Massive MIMO Channel Estimation with Convolutional Neural Network Structures

Leopoldo Carro-Calvo[1], Alejandro de la Fuente[1], Antonio Melgar[1], and Eduardo Morgado[1]

[1]Affiliation not available

April 18, 2024

Massive MIMO Channel Estimation with Convolutional Neural Network Structures

Leopoldo Carro-Calvo, Alejandro de la Fuente,
Antonio Melgar, and Eduardo Morgado

***Abstract*—Massive multiple-input-multiple-output (mMIMO) enables a significant increase in capacity in fifth-generation (5G) communications systems, both in beamforming and spatial multiplexing scenarios, demanding highly accurate channel estimates. We present two models based on convolutional neural networks (CNNs) for 5G mMIMO channel estimation that differ in complexity and flexibility. The results achieved with both models are competitive compared to traditional methods, such as least squares (LS) which presents a poor estimate in the low signal-to-noise ratio (SNR) region, or minimum mean square error (MMSE) which requires prior statistical knowledge of the channel and noise estimation. Furthermore, the proposed CNN models outperform estimation structures based on conventional deep neural networks (DNNs). Our approach achieves results close to the MMSE estimates, improving them in the low SNR regime, and enabling them to a wide range of channel conditions, i.e., variability in time, frequency, and SNR, not requiring any prior channel statistics information. Furthermore, we present a deep analysis of the computational and cost complexity, demonstrating the suitability of the proposed models for real hardware structure implementation.**

***Index Terms*—5G, massive MIMO, channel estimate, convolutional neural networks, deep neural networks.**

I. Introduction

THE mobile connectivity is growing fast – total fifth-generation (5G) subscriptions passed the 1.5 billion mark in 2023, growing by 500 million in just one year. Commercial 5G networks allow the service providers in the top 5G markets to enjoy growing revenue correlated with growing subscription penetration. 5G subscriptions are forecast to reach 4.6 billion globally by the end of 2028, making up more than 50 percent of all mobile subscriptions, and becoming the dominant mobile access technology. The most common 5G services launched by service providers for consumers are based on enhanced mobile broadband (eMBB), such as gaming, or some augmented/virtual reality-based services [1].

This work was partly supported by the grants PID2020-115323RB-C32 (IRENE-STARMAN), TED2021-131624B-I00 (GERMINAL), TED2021-131975A-I00 (ANTHEM5G), and PID2022-136887NB-I00 (POLIGRAPH) funded by MCIN/AEI/10.13039/501100011033 and by the "European Union NextGenerationEU/PRTR"; and partly supported by Young Researchers R&D Projects F861 (AUTO-BA- GRAPH) and F858 (INCREASE-5G) funded by Community of Madrid and by University Rey Juan Carlos.

Leopoldo Carro-Calvo, Alejandro de la Fuente, and Eduardo Morgado are with the Department of Signal Theory & Communications, University Rey Juan Carlos. E-mail: leopoldo.carro, alejandro.fuente, eduardo.morgado@urjc.es.

Antonio Melgar is with the global CTIO unit, Telefónica Digital Innovation, and the Department of Signal Theory & Communications, University Rey Juan Carlos. E-mail: antonio.melgargonzalez@telefonica.com.

5G eMBB is considered the continuity of the multimedia-based service provided by the previous generations, whose main requirements concern extremely high peak data rates, large volumes of data traffic, support of high-speed mobility, and extensive coverage. Telecom operators explore different radio access network (RAN) features for the support of eMBB service such as new millimeter waves (mmWaves) band, high-order modulations, carrier aggregation (CA), and massive multiple-input-multiple-output (mMIMO) [2].

5G preserves orthogonal frequency division multiplexing (OFDM), already implemented in Long Term Evolution (LTE). OFDM consists of transmitting long-time symbols simultaneously but allocated in orthogonal subcarriers in the frequency domain, composing an OFDM symbol and allowing the system to separate each original long-time symbol at the receiver. The cyclic prefix (CP) addition, which is larger than the delay spread (DS) of the channel, allows the system to mitigate intersymbol interference (ISI) [3]. Through the numerology concept [4, Sec. 4.2], 5G offers subcarrier spacing and subframe configuration flexibility to adapt the transmission to the different environments according to the frequency range, Doppler shift, and DS. The use of multiple narrow subchannels converts frequency-selective fading into flat fading. However, the resulting one-path channel responses modify the amplitude and phase of the transmitted signal. The channel response variability between subcarriers and OFDM symbols depends on the DS and Doppler shift of the environment, respectively. The phase and amplitude shift produced by the channel response together with the additive white Gaussian noise (AWGN) at the receiver could increase the bit error rate (BER) of the system. Through the equalization procedure, the receiver reverts the effect of the channel made over the transmitted signal. The equalization behavior depends on the assumption of accurate channel knowledge, acquired through the channel estimation procedure. In addition to equalization, in mMIMO systems, accurate channel estimation is crucial for the combining/precoding strategy to concentrate the radiation pattern from/toward the target mobile station (MS), both in beamforming and spatial multiplexing scenarios [5].

Traditional channel estimation acquires channel information from pilot signals allocated in some OFDM symbols and subcarriers. 5G OFDM systems reserve some resources for pilot allocation and others for data transmission. 5G estimates the channel response of pilots via traditional least squares (LS) or minimum mean square error (MMSE) channel estimators. The channel information of resources reserved for data trans-

mission is commonly estimated through linear interpolation methods [6]. The MMSE algorithm performs well in terms of mean square error (MSE) but requires high complexity. The LS estimator has low complexity, but its performance is not as good as the MMSE algorithm, especially when the MS presents weak coverage. In mMIMO, the transmitter can send pilots allocated in every antenna, adding an extra spatial dimension to the typical 2D time-frequency grid of OFDM systems. In this case, the system performs channel estimates for each antenna independently, in the same way as in the frequency-time grid of single-antenna OFDM systems. Nevertheless, the transmitter can also send pilots in some antennas and perform an additional spatial interpolation to estimate the channel response of the not transmitting antennas during pilot transmission [7].

A. Related Literature

The channel estimation procedure determines the global performance of mMIMO-OFDM systems. Some topic research focuses on analyzing efficient interpolation methods: linear interpolation [8], polynomial interpolation [9], polar linear interpolation [10], two-dimensional Wiener filtering [11], matrix factorization [12], high-speed adaptive interpolation [13], among others. Other research focuses on reducing the complexity of MMSE channel estimator: a simplified linear minimum mean square error (LMMSE) algorithm using the Fourier Transform and an appropriate training-sequence-aided is proposed in [14]; a novel low-rank LMMSE method to simplify the filtering matrix is presented in [15]; and in [16], the complexity of the MMSE channel estimator is reduced by the simplification of the matrix inversion and matrix-vector multiplication. Furthermore, mMIMO channel estimate is ridden with several challenges such as pilot sequence design [17], pilot overhead [18], and pilot contamination [19], [20].

Deep learning is used in extensive scientific fields for two main reasons: *(i)* deep learning is an efficient method to analyze data by identifying patterns and learning underlying structures, and *(ii)* the development of graphic processing units (GPUs) and increasingly specialized chips allowing to parallelize the execution of deep learning architectures, and achieving impressive computational throughput and energy efficiency [21], [22]. The physical layer of wireless communication systems integrates deep learning algorithms to address different tasks, including MIMO signal detection [23], error-correcting codes [24], and channel resource allocation [25].

Most recent research includes deep learning algorithms during the channel estimation procedure. Deep neural network (DNN) is one of the most popular deep learning techniques used for channel estimate [26]–[29]. DNN is a supervised machine learning technique that tries to estimate the non-linear mapping between the input and the output signals. The non-linear mapping is defined as a concatenation of layers, each layer is composed of a learnable vector linear transformation, followed by a preselected scalar non-linearity [29]. Another well-known deep learning tool is long short-term memory (LSTM) network which is defined with a special architecture capable of learning correlation over time. The main feature of LSTM networks is the capability of tracking the time-varying channels [30], [31]. Convolutional neural network (CNN) is a type of deep learning method commonly used for processing data with grid patterns. CNN is used for several visual applications such as image classification, due to its ability to extract features from the images [32]–[34]. The 3D geometry of mMIMO-OFDM channels makes CNN algorithm suitable for channel estimate. CNNs can extract patterns from the correlation in any of the three dimensions (frequency, time, and space). In [35], the authors design a 2D CNN and a 3D CNN model for handling spatial correlation. The CNN estimator interpolates the channel values of pilots for estimating the channel of the full OFDM resources. In [36], the designed dual CNN exploits the advantages of working both in the spacial-frequency domain and the benefits of the angle-delay domain.

B. Contributions

As far as we know, the previous work analyzes different deep learning structures to optimize the number of layers and hyperparameters providing the best estimation quality or computational complexity. Previous research examines deep learning models employing input data which often includes some additional knowledge of the channel. Motivated by the above considerations, in this work, we consider a CNN-based approach that only uses the minimal and always available information about the channel (the LS-estimated pilot channel state information (CSI)). We describe the proposed CNN-based structures for the estimation model and analyze their performance in the accuracy of channel estimation and the computational complexity.

- We propose two different CNN-based estimation models. We incorporate a preprocessing network, and an attention-like network to enhance the performance in the full range of channel conditions. We analyze and compare these models, considering channel estimation quality, computational complexity, and flexibility.
- We employ a training and validation framework capable of generalizing well for different channel properties, i.e., DS, Doppler frequency, and signal-to-noise ratio (SNR). The proposed CNN-based estimation models do not require prior statistical knowledge of the channel.
- We provide a deep analysis of the computational and cost complexity of the CNN-based models, proposing a hardware structure for a real implementation.
- We analyze the estimation accuracy in channels with different properties (i.e., scattering model and SNR). We evaluate the performance of the estimation models in terms of MSE and spectral efficiency (channel capacity of mMIMO systems) and compare them with benchmark techniques such as LS, MMSE, and DNN. Extensive simulated results and comparisons allow us to confirm the robustness and efficiency of our proposed approach.

II. System Model

Let us consider an OFDM single-cell consisting of one base station (BS) equipped with a mMIMO uniform linear array

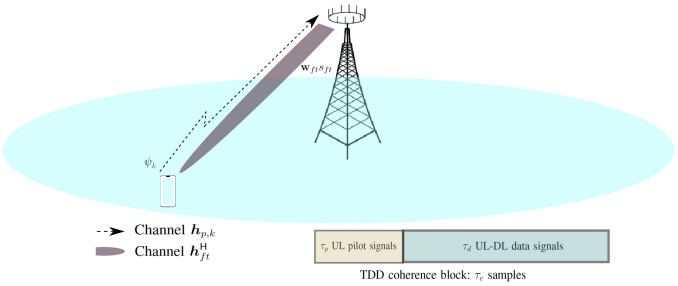


Fig. 1: System model of a single mMIMO BS and a single-antenna MS. Illustration of the UL mMIMO pilot transmission and DL mMIMO data transmission.

(ULA) composed of N_a antennas. We consider the link level between the BS and a single-antenna MS. We characterize the MS by the channel vector $\boldsymbol{h}_{ft} \in \mathbb{C}^{N_a \times 1}$ between the BS and the single-antenna MS on the subcarrier f and the OFDM symbol t. To simplify the channel estimation procedure, we assume that the BS works in time division duplexing (TDD) mode to benefit from channel reciprocity. TDD allows the BS to estimate uplink (UL) and downlink (DL) channels from the UL pilots, limiting the signaling overhead due to the channel information feedback required in frequency division duplexing (FDD) mode. FDD is not recommended in mMIMO systems because the signaling overhead increases drastically with the number of antennas. FDD operation can be employed in some certain cases [37]. Each TDD frame is divided into two phases, namely, the UL training phase and the UL-DL payload data transmission phase, whose lengths, measured in samples, are denoted as τ_p and τ_d, respectively. We denote the coherence block length as $\tau_c = \tau_p + \tau_d$. Figure 1 illustrates a single-cell mMIMO system model with a single-antenna MS. The UL mMIMO pilot transmission and the DL mMIMO data delivery are specifically shown.

A. Channel Estimation

Let us suppose that the BS delivers a service to K single-antenna MSs. We denote the set of MSs as $\mathcal{K} = \{1, \ldots, K\}$. Let $\boldsymbol{\psi}_k \in \mathbb{C}^{\tau_p \times 1}$ be the pilot sequence assigned to the MS k, with $\|\boldsymbol{\psi}_k\| = \boldsymbol{\psi}_k^{\mathsf{H}} \boldsymbol{\psi}_k = 1$. To exploit mMIMO potential, the BS needs to estimate the channel vector of MS k from its pilot sequence. Ideally, pilot sequences should be mutually orthogonal. Since τ_p mutually orthogonal UL pilot sequences can be provided with a pilot length τ_p, there is no interference (i.e., pilot contamination) if $K \leq \tau_p$.

The signal received at the BS during the UL pilot transmission, on the subcarrier f_p and the OFDM symbol t_p, is expressed as

$$\boldsymbol{R}_p^{UL} = \sqrt{P_p} \sum_{k \in \mathcal{K}} \boldsymbol{h}_{p,k} \boldsymbol{\psi}_k^{\mathsf{T}} + \boldsymbol{N}, \tag{1}$$

where $\boldsymbol{h}_{p,k} \in \mathbb{C}^{N_a \times 1}$ denotes the channel vector between the single-antenna MS k and the BS on the subcarrier f_p and the OFDM symbol t_p reserved for pilot transmission, P_p is the per pilot-symbol transmit power of every MS, and $\boldsymbol{N} \in \mathbb{C}^{N_a \times \tau_p}$ is the AWGN with i.i.d. elements distributed as $\mathcal{N}_{\mathbb{C}}(0, \sigma_{UL}^2)$. To estimate the channel of MS k, the BS projects the received UL training signal on the corresponding pilot sequence $\boldsymbol{\psi}_k^*$ to obtain

$$\boldsymbol{r}_{p,k}^{UL} = \boldsymbol{R}_p^{UL} \boldsymbol{\psi}_k^* = \sqrt{\tau_p P_p} \boldsymbol{h}_{p,k} + \boldsymbol{N} \boldsymbol{\psi}_k^*, \tag{2}$$

where $\boldsymbol{N} \boldsymbol{\psi}_k^* \sim \mathcal{N}_{\mathbb{C}}(0, \sigma_{UL}^2 \tau_p \mathbf{I}_{N_a})$ is the noise component that modifies randomly the pilot signal. To achieve channel estimate from the pilot signal, the BS can perform LS estimator that minimizes the squared deviation $\Big($i.e., minimizes $\mathbb{E}\left\{\|\boldsymbol{r}_{p,k}^{UL} - \sqrt{\tau_p P_p} \hat{\boldsymbol{h}}_{p,k}^{LS}\|^2\right\}\Big)$ [38, Sec. 3.4.1]. We can formulate the LS estimator as

$$\hat{\boldsymbol{h}}_{p,k}^{LS} = \frac{1}{\sqrt{\tau_p P_p}} \boldsymbol{r}_{p,k}^{UL}. \tag{3}$$

Simplicity is the main feature of the LS algorithm, but (3) does not consider the presence of the noise component. The LS channel estimator is exposed to significant estimation error, especially when the noise is comparable to the power transmission of pilots. To address with this issue, the MMSE estimator minimizes the MSE $\Big($i.e., minimizes $\mathbb{E}\left\{\|\boldsymbol{h}_{p,k} - \hat{\boldsymbol{h}}_{p,k}^{MMSE}\|^2\right\}\Big)$ [38, Sec. 3.2]. We can formulate the MMSE estimator as

$$\hat{\boldsymbol{h}}_{p,k}^{MMSE} = \boldsymbol{A}_k \boldsymbol{r}_{p,k}^{UL}, \tag{4}$$

where $\boldsymbol{A}_k$ is the smoothing matrix that suppresses the noise of $\boldsymbol{r}_{p,k}^{UL}$, that is given by

$$\boldsymbol{A}_k = \sqrt{\tau_p P_p} \boldsymbol{S}_k \left(\tau_p P_p \boldsymbol{S}_k + \sigma_{UL}^2 \mathbf{I}_{N_a}\right)^{-1}, \tag{5}$$

where $\boldsymbol{S}_k \in \mathbb{C}^{N_a \times N_a}$ is the positive semi-definite spatial covariance matrix of the channel between the MS k and the BS, independent of t and f, capturing the macroscopic propagation effects, namely the large-scale fading, which incorporates path-loss, shadow fading, and spatial correlations. Thus, MMSE estimator requires prior knowledge of the channel statistics (i.e., $\boldsymbol{S}_k$) to operate. The BS acquires this knowledge through more pilot transmission, reducing the available resources for data transmission. Furthermore, the MMSE estimator handles many complex operations that increase with the number of antennas [38, Sec. 3.4]. These constraints make MMSE estimator hard to implement in practical mobile scenarios.

We define the effective SNR perceived at the BS during the pilot transmission of MS k as

$$\gamma_k^{UL} = \frac{\tau_p P_p \beta_k}{\sigma_{UL}^2}, \tag{6}$$

where $\beta_k = \frac{1}{N_a} tr\left(\boldsymbol{S}_k\right)$ is the average channel gain. A high γ_k^{UL} is desirable to achieve a good channel estimation performance, especially when the BS performs the LS estimator that is extremely susceptible to noise errors. Note that in high-quality channels, i.e., negligible noise ($\sigma_{UL}^2 \sim 0$) or when $\gamma_k^{UL} \sim \infty$, the LS and MMSE estimators are equivalent $\left(\hat{\boldsymbol{h}}_{p,k}^{LS} \approx \hat{\boldsymbol{h}}_{p,k}^{MMSE}\right)$.

B. Channel Model

Clustered delay line (CDL) fading channel models characterize 5G mMIMO transmissions for the full frequency range from 0.5 GHz to 100 GHz with a maximum bandwidth of 2 GHz [39]. These models are based on scattering clusters to define the small-scale channel characteristics in the azimuth and the zenith dimensions. Each CDL model provides a small-scale fading description that can be scaled in delay and angle, so that the model achieves a desired normalized DS and angular spread (AS), respectively, according to the fast fading simulation environment. The Doppler frequency allows adapting the channel to manifold time-selectivity scenarios according to the relative movement of the MS concerning the BS. The spatial correlation of the channels between the different BS antennas and the single-antenna MS depends on the departure/arrival angles, ASs, and the spacing between antennas. Some of these parameters are fixed in the CDL models, i.e., the departure/arrival azimuth/zenith angle, and others are defined in the simulation setup, i.e., the DS, the ASs, the Doppler frequency, and the antenna element spacing. We employ the CDL-A channel model to represent non-line of sight (NLOS) channel profile (see the channel profile parameters in [39, Tab. 7.7.1.1]). Among the parameters that can be defined in the simulation setup, we fix the ASs and the antenna element spacing. These values will be detailed in Table IV (Section V). We employ different DSs and Doppler frequencies to achieve a variety of channel selectivity in time and frequency domains. Table V (Section V) will detail these channel configurations. Finally, the large-scale fading affects the expected SNR. The path-loss and shadowing determine the power of the received signal and, consequently, the SNR experienced by the MS. The higher the SNR, the better the coverage.

C. Combining/Precoding and Spectral Efficiency

For simplicity, from here on, we avoid the explicit use of subscript k and we consider the data transmission in the DL. Due to the channel reciprocity of TDD, the BS can use the knowledge acquired during the channel estimation procedure in the UL to design the beamforming vector $\mathbf{w}_{ft} \in \mathbb{C}^{N_a \times 1}$ that serves the MS in the DL. The conjugate beamforming (CB) is the optimal linear combination that selects the combining/precoding vector that maximizes the SNR received at the MS and increases the perceived throughput. The CB combining/precoding vector is defined as

$$\mathbf{w}_{ft} = \sqrt{P_d} \frac{\hat{\boldsymbol{h}}_{ft}}{\sqrt{\mathbb{E}\{\|\hat{\boldsymbol{h}}_{ft}\|^2\}}}, \tag{7}$$

where $\mathbb{E}\left\{\|\mathbf{w}_{ft}\|^2\right\} = P_d$, and P_d denotes the available transmit power at the BS. The received signal by the MS is expressed as

$$r_{ft}^{DL} = \mathbb{E}\left\{\boldsymbol{h}_{ft}^{\mathsf{H}} \mathbf{w}_{ft}\right\} s_{ft} + n, \tag{8}$$

where s_{ft} is the data signal sent to the MS, $\mathbb{E}\left\{\boldsymbol{h}_{ft}^{\mathsf{H}} \mathbf{w}_{ft}\right\}$ is the beamformed channel, and $n \sim \mathcal{N}_{\mathbb{C}}(0, \sigma_{DL}^2)$ is the AWGN present at the MS. From the received signal, we can derive a closed-form approximation of the lower bound spectral efficiency (SE) [40, Eq. (4)] perceived by the user as

$$\text{SE} \geq \left(1 - \frac{\tau_p}{\tau_c}\right) \log_2 \left(1 + \frac{P_d/\sigma_{DL}^2}{1 + \sigma_\varepsilon^2 \left(P_d/\sigma_{DL}^2\right)} \mathbb{E}\left\{\frac{\boldsymbol{h}_{ft}^{\mathsf{H}} \boldsymbol{h}_{ft}}{\|\boldsymbol{h}_{ft}\|}\right\}^2\right), \tag{9}$$

where σ_ε^2 denotes the MSE of the channel estimate $\left(\mathbb{E}\left\{\|\boldsymbol{h}_{ft} - \hat{\boldsymbol{h}}_{ft}\|^2\right\}\right)$.

III. Convolutional Neural Networks for Massive MIMO Channel Estimation

Machine learning aims to find the optimal relationship between an input $\mathbf{x}$ and its output $\hat{\mathbf{y}}$, using a function $\hat{\mathbf{y}} = f_{\hat{\boldsymbol{\theta}}}(\mathbf{x})$ that depends on input $\mathbf{x}$ and model parameters $\hat{\boldsymbol{\theta}}$. In supervised learning, these parameters are refined with labeled data. Given the complexity of wireless channels, we propose CNNs because of their efficiency in handling data with grid-like structures through convolution operations. CNNs are adept at feature extraction and classification [41], with a subset specializing in generative tasks to create data matching specific patterns [42], [43]. A key element for generative CNNs is the Conv2DTranspose layer, also termed transposed convolution or deconvolution[1], which upscales lower-resolution inputs into higher-resolution outputs, using activation functions like rectified linear activation function (ReLU) to enhance feature representation and improve output quality.

Building on the capabilities of CNNs to manage complex wireless channel data, we present two innovative models designed to improve mMIMO channel estimation accuracy. The first model leverages a CNN-based framework for preprocessing spatial information, and the second incorporates an attention mechanism to enhance reconstruction accuracy. The next sections further describe the methodologies and advantages of each model, highlighting the strategic use of CNNs for effective channel estimation.

A. SW-NN+CNN Model

On this first model, we propose a CNN-based approach that harnesses the spatial characteristics of mMIMO channels to enhance channel estimate accuracy. Our methodology involves two key steps: preprocessing spatial pilot information and channel reconstruction using a generative CNN. The subsequent sections will delve into the intricacies of the preprocessing and generative networks, showcasing their effectiveness and benefits in the context of channel estimates.

Preprocessing Network: We introduce a preprocessing with a SW-NN to exploit the spatial correlation between nearby antennas. The network processes pilot signals received across the N_a BS antennas within an OFDM resource block, aiming to estimate channel characteristics efficiently.

[1]Must be noted that despite "deconvolution" is often used colloquially to refer to the operation performed by the *Conv2DTranspose*, is not a strict mathematical deconvolution but rather an operation that increases the spatial resolution of the data.

TABLE I: Detailed preprocessing network structure parameters.

Layer	Function	Input size	Output size	Details
Input layer	LS pilot channel estimation	-	$N_{pf} \times N_{pt} \times N_a$	Processes $\hat{\boldsymbol{h}}_{p,k}^{LS}$ estimations for N_{pf} subcarriers, N_{pt} OFDM symbols, across N_a antennas.
shared-weight neural network (SW-NN) dense layers	Data compression	$N_{pf} \times N_{pt} \times N_a$	$N_{pf} \times N_{pt} \times N_c$	Dense layers with ReLU, compressing data from N_a to N_c ($N_c < N_a$), using weight sharing across all pilot instances.
Flatten layer	Data transformation	$N_{pf} \times N_{pt} \times N_c$	Linear vector	Converts multi-dimensional output into a single linear vector for further processing.
Final dense layer	Channel characteristic compression	Linear vector	$N_{pf} \times N_{pt}$	Applies a linear combination and ReLU activation to generate a compact representation of the channel characteristics.

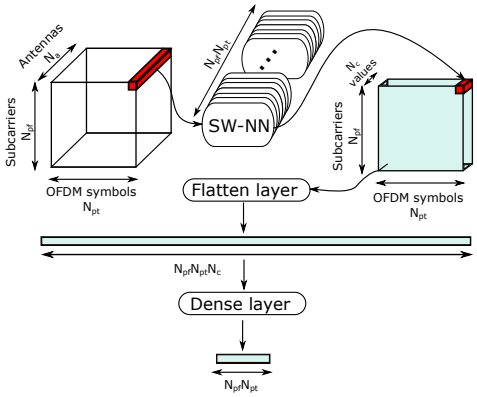


Fig. 2: Structure of the preprocessing network: input and output mapping.

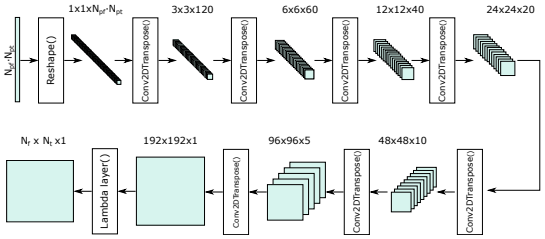


Fig. 3: Structure of the generative network: input and output mapping.

The structure of this neural network, presented in Figure 2 and described in Table I, begins with an input layer that feeds pilot signal-derived channel estimates into the network and progresses through the SW-NN dense layers for data compression. The flatten layer then transforms the data into a linear vector, effectively preparing it for the concluding phase executed by the final dense layer, which compacts the channel's characteristics into an efficient representation.

Generative Network: The condensed data from the preprocessing network is input into a CNN designed for channel reconstruction. The architecture of our generative CNN detailed in Figure 3 and Table II, initiates with the reshape layer to format the preprocessing output for convolution. It then progresses through Conv2DTranspose layers for upsampling, incorporating non-linearity with ReLU activation. This sequence leads to a final Conv2DTranspose layer for precise channel estimation in the time-frequency domain, concluded by spatial cropping to tailor the CNN output to the exact dimensions required, effectively encapsulating the network channel reconstruction capabilities.

In a practical mMIMO communication system with N_a BS antennas, the BS must estimate N_a distinct channels. We replicate the preprocessing network structure N_a times to accommodate this requirement. Each replication learns a unique relationship between the channel from a specific antenna and the pilot information. The outputs of these N_a neural networks are then passed through the same CNN-based generative network, resulting in N_a different channel estimations.

B. SW-NN+AN+CNN Model

In this second model, we add an attention-like network between the preprocessing network and the generative CNN network to improve the reconstruction capability of the system.

Attention-like Network: The attention mechanism architecture, depicted in Figure 4 and summarized in Table III, starts with a combined input expansion layer. This layer enriches the input data by integrating SW-NN outputs and raw pilot signals. Through a series of lambda and dense layers, the network modulates and transforms the data, facilitating the refinement of channel information.

The proposed network structure is not a traditional attention mechanism like the one used in sequence-to-sequence models for natural language processing [44]. Nevertheless, it shares some key aspects:

- Selective processing: The network selectively processes different input parts based on learned patterns. Specifically, it applies sigmoid activations to control the influence of varying input data segments, effectively attending to or ignoring specific information.
- Modulation: Using sigmoid activations to modulate the impact of data components resembles the attention mechanism. The sigmoid activations determine the weights or importance of different parts of the input data, allowing the network to focus on relevant information while suppressing less relevant parts.
- Adaptability: Controlling the influence of different input segments is a characteristic commonly associated with

TABLE II: Detailed generative network structure parameters.

Layer	Function	Input size	Output size	Details
Reshape layer	Data reformatting	$N_{pf} \times N_{pt}$	$N_{pf} \times N_{pt} \times 1$	Reformats preprocessing network output for convolutional processing.
Conv2DTranspose Layers	Upsampling	Variable	Variable	Series of layers that increase spatial dimensions using ReLU activation, tailored to the channel reconstruction process.
Final Conv2DTranspose layer	Channel estimation	Variable	$K_f N_f \times K_t N_t \times 1$	Applies linear activation to complete the channel estimation in the time-frequency domain ($K_f, K_t \in \mathbb{N}$).
Spatial cropping	Output formatting	$K_f N_{pf} \times K_t N_{pt} \times 1$	$N_f \times N_t \times 1$	Adjusts the final output dimensions to ensure compatibility with system requirements.

TABLE III: Detailed attention-like network structure parameters.

Layer	Function	Input Size	Output Size	Details
Combined input exp. layer	Linear transformation	$2(N_{pf} \times N_{pt})$	$4(N_{pf} \times N_{pt})$	Doubles the input size by performing a linear transformation, preparing the data for further processing.
Lambda layer (1)	Activation, element-wise multiplication	$4(N_{pf} \times N_{pt})$	$2(N_{pf} \times N_{pt})$	Applies sigmoid activation to the second half of the input and multiplies it by the first half, controlling the influence of data.
Dense layer (1)	Linear transformation	$2(N_{pf} \times N_{pt})$	$2(N_{pf} \times N_{pt})$	Performs a linear transformation without altering the input size, maintaining data dimensionality.
Lambda layer (2)	Activation, element-wise multiplication	$2(N_{pf} \times N_{pt})$	$N_{pf} \times N_{pt}$	Same behaviour to the first lambda layer.
Dense layer (2)	Linear transformation to desired output	$N_{pf} \times N_{pt}$	$N_{pf} \times N_{pt}$	Transform the data representation maintaining the desired output size.

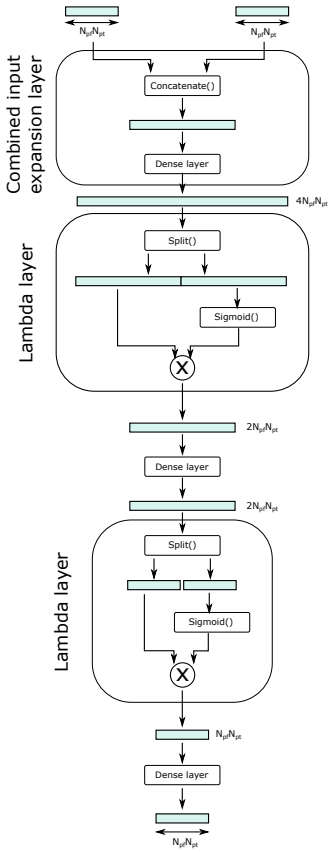


Fig. 4: Attention-like network structure.

attention mechanisms. In attention, the model learns which parts of the input to attend to, and similarly, in this network, the sigmoid activations adapt to the data characteristics during training.

Figure 5 presents the complete architecture of the SW-NN+AN+ CNN model. The intermediary network introduced in this architecture offers several advantages in the context of channel estimates. First, it facilitates enhanced information fusion by combining the learned features from the SW-NN and the raw unprocessed pilot data from the target antenna. This fusion is achieved using sigmoid activations, enabling the network to merge adaptively to the valuable features from both sources. This adaptability can lead to more comprehensive and informative representations of the channel. This addition can provide potentially finer-grained information about the characteristics of the channel, allowing for more accurate channel estimates.

Moreover, sigmoid activations within this network contribute to adaptive control over the influence of different data components. This adaptability is particularly advantageous in scenarios where certain components may contain noise or irrelevant information. By modulating the impact of various data sources, the network can effectively filter out undesirable elements, contributing to more robust channel estimations. However, it is essential to consider that this intermediary network does increase the complexity of the architecture. While complexity can sometimes lead to improved performance, it also poses the risk of overfitting. Nevertheless, the intermediary network can counterbalance this risk by allowing the neural network to learn how to appropriately balance the shared information and antenna-specific data, thus potentially mitigating overfitting concerns.

IV. Computational Complexity and Cost for the Neural Estimation Models

Assessing the computational complexity of channel estimation models is interesting for evaluating their efficiency. This

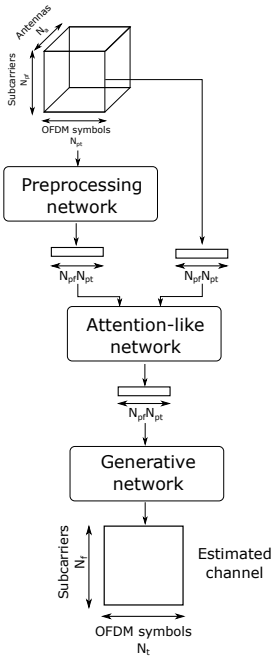


Fig. 5: The complete pipeline of the proposed CNN-based model.

complexity reflects the quantity of computational resources an algorithm requires, usually measured in time (time complexity) or storage (space complexity). The complexity varies with the input data size. A thorough understanding of this aspect is essential for estimating how an increase in problem size might impact the execution time of a model, as well as for comparing it against other models.

To establish a benchmark for comparison, we first introduce the time complexities of two methodologies that will serve as references for evaluating the performance of our models in the results section. These methodologies are the MMSE method and the DNN-based channel estimators, as detailed in [29, Model 2]:

- MMSE: Well-known for necessitating the computation of an inverse matrix, the MMSE method generally exhibits a cubic complexity, denoted as $O(n^3)$. Specifically, in our analysis where the MMSE algorithm is applied to the frequency domain and iterated over the time domain, we denote its complexity as $O(N_{pf}^3 N_{pt})$.
- DNN-based: Employing a deep multilayer perceptron (MLP) with several layers, this model is tailored for operations by dividing any given channel into smaller, fixed-size blocks. Consequently, its computational complexity, concerning the channel frequency and temporal dimensions, is determined to be $O(N_{pf}N_{pt})$.

A. Computational Complexity

We divide the analysis into different model components separately to gain insights into the computational complexity of the proposed estimation models.

1) Preprocessing Network: It comprises two fully connected neural networks. The SW-NN processes all system pilots, generating a concise representation for each. The second structure combines the outputs of the SW-NN to encapsulate them into a vector representing the channel. In a general form, the number of multiplications within each layer of a fully connected neural network can be expressed as $N_{in}N_{out}$, where N_{in} denotes the number of input values, and N_{out} is the number of output values.

In the preprocessing network, the initial analysis is applied to signals from different antennas, with the main parameters being N_a and the compact representation size N_c. This initial analysis is executed for each pilot, totaling $N_{pf}N_{pt}$ operations. Thus, the computational complexity of the SW-NN can be summarized as $O\left(N_a N_c N_{pf} N_{pt}\right)$. Subsequently, all values are merged and processed by a fully connected layer with an output size of $N_{pf}N_{pt}$. The complexity of this layer is $O\left(N_c N_{pf}^2 N_{pt}^2\right)$. Therefore, the preprocessing network computational complexity exhibits a quadratic relationship with the number of pilots.

2) Generative Network: In a general context, the number of multiplications in a transposed convolution layer can be described as $N_{ch,in}F_xF_yT_{x,out}T_{y,out}N_{ch,out}$, where $N_{ch,in}$ represents the number of input channels, F_x and F_y denote the filter size in both dimensions, $T_{x,out}$ and $T_{y,out}$ are the output channel sizes, and $N_{ch,out}$ is the number of output channels.

The number of operations to upscale data from the initial to the final size depends on the upsampling step and the number of layers. In our proposed model, we fix the filter size, allowing only the number of layers, denoted by N_L, to increase with the output data size. Since the final data size is $N_f N_t$ with a single channel and the initial number of channels is $N_{pf}N_{pt}$, we can express the total complexity as a function of these elements and the number of layers N_L as $O\left(N_L N_{pf} N_{pt} N_f N_t\right)$.

The number of layers N_L can also be expressed as a function of the output size. We can obtain the relationship between the output size and the number of layers using this expression $N_{out} = N_0 U^{N_L}$, where N_0 is the initial input size, U is the constant upsampling step, and N_L is the number of layers. Thus, we can obtain the number of layers as $N_L = \log\left(\frac{N_{out}}{N_0}\right)/\log(U)$. Therefore, the number of layers needed exhibits a logarithmic relationship with the output size. We can reformulate the complexity of this layer using only the number of channels and the output shape as $O\left(\log\left(\frac{N_f N_t}{N_{pf}N_{pt}}\right) N_{pf}N_{pt}N_f N_t\right)$.

3) Attention-like Network: It assembles linear combination layers applying a sigmoid activation function, followed by element-wise multiplication. The linear combination structure resembles fully connected layers, yielding a complexity of $O(N_{in}N_{out})$.

The computational complexity of the sigmoid function and the element-wise multiplication are independent of problem size and can be considered linear-dependant with the input size $O(N_{in})$. Consequently, the most complex layers are the linear combinations. The channel representative vector defines the input size as $N_{pf}N_{pt}$ in the proposed structure. The output of every layer is designed to be a multiple of the input size,

resulting in a computational complexity of this network given by $O\left(N_{pf}^2 N_{pt}^2\right)$.

B. Analysis of Computational Complexity Across Models

1) Complexity of the SW-NN+CNN Model: This model is constructed from two principal components: the preprocessing network, with a complexity of $O(N_c N_{pf}^2 N_{pt}^2)$, and the generative network, with a complexity of $O\left(\log\left(\frac{N_f N_t}{N_{pf} N_{pt}}\right) N_{pf} N_{pt} N_f N_t\right)$. Given that the ratios between N_f, N_t and N_{pf}, N_{pt} are fixed due to the constraints imposed by channel variability, the overall computational complexity can be effectively reduced to $O(N_c N_{pf}^2 N_{pt}^2)$.

2) SW-NN+AN+CNN Model Complexity: This second model incorporates the attention-like network alongside the previous two networks. The addition of the aggregated network complexity of $O\left(N_{pf}^2 N_{pt}^2\right)$, increases the coefficient by a marginal constant factor, leading to an overall complexity of $O\left((N_c+1) N_{pf}^2 N_{pt}^2\right)$, which simplifies back to $O\left(N_c N_{pf}^2 N_{pt}^2\right)$, mirroring the complexity of the SW-NN+CNN model.

Through this comparative analysis, we have established that the computational complexity of the SW-NN+CNN and SW-NN+AN+CNN models are identical. This equivalence, however, overlooks the practical impacts of constant factors that could significantly influence the real-world performance of the models. Therefore, while our complexity analysis offers a foundational understanding of the scalability of these models, it also prompts the need for a deeper examination of their practical deployment. The forthcoming subsection will address these considerations, offering a comprehensive overview of the technological feasibility of deploying these advanced neural network structures.

C. Computational Cost: Hardware Needs and Working Clock Frequency

A clear distinction must be drawn between computational complexity and computational cost. Computational complexity, expressed through "Big O notation," offers a high-level perspective on how the resource requirements of an algorithm scale with input size, aiding in the identification of computational bottlenecks and scalability assessment. However, it does not account for real-world resource consumption. On the other hand, computational cost delves into the practical aspects of execution, considering hardware, software, and implementation specifics. It is crucial to recognize that an algorithm with a favorable complexity order, such as $O(n)$, can incur high computational costs under certain conditions, as factors like hardware efficiency, memory usage, and cache management come into play. A comprehensive algorithm's performance evaluation necessitates a dual focus on computational complexity and cost, ensuring a holistic understanding of its behavior and efficiency.

For instance, the literature indicates that the real and imaginary parts can be estimated jointly [45]. However, to enhance computational efficiency, our models are designed to estimate either the real or the imaginary part of the channel independently, leveraging the corresponding component of the pilot information This necessitates using the model twice for a full channel estimation: once for the real part and once for the imaginary part. Opting for this bifurcated approach is a strategic decision aimed at reducing computational complexity (noting that complexity increases with the square of the number of pilots, which would quadruple the cost if both real and imaginary parts were analyzed simultaneously, as opposed to just doubling the number of model evaluations) while taking advantage of the independence between the real and imaginary components. This methodology offers efficiency benefits, speeding up the channel estimation process. Although certain channel characteristics, such as SNR estimation, might be slightly affected, this strategy is ultimately advantageous, ensuring both precision and computational efficiency.

To carry on this analysis in a standardized way, we proposed a micro neural-core structure to parallelize the execution of convolutional networks and assess the hardware requirements (chip complexity and clock frequency). The core concept behind this micro-neural structure is to enable parallel processing of the convolution operation[2]. Convolutions are performed on data blocks with dimensions $F_x \times F_y$. Our proposed structure efficiently computes the product of the $F_x \times F_y$ filter weights with the $F_x \times F_y$ input data block in parallel. Subsequently, the results are combined using a merging structure, significantly reducing the addition process to logarithmic time, as detailed in [46]. From now on, we will refer to these structures as logarithmic addition units (LAUs) to improve text clarity.

Assuming that this implementation takes two clock cycles for multiplication and one clock cycle for addition, the total number of clock cycles per operation is $2 + \log_2(N_{in})$, where $N_{in} = F_x F_y$, aligning the input size with the filter size. In our pursuit of optimization for speed, we introduce a pipeline structure that allows concurrent computation of products and additions. In this setup, the clock cycles required for the operation can be expressed as $CLK(F_x F_y) = \max\{2, \lceil log_2(F_x F_y) \rceil\}$.

We propose replicating the product-sum structure M times to mitigate costly data transfers. This allows the incorporation of another LAU that will reduce the outputs from the M channels to a final value. Leveraging the pipeline technique mentioned earlier, the number of clock cycles is given by $CLK(F_x F_y) = \max\{2, \lceil log_2(F_x F_y) \rceil, M\}$. The illustrated micro neural-core structure is presented in Figure 6.

Another important aspect is the computational cost of the sigmoid function. Although it is an operation $O(n)$, each application of this operation is costly, and even specific hardware approximations usually need two or three clock cycles to compute it. In case of linear or ReLU functions are not sufficient, it could be interesting to add the system a set of parallel sigmoid computation units (PSCUs). Note that this parallelization can be easily carried on with specific hardware

[2]Although the structure is designed to parallelize and reduce data transfer on convolutional networks, this structure can be also applied to compute efficiently fully connected outputs.

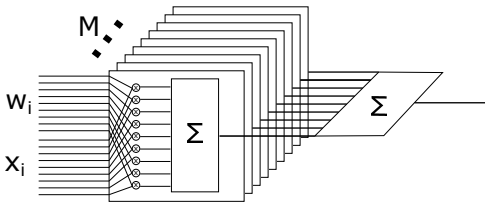


Fig. 6: This diagram illustrates the micro neural-core configuration, in which $\mathbf{M}$ denotes the count of product-sum modules aligned in parallel. The notation $\mathbf{W_i}$ refers to the neural network's weights, and $\mathbf{X_i}$ represents the respective data values

or by software employing classic multicore processing.

We illustrate this concept in Section V. An example allows us to demonstrate the computational cost associated with clock cycles and the number of micro neural-cores and LAUs required for the model implementation while addressing a specific channel estimation problem.

V. Experiments and Numerical results

In this Section, we present some numerical results to evaluate the performance of the proposed CNN-based channel estimators. The simulated environment considers the link level in the DL between the BS, equipped with 64 antenna ULA, and the single-antenna MS. Table IV illustrates the main system parameters and channel configuration, showing the time/frequency/spatial structure of the channels used in our proposed system. These parameters offer insight into the number of resources employed in the time, frequency, and spatial domains, apart from the pilot resource allocation.

We explain how we curated our training and validation datasets in the following Subsections. Then, we provide insights into the implementation of our models and the selection of hyperparameters. Finally, we offer a complete view of the effectiveness of our approach in this context with numerical results regarding complexity and estimation performance.

A. Building the Dataset: Training and Validation Sets

We conduct training on our neural network using a diverse dataset sourced from various mMIMO channels, each characterized by small-scale fading parameters, namely DS and Doppler frequency, along with large-scale fading parameters such as the average SNR at the receiver.

Our training dataset includes samples with pseudorandomly assigned values for DS, ranging from 32 ns, reflecting low-frequency selective channels, to 650 ns, reflecting high-frequency selective channels. Furthermore, the dataset includes varying Doppler frequencies, extending from 5 Hz (representing low mobility channels) to 480 Hz (representing high mobility channels). The range of the SNR in our dataset spans from -10 dB (characterizing bad coverage MSs) to 20 dB (representing MSs with good coverage). For the sake of simplicity, we consider that the MS during the DL data transmission will also experience $\gamma_k^{DL} = \frac{p\beta}{\sigma^2_{DL}} = \gamma_k^{UL}$. That is if the UL pilots that the BS use for channel estimation present $\gamma_k^{UL} = 5$ dB, we consider that the MS during the DL data transmission will also experience $\gamma_k^{DL} = 5$ dB.

All training samples feature an antenna spacing of $\lambda/2$, with λ denoting the wavelength (established at an operating frequency of 3.5 GHz using 5G numerology 1) to ensure consistency. Moreover, the dataset adheres to specific values for the normalized ASs that collectively determine the spatial correlation, including an azimuth spread of departure angles at 5°, an azimuth spread of arrival angles at 11°, and a zenith spread for both departure and arrival angles at 3°.

Within the context of our extensive training dataset, we encompass a broad spectrum of channel conditions. The entire range of DSs, Doppler frequencies, and SNRs is subdivided into 125 distinct subregions, each taking the form of a $\frac{1}{5}\times\frac{1}{5}\times\frac{1}{5}$ cube. A random point is generated using a uniform distribution within each subregion. Consequently, we obtain 125 distinct data points, with each point residing inside one of the 125 defined cubes, collectively covering the full range of DSs, Doppler frequencies, and SNRs. This strategy ensures comprehensive coverage of various channel behaviors and expedites the training process by facilitating the representation of diverse channel conditions.

In contrast, to improve clarity on the model's behavior the validation dataset includes samples with predefined values of DS, Doppler frequency, and SNR as detailed in Table V. These independent validation samples exhibit heterogeneous characteristics, ranging from the less selective channels represented by $S1$ to the highly selective channel denoted as $S7$, including specific SNR values spanning from -5 dB to 20 dB. This validation dataset enables us to assess the network capacity to generalize estimates across a wide range of channel conditions.

B. Models Implementation and Hyperparameters

In this Section, we will explore the two separate models for channel estimate, looking at their structures and the hyperparameters that control their behavior.

The preprocessing network is a crucial component of both SW-NN+CNN and SW-NN+AN+CNN models, responsible for condensing and processing pilot information from multiple antennas. The most important hyperparameter for this network is the value of N_c that denotes the initial compression level applied to the pilot information. In this work, we set N_c to 8. Table VI summarizes the whole preprocessing network structure and the hyperparameter values used.

The second shared block in SW-NN+CNN and SW-NN+AN+CNN models is the generative network, a pivotal component responsible for the channel reconstruction from a set of $N_{pf}N_{pt}$, that is, $13 \times 11 = 143$, input values. Several critical hyperparameters require meticulous attention to ensure optimal performance Within this generative network. These parameters include strides, filter size, and the number of "feature maps" on each layer. You can find the specific values set for these parameters in Table VII where the third dimension of the output size represents the number of feature maps.

TABLE IV: System parameters and channel configuration.

Identifier	Description	Value
f_c	Carrier frequency	3.5 GHz
μ	5G numerology	1
T_{OFDM}	OFDM symbol time duration	35.7 μs
Δ	Subcarrier spacing	30 KHz
N_a	Number of BS antennas	64
N_f	Number of frequency subcarriers in the target block	145
N_t	Number of OFDM symbols in the target block	141
N_{pf}	Number of pilots in the frequency domain in the target block	13
N_{pt}	Number of pilots in the time domain in the target block	11
CDL	Clustered delay line model	CDL-A
ASD	Scaled root mean square azimuth spread of departure angles	5°
ASA	Scaled root mean square azimuth spread of arrival angles	11°
ZSD	Scaled root mean square zenith spread of departure angles	3°
ZSA	Scaled root mean square zenith spread of arrival angles	3°
δ	BS antenna element spacing	$\lambda/2$

TABLE V: Predefined values of the validation dataset.

Validation samples	DS	Doppler frequency
S1	32 ns	5 Hz
S2	100 ns	30 Hz
S3	240 ns	60 Hz
S4	410 ns	90 Hz
S5	650 ns	120 Hz
S6	650 ns	240 Hz
S7	650 ns	480 Hz

TABLE VI: Implemented preprocessing network structure.

Layer	Input size	Output size	Act. func.
Dense layer 1	N_a (64)	32	ReLU
Dense layer 2	32	16	ReLU
Dense layer 3	16	N_c (8)	ReLU
Combination & flatten layer	$N_cN_{pf}N_{pt}$ (1144)	$N_cN_{pf}N_{pt}$ (1144)	None
Dense layer 4	$N_cN_{pf}N_{pt}$ (1144)	$N_{pf}N_{pt}$ (143)	ReLU

- **Strides**: Integral elements to the time-frequency resolution of the network. They determine how much the time-frequency dimensions are increased or decreased at each layer. Properly setting the strides is essential for controlling the trade-off between computational efficiency and the level of detail in the generated channel. Smaller strides preserve finer details, while larger strides reduce time-frequency dimensions and can speed up processing.
- **Filter Size**: In a generative network, filter size is key to defining the range of input data processed. Larger filters contribute to generating broader, more general features for the overall structure. Smaller filters excel in creating detailed, fine-grained features, thus increasing the precision and intricacy of the patterns generated.
- **Number of Feature Maps**: Variety of data matrices processed in each convolutional layer, essential for capturing diverse patterns for channel regeneration.

The attention-like network, unique to SW-NN+AN+CNN model, operates based on several essential hyperparameters crucial to ensure optimal performance. These parameters encompass aspects like intermediate data size and the number of layers. Table VIII reviews these specific parameter values.

Configuring Training Key Parameters and Settings: We have adopted a systematic and well-organized approach involving an exhaustive examination of the key parameters and their associated values. Our primary goal during this training phase is to enhance the model performance while using efficiently the available computational resources. Table IX details the most relevant parameters and their respective values to ensure a straightforward and concise explanation of our training process.

The training process commences with the specified initial learning rate of 0.001. At each iteration, the learning rate is intelligently decreased by a 0.999 factor, effectively reducing it by 0.1% to enhance model convergence and fine-tuning. Each iteration loads a data block comprising 125 channel examples, ensuring the model is exposed to a wide range of training data. During model training, we use a batch size of 32, allowing us to process simultaneously 32 examples. The chosen objective function for this training endeavor is the MSE, a metric that quantifies the squared differences between predicted and actual values. This function guides the model optimization process,

TABLE VII: Generative network structure.

Layer	Output size	Filter size	Strides	Act. func.
Reshape Layer	(1, 1, 143)	-	-	None
Conv2DTr layer (1)	(3, 3, 120)	(3, 3)	3	ReLU
Conv2DTr layer (2)	(6, 6, 60)	(3, 3)	2	ReLU
Conv2DTr layer (3)	(12, 12, 40)	(3, 3)	2	ReLU
Conv2DTr layer (4)	(24, 24, 20)	(3, 3)	2	ReLU
Conv2DTr layer (5)	(48, 48, 10)	(3, 3)	2	ReLU
Conv2DTr layer (6)	(96, 96, 5)	(3, 3)	2	ReLU
Conv2DTr layer (7)	(192, 192, 1)	(3, 3)	2	Linear
Lambda layer	(145, 141, 1)	-	-	None

TABLE VIII: Attention-like network structure with detailed data processing.

Layer	Output size	Data processing (details)	Activ. function
Input layer	$2N_{pf}N_{pt}$ (286)	-	-
Dense layer 1	$4N_{pf}N_{pt}$ (572)	-	Linear
Lambda layer 1	$2N_{pf}N_{pt}$ (286)	Passes the second half through sigmoid. Multiplies it with the first half.	Custom
Dense layer 2	$2N_{pf}N_{pt}$ (286)	-	Linear
Lambda layer 2	$N_{pf}N_{pt}$ (143)	Passes the second half through sigmoid. Multiplies it with the first half.	Custom
Dense layer 3	$N_{pf}N_{pt}$ (143)	-	Linear

TABLE IX: Training parameters.

Parameter	Value	Description
Initial learning rate	0.001	The initial rate at which the model learns.
Learning rate decrease	0.999	The rate at which the learning rate diminishes at each iteration, reducing by 0.001 (0.1%).
Number of iterations	10000	The total number of training iterations.
Data block size per iteration	125	The size of the data block loaded at each iteration containing 125 channel examples.
Batch size	32	The number of examples processed in each training batch.
Objective function	MSE	The mathematical function used to measure the dissimilarity between predicted and actual values.

TABLE X: Computational cost and clock cycles needed for the preprocessing network.

Layer id	#μ neural-cores	# LAUs	Clock cycles
Layer 1	3616	0	16
Layer 2	904	0	4
Layer 3	226	0	4
Layer 4	2020	1	12
Total	6766	1	36

facilitating its quest to estimate channels accurately.

C. Computational Complexity Results

In this Subsection, we compute the hardware requirements and the necessary clock frequency for operation. Note that these initial calculations are for one channel and one of the complex components of the channel. So the results will be later multiplied by the number of antennas (i.e., $N_a = 64$) and by two (for the real and imaginary parts).

Since our model filter size is 3×3 we set the micro neural-cores input size of 9. The value of M is also set to 9, so all the LAUs inside the micro neural-cores maintain the same structure. This implies that a micro neural-core can carry on 81 multiplications and perform all summations on 4 clock cycles. For the external LAUs, maintaining a 9 input structure, the 9 values summation is done in 4 clock cycles. We also consider 32 PSCUs to speed up the data processing for the attention-like network. It is necessary to set in the system the number of micro neural-cores and LAUs to calculate the clock cycles required for each layer. For the computational cost analysis, we employ 1024 micro neural-cores for the main neural computation, and 128 LAUs to facilitate the final data summation.

1) Preprocessing Network: The computational cost of the preprocessing network, measured in micro neural-cores and LAUs hardware units, is shown in Table X. These values are then translated into clock cycles, considering the available units and the clock cycles per unit.

2) Generative Network: Table XI outlines the computational cost associated with the generative network. Similarly, this cost is converted to clock cycles.

3) Attention-like Network: We should consider the time required to calculate the sigmoid activation function. For our analysis, we assume that it takes 2 clock cycles to perform this function. Furthermore, it is essential to consider the multiplication process of vectors, which can be highly time-consuming if parallelization is not implemented. Table XII provides an overview of the computational cost for the attention-like network. This cost is measured in micro neural-cores, LAUs, products, and PSCUs hardware units, and converted into clock cycles.

TABLE XI: Computational cost and clock cycles needed for the generative network.

Layer id	#μ neural-cores	# LAUs	Clock cycles
Layer 1	17280	1080	104
Layer 2	30240	2160	188
Layer 3	40320	0	160
Layer 4	57600	0	228
Layer 5	69120	0	272
Layer 6	92160	0	360
Layer 7	21025	0	80
Total	327745	3240	1392

4) Global Clock Cycles and System Frequency: Using only the basic parallelization previously described, Table XIII presents a comparison of the clock cycles needed to compute the complete channel estimation using the proposed CNN-based models and the DNN-based in [29, Model 2]. The proposed models must be able to realize channel estimate each $N_t T_{OFDM} = 5$ ms, so the clock frequency both for SW-NN+CNN and SW-NN+AN+CNN models is given by $f_{clk} = 2 \times \frac{CLK}{5 \times 10^{-3}}$ Hz, where CLK is the clock cycle needed to carry on an evaluation of the model. Note that we double the number of clock cycles due to the real and imaginary parts estimates. Upon examination of the results, we observe the proposed models induce an increase in the total clock cycle number. Specifically, the SW-NN+CNN model demonstrates a slight elevation of 3.6% in total clock cycles, and the more complex SW-NN+AN+CNN model necessitates a more considerable increase of almost 47%.

Table XIV illustrates the clock frequency required for the channel estimates using both SW-NN+CNN and SW-NN+AN+CNN models per BS antenna and MS in comparison with the utilization of the DNN-based model in [29, Model 2].

D. Channel Estimation Results

In this subsection, we present numerical results to evaluate the channel estimation performance with the proposed CNN-based models, i.e., SW-NN+CNN and the SW-NN+AN+CNN. We analyze the channel estimation results based on the MSE and the SE (bit/s/Hz), comparing with the benchmark solutions consisting of LS-based, MMSE-based, and DNN-based [29, Model 2] channel estimators. For reference, we also include the upper limit representing the theoretical maximum achievable SE under the assumption of perfect channel knowledge.

Figure 7 illustrates the MSE between the actual and the estimated channel using the benchmark estimators, i.e., LS-based, MMSE-based, and DNN-based, and the proposed SW-NN+CNN and SW-NN+AN+CNN estimators. Figure 7a shows the results achieved in a low-variable (time and frequency) channel (S1 validation samples) while Figure 7b

TABLE XII: Computational cost and clock cycles needed for the attention-like network.

Layer id	#μ neural-cores	# LAUs	# Products	# PSCUs	Clock cycles
Layer 1	2288	2020	-	-	80
Layer 2	-	-	286	9	304
Layer 3	1144	1010	-	-	48
Layer 4	-	-	143	5	153
Layer 5	286	4	-	-	12
Total	3718	3034	429	14	597

TABLE XIII: Comparison of computational resources (clock cycles) required for channel estimation with DNN-based in [29, Model 2] and the proposed CNN-based models on the same hardware (1024 μ neural-cores + 128 LAUs).

Model	MLP	Preproc. Netw.	Interpolation	Gen. Netw.	Att-like Netw.	Total clk cycles
DNN	1298 (118x11)	-	80	-	-	1378
SW-NN+CNN	-	36	-	1392	-	1428
SW-NN+AN+CNN	-	36	-	1392	597	2025

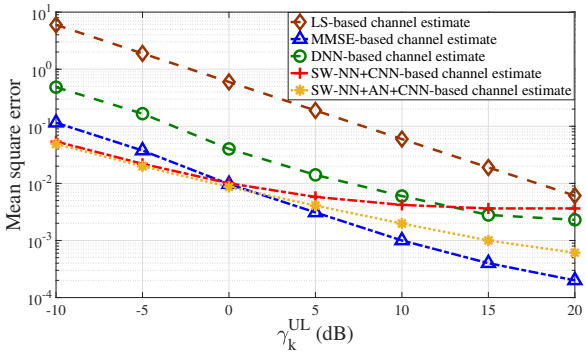
TABLE XIV: Comparison of working clock frequency required for channel estimation with DNN-based in [29, Model 2] and the proposed CNN-based models.

Model	1 ant. & 1 MS	64 ant. & 1 MS	64 ant. & 8 MSs	64 ant. & 64 MSs
DNN	551 KHz	35 MHz	282 MHz	2.2 GHz
SW-NN + CNN	571 KHz	37 MHz	293 MHz	2.3 GHz
SW-NN + AN + CNN	810 KHz	52 MHz	415 MHz	3.3 GHz

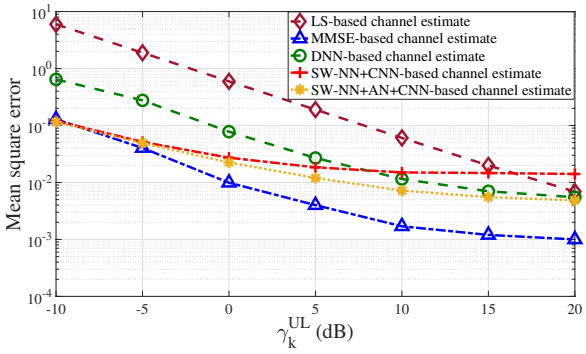
presents the results in a high-variable channel (S5 validation samples). We observe that the proposed CNN-based models outperform all the benchmark estimators in the low SNR regime, both in S1 and S5 scattering channels. The MSE of the SW-NN+AN+CNN model maintains lower than the benchmark solutions, except the MMSE estimator, independently of the channel variability and SNR experienced by the MS. However, the MSE achieved by the SW-NN+CNN model does not continue improving with better SNR and presents poorer results than DNN-based estimation in a high SNR regime. These results demonstrate the benefit of including the attention-like network in the proposed models to enhance the performance with high SNR, at the time that keeps excellent estimates with low SNR.

Intending to validate the proposed estimators with different channel conditions, Figure 8 shows the MSE achieved for the predefined channels detailed in Table V. First, we observe how both models achieve better estimates in low-variable channels. We notice that the SW-NN+CNN model achieves only slight improvements in the estimates with the enhancement of SNR conditions. However, the SW-NN+AN+CNN model allows the system to obtain significantly enhanced estimates when the MS experiences high SNR conditions.

The MSE results have been complemented with the analysis of the channel estimation accuracy impact on the spectral efficiency. Figure 9a shows the SE in a low-selectivity channel (S1). We observe the SE achieved with the proposed CNN-based estimators within the low-SNR range (from -10 to 0 dB) is extremely close to the perfect channel knowledge SE, i.e., 99.75% and 99.78% for the SW-NN+CNN and SW-NN+AN+CNN channel estimation models, respectively. These results are significantly better than LS-based SE in the same low-SNR region, i.e. 83.33%, slightly better than the DNN-based estimation, i.e., 98.37%, and even better than MMSE-based estimation, i.e., 99.6%. Examining the high-SNR region (ranging from 10 to 20 dB), we notice that the proposed CNN-based channel estimators obtain SE of 98.06% and 99.52% for SW-NN+CNN and SW-NN+AN+CNN models, respectively. We appreciate a slight performance degradation in the SW-

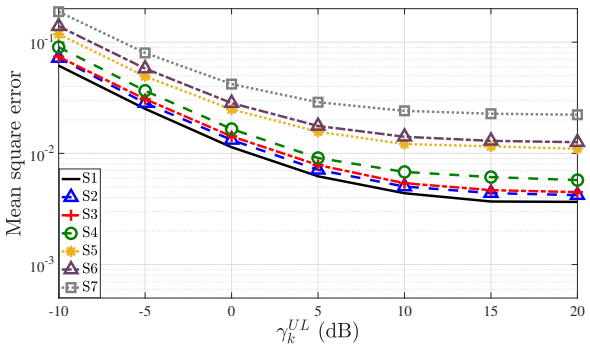


(a) Low-time and frequency selectivity channel (S1)

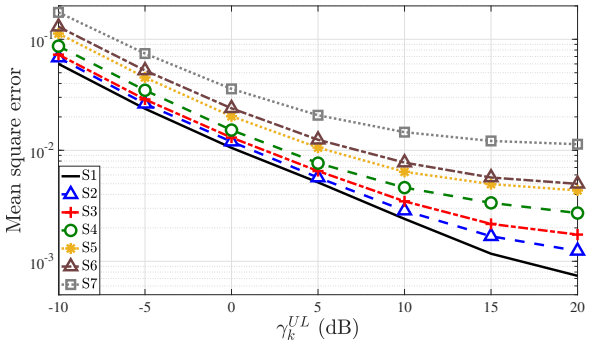


(b) Medium-time and high-frequency selectivity channel (S5)

Fig. 7: MSE between the actual and the estimated channel by the LS-, MMSE-, DNN-, SW-NN+CNN-, and SW-NN+AN+CNN-based models for different γ_k^{UL}. Low-time and frequency variability vs. medium-time and high-frequency variability channels (S1 vs. S5).

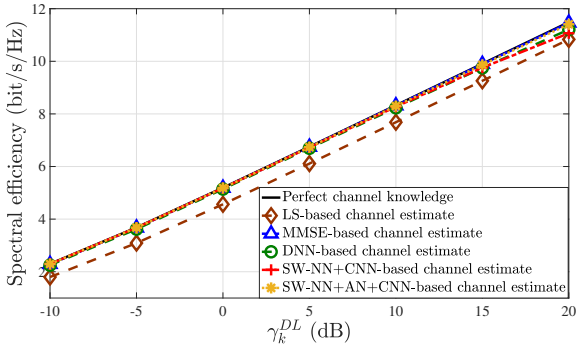


(a) SW-NN+CNN model

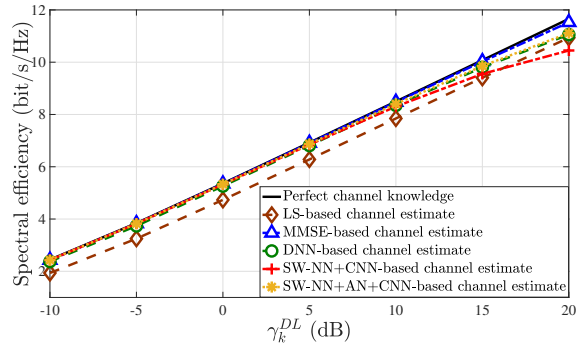


(b) SW-NN+AN+CNN model

Fig. 8: MSE between the actual and the estimated channel by the SW-NN+CNN- and SW-NN+AN+CNN-based estimation for different γ_k^{UL} and different channel selectivity.



(a) Low-time and frequency selectivity channel (S1)



(b) Medium-time and high-frequency selectivity channel (S5)

Fig. 9: Spectral efficiency achieved by perfect channel knowledge and the estimated channel by the LS-, MMSE-, DNN-, SW-NN+CNN-, and SW-NN+AN+CNN-based models for different γ_k^{DL}. Low-time and frequency variability vs. medium-time and high-frequency variability channels (S1 vs. S5).

NN+CNN model at a high SNR regime. We should remark that the SW-NN+AN+CNN model exhibits SE results better than LS- and DNN-based estimators, similar to MMSE-based, and extremely close to perfect channel knowledge results (i.e., 99.6%) for the full range of the assessed SNRs. On average, for the full range of SNRs analyzed in the low selectivity channel, the SW-NN+AN+CNN model results in approximately 0.74%, 1.32%, and 11.31% higher SE than the SW-NN+CNN-, the DNN-, and the LS-based channel estimation, respectively.

Figure 9b shows the SE achieved in a high-selectivity channel (S5). Observing the low-SNR region, we note that SE achieved by the CNN-based channel estimation models continues very close to perfect channel knowledge SE, i.e., 99.43%. These results outperform both LS- and DNN-based estimation SE, which result in 83.98% and 97.46%, respectively. Besides, the proposed CNN-based estimators' performance is similar to MMSE-based estimation in this region (i.e., 99.58%). In the high-SNR region, the proposed models achieve spectral efficiencies of 94.05% and 97.34% for SW-NN+CNN and SW-NN+AN+CNN models, respectively. We should emphasize that the SW-NN+AN+CNN model outperforms both LS- (93.17%) and DNN-based (96.78%) approaches in this region and channel model. However, the SW-NN+CNN model performance is degraded in the high-SNR region of the high-variable channel. We should remark that the proposed SW-NN+AN+CNN model outperforms the benchmark LS- and DNN-based estimation both in low and high SNR regimes, and only results in a slightly worse SE than MMSE-based estimation and perfect channel knowledge when the user experiences a high SNR. On average, for the full range of SNR analyzed in the high selectivity channel, the SW-NN+AN+CNN model results in approximately 1.67%, 1.28%, and 9.82% higher SE than the SW-NN+CNN-, the DNN-, and the LS-based channel estimation, respectively.

VI. Conclusions

This work presents two CNN-based models to estimate the channel between the BS antennas and an MS in mMIMO systems. Both models employ preprocessing networks based on SW-NN and generative CNN. The proposed models differ because of the introduction of an attention-like network to enhance the high-SNR region performance. The presented results show how the proposed CNN-based models can improve the channel estimates obtained by LS- and DNN-based estimation, independently of the channel scattering characteristics in the low SNR regime. The attention-like network in the SW-NN+AN+CNN model, allows this estimator to achieve better estimates than LS- and DNN-based approaches in all ranges of channel scattering behavior and SNR. The proposed

SW-NN+AN+CNN model exhibits close to perfect channel knowledge SE in all the assessed environments.

This work also analyzes the computational and cost complexity of the proposed estimation models. We present a deep complexity study split into the different network structures in the proposed model. The complexity analysis leads to the hardware requirements and the working clock frequency needed. We proposed a neural-core structure to parallelize the execution obtaining the clock frequency required for the proposed models, depending on the system number of MSs and BS antennas.

REFERENCES

[1] Ericsson, "Ericsson Mobility Report," June 2023, [Online] Available at: www.ericsson.com/en/reports-and-papers/mobility-report/reports/june-2023.

[2] ——, "5G wireless access: an overview," White Paper, April 2020, [Online] Available at: www.ericsson.com.

[3] F. Frederiksen and R. Prasad, "An overview of OFDM and related techniques towards development of future wireless multimedia communications," in *Proceedings RAWCON 2002. 2002 IEEE Radio and Wireless Conference (Cat. No.02EX573)*, 2002, pp. 19–22.

[4] 3GPP, "NR; Physical channels and modulation," 3GPP Technical specification 38.211, December 2022.

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

[6] J. van de Beek, O. Edfors, M. Sandell, S. K. Wilson, and P. O. Borjesson, "On channel estimation in OFDM systems," in *1995 IEEE 45th Vehicular Technology Conference. Countdown to the Wireless Twenty-First Century*, vol. 2, 1995, pp. 815–819 vol.2.

[7] A. Mayouche, A. Metref, and J. Choi, "Downlink Training Overhead Reduction Technique for FDD Massive MIMO Systems," *IEEE Signal Processing Letters*, vol. 25, no. 8, pp. 1201–1205, 2018.

[8] X. Dong, W.-s. Lu, and A. C. Soong, "Linear Interpolation in Pilot Symbol Assisted Channel Estimation for OFDM," *IEEE Transactions on Wireless Communications*, vol. 6, no. 5, pp. 1910–1920, 2007.

[9] O. O. Ogundile and D. J. J. Versfeld, "Performance evaluation of polynomial based channel estimation for OFDM systems on time-varying frequency-selective fading channels," in *AFRICON 2015*, 2015, pp. 1–6.

[10] X. Chen and M. Jiang, "Enhanced Adaptive Polar-Linear Interpolation Aided Channel Estimation," *IEEE Wireless Communications Letters*, vol. 8, no. 3, pp. 693–696, 2019.

[11] G. Liu, L. Zeng, H. Li, L. Xu, and Z. Wang, "Adaptive Interpolation for Pilot-Aided Channel Estimator in OFDM System," *IEEE Transactions on Broadcasting*, vol. 60, no. 3, pp. 486–498, 2014.

[12] N. Suga, R. Sasaki, and T. Furukawa, "Channel Estimation Using Matrix Factorization Based Interpolation for OFDM Systems," in *2019 IEEE 90th Vehicular Technology Conference (VTC2019-Fall)*, 2019, pp. 1–5.

[13] A. Osinsky, R. Bychkov, A. Ivanov, and D. Yarotsky, "Adaptive Channel Interpolation in High-Speed Massive MIMO," in *2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring)*, 2021, pp. 1–5.

[14] F. Hu and J. Li, "A Simplified LMMSE Channel Estimation Algorithm for OFDM Systems," in *2009 International Conference on Management and Service Science*, 2009, pp. 1–4.

[15] R. Tang, X. Zhou, and C. Wang, "A novel low rank LMMSE channel estimation method in OFDM systems," in *2017 IEEE 17th International Conference on Communication Technology (ICCT)*, 2017, pp. 249–253.

[16] X. Wang, X. Shen, F. Hua, and Z. Jiang, "On Low-Complexity MMSE Channel Estimation for OCDM Systems," *IEEE Wireless Communications Letters*, vol. 10, no. 8, pp. 1697–1701, 2021.

[17] M. Soltanalian, M. M. Naghsh, N. Shariati, P. Stoica, and B. Hassibi, "Training Signal Design for Correlated Massive MIMO Channel Estimation," *IEEE Transactions on Wireless Communications*, vol. 16, no. 2, pp. 1135–1143, 2017.

[18] M. Kuriyama and T. Ohtsuki, "Low Complexity and High Accuracy Channel Interpolation with Dividing URA into Small URAs for 3D Massive MIMO," in *2019 IEEE 90th Vehicular Technology Conference (VTC2019-Fall)*, 2019, pp. 1–5.

[19] M. Ju, L. Xu, L. Jin, and D. D. Huang, "Data aided channel estimation for massive MIMO with pilot contamination," in *2017 IEEE International Conference on Communications (ICC)*, May 2017, pp. 1–6.

[20] A. Khansefid and H. Minn, "On Channel Estimation for Massive MIMO With Pilot Contamination," *IEEE Communications Letters*, vol. 19, no. 9, pp. 1660–1663, 2015.

[21] Y. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks," *IEEE Journal of Solid-State Circuits*, vol. 52, no. 1, pp. 127–138, 2017.

[22] Y. Chen, Y. Xie, L. Song, F. Chen, and T. Tang, "A Survey of Accelerator Architectures for Deep Neural Networks," *Engineering*, vol. 6, no. 3, pp. 264–274, 2020.

[23] N. Samuel, T. Diskin, and A. Wiesel, "Deep MIMO Detection," 2017.

[24] E. Nachmani, Y. Beery, and D. Burshtein, "Learning to Decode Linear Codes Using Deep Learning," 2016.

[25] U. Challita, L. Dong, and W. Saad, "Deep Learning for Proactive Resource Allocation in LTE-U Networks," in *European Wireless 2017; 23th European Wireless Conference*, 2017, pp. 1–6.

[26] Y. Yang, F. Gao, X. Ma, and S. Zhang, "Deep Learning-Based Channel Estimation for Doubly Selective Fading Channels," *IEEE Access*, vol. 7, pp. 36 579–36 589, 2019.

[27] L. Ge, Y. Guo, Y. Zhang, G. Chen, J. Wang, B. Dai, M. Li, and T. Jiang, "Deep Neural Network Based Channel Estimation for Massive MIMO-OFDM Systems With Imperfect Channel State Information," *IEEE Systems Journal*, pp. 1–11, 2021.

[28] J. Zhang, X. Ma, J. Qi, and S. Jin, "Designing Tensor-Train Deep Neural Networks For Time-Varying MIMO Channel Estimation," *IEEE Journal of Selected Topics in Signal Processing*, vol. 15, no. 3, pp. 759–773, 2021.

[29] A. Melgar, A. de la Fuente, L. Carro-Calvo, O. Barquero-Pérez, and E. Morgado, "Deep Neural Network: An Alternative to Traditional Channel Estimators in Massive MIMO Systems," *IEEE Transactions on Cognitive Communications and Networking*, vol. 8, no. 2, pp. 657–671, 2022.

[30] Z. Wang, F. Pu, X. Yang, N. Chen, Y. Shuai, and R. Yang, "Online LSTM-Based Channel Estimation for HF MIMO SC-FDE System," *IEEE Access*, vol. 8, pp. 131 005–131 020, 2020.

[31] Z. Xiao, Z. Zhang, C. Huang, C. Zhong, and X. Chen, "GPAE-LSTMnet: A Novel Learning Structure for Mobile MIMO Channel Prediction," in *2021 IEEE 32nd Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2021, pp. 1–6.

[32] G. Shrestha, Deepsikha, M. Das, and N. Dey, "Plant Disease Detection Using CNN," in *2020 IEEE Applied Signal Processing Conference (ASPCON)*, 2020, pp. 109–113.

[33] L. Yuan, "Remote Sensing Image Classification Methods Based on CNN: Challenge and Trends," in *2021 International Conference on Signal Processing and Machine Learning (CONF-SPML)*, 2021, pp. 213–218.

[34] S. Patel, "An Overview and Application of Deep Convolutional Neural Networks for Medical Image Segmentation," in *2023 Third International Conference on Artificial Intelligence and Smart Energy (ICAIS)*, 2023, pp. 722–728.

[35] B. Marinberg, A. Cohen, E. Ben-Dror, and H. H. Permuter, "A Study on MIMO Channel Estimation by 2D and 3D Convolutional Neural Networks," in *2020 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, 2020, pp. 1–6.

[36] P. Jiang, C.-K. Wen, S. Jin, and G. Y. Li, "Dual CNN-Based Channel Estimation for MIMO-OFDM Systems," *IEEE Transactions on Communications*, vol. 69, no. 9, pp. 5859–5872, 2021.

[37] J. Nam, J.-Y. Ahn, A. Adhikary, and G. Caire, "Joint spatial division and multiplexing: Realizing massive MIMO gains with limited channel state information," in *2012 46th Annual Conference on Information Sciences and Systems (CISS)*, 2012, pp. 1–6.

[38] E. Björnson, J. Hoydis, and L. Sanguinetti, "Massive MIMO Networks: Spectral, Energy, and Hardware Efficiency," *Foundations and Trends® in Signal Processing*, vol. 11, no. 3-4, pp. 154–655, 2017.

[39] 3GPP, "Study on channel model for frequencies from 0.5 to 100 GHz," 3GPP Technical Report 38.901, March 2022.

[40] W. A. Al-Hussaibi and F. H. Ali, "A Closed-Form Approximation of Correlated Multiuser MIMO Ergodic Capacity With Antenna Selection and Imperfect Channel Estimation," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 6, pp. 5515–5519, 2018.

[41] W. Rawat and Z. Wang, "Deep convolutional neural networks for image classification: A comprehensive review," *Neural Computation*, vol. 29, no. 9, pp. 2352–2449, 2017.

[42] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai, and T. Chen, "Recent advances in convolutional neural networks," *Pattern Recognition*, vol. 77, pp. 354–377, 2018.

[43] H. Gao, H. Yuan, Z. Wang, and S. Ji, "Pixel Transposed Convolutional Networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 5, pp. 1218–1227, 2020.

[44] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is All you Need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017.

[45] C. Qing, L. Dong, L. Wang, J. Wang, and C. Huang, "Joint model and data-driven receiver design for data-dependent superimposed training scheme with imperfect hardware," *IEEE Transactions on Wireless Communications*, vol. 21, no. 6, pp. 3779–3791, 2022.

[46] F. Shehzad, M. Rashid, M. H. Sinky, S. S. Alotaibi, and M. Y. I. Zia, "A Scalable System-on-Chip Acceleration for Deep Neural Networks," *IEEE Access*, vol. 9, pp. 95 412–95 426, 2021.