LSTM are calculated as

$$P_{\text{lstm}} = 4(I \times J + J + J) + J \times K + K. \quad \hfill (24)$$

Similarly, for a deep LSTM, parameters can be calculated as

$$P_{\text{dlstm}} = 4 \left( I \times J^{(1)} + J^{(1)} \times J^{(1)} + J^{(1)} \right)$$

$$+ \sum_{l=2}^{L} 4 \left( J^{(l-1)} \times J^{(l)} + J^{(l)} \times J^{(l)} + J^{(l)} \right)$$

$$+ J^{(L)} \times K + K. \quad \hfill (25)$$

The number of hidden layers and their neurons depend on the optimization problem. Generally, the same number of $J$ is used in all hidden layers. Let us denote the size of a mMIMO configuration by $M = N_F \times N_t$ and the scale of the channel predictor by $C = 2\tau J$. Then the one-step prediction complexity of the RNN channel predictor is $O(MC)$. The training complexity depends on the number of training examples (samples), denoted by $N_e$, and the number of training epochs, represented by $N_e$. Therefore, the training complexity is $O(MCN_eN_e)$. Likewise, the complexity of the LSTM-based channel predictor is $O(P_{\text{lstm}})$.

Further, the number of floating-point operations for single-step prediction of LSTM network is 8.4 M, which are very few compared to state-of-the-art, i.e., CsiNet (58.52 M) [30]. Furthermore, CsiNet comprises 32 layers, and CsiFB-PNet comprises 4 layers. This comparison shows that a bigger NN architecture is required to design CsiNet and, correspondingly, resources to train it. Besides, training the LSTM network on track-1 of the measurement campaign took only 52 minutes. The code is executed on TensorFlow 2.3.0, Windows 10 platform running on a core i5 laptop with eight logical processors, clocked at 1.70 GHz with 8 GB of installed memory (RAM).

VI. CONCLUSION AND FUTURE OUTLOOKS

Motivated by the inaccurate CSI acquisition at the gNB due to compression, this paper addressed an ML-based mMIMO CSI feedback mechanism named CsiFB-PNet. The presented results corroborated the validity of CsiFB-PNet compared to state-of-the-art solutions. Specifically, the evaluated results on empirical data and the 3GPP channel model showed that CsiFB-PNet can reduce OTA overhead and enhance CSI feedback accuracy. We have learned from the results, evaluated using empirical data, that in a high-compression scenario, CsiFB-PNet brings huge gains while requiring minimum OTA overhead bits. Besides, we have concluded that the underlying channel model of the empirical data is evolving as an AR process of order 1, justifying the use of $\tau = 2$ (given in Table I) to make the one-step-ahead prediction. Furthermore, the feedback from the UE is not necessary for a 3GPP channel model as the prediction at the UE is approximately the same as the estimated channel. Numerically, we observed a gain of approximately 3.25 dB in CsiFB-PNet compared to training the ML model using compressed CSI. Thus, exploiting ML training only at the UE brings tremendous benefits.

Future works include evaluating the performance of CsiFB-PNet exploiting 3GPP CSI reporting schemes, i.e., type-I and type-II CSI feedback. Also, CsiFB-PNet can be extended towards multi-step channel prediction to reduce the overhead further. Thus, evaluating an AI/ML model against the training complexity of multi-step channel prediction can be one of the possible future directions.

REFERENCES

[1] 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, Feb. 2014.

[2] T. L. Marzetta, “Noncooperative cellular wireless with unlimited numbers of base station antennas,” IEEE Trans. Wireless Commun., vol. 9, no. 11, pp. 3590–3600, Nov. 2010.

[3] J. Zhang, C.-K. Wen, S. Jin, X. Gao, and K.-K. Wong, “On capacity of large-scale MIMO multiple access channels with distributed sets of correlated antennas,” IEEE J. Sel. Areas Commun., vol. 31, no. 2, pp. 133–148, Feb. 2013.

[4] J. Kazemitabar and H. Jalafkhani, “Multiuser interference cancellation and detection for users with more than two transmit antennas,” IEEE Trans. Commun., vol. 56, no. 4, pp. 574–583, Apr. 2008.

[5] F. Björnson, J. Hoydis, and L. Sanguinetti, “Massive MIMO has unlimited capacity,” IEEE Trans. Wireless Commun., vol. 17, no. 1, pp. 574–590, Jan. 2018.
[6] T. O’Shea and J. Hoydis, “An introduction to deep learning for the physical layer,” IEEE Trans. Cogn. Commun. Netw., vol. 3, no. 4, pp. 563–575, Dec. 2017.

[7] M. K. Shehzad, L. Rose, M. M. Butt, I. Z. Kovacs, M. Assaad, and M. Guizani, “Artificial intelligence for 6G networks: Technology advancement and standardization,” IEEE Veh. Technol. Mag., vol. 17, no. 3, pp. 16–25, Sep. 2022.

[8] A. Haydari and Y. Yilmaz, “Deep reinforcement learning for intelligent transportation systems: A survey,” IEEE Trans. Intell. Transp. Syst., vol. 23, no. 1, pp. 1–32, Jan. 2022.

[9] C.-K. Wen, W.-T. Shih, and S. Jin, “Deep learning for massive MIMO CSI feedback,” IEEE Wireless Commun., vol. 7, no. 5, pp. 748–751, Oct. 2020.

[10] M. A. Cheema, M. K. Shehzad, H. K. Qureshi, S. A. Hassan, and H. Jung, “A drone-aided blockchain-based smart vehicular network,” IEEE Trans. Intell. Transp. Syst., vol. 22, no. 7, pp. 4160–4170, Jul. 2021.

[11] H. Shirani-Mehr and G. Caire, “Channel state feedback schemes for multiuser MIMO-OFDM downlink,” IEEE Trans. Commun., vol. 57, no. 9, pp. 2713–2723, Sep. 2009.

[12] “Study on new radio access technology physical layer aspects; (Release 14),” 3GPP, Sophia Antipolis, France, Rep. TR 38.802, Sep. 2017. [Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3066.

[13] D. J. Love, R. W. Heath, W. Sanipatch, and M. L. Honig, “What is the value of limited feedback for MIMO channels?” IEEE Commun. Mag., vol. 42, no. 10, pp. 54–59, Oct. 2004.

[14] M. B. Khalilisari, Y. Song, T. Yang, and G. Caire, “FDD massive MIMO channel training: Optimal rate-distortion bounds and the spectral efficiency of ‘one-shot’ schemes,” IEEE Trans. Wireless Commun., vol. 22, no. 9, pp. 6018–6032, Sep. 2023.

[15] X. Rao and V. K. Lau, “Distributed compressive CSIT estimation and feedback for FDD multi-user massive MIMO systems,” IEEE Signal Process. Lett., vol. 62, no. 12, pp. 3261–3271, Jun. 2014.

[16] P.-H. Kuo, H. Kung, and P.-A. Ting, “Compressive sensing based channel feedback protocols for spatially-correlated massive antenna arrays,” in Proc. IEEE Wireless Commun. Netw. Conf. (WCNC), 2012, pp. 492–497.

[17] P. Cheng and Z. Chen, “Multidimensional compressive sensing based analog CSI feedback for massive MIMO-OFDM systems,” in Proc. IEEE 89th Veh. Technol. Conf. (VTC2014-Fall), 2014, pp. 1–6.

[18] B. Lee, J. Choi, J.-Y. Seol, D. J. Love, and B. Shim, “Antenna grouping based feedback compression for FDD-based massive MIMO systems,” IEEE Trans. Commun., vol. 63, no. 9, pp. 3261–3274, Sep. 2015.

[19] L. Tucker, “Some mathematical notes on three-mode factor analysis,” in Proc. Mathematics, Psychology, and Philosophy of Science, A J. Issued Courant Inst. Math. Sci., vol. 27, no. 1, pp. 200–204, Jan. 2023.

[20] D. L. Donoho, A. Maleki, and A. Montanari, “Message-passing algorithms for sparse signal recovery,” in Proc. IEEE 79th Veh. Technol. Conf. (VTC Spring), 2014, pp. 1549–1554, Apr. 2023.

[21] Y. Sun, W. Xu, L. Liang, N. Wang, G. Y. Li, and X. You, “A lightweight deep network for efficient CSI feedback in massive MIMO systems,” IEEE Wireless Commun. Lett., vol. 10, no. 8, pp. 1840–1844, Aug. 2021.

[22] Y. Sun, W. Xu, L. Fan, G. Y. Li, and G. K. Karagiannidis, “AnNet: An efficient deep learning approach for feedback compression of estimated CSI in massive MIMO systems,” IEEE Wireless Commun. Lett., vol. 9, no. 12, pp. 2192–2196, Dec. 2020.

[23] B. Zhang, H. Li, X. Liang, X. Gu, and L. Zhang, “Multi-task training approach for CSI feedback in massive MIMO systems,” IEEE Commun. Lett., vol. 27, no. 1, pp. 200–204, Jan. 2023.

[24] J. Guo, C.-K. Wen, and S. Jin, “Deep data hiding-based CSI feedback overload elimination: An initial investigation,” in Proc. IEEE Int. Conf. Commun. (ICC), 2022, pp. 718–722, Mar. 2023.

[25] J. Shen, X. Liang, X. Gu, and L. Zhang, “Clustering algorithm-based quantization method for massive MIMO CSI feedback,” IEEE Wireless Commun. Lett., vol. 11, no. 10, pp. 2155–2159, Oct. 2022.

[26] S. Mourya, S. Amuru, and K. K. Kuchi, “A spatially separable attention mechanism for massive MIMO CSI feedback,” IEEE Wireless Commun. Lett., vol. 12, no. 1, pp. 40–44, Jan. 2023.

[27] J. Cui, Q. Hu, T. Fan, W. Guo, and S. Wang, “IALNet: An integration attention lightweight neural network for massive MIMO CSI feedback,” IEEE Wireless Commun. Lett., early access, Feb. 28, 2023, doi: 10.1109/LWC.2023.3249739.

[28] Z. Yin, W. Xu, R. Xie, S. Zhang, D. W. K. Ng, and X. You, “Deep CSI compression for massive MIMO: A self-information model-driven neural network,” IEEE Trans. Wireless Commun., vol. 21, no. 10, pp. 8872–8886, Oct. 2022.

[29] Y. Zhang, X. Zhang, and Y. Liu, “Model-driven lightweight network for CSI feedback in time-varying massive MIMO systems,” in Proc. IEEE Int. Conf. Commun. (ICC), 2023, pp. 4961–4966.

[30] M. K. Shehzad, L. Rose, and M. Assaad, “Dealing with CSI compression to reduce losses and overhead: An artificial intelligence approach,” in Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops), 2021, pp. 1–6.

[31] C. Min, N. Chang, J. Cha, and J. Kang, “MIMO-OFDM downlink channel prediction for IEEE802.16e systems using Kalman filter,” in Proc. IEEE Wireless Commun. Netw. Conf. (WCNC), 2007, pp. 942–946.

[32] R. O. Adegun, P. D. Teal, and P. A. Dmochowski, “Parametric channel prediction for narrowband MIMO systems using polarized antenna arrays,” in Proc. IEEE 79th Veh. Technol. Conf. (VTC Spring), 2014, pp. 1–5.

[33] K. Baddour and N. Beaulieu, “Autoregressive modeling for fading channel simulation,” in Proc. IEEE Wireless Commun. Lett. (IEEE TWC), vol. 4, no. 4, pp. 1650–1662, Jul. 2005.

[34] M. K. Shehzad, L. Rose, and M. Assaad, “A novel algorithm to report CSI in MIMO-based wireless networks,” in Proc. IEEE Int. Conf. Commun. (ICC), 2021, pp. 1–6.

[35] M. K. Shehzad, L. Rose, S. Wesemann, and M. Assaad, “ML-based massive MIMO channel prediction: Does it work on real-world data?” IEEE Wireless Commun. Lett., vol. 11, no. 4, pp. 811–815, Apr. 2022.

[36] W. Jiang and H. D. Schotten, “Deep learning for fading channel prediction,” IEEE Open J. Commun. Soc., vol. 1, pp. 320–332, 2020.
M. Karam Shehzad (Member, IEEE) received the B.Eng. degree (Hons.) in electrical and electronic engineering from the University of Bradford, Bradford, U.K., in 2016, the M.S. degree in electrical engineering from the National University of Sciences and Technology (NUST), Islamabad, Pakistan, in 2019, and the Ph.D. degree in networks, information and communication sciences from the CentraleSupélec, University of Paris-Saclay, Paris, France, in 2023. He is a Senior Research Specialist with Nokia Standards, Paris, France. He worked with the Huawei France Research Center, Sigfox, and Thales Communications and Security, implementing the first IoT dedicated network, studying MIMO-related problems, and contributing to several standards. He is currently an ITU-R and ETSI delegate investigating telecommunication coexistence issues and dynamic spectrum allocation. His interests span from the field of AI-ML applied to channel acquisition, to physical layer security and game theory. Furthermore, he is a Lead Editor of the Internet of Things Series of IEEE Communications Magazine.

Mohamad Assaad (Senior Member, IEEE) received the M.Sc. and Ph.D. degrees in telecommunications from Telecom ParisTech, Paris, France, in 2002 and 2006, respectively. Since 2006, he has been with the Telecommunications Department, Centrale-supélec, University of Paris-Saclay, where he is currently a Professor. He is also a Researcher with the Laboratoire des Signaux et systèmes (CNRS) and was the 5G Chair Holder from 2017 to 2021. He has coauthored one book and more than 150 publications in journals and conference proceedings. His research interests include 5G and 6G systems, fundamental networking aspects of wireless systems, age of information and semantic communications, resource optimization, and machine learning in wireless networks. He has served as a TPC member or the TPC co-chair for various top-tier international conferences, including the TPC Co-Chair for IEEE WCNC’21 and IEEE Globecom’20 Mobile, and the Wireless Networks Symposium Co-Chair. He has also served as an editor for several international journals, including an Editor for the IEEE WIRELESS COMMUNICATIONS LETTERS from 2017 to 2022 and a Guest Editor for the IEEE TRANSACTIONS ON NETWORK SCIENCE AND ENGINEERING. He has also given in the past successful tutorials on topics related to 5G systems, and Age of Information at various IEEE Conferences.

Syed Ali Hassan (Senior Member, IEEE) received the B.E. degree in electrical engineering from NUST, Pakistan, in 2004, the first M.S. degree in Electrical Engineering from the University of Stuttgart, Germany, in 2007, the second M.S. degree in mathematics from Georgia Tech in 2011, and the Ph.D. degree in electrical engineering from the Georgia Institute of Technology, Atlanta, USA, in 2011. He is currently working as a Professor with SEECS, NUST, where he is the Director of the Information Processing and Transmission Lab, which focuses on various aspects of theoretical communications. He was a Visiting Professor with Georgia Tech in Fall 2017. He also held industry positions, in Cisco Systems Inc. CA, USA, and the Center for Advanced Research in Engineering, Islamabad, Pakistan. His broader area of research is signal processing for communications with a focus on cooperative communications for wireless networks, stochastic modeling, estimation and detection theory, and smart grid communications.