POLITECNICO DI TORINO
Repository ISTITUZIONALE

A Novel Deep Learning Approach to CSI Feedback Reporting for NR 5G Cellular Systems

Original
A Novel Deep Learning Approach to CSI Feedback Reporting for NR 5G Cellular Systems / Zimaglia, Elisa; Riviello, Daniel G.; Garello, Roberto; Fantini, Roberto. - ELETTRONICO. - (2020), pp. 47-52. (Intervento presentato al convegno 2020 IEEE Microwave Theory and Techniques in Wireless Communications (MTTW) tenutosi a Riga, Latvia nel October 2020) [10.1109/MTTW51045.2020.9245055].

Availability:
This version is available at: 11583/2854076 since: 2020-11-28T16:59:07Z

Publisher:
Institute of Electrical and Electronics Engineers Inc.

Published
DOI:10.1109/MTTW51045.2020.9245055

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright

(Article begins on next page)

12 October 2024

A Novel Deep Learning Approach to CSI Feedback Reporting for NR 5G Cellular Systems

Elisa Zimaglia*, Daniel G. Riviello†, Roberto Garello†, Roberto Fantini*
*TIM S.p.A., Torino, Italy
{elisa.zimaglia, roberto.fantini}@telecomitalia.it
†Department of Electronics and Telecommunications (DET), Politecnico di Torino, Italy
{daniel.riviello, roberto.garello}@polito.it

***Abstract*—In this paper, we study 5G Channel State Information feedback reporting. We show that a Deep Learning approach based on Convolutional Neural Networks can be used to learn efficient encoding and decoding algorithms. We set up a fully compliant link level 5G-New Radio simulator with clustered delay line channel model and we consider a realistic scenario with multiple transmitting/receiving antenna schemes and noisy downlink channel estimation. Results show that our Deep Learning approach achieves results comparable with traditional methods and can also outperform them in some conditions.**

***Index Terms*—5G, New Radio, Deep Learning, Convolutional Neural Network, CSI reporting.**

I. Introduction

Channel sounding is fundamental to completely exploit the potential gain offered by 5G Massive MIMO [1], [2], [3], [4], [5]. When Frequency Division Duplexing (FDD) solutions are considered, it is important to find an efficient way to provide Channel State Information (CSI) feedback to the transmitter. To reduce feedback overhead, most common solutions work with codebooks or vector quantization, but the process is still quite complex [3], [5]. In this paper we show that Deep Learning (DL) can provide an alternative solution to this important practical problem, achieving performances similar to those of traditional methods, and better under some conditions.

It is known that Machine Learning can be usefully applied to upper layers of mobile networks, for example for Self-Organizing Networks (SONs) [6] and resource management [7], [8]. Recently, many researchers have started to investigate the possible benefits achievable by Deep Learning (DL) application to the physical layer, too [9]. The DL solution proposed in this article is based on the implementation of a convolutional neural network. We start from the work presented in [3] and we extend it to a practical scenario characterized by:

1) 5G-New Radio (NR) fully compliant simulator.
2) Clustered delay channel model.
3) MIMO support, with multi-receiving antenna schemes.
4) Noisy downlink channel estimation.

To study the problem we use a proprietary MATLAB-based New Radio Link Simulator software developed in Telecom Italia laboratories. The simulated Physical Downlink Shared Channel (PDSCH) chain is depicted in Fig. 1. The colored blocks are implemented in C language to speed-up the simulations. The tool provides a bit level processing, fully compliant with the 5G-NR physical layer as described in the 38 series of the 3GPP specifications [10]–[13]. This allows to evaluate the link level performances of 5G-NR point-to-point communications, by considering a gNodeB[1] [14] and a single user equipped with multiple antennas (i.e., SU-MIMO).

The 3GPP clustered delay line (CDL) model is adopted, which is based on the description of the main departure and arrival directions of the signal in the space and the number of clusters corresponds to the number of channel reflections. In particular, we consider the CDL-B profile, that is specific of Non-Line-of-Sight (NLOS) transmissions and can be used for sub 6 GHz frequency, where Line-of-Sight (LOS) is not mandatory.

The paper is organized as follows. In Sec. II we present the Deep Learning model used to study the CSI feedback problem. In Sec. III we discuss the simulation results and we show that the DL approach can outperform the traditional ones. Sec. IV concludes the article.

II. A Deep Learning-based CSI feedback for New Radio

When working in Frequency Division Duplexing (FDD) mode, the user equipment (UE) estimates the downlink Channel State Information and reports it to the base station (BS) through feedback links: this information is exploited at the network side to compute the optimal precoder for MIMO transmission. As remarked in [3], when the number of antennas in transmission and reception considerably increases, the feedback overhead may become excessive. To solve this issue, the authors of [3] propose a DL-based CSI sensing and recovery mechanism, named CsiNet. This is a very innovative proposal since other relevant works treated DL-based approaches for CSI encoding, but none of them considered the recovery stage. The CsiNet model consists of:

- an encoder, that learns a mapping from the channel matrices to the compressed codewords;
- a decoder, that learns the inverse mapping, from the compressed codewords to the original CSI.

Summarizing, the UE applies the encoder function to transform the channel matrices into codewords, while at the BS the decoder is used to recover the original CSI information.

[1]node providing NR user plane and control plane protocol terminations towards the user equipment, and connected via the NG interface to the 5GC.

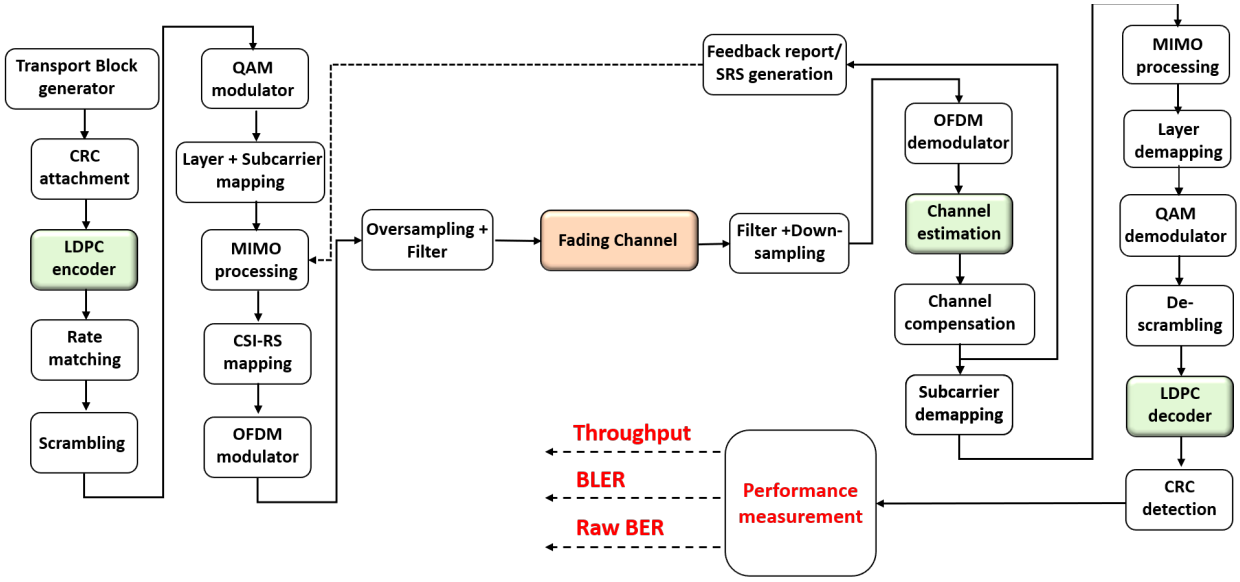


Figure 1: New Radio link-level simulator scheme

In this paper, we introduce a novel DL-based solution for CSI information reduction and recovery in a 5G context. In particular, we implement a deep convolutional neural network (CNN), named NR-CsiNet, which substantially extends the structure of the model presented in [3], but with some variations and new features that aim at generalizing the approach to a wider and more practical range of use cases: first of all, we try to make the model applicable also to multi-receiving antenna scenarios, overcoming the restriction imposed by paper [3], in which the study is limited to single receiving antenna communications; moreover, we consider also the downlink channel estimation, whereas this additional step is beyond the scope of [3].

A similar approach for CSI feedack reporting and information reduction has been adopted in [15], where the authors propose a CSI compression feedback algorithm based on DL, suitable for single-user and multi-user scenarios in massive MIMO systems. Some differences arise w.r.t. our work: first, the simulation environment in [15] is not 5G-NR compliant since the transmitting and receiving chain does not include some 5G processing blocks; secondly, a flat Rayleigh fading channel model is adopted, while in our work we implement a 3GPP CDL model; thirdly, the CNN proposed in [15] presents a different layered structure w.r.t. our NR-CsiNet; finally, results in [15] are presented with a fixed compression ratio ($1/4$), while we provide results with four different compression ratios (from $1/15$ to $1/120$).

For our experiments, we use the Tensorflow's Keras API [16] to build and train our CNN. Inside the link level simulator, presented in Sec. I, we substitute the traditional feedback reporting blocks with our DL-based alternatives.

A. System model

In [3], the authors consider a Multiple-Input-Single-Output (MISO) system with $N \gg 1$ transmitting antennas at the BS and a single receiving antenna at the UE. This paper instead presents a 5G-NR compliant MIMO system, with M antennas at the receiver side. $\widetilde{C}$ indicates the number of subcarriers considered for the OFDM-based system. The signal received at the m-th receiving antenna on the c-th subcarrier can be expressed as

$$y_{m,c} = \mathbf{h}_{m,c}^H \mathbf{v}_c x_c + z_{m,c} \tag{1}$$

where:

- $\mathbf{h}_{m,c}$ is the complex channel vector, of size $N \times 1$, corresponding to the m-th receiving antenna and to the c-th subcarrier;
- $\mathbf{v}_c$, of size $N \times 1$, is the complex precoding vector;
- x_c is the complex data symbol transmitted on the c-th subcarrier;
- $z_{m,c}$ denotes the additive noise on the c-th subcarrier affecting the m-th receiving antenna.

The BS selects the proper precoding vector $\mathbf{v} = \{\mathbf{v}_c : c = 1, \ldots, \widetilde{C}\}$ on the basis of the CSI feedback information $\mathbf{H}_c = [\mathbf{h}_1, \ldots, \mathbf{h}_M] \in \mathbb{C}^{N \times M}$ it receives. Thus, $N \times M \times \widetilde{C}$ feedback values must be sent to the BS and, in some cases, this could be an excessive quantity.

For each c-th subband, the M $y_{m,c}$ received symbols are then combined at the receiver side and the final estimate $\hat{y}_c$ is derived as follows:

$$\hat{y}_c = \mathbf{u}_c^H \mathbf{y}_c \tag{2}$$

where $\mathbf{u}_c$ is an $M \times 1$ combiner and $\mathbf{y}_c = [y_{1,c}, \ldots, y_{M,c}]^T$ is an $M \times 1$ vector including all the received symbols on the c-th subcarrier.

B. Downlink channel estimation

Paper [3] addresses the compression task, proposing a DL-based model which is able to compress and decompress an ideal downlink channel matrix minimizing the reconstruction

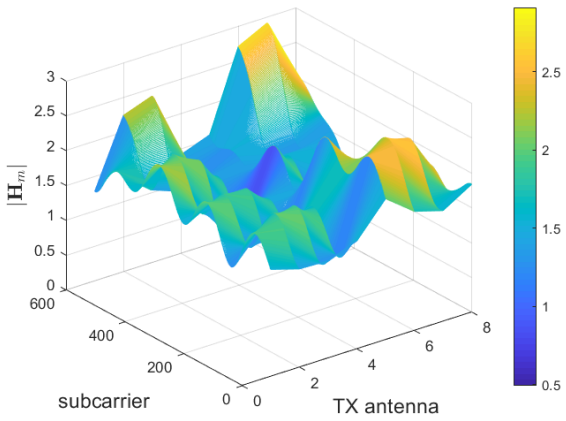


Figure 2: Module of channel matrix $|\mathbf{H}_m|$

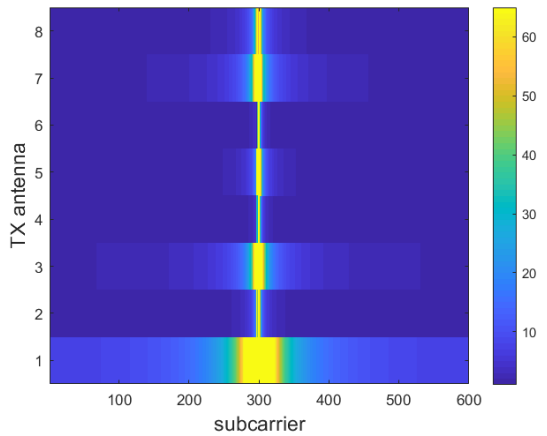


Figure 3: Module of DFT-transofrmed channel matrix $|\widetilde{\mathbf{H}}_m|$

error. With respect to this work, we introduce an important novelty: instead of performing compression and decompression operations on the ideal channel realizations at the output of the CDL channel block, our model learns efficient algorithms which are able to work on noisy channel matrices, estimated from specific reference signals, called Channel State Information Reference Signals (CSI-RS) [17]. This means that our solution addresses also the channel estimation task, removing the noisy contribution to encode and reconstruct the ideal channel matrices with the minimum error. Please note that our NR simulator does not implement the uplink transmission chain and thus, in all the experiments performed, the noise affecting the uplink direction is not taken into account.

For clarity, we have to distinguish two different instances of the channel matrix $\mathbf{H}$:

- an ideal channel matrix $\mathbf{H}^{id}$, which represents the reconstruction target of our NR-CsiNet model and is generated by the CDL module of the software simulator;
- a noisy estimate of the perfect channel matrix $\mathbf{H}^{\text{CSI-RS}}$, which represents the input of the NR-CsiNet model. It is obtained by means of a simple interpolation process operated on the known CSI-RS pilots for each transmitting/receiving antenna pair; in particular, we exploit a MATLAB predefined library to perform a cubic Radial Basis Function (RBF) interpolation on the real and the imaginary parts separately. Varying the SNR level of the simulation, we collect different sets of $\mathbf{H}^{\text{CSI-RS}}$ instances and each of them is used to train a different NR-CsiNet model.

Since all the processing steps described in the following subsection have to be performed on both $\mathbf{H}^{id}$ and $\mathbf{H}^{\text{CSI-RS}}$, we decide to lighten the notation by collapsing the two channel instances in a single matrix $\mathbf{H}$.

C. DFT-based preprocessing of CSI information

To sparsify the channel matrix $\mathbf{H}$ in the angular-delay domai, the authors of [3] includes a 2D-Discrete Fourier Transform (DFT) preprocessing operation. Our model extends this DFT-based preprocessing, repeated for each receiving antenna, as follows:

$$\widetilde{\mathbf{H}}_m = \mathbf{F}_d \mathbf{H}_m \mathbf{F}_a^H \tag{3}$$

where $\mathbf{F}_d$ and $\mathbf{F}_a$ are $\widetilde{C} \times \widetilde{C}$ and $N \times N$ DFT matrices respectively and $\mathbf{H}_m$ is the $\widetilde{C} \times N$ spatial-frequency channel matrix relative to the m-th receiving antenna. In Fig. 2 we provide an example of $\mathbf{H}_m$, with $N = 8$ and $\widetilde{C} = 600$. It results evident that only a small portion of the matrices $\widetilde{\mathbf{H}}_m$ contains significant values, while a consistent portion of it consists of elements close to zero, since the interarrival time between rays lies within a limited interval (Fig. 3). In particular, only the central $C < \widetilde{C}$ columns of $\widetilde{\mathbf{H}}_m$ are relevant, allowing us to remove the remaining portion of the matrix. So, after performing a truncation, $\widetilde{\mathbf{H}}_m$ is reduced to a matrix of size $C \times N$, denoted as $\widetilde{\mathbf{H}}_m^t$. Each matrix $\widetilde{\mathbf{H}}_m^t$ is then brought back in the spatial-frequency domain, by means of an Inverse DFT (IDFT) operation, obtaining $\mathbf{H}_m^t$ matrices. Despite not foreseen by the authors of [3], we decide to introduce this additional step in order to let our CNN work with smoother channel "images", similar to the realization shown in Fig. 2. This is justified by the fact that images characterized by crisp and sharp lines, as in the case of Fig. 3, are generally more difficult to compress and reconstruct. Finally, real and imaginary parts of each truncated matrix $\mathbf{H}_m^t$ are scaled to range $[0, 1]$. In a specular way, a rescaling operation, followed by a DFT and an inverse DFT operation, are performed on the matrices at the output of the decoder.

D. Multiple antennas at the receiving side

Another interesting novelty of our approach with respect to the one proposed in [3] is its applicability to MIMO scenarios. Even if dealing with multiple receiving antennas, our model maintains only two input channels, one for the real and one for the imaginary parts of matrices $\mathbf{H}_m^{\text{CSI-RS},t}$; in other words, matrices relative to distinct receiving antennas are conveyed through the CNN on the same input channels. This approach

allows us to avoid training a different CNN model for each distinct antenna in reception; another considerable benefit, especially in massive MIMO contexts, where $M \gg 1$, is that the number of learnable parameters of the CNN does not increase with the number of receiving antennas (M), preventing from overfitting phenomena. Lastly, it is very likely that our model will be able to exploit the spatial correlation between the different receiving antennas, allowing us to adopt smaller training datasets to produce efficient and generalized network models.

E. NR-CsiNet encoder

Our NR-CsiNet encoder (Fig. 4a) is essentially a deep convolutional neural network, whose input consists of the real and imaginary parts of the truncated matrices $\mathbf{H}_m^{\text{CSI-RS},t}$, where m is the index of the receiving antenna. The T variable introduced in Fig. 4a is $T = 2 \cdot C \cdot N$. The NR-CsiNet encoder adopts the structure of the CsiNet encoder proposed in [3]: the first layer of the CNN is a convolutional layer with $3 \times 3 \times 2$ filters, that generates 2 feature maps. Then, the 2 feature maps are vectorized performing a reshaping operation. Finally, a fully connected layer outputs a vector (codeword) $\mathbf{s}$ of size R for each receiving antenna, so that the comprehensive compression ratio results equal to $K = \frac{M \cdot R}{\widetilde{C} \cdot N}$.

F. NR-CsiNet decoder

The NR-CsiNet decoder (Fig. 4b) is a CNN, which takes as input the codeword $\mathbf{s}$, generated by the encoder and sent to the BS. It utilizes the implementation scheme of the CsiNet decoder presented in [3]: the first layer is a fully connected layer that outputs an initial estimation of the real and imaginary parts of the matrix $\mathbf{H}_m^{id,t}$. Then, these initial estimates are fed into a series of RefineNet units, whose purpose is to progressively refine the reconstruction. Each RefineNet unit consists of an input layer and three convolutional layers with a kernel of size 3×3: the second layer outputs 8 feature maps, the third 16 while the fourth generates the final reconstruction of the $\mathbf{H}_m^{id,t}$ matrices. Exploiting a zero-padding operation, the intermediate feature maps produced inside a RefineNet unit result in having the same size as the input channel matrix ($C \times N$). Moreover, each layer is followed by a Rectified Linear Unit (ReLU) [18], [19] and a Batch Normalization layer [20], [19]. After performing several experiments, we can conclude that for our NR-CsiNet model, as for the CsiNet one [3], adding further RefineNet units would not significantly improve the reconstruction quality, leading instead to a larger complexity. After the series of RefineNet units, a final convolutional layer is followed by a softmax layer [21], [19], which scales the values to the $[0, 1]$ range.

G. Training process

By exploiting the same notation adopted in [3], we can denote the set of trainable parameters as $\mathbf{\Theta} = \{\mathbf{\Theta}_{en}, \mathbf{\Theta}_{de}\}$. The truncated input and output of the NR-CsiNet model for the i-th patch are denoted as $\mathbf{H}_i^{\text{CSI-RS},t}$ and $\hat{\mathbf{H}}_i^t = f(\mathbf{H}_i^{\text{CSI-RS},t}; \mathbf{\Theta}) = f_{de}(f_{en}(\mathbf{H}_i^{\text{CSI-RS},t}; \mathbf{\Theta}_{en}); \mathbf{\Theta}_{de})$ respectively.

The loss function is defined as

$$L(\mathbf{\Theta}) = \frac{1}{T} \sum_{i=1}^{T} ||f(\mathbf{H}_i^{\text{CSI-RS},t}; \mathbf{\Theta}) - \mathbf{H}_i^{id,t}||_2^2 \tag{4}$$

where T denotes the number of training samples and $||.||_2$ is the Euclidean norm operator.

III. Simulation results

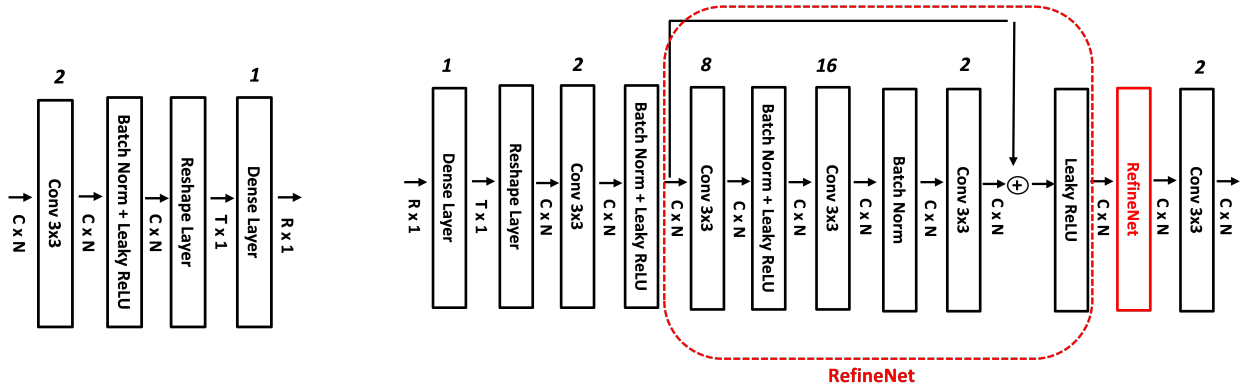
This section presents the results achieved by the NR-CsiNet model described in the previous sections. The NR link level simulator presented in Sec. I is used to carry out simulation trials in order to collect performance statistics. For these simulations, the blocks for CSI feedback definition and reporting are substituted with our CNNs. The precoding block at the transmitter side takes in input the channel matrix at the output of our decoder, reconstructed from the compressed feedback sent back by the UE; a Single Value Decomposition (SVD) operation is then performed and the eigenvector correspondent to the higher eigenvalue of the matrix is selected as precoding vector for the next downlink transmission. At the receiver side, the spatial diversity of the M branches is exploited by implementing a Maximal Ratio Combiner (MRC).

Tab. I collects the main parameters of interest concerning the simulator settings:

Table I: Simulation parameters

NR Simulator parameters	
System bandwidth	5 MHz
Time slot	1 ms
Carrier frequency	3.64 GHz
(Modulation, Coding rate)	(64-QAM, 0.694)
TBS	19968 bit
$\widetilde{C}$	300
Transmission layers	1
Codewords	1
CSI reporting Type	Type I
Channel model	CDL-B
Delay Spread model	Nominal

- a time slot is the time required to transmit a Transport Block of data, which corresponds to the transmission time of 14 OFDM symbols;
- TBS is the Transport Block Size, indicating the number of bits transmitted in 1 slot;
- $\widetilde{C}$ is the number of subcarriers used for data transmission; it can be computed by subtracting the number of subcarriers composing the band guards from the total number of OFDM subcarriers: in this case $\widetilde{C} = 512 - 106 - 106$.
- delay spread model defines the delay profile, which determines the scaling factor for the time of arrival of each cluster signal, this way creating a shorter or longer delay spread.



(a) NR-CsiNet encoder scheme

(b) NR-CsiNet decoder scheme

Figure 4: NR-CsiNet scheme

We decide to keep the system bandwidth quite small and the working frequency low, as it can be observed in Tab. I; this choice is dictated uniquely by computational complexity issues, since simulations with a larger bandwidth would last much longer and the simulation time would increase proportionally. The hyperparameters adopted for the training and testing process of our model instead are listed in Tab. II.

Table II: NR-CsiNet model parameters

NR-CsiNet parameters	
Training set	1000 sim of 100 slots
Validation set	100 sim of 100 slots
Testing set	100 sim of 100 slots
Learning rate	0.001
Loss function	MSE
Optimizer	Adam
Epochs	300
RefineNet units	2
K_{DFT}	3

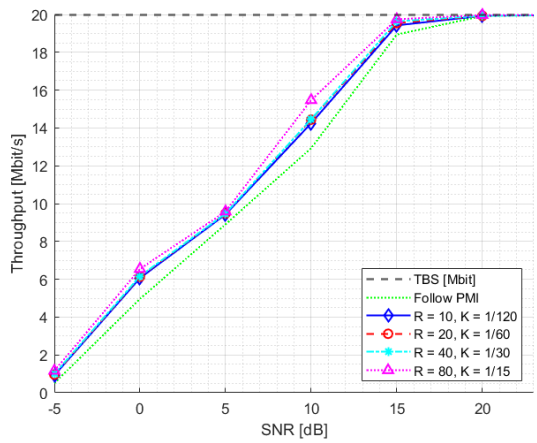
The DFT-compression factor K_{DFT} is defined as:

$$K_{\text{DFT}} = \frac{\widetilde{C}}{C} \tag{5}$$

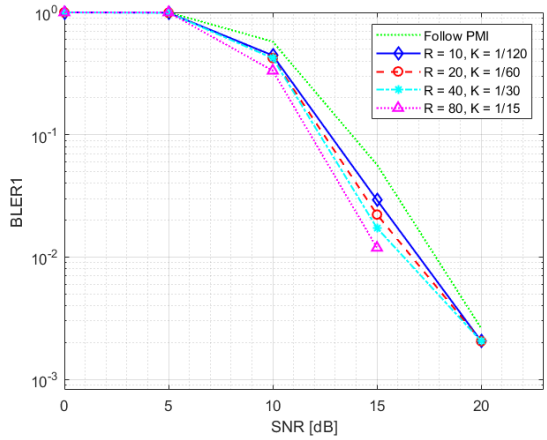
where $\widetilde{C}$ and C are the same parameters defined in Sec. II. Please notice that this parameter should not be confused with the final compression factor K, introduced in Subsec. II-E,

For our simulations, we consider a scenario with $N = 8$ and $M = 2$, where the transmitting antenna system is of a 2×2 planar array made of dual-polarized antenna elements, while the receiver equipment consists of a single dual-polarized antenna. Clearly, we are not considering a Massive MIMO scenario, since the number of antennas is quite small. An increase in the number of antennas would cause a proportional and unmanageable increase in simulation times; for this reason, we decide to leave experiments with larger N and/or M for future studies.

For our simulations, a single value of DFT-compression factor is considered ($K_{\text{DFT}} = 3$), while the SNR values selected for the training process are two, i.e., 10 dB and 20 dB. We



(a) Throughput statistics



(b) BLER1 statistics

Figure 5: Performance statistics for NR-CsiNet model trained at SNR = 20 dB

will show only the results obtained with the NR-CsiNet model trained at 20 dB because experiments have led us to conclude

that the system performances are almost independent of the SNR level at which training data are collected.

Fig. 5 relative to throughput and BLER1 (i.e., Block Error Rate at the first transmission attempt) are zoomed to the SNR ranges $[-5, 23]$ dB and $[0, 23]$ dB respectively for resolution improvement, but we test the model performances over a wider SNR range ($[-5, 35]$ dB). These graphs are useful to compare our NR-CsiNet model with a PMI-based reporting technique, that we refer to as follow PMI: adopting the follow PMI procedure, the UE computes the Precoding Matrix Indicator (PMI) [22], starting from some measurements of the channel based on CSI-RS. The PMI selected, which is nothing but an index pointing to a specific precoding vector inside the precoding codebook, is transmitted to the BS, which adopts the precoding vector corresponding to the PMI index received.

It seems reasonable to conclude that, in 8×2 scenarios, our DL-based approach outperforms the follow PMI algorithm, even when dealing with noisy channel estimates instead of perfect channel matrices. In particular, the plots show that above 20 dB of SNR, both NR-CsiNet and follow PMI throughput curves reach the maximum limit imposed by the Transport Block Size (TBS); at lower SNR, instead, all the variants of our DL solution provide better performances.

It is worth noting that for all our experiments we adopt a single variant of channel model CDL-B presented in Sec.I: simulation results have shown that this type of channel model usually presents a dominant cluster, which implicitly defines a preferential direction for signal transmission. Therefore, the follow PMI is expected to achieve good performance: in fact, among a set of possible beams, the PMI index simply identifies the one pointing to the receiver with more precision. Our guess is that, in the case of "richer" channel realizations where multipath is not dominated by a single cluster of paths, our DL-based approach has all the potential to further gain ground on the follow PMI algorithm.

IV. Conclusions

In this paper, we have investigated the potential use of neural networks as an alternative to the traditional feedback reporting block implemented inside a comprehensive New Radio link-level simulator. Starting from the CNN proposed in [3], we have developed a NR-CsiNet model that is applicable also to MIMO systems and able to deal with the channel estimation task. Simulation results, obtained by a 5G-NR compliant physical layer simulator with CDL channel model, have shown that our DL-based feedback system can outperform a system that implements a traditional PMI-based reporting technique, for a wide range of SNR values, as shown in Fig. 5.

Our work might be a starting point for future experiments, in which different and more realistic simulation conditions will be considered: a larger number of antennas, real channel realizations instead of mathematical models, multiple transmission layers, higher carrier frequencies and larger system bandwidths, suitable for 5G-based communication systems. We hope that our conclusions will provide useful guidelines to address future research on this topic, which is attracting considerable interest.

References

[1] E. Dahlman, S. Parkvall, and J. Skold, *5G NR: The Next Generation Wireless Access Technology*. Elsevier Science, 2018.

[2] H. Xie, F. Gao, S. Jin, J. Fang, and Y. Liang, "Channel Estimation for TDD/FDD Massive MIMO Systems With Channel Covariance Computing," *IEEE Transactions on Wireless Communications*, vol. 17, no. 6, pp. 4206–4218, 2018.

[3] C. Wen, W. Shih, and S. Jin, "Deep Learning for Massive MIMO CSI Feedback," *IEEE Wireless Communications Letters*, vol. 7, no. 5, pp. 748–751, 2018.

[4] Y. Han, T. Hsu, C. Wen, K. Wong, and S. Jin, "Efficient Downlink Channel Reconstruction for FDD Multi-Antenna Systems," *IEEE Transactions on Wireless Communications*, vol. 18, no. 6, pp. 3161–3176, 2019.

[5] T. Wang, C. Wen, S. Jin, and G. Y. Li, "Deep Learning-Based CSI Feedback Approach for Time-Varying Massive MIMO Channels," *IEEE Wireless Communications Letters*, vol. 8, no. 2, pp. 416–419, 2019.

[6] C. V. Murudkar and R. D. Gitlin, "Optimal-Capacity, Shortest Path Routing in Self-Organizing 5G Networks using Machine Learning," in *2019 IEEE 20th Wireless and Microwave Technology Conference (WAMICON)*, pp. 1–5, 2019.

[7] S. Khan Tayyaba, H. A. Khattak, A. Almogren, M. A. Shah, I. Ud Din, I. Alkhalifa, and M. Guizani, "5G Vehicular Network Resource Management for Improving Radio Access Through Machine Learning," *IEEE Access*, vol. 8, pp. 6792–6800, 2020.

[8] F. Hussain, S. A. Hassan, R. Hussain, and E. Hossain, "Machine Learning for Resource Management in Cellular and IoT Networks: Potentials, Current Solutions, and Open Challenges," *IEEE Communications Surveys Tutorials*, vol. 22, no. 2, pp. 1251–1275, 2020.

[9] T. Wang, C. Wen, H. Wang, F. Gao, T. Jiang, and S. Jin, "Deep learning for wireless physical layer: Opportunities and challenges," *China Communications*, vol. 14, no. 11, pp. 92–111, 2017.

[10] 3GPP, "NR; Physical channels and modulation," Technical Specification (TS) 38.211, 3rd Generation Partnership Project (3GPP), 06 2019. Version 15.6.0.

[11] 3GPP, "NR; Multiplexing and channel coding," Technical Specification (TS) 38.212, 3rd Generation Partnership Project (3GPP), 06 2019. Version 15.6.0.

[12] 3GPP, "NR; Physical layer procedures for control," Technical Specification (TS) 38.213, 3rd Generation Partnership Project (3GPP), 06 2019. Version 15.6.0.

[13] 3GPP, "NR; Physical layer procedures for data," Technical Specification (TS) 38.214, 3rd Generation Partnership Project (3GPP), 06 2019. Version 15.6.0.

[14] 3GPP, "NR; NR and NG-RAN Overall Description," Technical Specification (TS) 38.300, 3rd Generation Partnership Project (3GPP), 07 2020. Version 16.2.0.

[15] Y. Liao, H. Yao, Y. Hua, and C. Li, "CSI Feedback Based on Deep Learning for Massive MIMO Systems," *IEEE Access*, vol. 7, pp. 86810–86820, 2019.

[16] "Tensorflow Keras module." https://www.tensorflow.org/api_docs/python/tf/keras.

[17] 3GPP, "Evolved Universal Terrestrial Radio Access (E-UTRA); Physical channels and modulation (Release 10)," Technical Specification (TS) 36.211, 3rd Generation Partnership Project (3GPP), 02 2013. Version 10.7.0.

[18] "ReLU Keras API." https://keras.io/api/layers/activation_layers/relu/.

[19] A. C. Ian Goodfellow, Yoshua Bengio, *Deep learning*. Adaptive computation and machine learning, Cambridge (Mass.): MIT Press, 2016.

[20] "Batch normalization Keras API." https://keras.io/api/layers/normalization_layers/batch_normalization/.

[21] "Softmax Keras API." https://keras.io/api/layers/activation_layers/softmax/.

[22] 3GPP, "NR; Physical layer procedures for data (Release 15)," Technical Specification (TS) 38.214, 3rd Generation Partnership Project (3GPP), 03 2020. Version 15.9.0.