A deep learning-based approach to 5G-new radio channel estimation

A Deep Learning-based Approach to 5G-New Radio Channel Estimation

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 present a deep learning-based technique for channel estimation. By treating the time-frequency grid of the channel response as a low-resolution 2D-image, we propose a 5G-New Radio Convolutional Neural Network, called NR-ChannelNet, which can be properly trained to predict the channel coefficients. Our study employs a 3GPP-compliant 5G-New Radio simulator that can reproduce a realistic scenario by including multiple transmitting/receiving antenna schemes and clustered delay line channel model. Simulation results show that our deep learning approach can achieve competitive performance with respect to traditional techniques such as 2D-MMSE: indeed, under certain conditions, our new NR-ChannelNet approach achieves remarkable gains in terms of throughput.**

***Index Terms*—5G, New Radio, Channel Estimation, Deep Learning, Convolutional Neural Network.**

I. Introduction

Future wireless networks will be expected to support very high data rates and new applications that will pave the way for a new radio technology paradigm [1]. This innovative paradigm can be identified with a machine learning-based approach, aiming at satisfying all the different requirements of Next-Generation wireless networks by means of an intelligent adaptive learning process. The purpose of these new technologies is to learn the colorful characteristics of the system under service and autonomously determine the optimal configurations.

Even though many studies have started to explore all the possible Machine Learning (ML) applications for the upper layers of wireless communication systems, in recent years much research has been devoted to introduce innovative Deep Learning-based architectures into several processing blocks [2]: the idea is to outperform conventional communication algorithms in the emerging complex scenarios [2]–[4], characterized by lack of information about the channel model, high data rates and accurate processing requirements.

Channel estimation is a challenging problem in wireless systems: the transmitted information is subject to highly random distorting effects, such as reflection, scattering, diffraction, which considerably limit the performance of the system; moreover, the mobility of transmitters and receivers causes rapid changes in the channel response over time. All these factors make the channel estimator a very sensitive block of every wireless communication system.

DeModulation-Reference Signals (DM-RS) [5] have been introduced in 5G New Radio (5G-NR) to estimate the radio channel at the receiver side for the associated physical channel demodulation. Given these reference signals, there exist several conventional pilot-based estimation techniques that exploit the known-values in the time-frequency grid, corresponding to the pilots, to derive all the unknown values of the channel response.

To enhance these traditional approaches this paper proposes the introduction of a deep learning (DL)-based solution in the channel estimation process. We start from the work presented in [6] and we extend it to a practical scenario with: i) 5G-New Radio (NR) fully compliant simulator; ii) clustered delay line channel model; iii) MIMO support, with multi-transmitting/receiving antenna schemes. We compare our approach against a simplified version of a traditional 2D-Minimum Mean Square Error (2D-MMSE) channel estimation algorithm developed in TIM laboratories.

All the experiments described in this paper have been conducted through a MATLAB-based New Radio Link Simulator software [7], developed in TIM laboratories. The purpose is to model a radio interface fully compliant with 3GPP specifications [5], [8]–[10] and to evaluate the link level performance of 5G-based point-to-point communications.It is in particular a link level simulator and it targets scenarios with a single next-generation base station or gNodeB and a single UE, which can be equipped with multiple antennas

We adopt the clustered delay line (CDL) channel model, proposed by the the 3GPP WG RAN group for NR-compliant link simulators [11], [12]. The model 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 select the CDL-B profile. This CDL profile is specific for Non-line-of-sight (NLOS) transmissions and it 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 describe the system model and one of the traditional channel estimation algorithms for wireless OFDM systems. In Sec. III we present the Deep Learning model used to study the channel estimation problem in 5G-NR. In Sec. IV we discuss the simulation results and we show that the DL approach can outperform the traditional ones. Sec. V concludes the article.

II. System model

In an OFDM MIMO system, we denote with C the number of subcarriers used for data transmission, S the number of OFDM symbols transmitted in a time slot, while N and M represent the number of transmitting and receiving antennas respectively. The described model represents a single layer MIMO commu-

nication system, where spatial diversity is adopted both at the transmitting and receiving sides. By taking a snapshot of an s-th OFDM symbol and a c-th subcarrier, we can express the estimated symbol $\hat{y}$ as:

$$\hat{y} = \mathbf{u}^H \boldsymbol{\mathcal{H}} \mathbf{v} x + \mathbf{u}^H \mathbf{z} \tag{1}$$

where:

- $\boldsymbol{\mathcal{H}}$ is the complex MIMO channel matrix, of size $M \times N$, corresponding to the s-th OFDM transmitted symbol on the c-th subcarrier;
- $\mathbf{v}$, of size $N \times 1$, is the complex precoding vector;
- x is the complex s-th data symbol transmitted on the c-th subcarrier;
- $\mathbf{z}$ denotes the $M \times 1$ additive noise vector affecting the M receiving antennas.
- $\mathbf{u}$ is an $M \times 1$ combiner.

A. A simplified 2D-MMSE channel estimation algorithm

Traditional pilot-based channel estimation techniques are based on the multiplexing of pilots symbols (i.e., known symbols) into the transmitted data: these symbols are scattered on the time-frequency grid and the channel coefficients in neighboring positions can be recovered through a two-dimensional interpolation [13].

In OFDM systems, the optimal linear channel estimator is the 2D-MMSE Wiener filter, which is able to minimize the error between the estimated channel frequency response (CFR) and the original one [13]–[15]. This estimator combines the set of K surrounding pilots to estimate the channel coefficients in a specific position on the time-frequency grid, which basically consists of a two-dimensional convolution operation.

As $\boldsymbol{\mathcal{H}}$ refers to the typical MIMO channel matrix for a specific OFDM symbol s and subcarrier c, we consider now a single transmitting/receiving antenna pair (n, m) and we refer to the relative time-frequency channel grid matrix as $\mathbf{H}$, of size $S \times C$. At the receiver side, the CFR is only known in correspondence of the pilots: since for coherent demodulation the whole channel coefficients must be computed, the remaining $\mathbf{H}(s, c)$ entries have to be estimated by interpolating the known $\mathbf{H}(s_p, c_p)$ coefficients:

$$\mathbf{H}(s, c) = \sum_{(s_p, c_p) \in P} w^{s,c}_{s_p, c_p} \mathbf{H}(s_p, c_p) \tag{2}$$

where P is the set of the positions that accommodate the nearest pilots with respect to the position (s, c). The filter coefficients $w^{s,c}_{s_p,c_p}$ are computed by taking into account the autocorrelation function $R_{HH}(\Delta s, \Delta c) = E\{H(s, c)H^*(s - \Delta s, c - \Delta c)\}$: the idea is that the more a pilot is far from the position to estimate, the less it contributes to the estimation, since the correlation between the pilot symbol and the estimated symbol decreases.

The solution of (2) can be expressed as

$$\mathbf{h} = \mathbf{R}_{hp} \mathbf{R}^{-1}_{pp} \mathbf{p} \tag{3}$$

where $\mathbf{p}$ is a vector of size $K \times 1$ containing the K pilot symbols which are taken into account, K is generally smaller than the total number of transmitted pilots to reduce the complexity of the estimation; $\mathbf{h}$ is a vector of size $Q \times 1$ where Q represents the number of coefficients that have to be estimated; $\mathbf{R}_{hp} = \mathrm{E}\left[\mathbf{h}\mathbf{p}^H\right]$ is the cross-covariance matrix between the CFR estimated in positions (s, c), without pilot, and the CFR estimated in pilot positions (s_p, c_p), $\mathbf{R}_{pp} = \mathrm{E}\left[\mathbf{p}\mathbf{p}^H\right]$ is the auto-covariance matrix of $\mathbf{p}$.

The general version of the 2D-MMSE estimator is typically too complex to be actually implemented. The complexity of the original algorithm can be reduced by implementing a low-rank 2D-MMSE filter, which decreases the rank of the 2D filter by decreasing the set of pilots P. Given a channel coefficient, the contribution of each pilot for its estimation diminishes as the pilot distances in frequency. So, the inclusion of very distant pilots in the time-frequency grid does not significantly improve the estimator performance. Therefore only the neighboring pilots can be used to estimate a certain value of the channel coefficient. The NR link level simulation platform developed in TIM laboratories attains this simplification by implementing a sliding window that includes only the surrounding pilot symbols in the grid. The window size determines the accuracy of the estimation: experiments have shown that a window size between 3 and 7 Resource Blocks is enough to guarantee a comparable performance to the full 2D-MMSE estimator, especially in case of fast channel variations.

III. Deep Learning-based channel estimation

To enhance the traditional estimators described in the previous section, many studies have proposed the introduction of deep learning-based algorithms in the channel estimation process. The idea is to treat the time-frequency grid of the channel response as a low-resolution 2D-image, whose pixels are known only at the pilot positions. Our DL-based approach is based on two different Convolutional Neural Networks (CNN) [6] and consists of two different phases:

- an image Super Resolution CNN (SRCNN) [16], which enhances the resolution of the low-resolution input image and transforms it into an high-resolution image, by estimating the channel response values at all positions without pilots;
- a denoising CNN (DnCNN) [17], which implements an Image Restoration (IR) algorithm to remove or reduce the noise affecting an image.

A. Image Super-Resolution Convolutional Neural Network

A deep CNN [16] takes a low-resolution image as input and outputs the high-resolution version. Similarly to a Sparse-Coding-based method, the SRCNN embeds two different steps:

- patch extraction and representation, formulated as convolutional layer: several patches are extracted from the low-resolution image and mapped onto high-dimensional vectors, whose number of dimensions corresponds to the number of collected feature maps;
- dictionary-based non-linear mapping: the high-dimensional vectors output by the previous step are mapped onto other high-dimensional vectors through a non-linear operation. The resulting vectors are representations of high-resolution patches and store another set of feature maps. Note that dictionaries are not explicitly learned, since they are implicitly achieved through hidden layers;

- reconstruction: in this last phase, the final high-resolution image is generated by aggregating the high-resolution patch-wise representations.

The only foreseen pre-processing operation is an interpolation of the input low-resolution image, in order to upscale it to the desired size.

B. Image Denoising Convolutional Neural Network

The DnCNN proposed in [17] is able to handle Gaussian denoising with unknown noise level; it isolates the noise in the noisy input image by means of a feed-forward convolutional network. It is designed to predict not the denoised image $\hat{\mathbf{x}}$ but the residual image $\hat{\mathbf{v}}$, defined as the difference between the noisy observation and the latent clean image: in other words, in the hidden layers of the network the latent clean image is implicitly removed, so that the inputs at each layer are characterized by a Gaussian distribution, a lower correlation and a negligible relationship with the image content [17]. Also batch normalization is introduced to enhance the training performance: it is an efficient solution to the internal covariate shift, i.e., the changes in the distributions of internal nonlinearity inputs during training [17], such solution is achieved through the mini-batch gradient descent algorithm. Since residual learning and batch normalization can benefit from each other, the proposed DL-based solution shows a consequent speeding up in training and an improvement of denoising performance [17].

A DnCNN with depth D is characterized by three different types of layers:

- the first layer consists of a convolutional layer followed by a Rectified Linear Unit (ReLU): this layer generates 64 feature maps from 64 filters of size $3 \times 3 \times c$, where c is the number of image channels; the ReLU is then employed for nonlinearity;
- layers from 2 to $D-1$ consist of a convolutional layer followed by a batch normalization layer and a ReLU: these layers adopt 64 filters of size $3 \times 3 \times 64$;
- the last layer simply consists of a convolutional layer with c filters of size $3 \times 3 \times 64$.

C. A ChannelNet model for NR

A pipeline for DL-based channel estimation, called ChannelNet, was proposed in [6] with the purpose of estimating the time-frequency response matrix $\mathbf{H}$ that characterizes the link between a single transmitter and a single receiver antenna, i.e., a SISO channel. Since the matrix $\mathbf{H}$ has complex values, it is represented as two 2D-images, one for the real values and one for the imaginary values. The values estimated at the pilot locations $\hat{\mathbf{h}}_p$ are considered as the low-resolution and noisy version of the channel image which must be mapped to its high-resolution version:

- first, an SRCNN network (Fig. 1a) takes as input the low-resolution interpolated images (real and imaginary part) and estimates the unknown values of the channel response matrix $\mathbf{H}$;
- secondly, a DnCNN implementation (Fig. 1b), cascaded with the SRCNN, removes the noise from the estimated images.

We introduce a channel estimation solution inspired by the ChannelNet model: specifically, our approach, named NR-ChannelNet, substantially maintains the same structure of the CNN illustrated in [6], as shown in Fig. 1, but it introduces some modifications in the training procedure and it attempts at generalizing the model to multiple use cases:

- first of all, we try to make the model applicable also to MIMO scenarios, differently from the work proposed in [6] which focuses only on the channel matrix along a single transmitter/receiver antenna pair;
- furthermore, we try to enrich our training datasets by performing link simulations at different levels of SNR: instead of fixing two values of SNR for the training process, we change it with a finer granularity, with the purpose of obtaining models with better performance also in scenarios where the quality of the channel significantly fluctuates.

We build and train the neural network models in Python with Keras, a TensorFlow's open-source high-level API. Once completed the training phase, we develop a deep learning-based channel estimation block within the TIM software link simulator [7]. This block basically substitutes the pre-existing low-rank 2D-MMSE estimator: it imports a pre-trained neural network model in MATLAB environment and predicts the channel estimate by exploiting the functionalities of the MATLAB Deep Learning tool.

D. Training process

Tab. I reports the configuration of the simulator parameters that are of particular interest in our experiments.

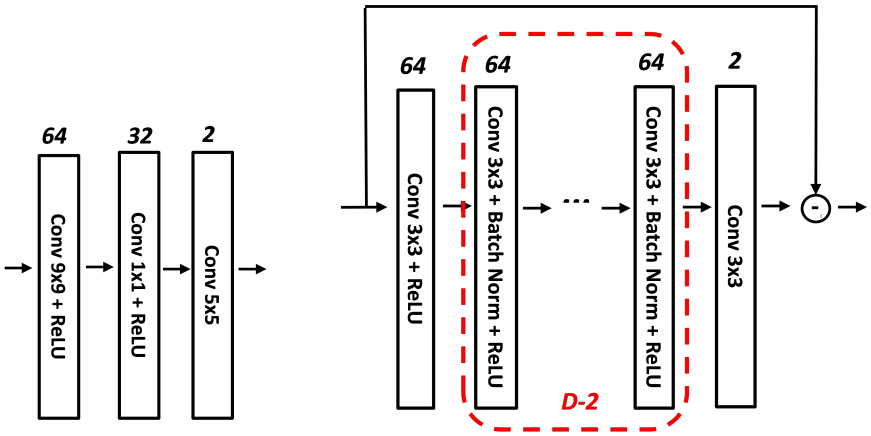
Table I: Simulation parameters for NR-ChannelNet training

Simulation parameters	
Transmission direction	Downlink
Carrier frequency	3.64 GHz
System bandwidth	1.4 MHz
Time slot	1 ms
(Modulation, Coding rate)	(64-QAM, 0.694)
Transport Block Size (TBS)	4736 bit
OFDM symbols per time slot (S)	14
Transmission layers (L)	1
Codewords	1
Channel Model	CDL-B

Please not that the choice of the system bandwidth (1.4 MHz), which is only provided by LTE standard, is imposed by a reduced computational complexity and simulation time.

As far as the transmitter and receiver antenna systems are concerned, we consider two different configurations, shown in Tab. II. N and M represent the number of transmitting and receiving antennas respectively, while N_1 and N_2 denote the number of antenna elements on the azimuth and elevation plane; N_{pol} instead indicates the number of polarizations for each antenna element.

It is worth noting that the channel matrix considered in our DL-based solution is different from the channel time-frequency



(a) SRCNN **(b)** DnCNN

Figure 1: NR-ChannelNet implementation scheme

Table II: Antenna systems configurations

Configuration	N	M	N_1	N_2	N_{pol}
8×1	8	1	2	2	2
32×2	32	2	4	4	2

response image $\mathbf{H}$ introduced in [6]: the matrix $\mathbf{H}$, in fact, contains the channel coefficients for a SISO link and therefore it has size $C \times S$, where S is the number of OFDM symbols transmitted in a single time slot; our matrix, referred to as $\mathbf{H}_{id}^{v}$, instead, represents the beamformed channel matrix, of size $M \times L \times C \times S$, where L is the number of transmission layers; but if we consider a single transmission layer, the size reduces to $M \times C \times S$. For each subcarrier c and OFDM symbol s, $\mathbf{H}_{id}^{v}$ is computed as the multiplication of the CDL-B channel matrix $\widetilde{\mathbf{H}}$, of size $M \times N$, by the beamforming vector $\mathbf{v}$, of size $N \times L$. As a consequence, in our model also a third dimension must be taken into account: in fact, while in [6] only single receiver antenna scenarios are considered, our model is aimed at working also with configurations where $M > 1$. This is accomplished by considering channel images associated to different receiving antennas as distinct data points on the same input channel: with this approach only 2D images are considered as input of the CNN.

E. Input pre-processing: interpolation and scaling

Before going through the first layer of the NR-ChannelNet model, all the matrices $\mathbf{p}_{DMRS}$ contained in the dataset are subject to a two-step preprocessing.

First, we place the values of $\mathbf{p}_{DMRS}$ on a time-frequency grid and we apply an interpolation algorithm on the real and imaginary part separately, to pre-compute the channel coefficients in positions where DM-RS pilots are not present; in particular, we adopt the Radial Basis Function (RBF) interpolation implemented in Python SciPy library and we obtain a complete interpolated input channel matrix. We can denote the matrix resulting from this interpolation step as $\mathbf{H}_{DMRS}^{v}$.

Once we obtain the matrix $\mathbf{H}_{DMRS}^{v}$, we rescale the values of both its real and its imaginary parts in the range $[0, 1]$. This is a crucial step in the preprocessing pipeline, since without data normalization, the objective function of the DL algorithm does not properly work in most cases. Data normalization or feature scaling is a technique that normalizes the range of values characterizing independent variables or features [18]. As explained in [19], when features are measured on different scales, the optimization algorithm will be governed by the feature with the broadest range, since the weights will be mostly optimized based on its errors. To counteract this issue, we perform a min-max normalization, according to the following expression:

$$\mathbf{x}_{norm} = \frac{\mathbf{x} - min(\mathbf{x})}{max(\mathbf{x}) - min(\mathbf{x})}. \tag{4}$$

The resulting matrix is denoted as $\hat{\mathbf{H}}_{DMRS}^{W}$.

While the authors of [6] consider the real and the imaginary parts of the input matrix $\mathbf{H}_{DMRS}^{v}$ and the desired output matrix $\mathbf{H}_{id}^{v}$ as distinct entries of the training dataset, we follow instead the approach of [20], i.e., we consider the real and the imaginary parts as distinct colors of the same image, conveyed through the network on two separate channels: this means that both the third dimension of the input layer and the number of feature maps at the output of the last convolutional layer must be doubled.

F. Variability of channel quality

The $\mathbf{H}_{DMRS}$ matrices are collected by simulating different conditions of the channel: in particular, we modify the SNR value in order to test the behavior of our DL-based model at different levels of channel quality.

In [6], the SNR range is divided into two regions and the ChannelNet is trained for two different SNR values: for low SNR values, the network is trained at 12 dB of SNR; for higher SNR values, the network is trained at 22 dB of SNR. Instead, we decide to use two different approaches:

Table III: NR-ChannelNet simulation parameters

Parameter	8×1 configuration	32×2 configuration
Training set	1000 sim of 100 slots	1000 sim of 100 slots
Validation set	100 sim of 100 slots	100 sim of 100 slots
Testing set	100 sim of 100 slots	100 sim of 100 slots
Learning rate	0.001	0.001
Loss function	MSE	MSE
Optimizer	Adam	Adam
SRCNN epochs	300	400
DnCNN epochs	200	200
DnCNN depth (D)	4	4

- on one hand, we select a set of different SNR values, $\{0, 10, 20, 50\text{dB}\}$, and we train a different NR-ChannelNet model for each of these values;
- in parallel, we define two SNR regions, $[0; 20]$ and $[20; 50]$ dB. For each region, we collect our training data varying the SNR inside the corresponding range of values: for the first SNR region, the SNR values considered for the simulations are 0 dB, 10 dB and 20 dB; for the second one instead, training data are collected at SNR 20 dB and 50 dB.

With this second approach, we attempt at improving the performance of our NR-ChannelNet in scenarios where the SNR oscillates significantly: by increasing the variability of the training data, we try to obtain a more generalizable model, less sensitive to possible channel variations.

IV. Simulation results

The performance of the NR-ChannelNet model, in all the illustrated variants, are evaluated after simulation campaigns conducted with the NR link level simulator presented in in the previous sections. Tab. I reports the main parameters of interest concerning the simulator settings, while Tab. III shows the hyperparameters adopted for the training and testing process of our models.

A. MISO scenario: 8 X 1 antenna configuration

The plots displayed in Fig. 2 are useful to summarize and compare the obtained results, showing the curves of throughput for all the different trained models. In particular, Fig. 2a shows the performance of the models trained with data collected at constant SNR levels; the curves shown in Fig. 2b, instead, are relative to NR-ChannelNet variants trained at mixed SNR levels.

Each plot also reports a horizontal straight line, representing the maximum throughput level that can be achieved given the selected Transport Block Size (TBS). As it can be observed, the throughput curves never reach the maximum possible value, even at very high SNR levels. The reason lies in the TBS computation: the NR link simulator adopted for our experiments calculates the TBS value only once, at the beginning of the simulation, without taking into account that, at each Channel State Information (CSI) reporting period, some resource elements in the time-frequency grids must be reserved for CSI-RS pilots and some redundant bits are sacrificed, with a consequent increase of the code rate and a decrease in the degree of protection.

It is evident that the SNR level at which the considered model as been trained has negligible impact on the performance; the only exception seems to be the NR-ChannelNet trained at 0 dB of SNR, which turns out to be the worst performing. This proves that it is sufficient to train a single NR-ChannelNet model provided that the training data are collected at sufficiently high SNR values. As a consequence, the idea of mixing data collected at different SNR levels, with the purpose of training models that are more insensitive to the channel quality oscillation, does not seem to be particularly useful. This is the reason why this training approach is not taken into account for the 32×2 configuration pattern. To conclude, it is possible to state that the NR-ChannelNet approach turns out to be absolutely competitive with the low-rank 2D-MMSE algorithm, even achieving visibly better performance within certain SNR ranges.

B. MIMO scenario: 32 X 2 antenna configuration

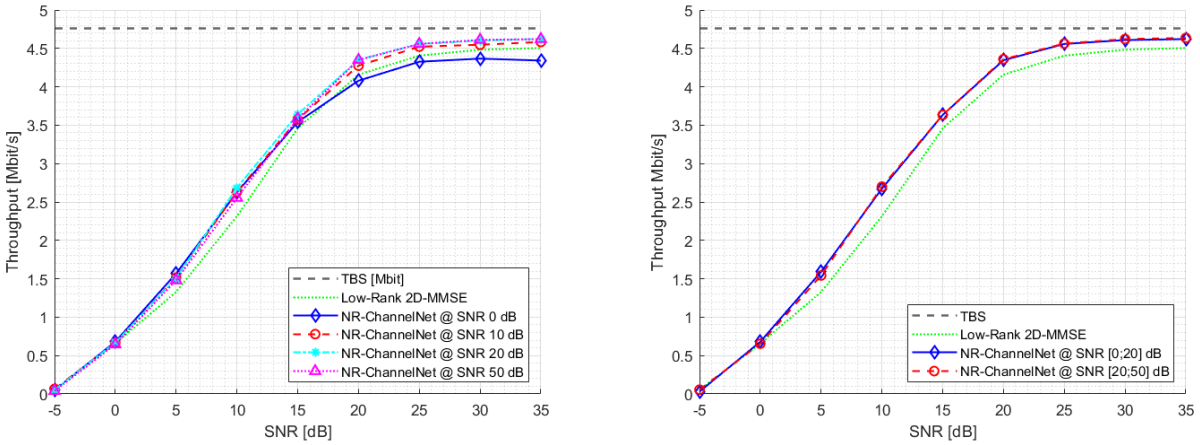
This subsection refers to simulation experiments performed with a 32×2 antenna configuration. With this second configuration, characterized by an increased number of antennas, we certainly expect better results: a larger number of transmitter and/or receiver antennas can provide additional diversity against fading on the radio channel [21], [22], [23]; not only this, but a greater availability of antenna elements results in a grater beamforming gain, due to an improved ability to orient the beam in the direction of the receiver. It is possible to find an effective match long at Fig 3.

As for the 8×1 configuration, the NR-ChannelNet model trained at 0 dB of SNR shows very poor performance. Our guess is that when the input channel images are particularly noisy the training set collected and fed to the convolutional network essentially contains only noise; as a consequence, larger datasets are needed to finalize the learning process and avoid overfitting phoenomena.

If we compare the results obtained with the two considered antenna configurations, it is quite evident that NR-ChannelNet models bring more significant improvements for the 8×1 scenario. In Fig. 3, relative to the 32×2 antenna configuration, it is possible to observe a slight throughput enhancement only at very low SNR values. This sounds reasonable since, when a single receiver antenna is available, the system performance is more sensitive to the channel estimation accuracy and thus there is more room for improvement.

V. Conclusions

In this paper, we have investigated the potential use of neural networks as an alternative to the traditional channel estimation block implemented within a 3GPP-compliant New Radio link simulator. Starting from the DL-based solution presented in [6], we have proposed a NR-ChannelNet model that is able to deal with channel estimation task in MIMO communication systems. Several experiments have been carried out by generating several CDL channel scenarios through a 5G physical layer software simulator. Results have shown that our deep learning approach for channel estimation can achieve absolutely competitive performance w.r.t. a traditional 2D-MMSE technique and visibly outperforms it in some cases.



(a) Discrete SNR values

(b) Mixed SNR values

Figure 2: NR-ChannelNet - Throughput - 8x1 configuration

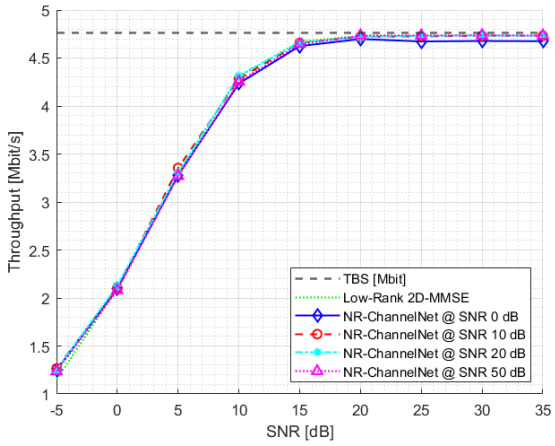


Figure 3: NR-ChannelNet - Throughput - 32x2 configuration

REFERENCES

[1] C. Jiang, H. Zhang, Y. Ren, Z. Han, K. Chen, and L. Hanzo, "Machine learning paradigms for next-generation wireless networks," *IEEE Wireless Commun.*, vol. 24, no. 2, pp. 98–105, 2017.

[2] 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, pp. 92–111, Nov 2017.

[3] T. J. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Trans. Cogn. Comm. & Networking*, vol. 3, no. 4, pp. 563–575, 2017.

[4] H. Huang, S. Guo, G. Gui, Z. Yang, J. Zhang, H. Sari, and F. Adachi, "Deep learning for physical-layer 5g wireless techniques: Opportunities, challenges and solutions," 2019.

[5] 3GPP, "Nr; multiplexing and channel coding," Technical Specification (TS) 38.212, 3rd Generation Partnership Project (3GPP), 06 2019. Version 15.6.0.

[6] M. Soltani, V. Pourahmadi, A. Mirzaei, and H. Sheikhzadeh, "Deep learning-based channel estimation," *IEEE Communications Letters*, vol. 23, pp. 652–655, April 2019.

[7] E. Zimaglia, D. G. Riviello, R. Garello, and R. Fantini, "A novel deep learning approach to csi feedback reporting for nr 5g cellular systems," in *2020 IEEE Microwave Theory and Techniques in Wireless Communications (MTTW)*, vol. 1, pp. 47–52, 2020.

[8] 3GPP, "Nr; physical channels and modulation," Technical Specification (TS) 38.211, 3rd Generation Partnership Project (3GPP), 06 2019. Version 15.6.0.

[9] 3GPP, "Nr; physical layer procedures for control," Technical Specification (TS) 38.213, 3rd Generation Partnership Project (3GPP), 06 2019. Version 15.6.0.

[10] 3GPP, "Nr; physical layer procedures for data," Technical Specification (TS) 38.214, 3rd Generation Partnership Project (3GPP), 06 2019. Version 15.6.0.

[11] 3GPP, "Nr; study on test methods," Technical Report (TR) 38.810, 3rd Generation Partnership Project (3GPP), 09 2018. Version 2.4.0.

[12] 3GPP, "Study on channel model for frequencies from 0.5 to 100 ghz," Technical Report (TR) 38.901, 3rd Generation Partnership Project (3GPP), 01 2018. Version 14.3.0.

[13] M. Sandell and O. Edfors, *A comparative study of pilot-based channel estimators for wireless OFDM*, vol. TULEA 1996:19. 1996.

[14] R. Nilsson, O. Edfors, M. Sandell, and P. O. Borjesson, "An analysis of two-dimensional pilot-symbol assisted modulation for ofdm," in *1997 IEEE International Conference on Personal Wireless Communications (Cat. No.97TH8338)*, pp. 71–74, 1997.

[15] A. R. James, R. S. Benjamin, S. John, T. M. Joseph, V. Mathai, and S. S. Pillai, "Channel estimation for ofdm systems," in *2011 International Conference on Signal Processing, Communication, Computing and Networking Technologies*, pp. 587–591, 2011.

[16] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 2, pp. 295–307, 2016.

[17] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising," *IEEE Transactions on Image Processing*, vol. 26, no. 7, pp. 3142–3155, 2017.

[18] S. Raschka, "About feature scaling and normalization and the effect of standardization for machine learning algorithms," 07 2014.

[19] S. Raschka, *Python Machine Learning*. Packt Publishing, 2015.

[20] C. Wen, W. Shih, and S. Jin, "Deep learning for massive mimo csi feedback," *IEEE Wireless Communications Letters*, vol. 7, pp. 748–751, Oct 2018.

[21] K. Sulonen, P. Suvikunnas, J. Kivinen, L. Vuokko, and P. Vainikainen, "Study of different mechanisms providing gain in mimo systems," in *2003 IEEE 58th Vehicular Technology Conference. VTC 2003-Fall (IEEE Cat. No.03CH37484)*, vol. 1, pp. 352–356 Vol.1, 2003.

[22] C. Gomez-Calero, J. Mora-Cuevas, L. Cuellar, R. Martinez, and L. De Haro, "Measurement of diversity gain and capacity on a mimo-ofdm channel comparing different types of antennas," in *2009 3rd European Conference on Antennas and Propagation*, pp. 1037–1041, 2009.

[23] Li Yonghua, Yang Bin, He Zhiqiang, and Wu Weiling, "Relation between beamforming and diversity in feedback mimo system," in *Proceedings. 2005 International Conference on Wireless Communications, Networking and Mobile Computing, 2005.*, vol. 1, pp. 115–118, 2005.