Tanzania Journal of Engineering and Technology 2023, **42**(**3**): 78-88
OPEN ACCESS articles distributed under Creative Commons Attribution Licence [CC BY-ND]
Websites: https://ajol.org/tjet; https://tjet.udsm.ac.tz

ISSN 1821-536X (**print**); ISSN 2619-8789 (**electronic**)
https://doi.org/10.52339/tjet.v42i3.910

Regular Research Manuscript

A Multitask Learning Framework for Pilot Decontamination in 5G Massive MIMO

Crallet M. Victor[1†], Alloys N. Mvuma[2], and Salehe I. Mrutu[3]

[1,3]University of Dodoma, Dodoma, Tanzania;

[2]Mbeya University of Science and Technology, Mbeya, Tanzania

[†]Corresponding author: victorcrallet@gmail.com, crallet.mbwelwa@udom.ac.tz

ORCID: https://orcid.org/0000-0002-2233-7128

ABSTRACT

Reference signals enable the acquisition of channel state information (CSI) for purposes such as channel estimation, beam selection, precoding, and symbol detection in 5G massive multiple-input multiple-output (MAMIMO) systems. Eventually, as more and more users and cells are added, orthogonal reference signals become few which leads to pilot contamination. Pilot contamination limits the performance and occurs when non-orthogonal reference signals occupy time-frequency resources that are alike. Learning-based techniques have been proposed to alleviate it. However, each can only learn to perform a single task namely pilot assignment, power allocation, pilot design, or de-noising for pilot decontamination. In addition, each learner can only be successful if postulated conditions are met. This study proposes a multitask learning framework that can be trained to dynamically select from the multitude of deep learning models which have been suggested for pilot decontamination. Under all signal-to-noise (SNR) ratios, experiments conducted on the deep residual learning aided channel estimator using the multitask learning framework showed minimum channel estimation errors compared to single-task learning.

ARTICLE INFO

First submitted: **June 1, 2023**

Revised: **Sep. 29, 2023**

Accepted: **Oct. 10, 2023**

Published: **Oct., 2023**

Keywords: *Pilot contamination, channel estimation, single-task learning, multitask learning, deep learning framework, routing-based model selection, massive MIMO.*

INTRODUCTION

Fifth-generation (5G) mobile communication technologies allow access to enhanced multimedia broadband services such as 3D games and television. However, these services consume high bandwidths which can be met by hyper-dense small cells, or millimeter-waves (mm-Waves). Hyper-dense small cells are favorable at low frequencies due to limited bandwidth and less spectral efficiency. MAMIMO increases spectral efficiency by allowing users to simultaneously transmit for them. At high frequencies, mm-Waves provide large bandwidths, thus, spectral efficiency is not an issue. Instead, severe attenuations are of primary concern due to small antenna apertures, blockage, oxygen absorption, rain, etc. MAMIMO compensates for these transmission losses by beam-forming the transmitted signals in the users' directions (Zhao et al., 2018). The MAMIMO technology requires the base stations to install antennas much larger than the number of active users for its

services (Marzetta, 2015). In addition, mechanisms for CSI acquisition and feedback must be incorporated. However, perfect CSI is impeded by pilot contamination in 5G MAMIMO systems (Fatema et al., 2017).

5G MAMIMO systems exhibit pilot contamination as a result of multiplexed non-orthogonal reference signals in the time-frequency domain. It is caused by inter-cell interference (ICI) leading to low-quality of CSI. In the frequency domain, the ICI due to users sharing time-frequency resources can be given by Equation (1) (Ferrante et al., 2017).

$$Y_j = \sum_{l=1}^{L} \sum_{k=1}^{K} \sqrt{p_{lk}\beta_{lk}} H_{lk} \odot X_{lk} + N_j \quad (1)$$

The terms in Equation (1) are as follows. Y_j gives the received 5G MAMIMO reference signal by the *jth* reference base station in the system. A reference signal sent by the *kth* user from the *lth* cell is given as X_{lk}. L gives the total number of cells. K denotes the number of users in each cell. Power allotted to the user is given by p_{lk}. H_{lk} is the channel used by users to transport information. β_{lk} represents the large-scale coefficient (LSC). N_j is the additive white Gaussian noise (AWGN) with zero mean and element-wise variance of σ_n^2. $\odot$ stands for element-wise multiplication.

For channel estimation, CSI can be acquired by using the least square (LS) algorithm, Y_j/X_{jk} which leads to Equation (2) for the *kth* user in the j^{th} cell. According, to Equation (2), $\widetilde{H}_{jk}^{LS}$ which provides the channel gains cannot be utilized without suppressing ICI given by the second term as it would degrade further processes that depend on the measured CSI. This ICI is known as pilot contamination and due to it the LS estimate for the k^{th} user in the j^{th} cell is contaminated by channels of other users. Downlink beamforming vectors computed by using the LS estimate would result in interference for users sharing the time-frequency resources (Larsson & Marzetta, 2014).

$$\widetilde{H}_{jk}^{LS} = H_{jk} + \sum_{l \neq j}^{L} \sqrt{\frac{p_{lk}\beta_{lk}}{p_{jk}\beta_{jk}}} H_{lk} + N_j \odot X_{jk}^{-1} \quad (2)$$

In 5G MAMIMO, pilot contamination is caused by uplink (UL) sounding references (SRSs) signals upon CSI acquisition. UL SRSs allocation strategies have been studied to relieve users from pilot contamination (Giordano et al., 2018). In a wider scope, deep learning techniques that deal with pilot contamination can be categorized as power allocation, pilot assignment, denoiser, and pilot design techniques (D'Andrea et al., 2019; Omid et al., 2021; Hirose et al., 2021; Jiang et al., 2021).

The power control scheme in (Xu et al., 2019) demonstrated that the deep neural network (DNN) fed with LSCs, β_{lk} can learn to allocate the power, p_{lk} for all users to alleviate pilot contamination. User positions were also used as inputs in (D'Andrea et al., 2019) for the same purpose. By using positions and corresponding pilot assignments obtained thru an exhaustive search, (Kim et al., 2018) trained the DNN to perform pilot assignments, X_{lk} to increase the network capacity. To cope with the changing ICI caused by pilot contamination, (Omid et al., 2021) designed the pilot assignment scheme using deep reinforcement learning. In (Balevi et al., 2020), the DNN denoised the received signal, then the least square estimate was carried out to eliminate pilot contamination. Denoising can be regarded as an act of training the neural network to understand the relationship between $\widetilde{H}_{jk}^{LS}$ and H_{jk}. Referring to Equation (2), it is clear how denoising reduces pilot contamination. Another denoiser was devised using deep residual learning conceived around a convolutional neural network (CNN) for channel estimation (Lim et al., 2021).

Authors of (Jiang et al., 2021) argued that channel sparsity can be exploited for denoising in the angle-delay domain while pilot contamination can be tackled well in the spatial-frequency domain. Then, they

presented the dual CNN with good channel estimation performance which is less complex. In an attempt to combat pilot contamination, pilot design strategies search for non-orthogonal pilots, X_{lk} to decrease the scarcity of pilots. As long as the DNN could maintain the orthogonality of pilots for users with close LSCs. One approach experimented with a DNN that accepted LSCs to provide pilots as outputs (Lim et al., 2021). The alternative, learned the pilots as weights using a two-layer network (TNN) for each user. Then, extracted weights represented the final pilots after learning (Chun et al., 2019).

Motivation

Power allocation schemes based on deep learning have shown the best performance with low complexity in comparison. However, when shadowing is taken into account performance can be poor (D'Andrea et al., 2019). Moreover, shadowing and path losses will never favor cell-edge users experiencing similar LSCs to multiple base stations. Reliance on pre-computed positions or AOAs can offset the performance of deep learning-aided pilot assignment schemes. Of course, performance bottlenecks will be vivid when it is not possible to compute positions or AOAs with a certain degree of accuracy (Omid et al., 2021; Kim et al., 2018).
Assumptions about the MAMIMO channel models can lead to wide performance deviations when estimating the channel by using denoising. Nevertheless, denoisers allow each base station to work independently. In addition, end-to-end learning without channel models can avert the deviations (Aoudia & Hoydis, 2018). Finally, TNNs as pilot designers are still constrained by CSI acquisition of the LSCs for all users in all cells at the base stations (Chun et al., 2019). Due to the above reasoning. Each deep learning scheme is optimal for the 5G MAMIMO system with pilot contamination under certain conditions. Therefore, this study proposes an adaptive approach that can align the conditions and the appropriate deep-learning strategy when pilot contamination arises.

Contributions

However, instead of training different networks for pilot decontamination. This paper recommends the multitask learning framework (MTL) that can be applied to realize the deep learning model that learns from many tasks while eliminating pilot contamination. The MTL framework can be tailored for different inputs, computation graphs, and outputs to capture various architectures proposed for 5G MAMIMO systems with pilot contamination (D'Andrea et al., 2019; Omid et al., 2021; Hirose et al., 2021; Jiang et al., 2021). The main contributions of this paper are as follows.

a) It presents the MTL framework, which can dynamically learn to perform power allocation, pilot allocation, pilot design, and denoising tasks to eliminate pilot contamination in 5G MAMIMO. With the framework; activations from multiple outputs can be merged (Misra et al., 2016), functions or layers can be chosen adaptively to make up different deep learning networks based on inputs, outputs, and tasks (Rosenbaum et al., 2017), and the right model for the task can be derived from the original model (Ahn & Kim, 2019) by the routing function.
b) The routing function is flexible as it can be fixed or learnable by using sampling-based learning (Ahn & Kim, 2019), reinforcement learning (Rosenbaum et al., 2017), etc. The routing function selects the function block, applies data to it, and gets the output back. By doing this, the routing function can purposely apply masks before propagating the data to the next function block to enable MTL through parameter sharing (Strezoski et al., 2019), sparse sub-networks (Pfeiffer et

al., 2023), pruning (Mallya & Lazebnik, 2018), etc. for parament and size efficient models.

c) It demonstrates how fixed routing can be exploited for channel estimation in 5G MAMIMO systems with pilot contamination given the two tasks, MTL framework and deep residual learning (DRL) based channel estimation (Lim et al., 2021).

d) Computer simulations have shown that the MTL framework has improved the performance of the DRL-based channel estimator under all SNRs.

The organization of the remainder of the paper is as follows. A description of the multitask learning framework comes next. Then, it applies the MTL framework using a fixed routing scheme for training the DRL-based channel estimator established by [13] under two tasks. It continues with performance evaluation. The conclusion of the paper is at the end.

Multi-task Learning Framework

Multi-task learning (MTL) is a technique that optimizes the learning model for more than one task. MTL allows experiments to be conducted on multiple tasks to boost the learning of each of them (Strezoski et al., 2019). As shown in Figure 1, the MTL framework proposed is essentially a routing network consisting of the sampler, router, function blocks, and repository. The sampler feeds the data to the framework. It also formats the data so they can be processed by the router to identify tasks based on statistical distribution, keys, etc. This way the router can apply the domain-specific information found in the signals linked to the tasks for selecting function blocks given an input instance (Rosenbaum et al., 2017).

The repository act as the host for all functions and their corresponding parameters. It also stores special non-parameterized functions and sub-networks. Sub-networks can be pre-trained or model-driven modules. In Figure 1, the function block, f_{di} is a unit that performs computation as a whole independently. Where: d and i stand for the current depth level and the selected function block respectively. This implies that each function block can be regarded as a special non-parameterized function, a sub-network, a parameterized function, or an entire deep neural network. Parameterized functions implement popular deep neural network layers such as convolution layers (Conv), fully connected layers, etc., and any operation with a learnable parameter. Non-parameterized functions include activation functions such as rectified linear unit (ReLU) and any other operation without learnable parameters (Goodfellow et al., 2016).

To perform computations using function blocks, the router is called recursively as follows. At the depth level, $\mathrm{d} = 1$ it has chosen f_{13}, f_{21} at the depth level, $\mathrm{d} = 2$, and f_{34} at the depth level, $\mathrm{d} = 3$. At any depth level, the router picks the matching function block according to the task. Multiple selections of function blocks are possible at any depth level if multiple outputs are needed. In addition, the function block may be a layer or an entire sub-network with multiple layers provided that it can accept the output of the preceding function block. For each *lth* depth level, multiple function blocks from the repository can be graded by the router based on a score conditioned on the training signal for learning multiple tasks simultaneously in four ways; input composition, function composition, parameter composition, and output aggregation (Pfeiffer et al., 2023).

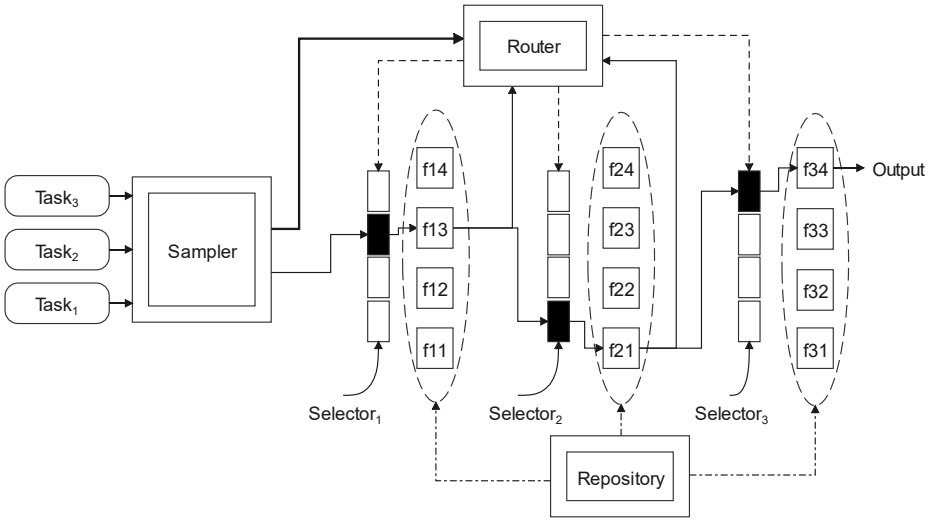

Figure 1: Multitask learning based on model selection.

Input composition gives the MTL framework to perform operations on inputs to learn a new task. Operations might include concatenation, merging, pruning, re-assembling, etc. This is a work of the sampler in collaboration with a router. Function decomposition is when the output of the previous function block is fed to the next function block. Parameter composition allows the MTL framework to join the parameters of the original function block with the new function block. Parameters might be element-wise added or can be made sparse to improve performance. Outputs of active function blocks are aggregated by using output composition.

The four ways for learning multiple tasks can be implemented in the router through hard routing and soft routing. Hard routing uses a binary vector at each depth level to represent the selected function block. That is if the router returns a score α_{di}, for each ith function block at the current depth level, d . $\alpha_{di} = \{0,1\}^{L_d}$ denotes the contribution of the chosen function block at the current depth level, given L_d as the total number of depth levels. For soft routing $\alpha_{di} \epsilon \lfloor 0,1 \rfloor^{L_d}$ such that at the depth level d, $\sum_i \alpha_{di} = 1$ represents a continuous probability distribution (Ahn & Kim, 2019; Pfeiffer et al., 2023). To determine which function blocks are active given the training signals, scores can be fixed or learned. Fixed scores are derived by using pre-defined function blocks for the task under consideration. This is a form of applying a fixed router for pilot decontamination based on the nature of the inputs. In cases with unknown function blocks for the given task. Scores can be learned as hard-learned routing or soft-learned routing with weighted function blocks at the corresponding depth level.

At this point, it's now obvious how fixed routing can be used to adapt the MTL framework for power allocation, pilot assignment, pilot design, and denoising in 5G MAMIMO systems with pilot contamination. It should be known that all function blocks with their parameters must be available in the repository. Non-parameterized function blocks are also part of the repository. Moreover, *PASS* and *SKIP* function blocks can be included to simplify the design of the routing algorithm in the router. The *PASS* function block feeds the next function block with un-modified data. The *SKIP* function block avoids all the function blocks at the current depth level. In the next section, the study illustrates how the MTL framework can be adopted for channel estimation in 5G MAMIMO systems with pilot contamination using the DRL-based channel estimator (Lim et al., 2021).

METHODS AND MATERIALS

Channel Estimation with the Proposed Multitask Framework

In (Lim et al., 2021), a deep residual learning approach was suggested for channel estimation to mitigate the effects of pilot contamination. The learning approach was made around the noise level estimator and the denoiser. This work conducted computer simulation experiments on the proposed approach under 5G MAMIMO system parameters given in Table 1. The experiments were implemented by the 5G and deep learning toolboxes. In each experiment, the user terminal (UT) sent the orthogonal frequency modulation (OFDM) grid with a known sounding reference signal (SRS) thru the 5G tapped delay line (TDL) channel. Line of sight (LOS) and non-line of sight (NLOS) 5G TDL channels were considered. Then, the base station acquired the CSI by using the received SRS and pre-known SRS. Data consisted of random binary sequences mapped to symbols using quadrature amplitude modulation with sixty-four constellation points (64QAM).

Table 1: 5G MAMIMO system parameters

Parameter	Value	Parameter	Value
Receive Antennas	128	Shadow fading deviation (LOS)	4dB
Transmit Antenna	1	Shadow fading deviation (NLOS)	7.82dB
Modulation	64QAM	Fast Fourier transform size	1024
5G Channel Model	TDL A-E	Cyclic prefix	Normal
AWGN model	Gaussian	Resource blocks (RBs)	52
User terminal velocity	30Km/h	Carriers per RB	12
Carrier frequency	4GHz	Slots	10
Subcarrier Spacing	15KHz	Symbols per slot	14
User terminal antenna height	1.5m	SRS period	2
Base station antenna height	10m	SRS period offset	0

Thereafter, each 64QAM symbol was assigned to the OFDM carrier and sent thru the 5G TDL channel to get the received data. Details on how CSI was estimated after SRS reception can be found in (MATLAB-Documentation, 2023). However, received SRSs and data were crucial in computing the output labels for training purposes using the approach in (Alnajjar & Abdallah, 2016).Initial experiments progressed with the DRL-based channel estimator detailed in (Lim et al., 2021) as it is. The experiments were then repeated with the same DRL but now there was no denoiser (DRL-No denoiser). Then, additional experiments were conducted with the same DRL again without the denoiser. For these additional experiments, the LSCs were not considered and the dimensions of the filters in the convolutional layers were increased (DRL-No denoiser and LCSs with new filters). Figure 2 provides the normalized mean square error (NMSE) results for these three experiments.

Unexpectedly, for SNR values above 15, the original learning approach did not give good NMSE. Repeating the experiments with the denoiser removed has an advantage in comparison. However, for SNR values below 15, NMSE performance was still poor. Increasing the size of filters and remaining with inputs that had no LSCs. Additional experiments led to remarkable NMSE performance for SNR values above 12.5. Judging from the NMSE results in Figure 2. The experiments revealed that the SNR, contents of the inputs, and the denoiser play a key role when it comes to the final NMSE performance. Contradicting NMSE performance given by these three

experiments led to the development of the more optimal DRL channel estimator based on the MTL framework (DRL-MTL).

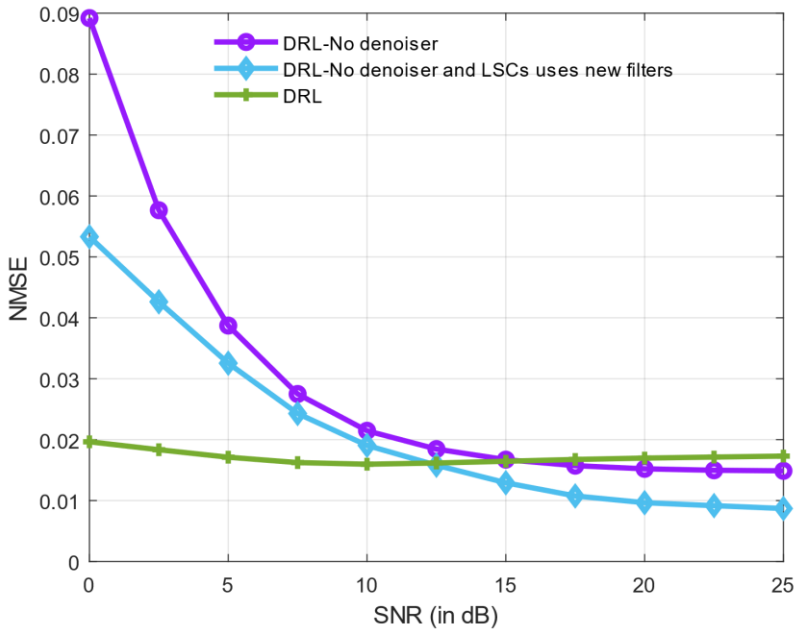

Figure 2: Performance of channel estimation.

The procedure for applying the MTL framework to arrive at the DRL-MTL-based channel estimator was as follows. Step 1, identification of the tasks. Step 2, the definition of the function blocks. Step 3, specification of how the function blocks are selected. Step 4, designation how the outputs are aggregated. The last step is the invention of the training method. For this study, Table 2 provides the tasks and function blocks at each depth level.

Table 2: Tasks and function blocks at each depth level

{Dimension, SNR}	Task	d=1	d=2	d=3	d=4	d=5	d=6
{3,<=10}	T1	Conv{3, 3x3} ReLU	Conv{3, 3x3} ReLU	Conv{3, 3x3} ReLU	Conv{3, 3x3} ReLU	Conv{2,3x 3} ReLU Conv{2,3x 3} ReLU	Denoising subnetwork
{2,>10}	T2	Conv{64 ,9x9} ReLU	Conv{64 ,9x9} ReLU	Conv{64 ,9x9} ReLU	Conv{64 ,9x9} ReLU	Conv{32,5 x5} ReLU	Pass

Parameters $\{n_f, f_x \times f_y\}$ in the Conv layer are as follows. n_f states the number of filters, f_x and f_y represent the width and height of the filter in the x and y dimension respectively. The denoising subnetwork is explained in (Lim et al., 2021). Task 1 was established if the received sounding reference signal has the SNR below or equal to 10. In addition, the LSCs and the channel gains must be available at the base station. Task 2 was designated by the received sounding reference signal with SNR above 10 when only the channel gains are present at the base station.

As per Table 2, the information about data dimensions was used to detect whether the

inputs have LSCs or not. Fixed hard routing was adopted for function block selection at each depth level. The router implemented a routing scheme with scores represented in Table 3. Depth level 6 stood for the output. Therefore, the aggregation of outputs was a selective one based on the score at this depth level.

Table 3: Routing scheme

	Task 1	**Task 2**
Depth level, d	Score, α_{di}	Score, α_{di}
1	[1 0]	[0 1]
2	[1 0]	[0 1]
3	[1 0]	[0 1]
4	[1 0]	[0 1]
5	[1 1 0]	[0 0 1]
6	[1 0]	[0 1]

The training phase had two parts. Forward and backward pass. Algorithm 1 offers the forward pass procedure in detail. The sampler fetches the training sample from the database, D_s , and gives the $\langle \widetilde{H}_{jk}^{LS}, H_{jk}, H^r, t \rangle$ as the outputs. $\widetilde{H}_{jk}^{LS}$ and H_{jk} are as stated in Equation (1). H^r was the signal that can be easily applied by the router to select the function block at each depth level. t was the task identifier. Given, the signal H^r, task identifier t, and the depth level d. The router iterated L_d times picking the function block that produced the output. Intermediate outputs were not utilized for routing in this study.

The backward pass is given by algorithm 2. It evaluated the loss function based on the task identifier. Then, it found the gradients for chosen function parameters in the routing path. Finally, it updated the chosen parameters. The complete training procedure is outlined by algorithm 3.

Algorithm 1: Forward pass

Input: D_s, s ; s is the current sample
D_s is the database with all the samples

Output: $p, \widehat{H}_{jk}, v$; p is the structure that holds parameters for all function blocks
$\widehat{H}_{jk}$ stands for the predicted channel for the k^{th} user at the j^{th} base station
v Initially empty, traces the function block used at each depth level

1: $\langle \widetilde{H}_{jk}^{LS}, H_{jk}, H^r, t \rangle \leftarrow sampler(D_s, s)$; Load data sample from the database
2: $v \leftarrow \{\}$
3: **for** $l = 1$ in L_d **do**
4: $i \leftarrow router(H^r, t)$; Determine the function block
5: **if** $d = 1$
6: $\widehat{H}_{jk} \leftarrow function_i(\widetilde{H}_{jk}^{LS})$; Compute using the selected function block
7: **else**
8: $\widehat{H}_{jk} \leftarrow function_i(\widehat{H}_{jk})$; Compute using the selected function block
9: **end if**
10: $v = v \cup i$; Capture the function block(s) used at each depth level
11: **end for**

Algorithm 2: Backward pass

Input: $\theta_{select}, H_{jk}, \widehat{H}_{jk}$; H_{jk} denotes the channel gains without pilot contamination
θ_{select} is the structure that contains the parameters for all selected function blocks in the routing path

Output: θ_{update}; The structure that holds updated parameters for all selected function blocks

1: $J \leftarrow loss_t(H_{jk}, \widehat{H}_{jk},)$; Compute the loss
2: $\Delta J \leftarrow gradient(J, \theta_{select})$; Compute the gradients
3: $\theta_{update} \leftarrow update(\Delta J, \theta_{select})$; Update the selected parameters

Algorithm 3: Training procedure

$\theta \leftarrow initializeFunctionParameters()$; Initialize all function parameters in the repository

for $s = 1$ to S **do;** S is the total number of samples in the database

1: $Perform\ forward\ pass\ using\ Algorithm\ 1$

2: $\theta_{select} = \text{getParameters}(v, \theta)$; Choose parameters used during forward computation

3: $Perform\ backward\ pass\ using\ Algorithm\ 2$

4: $\theta = \text{set Parameters}(v, \theta, \theta_{update})$; Update the repository with new updated parameters

end for

PERFORMANCE EVALUATION

The DRL-MTL-based channel estimator was trained by using LSC and channel gains for Task 1. Task 2 involved channel gains only. 5G MAMIMO system parameters outlined in Table 1 were called upon for synthesizing the training data via computer simulations. Training of the DRL-MTL followed Algorithm 3. The testing phase consisted of the forward pass given by Algorithm 1. The output label, H_{jk} was computed by using the channel estimator proposed in (Alnajjar & Abdallah, 2016). The mean square error (MSE) expressed by Equation (3) determined the loss function during training. $\widehat{H}_{jk}$ represents the estimated channel gains.

$$\mathcal{L}(\omega, b) = \frac{1}{|S|}\sum_{s=1}^{S}\sum_{r=1}^{N_{sc}}\sum_{q=1}^{N_{sy}} ||H_{jk} - \widehat{H}_{jk}||_2^2 \quad (3)$$

Where S represents the total number of samples in the database grouped by the same SNR. N_{sc} and N_{sy} denote the number of subcarriers and symbols. It should be known that H_{jk} and $\widehat{H}_{jk}$ dimensions are $N_{sc} \times N_{sy} \times N_r \times N_t \times 3$ for Task 1 rising from real, imaginary, and LSCs. $N_{sc} \times N_{sy} \times N_r \times N_t \times 2$ for Task 2 when LSCs are not considered. N_r and N_t stand for the number of receiving and transmitting antennas respectively. Figure 3 plots the NMSE performance versus SNR for the DRL-MTL and other three experiments as stated in Section 3 (see Figure 2).

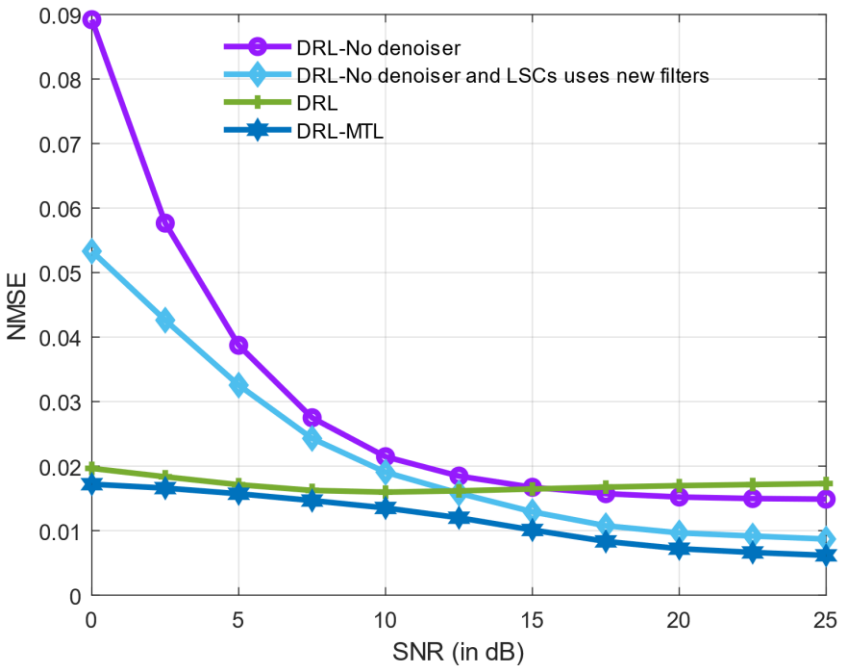

Figure 3: Performance of channel estimation with multitask learning.

The NMSE was expressed as in Equation (4). N_{Sample} indicates the number of observations per SNR.

$$NMSE = \frac{1}{N_r}\sum_{s=1}^{N_{sample}} \frac{||H_{jk}-\widehat{H}_{jk}||_F}{||H_{jk}||_F} \quad (4)$$

As shown in Figure 3, the DRL-MTL has low NMSE at all SNR values due to the following two reasons. First, function blocks that have more ability to learn the mapping from $\widetilde{H}_{jk}^{LS}$ to H_{jk} are activated accordingly at each depth level. Second, each path that consists of a series of composed function blocks is exposed to the knowledge it can attain and reproduce only. These factors that render the DRL-MTL powerful are all achieved by the simple hard-fixed routing strategy. The DRL-MTL performance can be enhanced further during training by applying masks on the filters. Masks would produce the DRL-MTL model with fewer parameters and small in size, because, Task 1 and Task 2 are sharing the parameters (Mallya & Lazebnik, 2018). Learned routing is also possible for improvements if information about SNRs is not available (Rosenbaum et al., 2017).

CONCLUSION AND RECOMMENDATION

This paper has proposed the MTL framework for pilot decontamination in 5G MAMIMO systems. The MTL framework can be used to selectively activate and train the function blocks suitable for the pilot assignment, power allocation, pilot design, and denoising to lower the effects of pilot contamination. As demonstrated by the NMSE results, the channel estimator produced by the MTL framework performed better under all SNR ranges. In the future, more investigation is required on learned routing to support more tasks, sharing parameters to decrease the size of the final MTL-based model, and ways for aggregating the outputs found in existing deep learning-based pilot decontamination schemes.

REFERENCES

Ahn, C., & Kim, E. (2019). Deep Elastic Networks with Model Selection for Multi-Task Learning. *ArXiv*.

Alnajjar, K. A., & Abdallah, S. (2016). Performance of low complexity receivers for massive MIMO with channel estimation and correlation. *2016 IEEE 3rd International Symposium on Telecommunication Technologies (ISTT)*, 1–5. https://doi.org/10.1109/ISTT.2016.7918074

Aoudia, F. A., & Hoydis, J. (2018). End-to-End Learning of Communications Systems Without a Channel Model. *Conference Record - Asilomar Conference on Signals, Systems and Computers*, *2018-Octob*, 298–303. https://doi.org/10.1109/ACSSC.2018.8645416

Balevi, E., Doshi, A., & Andrews, J. G. (2020). Massive MIMO channel estimation with an untrained deep neural network. *IEEE Transactions on Wireless Communications*, **19**(3), 2079–2090. https://doi.org/10.1109/TWC.2019.2962474

Chun, C. J., Kang, J. M., & Kim, I. M. (2019). Deep Learning-Based Joint Pilot Design and Channel Estimation for Multiuser MIMO Channels. *IEEE Communications Letters*, **23**(11), 1999–2003. https://doi.org/10.1109/LCOMM.2019.2937488

D'Andrea, C., Zappone, A., Buzzi, S., & Debbah, M. (2019). Uplink Power Control in Cell-Free Massive MIMO via Deep Learning. *2019 IEEE 8th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing, CAMSAP 2019 - Proceedings*, 554–558. https://doi.org/10.1109/CAMSAP45676.2019.9022520

Fatema, N., Xiang, Y., & Natgunanathan, I. (2017). Analysis of a semi blind pilot decontamination method in massive MIMO. *2017 27th International Telecommunication Networks and Applications Conference, ITNAC 2017*, *2017-Janua*, 1–6. https://doi.org/10.1109/ATNAC.2017.8215368

Ferrante, G. C., Geraci, G., & Quek, T. Q. S.

(2017). A Group-Blind Detection Scheme for Uplink Multi-Cell Massive MIMO. *IEEE Vehicular Technology Conference*, *2017-June*. https://doi.org/10.1109/VTCSpring.2017.8108326

Giordano, L. G., Campanalonga, L., Lopez-Perez, D., Garcia-Rodriguez, A., Geraci, G., Baracca, P., & Magarini, M. (2018). Uplink sounding reference signal coordination to combat pilot contamination in 5G massive MIMO. *IEEE Wireless Communications and Networking Conference, WCNC*, *2018-April*, 1–6. https://doi.org/10.1109/WCNC.2018.8377316

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press. http://www.deeplearningbook.org

Hirose, H., Tomoaki, O., & Guan, G. (2021). Deep Learning-Based Channel Estimation for Massive MIMO Systems With Pilot Contamination. *IEEE Open Journal of Vehicular Technology*, **2**(4), 67–77. https://doi.org/10.1109/OJVT.2020.3045470

Jiang, P., Wen, C., Jin, S., & Li, G. Y. (2021). Dual CNN based Channel Estimation for MIMO-OFDM Systems. *IEEE*, 1–14.

Kim, K., Lee, J., & Choi, J. (2018). Deep Learning based Pilot Allocation Scheme (DL-PAS) for 5G Massive MIMO System. *IEEE Communications Letters*, **7798**(c), 1–4. https://doi.org/10.1109/LCOMM.2018.2803054

Larsson, E. G., & Marzetta, T. L. (2014). Massive MIMO for Next Generation Wireless Systems. *IEEE Communications Magazine*, *February*, 186.

Lim, B., Yun, W. J., Kim, J., & Ko, Y. (2021). Joint Pilot Design and Channel Estimation using Deep Residual Learning for Multi-Cell Massive MIMO under Hardware Impairments. *ArXiv:2108.04485v1*, 1–12. http://arxiv.org/abs/2108.04485

Mallya, A., & Lazebnik, S. (2018). PackNet : Adding Multiple Tasks to a Single Network by Iterative Pruning. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7765–7773. https://doi.org/10.1109/CVPR.2018.00810

Marzetta, T. L. (2015). Massive MIMO: An introduction. *Bell Labs Technical Journal*, **20**, 11–12. https://doi.org/10.15325/BLTJ.2015.2407793

MATLAB-Documentation. (2023). *NR Uplink Channel State Information Estimation Using SRS*. https://www.mathworks.com/help/5g/ug/nr-uplink-channel-state-information-estimation-using-srs.html

Misra, I., Shrivastava, A., Gupta, A., & Hebert, M. (2016). Cross-Stitch Networks for Multi-task Learning. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2016-Decem*, 3994–4003. https://doi.org/10.1109/CVPR.2016.433

Omid, Y., Hosseini, S. M., Shahabi, S. M., Shikh-Bahaei, M., & Nallanathan, A. (2021). AoA-Based Pilot Assignment in Massive MIMO Systems Using Deep Reinforcement Learning. *IEEE Communications Letters*, **25**(9), 2948–2952. https://doi.org/10.1109/LCOMM.2021.3089234

Pfeiffer, J., Ruder, S., Vulić, I., & Ponti, E. M. (2023). Modular Deep Learning. *ArXiv*, 1–69.

Rosenbaum, C., Klinger, T., & Riemer, M. (2017). Routing Networks : Adaptive Selection of Non-Linear Functions for Multi-Task Learning. *ArXiv*, 1–16.

Strezoski, G., Noord, N. Van, & Worring, M. (2019). *Many Task Learning With Task Routing*. *Iccv*. https://doi.org/10.1109/ICCV.2019.00146

Xu, J., Zhu, P., Li, J., & You, X. (2019). Deep Learning-Based Pilot Design for Multi-User Distributed Massive MIMO Systems. *IEEE Wireless Communications Letters*, **8**(4), 1016–1019. https://doi.org/10.1109/LWC.2019.2904229

Zhao, L., Zhao, H., Zheng, K., & Xiang, W. (2018). *Massive MIMO in 5G Networks : Selected Applications*. Springer.