# Computational Efficiency of Deep Learning-Based Super Resolution Methods for 5G-NR Channel Estimation

David Góez* ‡, Esra A. Beyazıt*, Nina Slamnik-Kriještorac†,
Johann M. Marquez-Barja†, Natalia Gaviria‡, Steven Latré*, Miguel Camelo*

*University of Antwerp - imec, IDLab - Department of Computer Science, Belgium
†University of Antwerp - imec, IDLab - Faculty of Applied Engineering, Belgium
‡ Electronics Engineering Department, Universidad de Antioquia, Colombia

***Abstract*—The increasing demand for high-quality and efficient Channel Estimation (CE) in 5G New Radio (5G-NR) systems has prompted the exploration of advanced Deep Learning (DL) techniques. While traditional methods, such as Linear Interpolation (LI) and Least Squares (LS), provide reasonable accuracy and are practical for real-time physical layer processing, recent DL-based CE approaches have primarily focused on accuracy, often without evidence of real-time capabilities. In this paper, we present a comprehensive evaluation of DL-based Super-resolution (SR) methods for CE, comparing models like Super Resolution Convolutional Neural Network (SRCNN), ChannelNet, and Enhanced Deep Super-Resolution (EDSR) in both 1D and 2D convolutional architectures. We optimize these models using NVIDIA TensorRT to reduce computational complexity and latency. Our results show that the optimized 1D-EDSR model achieves the best performance with a Mean Squared Error (MSE) of 0.0126, outperforming all other models in terms of accuracy. However, the optimized 1D-EDSR model fails to meet real-time constraints due to additional computational overhead (0.6798 ms/sample). In contrast, the 1D-SRCNN model offers a balanced trade-off between MSE (0.01738) and inference time (0.0866ms/sample), achieving 40% higher accuracy than LS (0.0288) while maintaining the best energy efficiency (1.48 mJ/sample).**

***Index Terms*—5G New Radio, Channel Estimation, Deep Learning, Super-Resolution, Convolutional Neural Networks, Model Optimization, TensorRT Optimization, GPU Acceleration, Real-Time Inference**

## I. Introduction

The rapid evolution of wireless communication technologies has led to the development of the fifth-generation (5G) standard for cellular networks, driven by the demand for high-speed, low-latency, and reliable connectivity to support applications such as Augmented Reality (AR), Virtual Reality (VR), and Utra-High-Definition (UHD) video streaming. These applications require massive data throughput and real-time response, setting stringent requirements for the performance and reliability of 5G networks.

A crucial aspect of meeting these requirements is efficient and accurate Channel Estimation (CE), which involves predicting the channel state information (CSI) from received pilot signals. This process is essential for optimizing transmission strategies and ensuring robust communication links. Traditional methods like Linear Interpolation (LI) and Least Squares (LS) offer reasonable accuracy and are practical for real-time physical-layer processing in 5G New Radio (5G-NR) systems. However, they often fail to meet the stringent performance demands of emerging 5G applications, especially under rapidly changing channel conditions.

Recent advances in Deep Learning (DL) have led to the development of more sophisticated CE models that can handle complex data patterns and achieve higher accuracy. Super-resolution (SR)-based DL models, such as Super Resolution Convolutional Neural Network (SRCNN) [1], ChannelNet [2], and Enhanced Deep Super-Resolution (EDSR) [3], have shown promise in transforming low-resolution channel estimates into high-resolution ones, thereby improving overall system performance. However, most of these DL-based CE models are primarily designed to maximize accuracy, often overlooking considerations of computational complexity and real-time feasibility. While traditional methods like LI and LS are practical due to their simplicity and low computational demands, DL-based approaches require substantial processing power and memory, which limits their applicability in environments with strict real-time processing requirements or resource constraints.

In previous studies [4], [5], we developed methodologies to optimize DL models, reducing computational complexity and energy consumption while maintaining high performance. Building on this foundation, this work applies these optimization strategies specifically to state-of-the-art DL-based CE architectures in 5G environments, focusing on adapting the architectures for real-time performance under the stringent constraints of signal processing such as CE in modern wireless systems such as 5G-NR [6]. We evaluate whether these models can achieve the required balance between accuracy and efficiency for deployment in real-time wireless systems. More precisely, the main contributions of this work are threefold. First, we design and implement optimized versions of three SR-based DL architectures for CE, aiming to reduce computational complexity and inference time while maintaining high accuracy. Second, we validate the feasibility of these models by comparing their performance against their non-

optimized counterparts and traditional methods such as LI and LS. Finally, we provide a comprehensive performance evaluation, demonstrating that the optimized models meet the stringent execution time and energy efficiency requirements for CE in 5G-NR systems.

The remainder of this paper is organized as follows: Section II provides an overview of the system model and reviews traditional and DL-based CE methods, highlighting their advantages and limitations. Section III describes the proposed optimization techniques applied to state-of-the-art DL-based CE architectures, detailing the methodology used to improve computational efficiency and real-time performance. Section IV presents the experimental setup, performance results, and a comprehensive comparison of the optimized and non-optimized models. Finally, Section V concludes the paper and discusses potential future research directions.

## II. Background on CE Methods for Orthogonal Frequency-division Multiplexing (OFDM) systems

This research considers the downlink Single Input Single Output (SISO) OFDM system model for 5G-NR. In an OFDM system, for the $k_{th}$ time slot and the $i_{th}$ subcarrier, the received signal, $Y_{i,k}$ is defined as follows.

$$Y_{i,k} = H_{i,k}X_{i,k} + Z_{i,k} \tag{1}$$

where $X_{i,k}$ is the transmitted OFDM signal and $Z_{i,k}$ represents Additive White Gaussian Noise (AWGN) with variance $\sigma^2$. The considered OFDM subframe size is $N_S \times N_D$. The slot index $k$ range is given as $[0, N_D - 1]$, and the subcarrier index $i$ is given as $[0, N_S - 1]$. $H_{i,k}$ represents the $(i,k)$ element of the channel matrix $\mathbf{H} \in \mathbb{C}^{N_S \times N_D}$. Below, we will describe some traditional and practical methods for CE used in 5G-NR, as well as novel DL techniques that will be the focus of this study

### A. Least Square Method

In the LS method, the channel response is calculated for each pilot location [7]. The considered diagonal channel matrix, $\hat{\mathbf{H}}_p^{\text{LS}} \in \mathbb{C}^{N_p \times N_p}$, $\hat{\mathbf{H}}_p^{\text{LS}}$ can be calculated as follows.

$$\hat{\mathbf{H}}_p^{\text{LS}} = \arg\min_{\mathbf{H}_p} \|\mathbf{y}_p - \mathbf{H}_p\mathbf{x}_p\|_2^2, \tag{2}$$

where $\|\cdot\|_2$ is the Euclidean norm and $\hat{\mathbf{H}}_p^{\text{LS}}$ is the estimated channel matrix. $\mathbf{x}_p$ is the pilot signal vector and $\mathbf{y}_p$ is the observation vector.

### B. The Minimum Mean Square Error Estimation

Minimum Mean Square Error (MMSE) minimizes the mean square error between the actual and estimated channel responses. It incorporates the correlation between the channel responses at different pilot locations and the noise statistics, making it more robust than simpler methods like Least Squares (LS) estimation.

Mathematically, the channel response $\hat{H}$ at a pilot location can be estimated using the MMSE estimator as follows.

$$\hat{H}^{MMSE} = R_{hh}\left(R_{hh} + \frac{\sigma^2}{P}I\right)^{-1} Y, \tag{3}$$

where $R_{hh}$ is the autocorrelation matrix of the channel, and $P$ is the pilot power. Similar to LS method, interpolation is performed to obtain the whole channel response, $\hat{H}^{MMSE}$.

### C. Linear Interpolation

Linear Interpolation (LI) method is a commonly used method for estimating values between two known data points [8]. In the context of CE for communication systems, LI is applied to estimate the channel transfer function for non-pilot subcarriers based on the values from adjacent pilot subcarriers.

Mathematically, if the transfer function of the pilot subcarriers is given by $\hat{H}(m)$ and $\hat{H}(m+1)$ at the $m$-th and $(m+1)$-th subcarriers, respectively, the transfer function for a non-pilot subcarrier at index $l$ between these two pilots can be estimated as follows.

$$\hat{H}(l) = \left(1 - \frac{l}{L}\right)\hat{H}(m) + \frac{l}{L}\hat{H}(m+1), \tag{4}$$

where $L$ is the number of subcarriers between the two pilots, and $l = 1, 2, \ldots, L-1$. A larger weight is assigned to the closer pilot subcarrier, ensuring that the estimated transfer function is more influenced by nearby pilots.

### D. ChannelNet

ChannelNet is a DL-based imaging framework that can be adapted for CE in OFDM systems [1], [2]. It treats the channel time-frequency response as a two-dimensional image, aiming to estimate unknown channel values using known values at pilot locations. This framework uses image SR and Image Restoration (IR) techniques to enhance the resolution of the Low-Resolution (LR) pilot values and reduce noise effects, respectively.

Let $\mathbf{H} \in \mathbb{C}^{N_S \times N_D}$ represent the channel time-frequency response matrix. The observed pilot values, denoted as $\mathbf{H}_p^{LS}$, are treated as a low-resolution and noisy version of the channel image. ChannelNet employs a two-stage approach to estimate the complete channel image. In the first stage, an SR network SRCNN is used to enhance the resolution of $\mathbf{H}_p^{LS}$, producing an intermediate high-resolution estimate $\mathbf{K}$. In the second stage, a denoising IR network Denoising Convolutional Neural Network (DnCNN) further refines $\mathbf{K}$ to obtain the final channel estimate $\hat{\mathbf{H}}$ [9]. The set of all network parameters are denoted by $\Theta = \{\Theta_{\text{SR}}, \Theta_{\text{IR}}\}$ where $\Theta_{\text{SR}}$ and $\Theta_{\text{IR}}$ are set of parameters of SR and IR networks, respectively. If $f_{\text{SR}}$ and $f_{\text{IR}}$ denote the SR and IR functions, then the output of the ChannelNet can be defined as follows [10], [11].

$$\hat{\mathbf{H}} = f_{\text{IR}}(f_{\text{SR}}(\mathbf{H}_p^{LS}; \Theta_{\text{SR}}); \Theta_{\text{IR}}) \tag{5}$$

where $f_{\text{SR}}$ and $f_{\text{IR}}$ represent the SR and IR functions, which are parameterized by $\Theta_{\text{SR}}$ and $\Theta_{\text{IR}}$. The network is optimized by minimizing the Mean Squared Error (MSE) between the estimated channel $\hat{\mathbf{H}}$ and the actual channel $\mathbf{H}$ as follows.

$$\mathcal{L}(\Theta_{\text{SR}}, \Theta_{\text{IR}}) = \frac{1}{|T|}\sum_{\mathbf{H}_p^{LS} \in T} \|\hat{\mathbf{H}} - \mathbf{H}\|_2^2 \tag{6}$$

This approach allows ChannelNet to significantly enhance CE accuracy, making it a competitive alternative to traditional

methods such as LS and more resilient in dynamic channel conditions [12]. where $T$ is the number of training samples.

### E. Enhanced Deep Super-Resolution (EDSR)

The EDSR model is designed to improve the performance of SR tasks by optimizing the Deep Residual Networks (DRN) [9]. EDSR builds on the Residual Network (ResNet) architecture by removing unnecessary modules, such as batch normalization layers, which are found to limit the network's flexibility and performance. This simplification allows the network to focus on the essential parts of the SR task, improving accuracy and computational efficiency. This architecture was evaluated for CE in [3], demonstrating superior performance compared to traditional methods like LI and LS, as well as DL approaches such as ChannelNet and SRCNN.

Let $\mathbf{I}_{LR}$ represent the low-resolution input image and $\mathbf{I}_{HR}$ the high-resolution target image. The EDSR network is trained to learn the mapping function $f_\Theta$ such that $\mathbf{I}_{SR} = f_\Theta(\mathbf{I}_{LR})$. $\mathbf{I}_{SR}$ denotes the super-resolved output image and $\Theta$ represents the network parameters. The loss function used to train the network is typically the MSE between the super-resolved image and the high-resolution target image:

$$\mathcal{L}(\Theta) = \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{I}_{HR}^{(i)} - \mathbf{I}_{SR}^{(i)}\|_2^2, \tag{7}$$

where $N$ is the number of training samples.

## III. Optimizing SR-based Architectures for CE

In order to reduce the complexity of DL-based models for CE while still ensuring high performance in the task, we follow the procedure described in [4] but apply it to single-task models. As a reference, we will describe the complete approach using the ChannelNet architecture; however, this method can be applied to any state-of-the-art architectures, such as SRCNN and EDSR.

Let us recall the initial process for obtaining the LR input for any of the previously described models based on SR architectures. Initially, pilot symbols at both ends of the communication form an LR signal using linear interpolation of the channel response, $H_p$, at the pilot locations. The practical method to calculate $H_p$ is by using the elements of the received signal ($y_p$) and the known pilot symbols ($x_p$) according to $H_p = \frac{y_p}{x_p}$, which is equivalent to applying the LS method at each pilot location. After this, $\hat{H}p$ is interpolated to obtain $\hat{H}int$. $\hat{H}_{int}$ is then used as the input for the models. The input size is $N_S \times N_D \times 2$, where the last dimension represents the real and imaginary values.

As demonstrated in [4], a simple but robust strategy to reduce the complexity of Convolutional Neural Network (CNN)-based architectures for signal processing is to convert 2D-Convolutional (Conv) layers into One-Dimensional (1D) ones. However, this change implies that we can no longer use the extra dimension to represent the real and imaginary parts separately as different channels in the Two-Dimensional (2D) Conv layers. In this case, we can concatenate the real and imaginary dimensions, removing the final dimension of $\hat{H}_{int}$ and resulting in a shape of $(N_S, N_D \times 2)$.

Now, let us focus on the optimization techniques applied after the 2D to 1D conversion in the SR-based architectures. Deep Neural Network (DNN) optimization techniques generally involve multiple optimization steps targeting batch processing, layer structure and grouping, and operations in the DNN model for better performance on specific hardware. Among such techniques, the following are the ones with major impact on the optimization of ChannelNet, EDSR, and SRCNN architectures using NVIDIA Graphics Processing Unit (GPU) optimization framework TensorRT:[1].

1) **Layer Fusion:** Combines multiple layers into a single operation for improved computational efficiency. For example, this will allow that convolution, bias addition, and ReLU activation can be performed in one single step.
2) **Precision Calibration:** Adjusts computation precision (e.g., from FP32 to FP16) to balance performance and accuracy.
3) **Graph Optimizations:** Analyzes and optimizes the execution graph to remove redundant operations.
4) **Multi-Stream Execution:** Enables concurrent execution of multiple inference streams to optimize GPU resource utilization.
5) **Data Layout Transformation (Shuffle Layers):** Some layers can improve memory access patterns and computational efficiency during convolution operations by changing the data layout from Row major (batch, height, width, channel) to Channel major (batch, channel, height, width) and vice versa.

Certain optimizations may not be applicable depending on the target hardware. For instance, NVIDIA GPUs such as H100 and A100 have dedicate hardware for INT8 operations (INT8 Tensor Cores) while V100 does not, resulting in lower INT8 performance for the V100 compared to GPUs equipped with native INT8 Tensor Cores [2]. In this paper, we use FP16 precision calibration instead of INT8 since no major performance improvements were obtained with lower precision.

One modification to the CE method using the EDSR architecture, compared to the implementation in [13], is that we no longer require two separate parallel EDSR architectures to process the real and imaginary components of the channel response. Instead, a single EDSR architecture processes them as a single concatenated dimension.

## IV. Performance Results

### A. Dataset description

The dataset was generated using the MATLAB Data Synthesis for Channel Estimation in 5G code[3]. A summary of

[1]https://docs.nvidia.com/deeplearning/tensorrt/archives/tensorrt-803/best-practices/index.html

[2]https://docs.nvidia.com/deeplearning/tensorrt/support-matrix/index.html

[3]https://nl.mathworks.com/help/5g/ug/deep-learning-data-synthesis-for-5g-channel-estimation.html

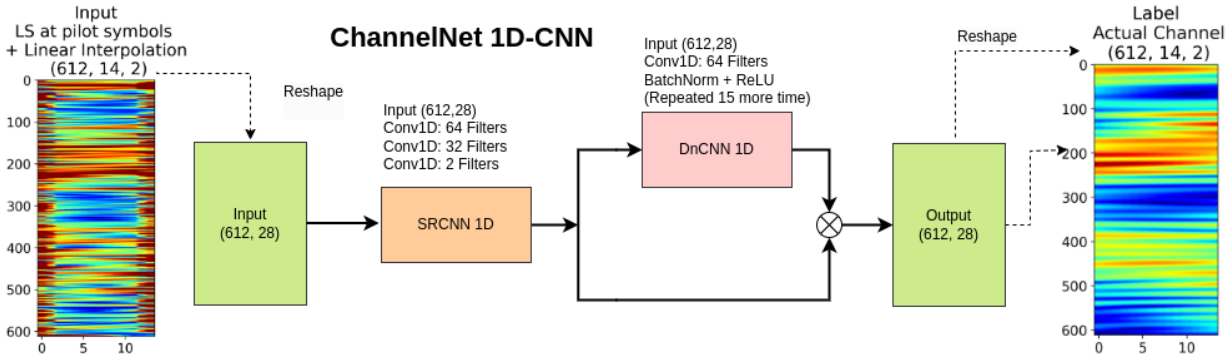


Figure 1. Example of the ChannelNet architecture using 1D-Conv for channel estimation.

Table I
MODEL ARCHITECTURE

| Feature | SRCNN-2D | SRCNN-1D | EDSR-2D | EDSR-1D | ChannelNet-2D | ChannelNet-1D |
|---|---|---|---|---|---|---|
| **Number of CNN layers** | 3 | 3 | 17 | 17 | 20 | 20 |
| **Parameters** | 14114 | 22780 | 306466 | 108796 | 574244 | 222744 |
| **Model size** | 55.13 KB | 88.98KB | 1.17 MB | 424.98 KB | 2.19 MB | 870.09 KB |
| **Architecture summary** | Input: (612, 14, 2)<br>Conv2D: 64 filters<br>Conv2D: 32 filters<br>Conv2D: 2 filters | Input: (612, 28)<br>Conv1D: 64 filters<br>Conv1D: 32 filters<br>Conv1D: 2 filters | Input: (612, 14, 2)<br>Conv2D: 64 filters<br>Optimized residual blocks<br>(Repeated 15 more times) | Input: (612, 28)<br>Conv1D: 64 filters<br>Optimized residual blocks<br>(Repeated 15 more times) | Input: (612, 14, 2)<br>SRCNN 2D<br>DnCNN 2D | Input: (612, 28)<br>SRCNN 1D<br>DnCNN 1D |

the parameters used to generate the dataset, such as the type of physical channel (e.g., Physical Downlink Shared Channel (PDSCH)), central frequency, Subcarrier Spacing (SCS), Cyclic Prefix (CP) type, number of Resource Blocks (RBs), code rate, and modulation, among others, is provided in Table II. Compared to the original code, some minor modifications were made. Specifically, the Signal-to-Noise Ratio (SNR) values ranged between 0 and 20 dB, instead of 0 to 10 dB, and we generated 1024 samples for each SNR value, resulting in a total of 11,264 samples (11 SNR values $\times$ 1024 samples per value), with a split of 70% for training, 15% for validation, and 15% for testing. The true labels are generated by assuming full channel state information, while the model inputs are obtained by applying LI over LS estimates on the pilot symbols, as utilized in ChannelNet and EDSR.

### *B. Baseline methods and models*

In order to compare the performance of the SRCNN, ChannelNet, and EDSR CE methods optimized using the steps described in Section III, the estimations of the following methods and models were also obtained. The two classic methods were LS+Linear Interpolation (LI), and Least Squares (LS) + Channel Impulse Response (CIR) for denoising + averaging the estimated noise + Linear Interpolation (LI) in the frequency direction and time direction. This approach is also known as the 5G-NR practical estimator in MATLAB. Additionally, the original versions of ChannelNet, SRCNN, which is derived from the first phase of ChannelNet, and EDSR were implemented using 2D-Conv layers. The only modification to the EDSR architecture was to use a single network where the real and imaginary parts of the input were treated as two separate channels in the 2D-CNN architecture.

Table II
CHANNEL AND 5G-NR PARAMETERS FOR DATASET GENERATION

| Parameter | Value/Description |
|---|---|
| Channel Profiles | TDL-A, TDL-B, TDL-C, TDL-D, TDL-E |
| Fading Distribution | Rayleigh |
| Number of Antennas (Tx, Rx) | 1, 1 |
| Carrier Configuration | 51 RBs, 30 kHz SCS, Normal CP |
| Sub-carriers Per RB | 12 |
| Symbols Per Slot | 14 |
| Slots Per Subframe | 2 |
| Slots Per Frame | 20 |
| Frame Duration | 10 ms |
| NFFT | 1024 |
| Transmission Direction | Downlink |
| PDSCH Configuration | PRB: 0-50, All symbols, Type A, 1 Layer |
| Modulation | 16QAM |
| DM-RS Configuration | Ports: 0, Type A Position: 2<br>Length: 1, Config: 2 |
| Delay Spread | 1 - 300 ns |
| Doppler Shift | 5 - 400 Hz |
| SNR | 0 - 20 dB, steps of 2 dB |
| Sample Rate | 30720000 |
| Noise | AWGN |
| Interpolation | Linear |
| Data set shape | [11264, 612, 14, 2] = [# samples, SCS$\times$RBs, symbols per slot, real & imaginary components] |

The table I summarizes the architecture and characteristics of six different models: the baselines SRCNN-2D, ChannelNet-2D, EDSR-2D, and the optimized versions SRCNN-1D, ChannelNet-1D, EDSR-1D. While in the following sections we will analyze more in details the different characteristics, it is important o highlight the following two general aspects.

- **Number of CNN Layers**: The SRCNN models have 3 layers, the EDSR models have 17 layers, and the ChannelNet models have 20 layers in both 2D and 1D architectures.

Table III
PERFORMANCE COMPARISON AMONG BASELINE MODELS

| Metrics | ChannelNet-2D | SRCNN-2D | EDSR-2D | LS | LI |
|---|---|---|---|---|---|
| MSE | 0.01369 | 0.02003 | 0.01256 | 0.0288 | 0.2232 |
| MAE | 0.0834 | 0.101 | 0.0791 | 0.119 | 0.271 |
| $R^2$ | 0.966 | 0.950 | 0.968 | 0.9282 | 0.4475 |

- **Architecture**: The SRCNN models use basic convolutional layers follow by Rectified Linear Unit (ReLU) activations, the EDSR models incorporate multiple repeated optimized residual blocks, e.g., batch normalization layers are removed, and the ChannelNet models combine SRCNN and the DnCNN networks, where the DnCNN contains 17 groups of Conv + batch normalization + ReLU layers training using residual learning.

### C. Performance of baseline models

Table III shows a comparative analysis of the performance metrics, MSE, Mean Absolute Error (MAE), and the coefficient of determination ($R^2$), across the 2D-Conv versions of ChannelNet, SRCNN, and EDSR, together with the traditional methods LI and LS CEs using the test dataset. The results shows that the 2D-EDSR model achieved the best performance across all metrics, with the lowest MSE (0.0126) and MAE (0.0791), as well as the highest $R^2$ score (0.968), indicating superior accuracy and reliability. The 2D-ChannelNet also performed well, showing a close second with an MSE of 0.0137, and an $R^2$ of 0.966. In contrast, the traditional LI method had the poorest performance, with significantly higher MSE (0.2232) and MAE (0.271), and a much lower $R^2$ (0.4475), indicating its limited capability in accurately estimating the channel. The LS method performed better than LI, but it still lagged behind the DL models.

In general, the EDSR model has an improvement about 37% over the SRCNN model, 8% over the ChannelNet model, 56% over the LS method, and 94% over the LI method. These results align with the finding of [3] and will be used as baseline for comparison against the optimized models. Finally, Figure 2 illustrates the effect of varying SNR levels on the MSE for the five different CE methods. It can be observed that the performance of each method remains relatively consistent with the results from Table III across the range of SNR values, where EDSR method outperforms the others.

### D. Computational requirements and inference performance

*1) Model size:* The architectures of SRCNN, EDSR, and ChannelNet models are given for both 1D and 2D cases in Table I. One can observe that the comparison between 1D and 2D CNN models reveals significant differences in computational demand in trainable parameters and memory size. Contrary to expectations, in the case of SRCNN models, the 1D version has around $61\%$ more trainable parameters and model size than the 2D version. This is primarily due to the larger number of input channels (28 in 1D vs. 2 in 2D), the filter size configuration, and its shallow architecture. Now,in the EDSR models, the 2D version has $53.31\%$ more trainable parameters and $49.85\%$ more model size than the 1D version.Finally, in the ChannelNet models, the difference is even clearer, with $157.92\%$ more trainable parameters and $151.58\%$ more memory size in the 2D version compared to the 1D. These results are aligned with the expectations of the model complexity of using 1D vs. 2D convolutions

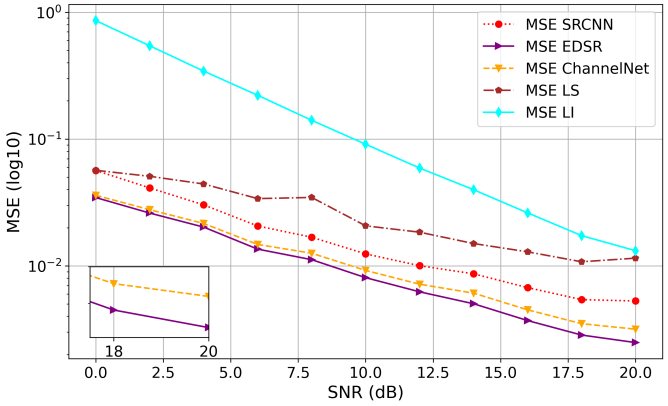


Figure 2. MSE vs. SNR comparison of the baseline CE methods

*2) Model inference time:* Moving now to the inference time, we evaluate all the models using a virtualized server in our facilities[4], which was equipped with 4 virtual cores, 64GB RAM, and a NVIDIA GPU Tesla V100-SXM2-32GB. This graphics card, equipped with 32 GB of memory, was running NVIDIA-SMI driver version 535.183.06 and CUDA version 12.2. The performance comparison in Table IV reveals significant differences in accuracy and inference time between 1D and 2D CNN models with and without TensorRT optimization. In general, we can see that the optimization with TensorRT reduces the inference time in up to 99% (e.g., 0.086 ms vs. 68 ms in SRCNN), with even imporvement on accuracy (0.01738 vs. 0.02003 in SRCNN, approx. 10% improvement). In other words, these models will not work in real environments using only GPU for inference.

When comparing the 1D and 2D versions of models optimized with TensorRT, it is evident that deeper models, such as ChannelNet and EDSR using 2D convolutions, exhibit better inference times (e.g., ChannelNet: 0.35 ms vs. 0.39 ms; EDSR: 0.33 ms vs. 0.67 ms). It can be hypothesized, based on the output from the TensorRT engine inspector[5], that deep 1D-CNNs deployed with TensorRT are slower than their 2D counterparts due to the additional computational overhead required for transforming 1D convolutions into 2D convolutions, including operations like dimension expansion and squeeze, which are necessary to adapt the data format for the optimized 2D convolution routines in TensorRT.

Now, can these models perform CE to satisfy the signal processing requirements of 5G-NR? Assuming that the deadline for completing CE is 1/3 of a Transmission Time Interval

[4]https://doc.ilabt.imec.be/ilabt/gpulab/

[5]https://docs.nvidia.com/deeplearning/tensorrt/api/python_api/infer/Core/EngineInspector.html

Table IV
PERFORMANCE COMPARISON OF THE DIFFERENT CE MODELS WITH AND WITHOUT TENSORRT OPTIMIZATION. INFERENCE WAS PERFORMED **ONE SAMPLE** AT A TIME TO SIMULATE THE SEQUENTIAL ARRIVAL OF THE 5G-NR WAVEFORM. MS = MILLI-SECONDS, MJ = MILLI-JULES

| **Model** | **TensorRT Optimization** | **MSE** | **Inference ms/sample** | | **Power consumption Mean (Watts)** | **Energy efficiency mJ/sample** |
|---|---|---|---|---|---|---|
| | | | **Mean** | **Standard deviation** | | |
| 1D-SRCNN | **Yes** | **0.01738** | **0.0866** | **0.0423** | 17.19 | **1.4894** |
| 2D-SRCNN | Yes | 0.02003 | 0.1059 | 0.0501 | 17.94 | 1.90 |
| 2D-SRCNN | No | 0.02003 | 68.0460 | 17.9107 | 0.10 | 6.8046 |
| 1D-ChannelNet | Yes | 0.01365 | 0.3984 | 0.0670 | 28.69 | 11.4295 |
| 2D-ChannelNet | Yes | 0.01369 | 0.3507 | 0.0696 | 79.6700 | 27.9414 |
| 2D-ChannelNet | No | 0.01368 | 73.1562 | 19.0223 | 0.5300 | 38.7728 |
| 1D-EDSR | Yes | **0.01220** | **0.6798** | 0.1911 | 22.6500 | 15.3976 |
| 2D-EDSR | Yes | 0.01256 | 0.3332 | 0.1221 | 86.7800 | 28.9152 |
| 2D-EDSR | No | 0.01256 | 71.6490 | 17.3806 | 0.3400 | 24.3607 |

(TTI) [6], where each TTI corresponds to the duration of 14 OFDM symbols, we find that all TensorRT-optimized models, except the 1D-EDSR, can meet this requirement, with an inference time per sample of $\leq 0.5$ ms.

*3) Model energy-efficiency:* However, the results in Table IV also indicate a trade-off between energy efficiency and accuracy among the models. Specifically, the optimized 1D-SRCNN model reduces energy consumption by 90% compared to the EDSR model (1.48 mJ vs. 15.39 mJ) and by 86% compared to the ChannelNet model (1.48 mJ vs. 11.42 mJ). These energy savings with 1D-SRCNN come at the cost of a 21% higher error compared to ChannelNet (0.0173 vs. 0.0136) and a 29% higher error rate compared to EDSR (0.0173 vs. 0.0122). This is expected, as SRCNN is not a deeper model.

## V. CONCLUSIONS AND FUTURE WORK

In this paper, we evaluated the computational efficiency and performance of DL-based SR methods for CE in 5G-NR systems. The study compared traditional methods such as LI and LS with advanced DL models like SRCNN, ChannelNet, and EDSR in both their 1D and 2D convolutional variants. The results demonstrated that the optimized version of the EDSR model achieved the best overall performance with the lowest MSE of 0.0126, outperforming all other models in terms of accuracy but failing to meet the expected latency requirements in its optimized 1D version. On the other hand, the optimized 1D-SRCNN model provided a better balance between MSE (0.01738) and inference time (0.0866ms/sample), achieving 40% higher accuracy than the practical LS method (0.01738 vs. 0.0288). It also exhibited the best energy efficiency with only 1.48mJ/sample), a reduction of ¿86% compared to ChannelNet and EDSR. However, the study also revealed that 1D-CNN models are generally less efficient than 2D-CNN models when optimized with TensorRT, due to the additional overhead required to adapt 1D convolutions to the optimized 2D format.

As future work, we will explore hybrid models that integrate 1D and 2D convolutions to provide a balance between computational efficiency, energy consumption, and model accuracy. Finally, further evaluation of these models under varying SCS, bandwidth, channel conditions and hardware configurations would provide deeper insights into their robustness and adaptability in diverse real-world scenarios.

## ACKNOWLEDGMENT

The dataset generation and model performance evaluation and analysis have been funded by the 6G-TWIN project, which has received funding from the Smart Networks and Services Joint Undertaking (SNS JU) under the EU's Horizon Europe research and innovation program under Grant Agreement No 101136314. Similarly, the design and development of the models presented in this paper has been performed within the European project 6G-XCEL, which has received funding from the Smart Networks and Services Joint Undertaking (SNS JU) under the European Union's Horizon Europe research and innovation program under Grant Agreement No 101139194. However, views and opinions expressed are those of the author(s) only and do not necessarily reflect those of the European Union or Smart Networks and Services Joint Undertaking. Neither the EU nor the granting authority can be held responsible for them.

## REFERENCES

[1] C. Dong *et al.*, "Image super-resolution using deep convolutional networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 2, pp. 295–307, 2015.

[2] M. Soltani *et al.*, "Deep learning-based channel estimation," *IEEE Communications Letters*, vol. 23, no. 4, pp. 652–655, 2019.

[3] H. Ye *et al.*, "Power of deep learning for channel estimation and signal detection in ofdm systems," *IEEE Wireless Communications Letters*, vol. 7, no. 1, pp. 114–117, 2018.

[4] D. Góez *et al.*, "Resource-efficient spectrum-based traffic classification on constrained devices," *IEEE Open Journal of the Communications Society*, 2024.

[5] D. Goez *et al.*, "A methodology to design quantized deep neural networks for automatic modulation recognition," *Algorithms*, vol. 15, no. 12, p. 441, 2022.

[6] S. A. Damjancevic *et al.*, "Channel estimation for advanced 5g/6g use cases on a vector digital signal processor," *IEEE Open Journal of Circuits and Systems*, vol. 2, pp. 265–277, 2021.

[7] M. O. Mendonça *et al.*, "Machine learning-based channel estimation for insufficient redundancy ofdm receivers using comb-type pilot arrangement," in *2022 IEEE Latin-American Conference on Communications (LATINCOM)*. IEEE, 2022, pp. 1–6.

[8] J. Zhang *et al.*, "Channel estimation based on linear interpolation algorithm in ddo-ofdm system," in *Asia Communications and Photonics Conference and Exhibition*, 2010, pp. 605–606.

[9] B. Lim *et al.*, "Enhanced deep residual networks for single image super-resolution," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2017, pp. 136–144.

[10] K. Zhang *et al.*, "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.

[11] A. Yang *et al.*, "Frequency-time division based deep learning for ofdm channel estimation," *arxiv. abs/2107.07161*, 2021.
[12] D. Luan and J. Thompson, "Low complexity channel estimation with neural network solutions," in *WSA 2021; 25th International ITG Workshop on Smart Antennas*. VDE, 2021, pp. 1–6.
[13] W. Shen *et al.*, "Deep learning for super-resolution channel estimation in reconfigurable intelligent surface aided systems," *IEEE Transactions on Communications*, vol. 71, no. 3, pp. 1491–1503, 2023.