Multi-Layer Perceptron for Channel State Information Estimation: Design Considerations

Andrew Oosthuizen*[†‡], Marelie H. Davel[†‡§], Albert Helberg[†]
[†]*Faculty of Engineering, North-West University*, South Africa.
[‡]*Centre for Artificial Intelligence Research (CAIR), South Africa.*
[§]*National Institute for Theoretical and Computational Sciences (NITheCS), South Africa.*

aj.oosthuizen.ao@gmail.com
marelie.davel@nwu.ac.za
albert.helberg@nwu.ac.za

Abstract—**The accurate estimation of channel state information (CSI) is an important aspect of wireless communications. In this paper, a multi-layer perceptron (MLP) is developed as a CSI estimator in long-term evolution (LTE) transmission conditions. The representation of the CSI data is investigated in conjunction with batch normalisation and the representational ability of MLPs. It is found that discontinuities in the representational feature space can cripple an MLP's ability to accurately predict CSI when noise is present. Different ways in which to mitigate this effect are analysed and a solution developed, initially in the context of channels that are only affected by additive white Guassian noise. The developed architecture is then applied to more complex channels with various delay profiles and Doppler spread. The performance of the proposed MLP is shown to be comparable with LTE minimum mean squared error (MMSE), and to outperform least square (LS) estimation over a range of channel conditions.**

Index Terms—**Channel State Information, Deep learning, Multi-Layer Perceptron, Long-Term Evolution**

I. Introduction

Channel state information (CSI) is an integral part of wireless communication standards, and combines all channel impairments into a single estimation. Standards such as long-term evolution (LTE) and WiFi use pilot symbols, known to both the receiver and transmitter, to obtain CSI with one of several estimation methods that vary in computational cost. CSI is used to quantify the quality of the channel, which can then be used to make decisions on modulation scheme selection and transmission frequencies. CSI can also be used to equalise received transmissions by applying the inverse of the channel conditions, thus restoring data to its transmitted state if the CSI is of adequate accuracy [1], [2].

As deep learning methods have gained traction in CSI estimation in recent years [3], the field has seen many implementations outperform statistical methods such as least square (LS) [4], maximum likelihood (ML) [5] and minimum mean squared error (MMSE) [6] estimation. These deep learning implementations presented the input features to the CSI estimation networks in several different ways and it is unclear how the representation, specifically, affects the training procedure or performance of neural networks.

The paper is organised as follows: Related work is presented in Section II. Section III discusses the experimental setup, including the dataset setup, MLP architecture and training protocol; and is followed by an analysis of the results in Section IV. Within Section IV, the feature representation, hyperparameter choices and observed performance are discussed.

II. Background

Deep learning has been successfully applied to the CSI estimation problem using different deep learning architectures, such as recurrent neural networks (RNNs), long short-term memory (LSTM) networks and convolutional neural networks (CNNs) [3]. The attractiveness of deep learning methods has been accredited to the computational efficiency of deep learning models (once trained) over techniques such as MMSE that use resource-consuming matrix calculations. In addition, Jiang et al. [7] state that the non-linear capabilities of deep learning networks play an important role in estimating non-linear channel conditions more effectively than linear statistical methods. Finally, the data-driven nature of deep learning is fitting for telecommunication implementations, as ExaBytes of data is transmitted yearly [8] and real-world conditions can easily be simulated to generate data.

Due to the data-driven nature of deep neural networks (DNNs) [9] and the ease of training a multi-layer perceptron (MLP) compared to other deep learning architectures, it is a natural choice to apply an MLP to this task. MLPs and deep neural networks have been shown to outperform LS methods and contend with MMSE methods under complex channel conditions [10], [11]. The task can be framed in different ways, with either pilot points data or an already generated CSI estimate of low quality as input. In related work, it was seen that this data can be presented in several different ways to DNNs, with main representations: as the angle and absolute values [12], as complex values directly [13], and as quadrature and in-phase components (QI) of this complex data [10].

More recently deep learning architectures have been implemented to reduce the overhead caused by CSI in resource allocation problems [14] and for estimation of CSI feedback for 5G implementations [15]. These two papers both utilise different architectures and input features with Godala et al. [15] using CNNs with real and imaginary input features and Khan et al. [14] using DNNs and angular input features.

DNNs have many hyperparameters that play a role in their performance. Some of these hyperparameters are related to the dataset, such as the presentation of features or sample size, while others are related to the architecture of the DNN. There are also hyperparameters that control how the networks are trained, such as optimisers and learning rates. When confronted with these many interacting variables, it is crucial to understand the impact of choosing one hyperparameter over the other. Due consideration should also be given to the specific range of hyperparameter values used when applying hyperparameter searches, as some hyperparameter values perform better in combinations with specific values of other hyperparameters.

This paper investigates the performance of an MLP in predicting the CSI of a single input, single output (SISO) channel setup under various LTE [16] channel conditions. Within this context, the following questions are addressed:

- How should features be represented for CSI estimation?
- How sensitive are results to MLP architectural choices?
- What order of performance can be expected for channels with different complexities?

In answering these questions, several observations are made with regard to which feature representations perform best, and how architectural choices influence the performance of MLPs for CSI estimation.

III. Experimental setup

Synthetic data is generated and used to train an MLP. The data generation process is described in Section III-A, the MLP architecture (including the varied elements) in Section III-B, and the hyperparameter optimization process in Section III-C.

A. Dataset setup

The initial dataset is generated using a SISO setup with no noise and no delay profile through the LTE toolbox available in Matlab®. The downlink transmission used in this study makes use of orthogonal frequency-division multiplexing (OFDM) with a bandwidth of 10MHz and short cyclic prefixes. To ensure that only the performance of the MLP is measured, no link level techniques are used and control frames are removed as well.

Each data sample consists of a single slot containing 4 pilot symbols. In Figure 1 we see that slots contain 7 OFDM signals with 12 sub-carriers, each depicting a Quadrature Phase Shift Keying (QPSK) modulated symbol. The input data is provided to the MLP from the four pilot symbols, extracting 2 features per symbol and then flattening the data into a single tensor with a width of 8. This process is also applied to the target data using the true CSI for the entire slot to produce a 168-wide tensor. It is thus expected for the MLP to estimate the true 168-dimensional CSI given the 8-dimensional input.

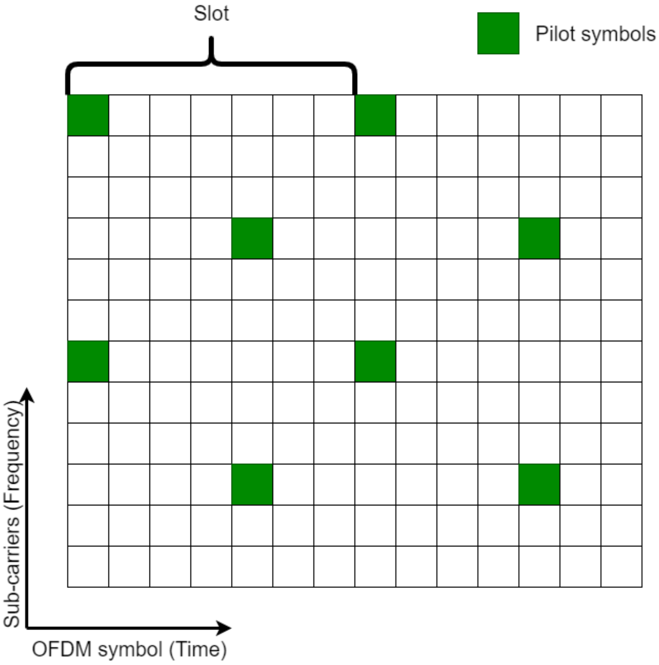

Fig. 1. LTE frames contain pilot symbols used for CSI estimation.

Using these principles, several datasets are created to train MLPs. The least complex dataset contains only primary multipath effects since no delay profile or noise is added to the channel. This is referred to as the 'noiseless' dataset. The 'noisy' dataset is similar to the noiseless dataset, except that it adds 20dB additive white Gaussian noise (AWGN) to the channel. To further test the MLP's ability, delay profiles are added to the noisy dataset making the channel more complex. Each delay profile incrementally increases the complexity of the multipath components sent over the channel. The Extended Pedestrian A model (EPA), Extended Vehicular A model (EVA) and Extended Typical Urban model (ETU) [17] respectively represent low, medium and high delay spread environments. These three datasets are called the 'delay profile' datasets. Lastly, a 'Doppler fading' dataset is constructed by adding Doppler fading to the EVA dataset. This process of increasing complexity is used to investigate the effect of different channel conditions on the MLP's performance.

Each dataset consists of a training set of 20 000 samples and a validation set of 5 000 samples. An unseen test for each dataset consisting of 5 000 samples is also generated. For each test set, the LS and LTE MMSE estimates are calculated as a comparative baseline. The LS estimate has low complexity and is obtained by linearly extrapolating the CSI from known pilot data, while the standardised LTE MMSE downlink estimation is obtained as described in [18]. These two methods give an indication of how well the MLP is performing against statistical CSI estimation methods.

B. MLP architecture

The deep neural network used in this paper is an MLP, as MLPs consist of fewer architectural hyperparameters than other neural networks such as CNNs or LTSMs. More complex architectures add an additional architecture selection process during training. This paper thus uses the simplest network to probe feature representation which other networks can further utilise.

The MLP network has a general architecture consisting of several layers of fully-connected neurons, each containing the same set amount of neurons per layer. Between each layer lies an activation function, as well as a batch normalisation layer [19]. The activation function introduces non-linearity into the model, enabling it to learn non-linear data distributions and perform complex regression tasks. TanH is a commonly used activation function for CSI prediction and has shown promising results over other activation functions for this task [20]. Batch normalisation is a concept that was developed by Ioffe et al. [19] originally "to reduce internal covariate shift" within neural networks but with additional benefits becoming apparent over time. This method enables accelerated training in neural networks by normalising activations between layers. Ioffe et al. also claim that less careful initialisation of networks is needed to obtain a successfully trained network if batch normalisation is present.

The depth (the number of hidden layers in a network) and width (the number of nodes in a hidden layer) of the network are considered structural hyperparameters in Section III-C. Structural hyperparameters control the size of a network and thus its representational capability. The representational capacity of a DNN is a measure of the complexity of the data distribution that the network is able to learn. It is important to make the network large enough to solve the problem, however, making the network too large makes it computationally expensive to train and has been known to create vanishing gradient problems.

C. *Training protocol*

All networks are trained using the Adam [21] optimiser and the mean square error (MSE) loss metric. Adam is selected for its ability to adapt the learning rate of the training algorithm for each parameter individually. MSE is used as loss function since this research is interested in the difference between the real CSI and estimated CSI. All networks are trained to convergence, and early stopping is used: the model is selected at the epoch of best validation accuracy.

Different hyperparameters are assessed by using a hyperparameter grid search in which all possible combinations of the selected hyperparameter values are applied. Each combination of hyperparameters is also trained over three random initialisation seeds to ensure that strong or weak initialisation does not affect trend analysis over the sweep. Presented loss and bit error rate (BER) results are the averages over the three random seeds of a specified hyperparameter set. Since exhaustive hyperparameter searches are conducted it is important to select a wide range of hyperparameter values that can be narrowed down if needed.

In the initial grid search, the protocol exhaustively search over possible combinations of depth {1, 2, 3, 4} and width {10, 100, 1 000}. At the same time the protocol also sweeps over batch size {16, 32, 64, 128} and learning rate{1e-02, 1e-03, 1e-04, 1e-05}, referred to as the training hyperparameters. This grid search thus contains 576 possible hyperparameter combinations, including initialisation seeds. These hyperparameter ranges are chosen as the structural hyperparameters provide sufficient representational capacity while still being computationally feasible to train. The learning rate is selected over a wide range of values, while the batch size selection remains large enough to see effect but still be memory conscious.

The best performing networks are chosen based on the lowest validation loss and are tested on the unseen test set. This protocol ensures that networks do not overfit training data and can generalise to unseen data.

IV. Analysis and results

A. *Feature representation*

Following Erdogmus et al. [12], the angle and absolute values are initially used as input features. The initial model is constructed using the noiseless dataset to train the MLP; this is done to create a baseline for comparison and to validate the training process. Note that, when a BER of 0 is calculated on a specific dataset, it is reported as δ. For the noisless dataset it is expected that the MLP obtains a BER of δ, as this is what LS obtains on these channel conditions.

When evaluating these networks on the test set, a loss of 4.61e-2 and BER of 8.33e-4 are obtained. These results are concerning as a simple LS implementation can obtain a BER of δ. When further analysing the results of the best performing MLPs, it is found that the network has difficulty estimating the CSI angle close to the discontinuity within the angle representation of the data. This discontinuity is found at the point where the CSI rotates past 3.14 (PI) or 180 degrees. Angles use a radian representation where the angle abruptly moves from 3.14 to -3.14 instead of continuing linearly to 3.15 when passing 180 degrees from the original starting point. Examining the angle component of OFDM symbols in Table I, it is observed that the network has difficulty correctly predicting the angle of the CSI when approaching PI. The incorrectly estimated sub-carriers are shown in bold in Table I.

TABLE I
Predicted CSI angle over two OFDM signals compared to the target CSI angle over the same OFDM signals

	Predicted CSI (OFDM symbol)		Target CSI (OFDM symbol)	
Sub-carrier	1	2	1	2
1	**1.36**	**1.35**	3.09	3.09
2	3.05	3.05	3.05	3.05
3	3.00	3.01	3.01	3.01
4	2.88	2.89	2.96	2.96
5	2.85	2.85	2.92	2.92
6	2.80	2.8	2.88	2.88
7	2.77	2.77	2.83	2.83
8	2.72	2.72	2.79	2.79
9	2.67	2.68	2.75	2.75
10	2.56	2.55	2.71	2.71
11	2.52	2.51	2.66	2.66
12	2.46	2.45	2.62	2.62

After removing batch normalisation from the MLP (reasoning as discussed in Section IV-B), a BER of δ is obtained on the noiseless data using the angle and absolute value data representation.

When experimenting with the same feature representation and the noisy dataset, another concerning behaviour is observed: As a baseline, the best performing network trained on noiseless data achieves a loss of 2.31e-1 and BER of 5.16e-3 on the noisy test set. After following the training protocol, the best performing networks (as measured using the validation set) are obtained. When applying these best performing networks to the test set, a loss of 1.28e-1 and BER of 1.72e-2 are observed. This behaviour characterises a dysfunctional training process as the loss is not correlated to the BER: it is expected that, when the predicted CSI looks more similar to the real CSI, shown as a lower MSE loss, the performance, shown as BER, would improve.

Examining the predicted CSI, it can be observed that adding noise to the channel again introduces the discontinuity problem, similar to the results in Table I. This is not unexpected, as the network now struggles to find the precise point at which the discontinuity exists due to the added noise. Ahmed et al. [22] explain how deep learning methods may struggle using data with non-euclidean features and perform much better on data presented in euclidean space. Therefore, it is proposed to change the extracted features from absolute value and angle to the QI of the CSI representation. This places the data representation on a continuous euclidean plane that may be interpreted better by the MLP.

The QI representation is implemented on the noisy dataset using the hyperparameter sweep protocol. Analysing the results, a decrease in the loss of the training hyperparameters are observed. Testing the best performing networks a loss of 5.23e-3 and BER of δ is obtained. These results show that changing the features used by the MLP has increased the MLP's ability to generate accurate CSI. It is assumed that this is the result of using a feature space of euclidean nature, which is more fitting to the MSE loss function.

B. Effect of hyperparameter choices

In this section, the effect of structural hyperparameters, training hyperparameters and the effect of batch normalisation in the training process is discussed.

In examining the structural hyperparameters when training the MLP on the noiseless data set using the angle and absolute representation, it was found that MLPs with deeper and wider layers perform better than smaller networks. These results (not shown here) indicate that increasing the representational ability of the MLP by providing a larger architecture decreases the loss on the validation set, thus lowering the BER. For the remainder of the experiments, moving on from the noiseless dataset, a single MLP architecture which consists of 4 layers of a 1 000 nodes each is used, to ensure consistent results and to simplify the training analysis. This decision can be made as it has been confirmed that the network has the representational ability to make CSI predictions.

The analysis in this paper always considers training hyperparameters in the hyperparameter sweeps. The initial sweeps (not shown here) on noiseless data, with an angle and absolute representation, showed that a learning rate of 1e-3 performed best over a range of runs. It was also found that performance increases as batch sizes become larger. This leads to the conclusion that increasing the amount of data processed before a gradient update (the batch size), improves results. Additional hyperparameter searches are thus done over the same learning rate range and an extended batch size range: {32, 128, 256, 512}. The batch size is extended as previous results indicate improved learning environments at higher batch sizes.

As stated in Section IV-A the results of the angle and absolute data representation on noiseless data are less than optimal, especially when compared to LS results, due to the presence of a discontinuity in the data representation. To help the MLP learn how to interpret the discontinuity found in the representational plane the decision to change hyperparameters is made. The activation function and batch normalisation layers can be altered when inspecting structural hyperparameters. There are no grounds for changing the TanH activation function as it has been widely used in the CSI field. Thus the batch normalisation layer is examined.

Santurkar et al. [23] suggest that batch normalisation does not succeed in reducing internal covariance shift but rather in smoothing the optimisation landscape that leads to more predictive gradient behaviour. This is done by reparameterising the underlying optimisation problem. Furthermore, smoothing the optimisation landscape may be why the MLP is having trouble identifying the discontinuity, as the exact position of where the MLP acknowledges the discontinuity may be a sharp point in the optimisation landscape that is smoothed over.

This hypothesis is tested by running a hyperparameter sweep on the noiseless dataset but removing batch normalisation layers from the MLP architecture. Table II depicts the architectural hyperparameters for a specific set of learning hyperparameters from the hyperparameter sweep. When examining the results, it is observed that the best performing networks have loss results that are better by several orders of magnitude, when compared with the batch normalisation sweeps. Applying this network on the test set, a BER of δ is obtained, which is the expected result from an MLP on this dataset.

TABLE II
BEST PERFORMING STRUCTURAL HYPERPARAMETERS TEST LOSS RESULTS FOR MLP TRAINED ON NOISELESS DATA AND NO BATCH NORMALISATION LAYERS

	Depth			
Width	1	2	3	4
10	8.41e-2	9.45e-2	1.27e-1	1.95e-1
100	5.41e-2	4.46e-2	2.54e-2	1.14e-3
1 000	4.71e-2	1.65e-3	9.49e-8	6.33e-8

From these results, it can be observed that the batch normalisation layer smooths the results over different hyperparameter combinations, further referred to as the hyperparameter landscape. This can be seen from the more diverse sweep results that are obtained when removing batch normalisation. However, it is speculated that this smoothing prevented the training protocol from finding a point in the hyperparameter landscape where the discontinuity is recognised.

The hypothesis is further investigated by applying the QI data representation and implementing a hyperparameter sweep on noisy data using networks with and without batch normalisation. If the assumptions thus far are correct, there should be no reason for batch normalisation to decrease the network's performance significantly. These results are reported on over different batch sizes and learning rates with and without batch normalisation in Table III and Table IV, respectively.

TABLE III
TEST LOSS RESULTS OVER GROUPED TRAINING PARAMETER FOR NOISY DATA WITH QI VALUES AS FEATURES

	Learning rate			
Batch size	1e-5	1e-4	1e-3	1e-2
32	5.28e-3	5.22e-3	5.49e-3	2.64e-2
128	5.32e-3	5.22e-3	5.49e-3	1.40e-2
256	5.33e-3	5.26e-3	5.56e-3	9.37e-3
512	5.36e-3	5.31e-3	5.56e-3	7.59e-3

TABLE IV
THE SAME SETUP AS IN TABLE III, EXCEPT WITH BATCH NORMALISATION

	Learning rate			
Batch size	1e-5	1e-4	1e-3	1e-2
32	5.47e-3	5.64e-3	5.37e-3	6.33e-3
128	5.47e-3	5.5e-3	5.36e-3	5.75e-3
256	5.48e-3	5.48e-3	5.47e-3	5.66e-3
512	5.46e-3	5.47e-3	5.57e-3	5.59e-3

When Analysing these tables, a smoother set of results in the same range as the MLP using a QI feature set and no batch normalisation is noted. From the results found in IV, batch normalisation is reintroduced into the MLP architecture as a smoother hyperparameter landscape is obtained, ensuring networks that perform in the expected range from the training process. In the results presented in Table III and Table IV it is seen that this is done at a negligible loss in accuracy but would provide a speedup in training [19].

Thus, batch normalisation is suspected of having a smoothing effect in the optimisation space. While this feature is generally desired in the optimisation process, it can potentially be damaging when implementing a non-euclidean data representation, which has particular optimisation points.

C. Observed performance

To test the ability of the MLP CSI estimator, the three delay profile datasets are applied. The networks are trained making use of the same training protocol as described in Section III-C except for the following, as motivated in Section IV-B:

- The structural hyperparameters are fixed at 4 layers and 1 000 neurons.
- Adapting the range of batches searched over to {32, 128, 256, 512}.

In Table V the BER results of the best performing network of each delay profile tested on all delay profiles (the corresponding delay profile as well as others) are displayed. It is observed that each network performs best on the data it was trained on, but does not fail when applied to other delay profiles, showing a degree of cross-condition generalisation. Comparing the results to an LS implementation it is noted that the network improves on the LS BER, especially on EPA and EVA data.

TABLE V
BER OF MLP NETWORKS TRAINED ON THE DELAY PROFILE DATASETS AND APPLIED TO THE SAME OR OTHER DELAY PROFILE TEST SETS

Trained on	EPA	EVA	ETU
EPA	3.97e-3	1.10e-2	3.63e-2
EVA	4.82e-3	6.88e-3	1.57e-2
ETU	6.69e-3	8.35e-3	1.16e-2
LS Method	1.20e-2	1.43e-2	3.28e-2

The channel is made more complex by applying the Doppler dataset. Following the training protocol described previously, the results in Table VI are obtained. This table shows that the networks outperform LS and can compete with MMSE at lower Doppler frequencies, even with minimal statistical information on the channel conditions. Finally, it is observed that the MMSE method obtains similar BER performance for all Doppler frequencies.

TABLE VI
BER OF DIFFERENT EQUALISATION METHODS ON THE DOPPLER DATASET WITH VARIOUS AMOUNTS OF DOPPLER SHIFT

Doppler (Max Hz)	MLP	LS	MMSE
50	6.99e-3	1.38e-2	8.6e-3
100	8.01e-3	1.45e-2	8.6e-3
200	9.13e-3	1.5e-2	8.6e-3

V. CONCLUSION

This paper implements an MLP for CSI estimation over several OFDM symbols under various LTE channel conditions for a SISO system. Through an extensive model development process, an MLP architecture was found capable of contending with MMSE methods and improving LS estimation results in complex LTE channel conditions. These results can however only be obtained when utilising the correct hyperparameters for selecting and training the MLP network. First, using wide hyperparameter sweeps it is found that a discontinuous feature representation can obtain results comparable to LS in less complex channel conditions at the cost of removing batch normalisation. However, when considering the effect the feature representation has on the results, this research motivates for the importance of using a continuous euclidean feature representation when using complex channel conditions.

ACKNOWLEDGEMENT

The authors gratefully acknowledge the financial support of this study by the Telkom CoE at the NWU and Hensoldt South Africa.

References

[1] D. F. Carrera, C. Vargas-Rosales, N. M. Yungaicela-Naula, and L. Azpilicueta, “Comparative study of artificial neural network based channel equalization methods for mmWave communications,” *IEEE Access*, vol. 9, pp. 41 678–41 687, 2021.

[2] Y. S. Cho, J. Kim, W. Y. Yang, and C. G. Kang, *MIMO-OFDM wireless communications with MATLAB*. IEEE Press, 2011.

[3] C. Luo, J. Ji, Q. Wang, X. Chen, and P. Li, “Channel state information prediction for 5G wireless communications: A deep learning approach,” *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 1, pp. 227–236, 2018.

[4] E. Karami, “Tracking performance of least squares MIMO channel estimation algorithm,” *IEEE Transactions on Communications*, vol. 55, no. 11, pp. 2201–2209, 2007.

[5] Z. Du, X. Song, J. Cheng, and N. C. Beaulieu, “Maximum likelihood based channel estimation for macrocellular OFDM uplinks in dispersive time-varying channels,” *IEEE Transactions on Wireless Communications*, vol. 10, no. 1, pp. 176–187, 2010.

[6] J. Ma, S. Zhang, H. Li, N. Zhao, and A. Nallanathan, “Iterative LMMSE individual channel estimation over relay networks with multiple antennas,” *IEEE Transactions on Vehicular Technology*, vol. 67, no. 1, pp. 423–435, 2017.

[7] Z. Jiang, S. Chen, A. F. Molisch, R. Vannithamby, S. Zhou, and Z. Niu, “Exploiting wireless channel state information structures beyond linear correlations: A deep learning approach,” *IEEE Communications Magazine*, vol. 57, no. 3, pp. 28–34, 2019.

[8] Ericsson, Ericsson mobility report, 2017, available: https://www.ericsson.com/assets/local/mobility-report/documents/2017/ericsson-mobility-report-june-2017.pdf.

[9] W. Jiang and H. D. Schotten, “Deep learning for fading channel prediction,” *IEEE Open Journal of the Communications Society*, vol. 1, pp. 320–332, 2020.

[10] 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.

[11] E. Balevi and J. G. Andrews, “Deep learning-based channel estimation for high-dimensional signals,” *arXiv preprint arXiv:1904.09346*, 2019.

[12] D. Erdogmus, D. Rende, J. C. Principe, and T. F. Wong, “Nonlinear channel equalization using multilayer perceptrons with information-theoretic criterion,” in *Neural Networks for Signal Processing XI: Proceedings of the 2001 IEEE Signal Processing Society Workshop (IEEE Cat. No. 01TH8584)*. IEEE, 2001, pp. 443–451.

[13] Y. Zhang, J. Wang, J. Sun, B. Adebisi, H. Gacanin, G. Gui, and F. Adachi, “CV-3DCNN: Complex-valued deep learning for CSI prediction in FDD massive MIMO systems,” *IEEE Wireless Communications Letters*, vol. 10, no. 2, pp. 266–270, 2020.

[14] H. Khan, M. M. Butt, S. Samarakoon, P. Sehier, and M. Bennis, “Deep learning assisted CSI estimation for joint URLLC and eMBB resource allocation,” in *2020 IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE, 2020, pp. 1–6.

[15] A. R. Godala, S. Kadambar, A. K. R. Chavva, and V. S. Tijoriwala, “A deep learning based approach for 5G NR CSI estimation,” in *2020 IEEE 3rd 5G World Forum (5GWF)*. IEEE, 2020, pp. 59–62.

[16] J. Zyren and W. McCoy, “Overview of the 3GPP long term evolution physical layer,” *Freescale Semiconductor, Inc., white paper*, vol. 7, pp. 2–22, 2007.

[17] “E-UTRA, base station (BS) radio transmission and reception,” 3GPP, May 2008.

[18] “Evolved universal terrestrial radio access E-UTRA; base station BS conformance testing,” 3GPP TS 36.141, July 2018.

[19] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in *International conference on machine learning*. PMLR, 2015, pp. 448–456.

[20] H.-C. Tsai, C.-J. Chiu, P.-H. Tseng, and K.-T. Feng, “Refined autoencoder-based CSI hidden feature extraction for indoor spot localization,” in *2018 IEEE 88th vehicular technology conference (VTC-Fall)*. IEEE, 2018, pp. 1–5.

[21] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2017, preprint on webpage at https://arxiv.org/abs/1412.6980.

[22] E. Ahmed, A. Saint, A. E. R. Shabayek, K. Cherenkova, R. Das, G. Gusev, D. Aouada, and B. Ottersten, “A survey on deep learning advances on different 3D data representations,” *arXiv preprint arXiv:1808.01462*, 2018.

[23] S. Santurkar, D. Tsipras, A. Ilyas, and A. Madry, “How does batch normalization help optimization?” *Advances in neural information processing systems*, vol. 31, 2018.

Andrew Oosthuizen received his B.Eng Computer and Electronic Engineering degree in 2020 and is currently a second year M.Eng student at the North-West University in South Africa. He is working as a dual member of the TeleNet research group in the Telkom CoE, and of MUST, a CAIR-affiliated research group at NWU specialising in deep learning.