Deep-learning Based Approach for Orthogonal Frequency-Division Multiplexing System

Philip Lee Hann Yung
School of Electrical and Electronic Engineering

Assoc Prof Teh Kah Chan
School of Electrical and Electronic Engineering

Assoc Prof A S Madhukumar

Abstract - Cyclic prefix (CP) is inserted as a guard band to protect against Inter-symbol interference (ISI) for orthogonal frequency-division multiplexing (OFDM) systems. However, adding CP consumes additional power and most importantly reduces the overall data rate in block pilot schemes. Current channel estimation methods are designed with CP in mind, hence removing CP will make data recovery more challenging. As deep learning (DL) can learn important features from data by itself, it is increasingly rising in popularity to being applied in many fields. Hence, it is interesting to investigate whether a DL approach is able to recover data without CP effectively. Simulation results show that the proposed DL pipeline performs data recovery with a lower bit-error rate (BER) with or without CP removal. Furthermore, there is evidence of robustness as data recovery by the same DL method does not suffer as much as conventional methods when CP is suddenly removed on a CP-trained DL model. Second-order channel statistics also do not need to be known by the DL approach due to the incorporation of training set diversity that has accounted for the various channel lengths.

Keywords - deep learning, neural networks, channel estimation, cyclic prefix, Rician channel, second-order channel statistics

1. INTRODUCTION

Orthogonal frequency-division multiplexing (OFDM) is a popular multicarrier modulation scheme [1] that is used as a basis in modern wireless local area networks (WLAN) and the fourth-generation long-term evolution (4G LTE) standard use in wireless broadband communication [2]. One of the features of OFDM is the duplication of a fixed length of symbols taken from the end of the OFDM time block and appending it at the start. This addition is called cyclic prefix (CP) and is shown in Fig. 1 [3].

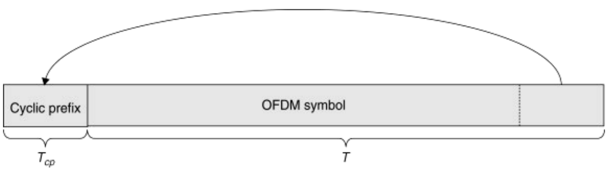

Figure 1: Addition of CP to OFDM symbols

Due to the addition of CP, OFDM is a highly effective in combating multipath fading. Inserting CP adds a guard band that protects against inter-symbol interference (ISI) [4, p. 24]. However, due to the overhead, there is a reduction in the overall data rate, and an increase in system power consumption. Removing CP directly while still using conventional channel estimation methods is not effective as these methods are designed with the presence of CP.

In this paper, a deep learning (DL) pipeline is introduced to reap the benefits of the higher data rate for OFDM signals without CP, while having a lower bit-error rate (BER) compared to existing channel estimation methods.

2. RELATED WORKS

DL approaches have been extensively used in computer vision and natural language processing. Recently, it has also entered the wireless communication field and been applied to different scenarios. This paper proposes a different approach compared to V.-H. Nguyen *et al.* [5], which predicted ISI-cancelled signals.

On the other hand, Hao Ye *et al.* [6] developed a DL architecture using TensorFlow 1.0 and Python [7] which only predicted 16 bits out of the total 128 bits as an initial effort to showcase deep neural networks (DNNs) as a possible end-to-end pipeline. A simpler implementation of the same paper [8] was used as reference during the simulation steps in this paper.

3. METHODOLOGY

Monte Carlo simulation experiments were conducted in Python and MATLAB to implement both the DL pipeline and the conventional OFDM system pipeline.

The highlight of this paper is to design a new DL pipeline that can perform channel estimation and simultaneously recover data from OFDM signals. Hence, it is assumed that the OFDM signals used in simulation have no synchronisation issues. Static conditions are also assumed so that Doppler shift effects are not present.

3.1 OFDM SYSTEM DESCRIPTION

The OFDM system parameters and system architecture used in this paper are detailed in Table 1 and Fig. 2, respectively.

Table 1: OFDM system parameters

Parameters	Specifications
Number of subcarriers	64
FFT size	64
CP length	16
Channel	Rician fading channel
Modulation type	QPSK
Channel estimation	LS or LMMSE or DL
Power delay profile	Exponential

Both pilot and data blocks are passed through a Rician fading channel, with additive white Gaussian noise (AWGN) being added at the receiver end due to thermal noise. The magnitude of AWGN is determined by the signal-to-noise ratio (SNR) which will be varied throughout the different simulation experiments.

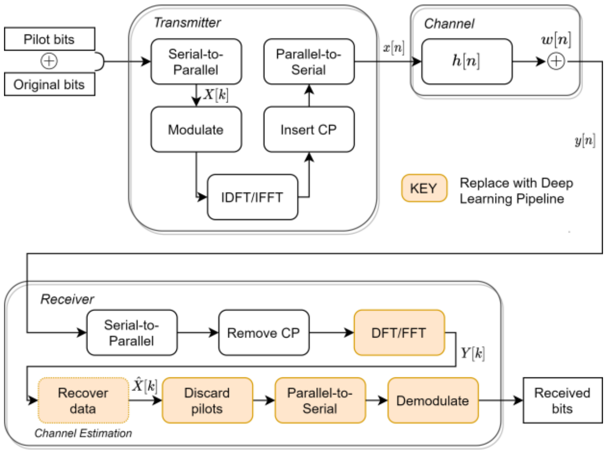

Figure 2: OFDM system architecture

3.2 DEEP LEARNING PIPELINE

DL is increasingly becoming popular in the applications of many fields. Compared to traditional machine learning (ML) techniques, DNNs are advantageous due to the lack of need to handcraft features that may be important to the output of the model [9]. This results in the deployment of DL pipelines without hard-coded domain knowledge.

As seen in Fig. 2, the DL pipeline seeks to replace the blocks in orange. Special care is taken during the generation of the train set and pre-processing of inputs to ensure the generalizability of the DL pipeline.

The DL pipeline is compared against the Least Squares (LS) and Linear Minimum Mean-Squared-Error (LMMSE) channel estimation methods.

3.2.1 PRE-PROCESSING OF INPUT DATA

After CP is removed, the pilot and data block consist of 64 time-domain OFDM complex symbols each. These complex symbols are separated into its real and imaginary parts as shown in Fig. 3, and finally concatenated to form a final input of 1D array with length 256.

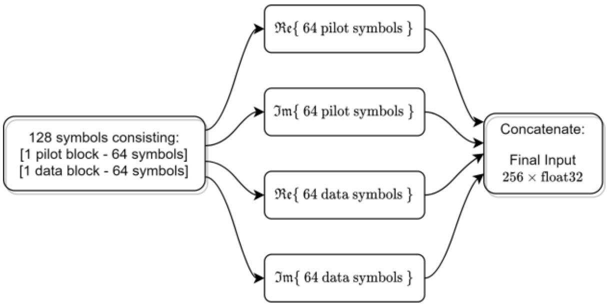

Figure 3: Pre-processing of input data

3.2.2 TRAIN SET

The train set is generated in real-time during simulation with the same channel statistics using a Rician fading channel.

To incorporate training set diversity, the Rician *K*-factor was varied from -40 to 20 dB, while the number of channel taps was varied from a length 3 to 10. This enables the DL pipeline to be able to generalise across these multiple scenarios of Rician fading, which is more applicable.

3.2.3 MODEL ARCHITECTURE

The inputs are fed into fully connected (FC) layers. Relu and sigmoid functions defined in equation (1) are used as activation functions to introduce non-linearity into the neural network.

$$f_{\text{relu}}(a) = \max(0, a), \quad f_{\text{sigmoid}}(a) = \frac{1}{1 + e^{-a}} \qquad (1)$$

The hidden FC layers uses the Relu function while the final FC layer uses the sigmoid function to predict the class probabilities of 0 or 1. Rounding is finally done to form the direct output bits.

The specifics of the two DL model layers are discussed in Sections 4 and 5.

3.2.4 MODEL TRAINING

The loss function used was binary cross-entropy as the DL model is only predicting binary classes – 1 or 0 representing the output bits. A custom BER metric was also used to track of the model performance during training and validation. Adam [10] was used as the optimizer as it has been found to converge the loss to a local minimum the fastest most of the time. A decaying learning rate scheduler was used alongside in Fig. 4 to prevent convergence instability during gradient descent.

3.3 PLOTS

Plots of BER against SNR (in dB) were generated for each simulation experiment. Three BER curves are plotted with Rician *K*-factors of -40 dB, 10 dB and 20 dB to test against various line of sight conditions.

4. INITIAL SIMULATION RESULTS

Fig. 4 shows a slightly modified DL model from the implementation in [8].

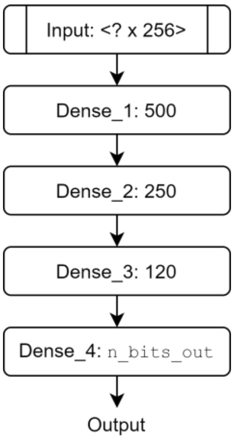

Figure 4: DL model

The *n_bits_out* can be modified as an initial proof of concept to attain a reasonable BER. We will use a SNR of 20 dB in the train set and validation set for reasons detailed in section 5.1, which were also carried out using DL model 1.

4.1 IMPACT OF *n_bits_out*

We first start by simulating OFDM signals with CP and varying the size of *n_bits_out*.

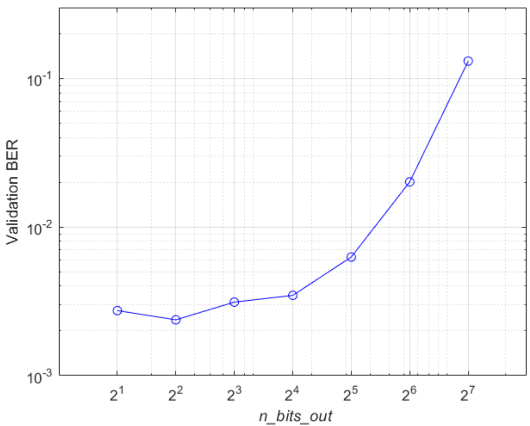

Figure 5: Validation BER vs *n_bits_out*

The results in Fig. 5 indicate that the BER can be reduced at the expense of reducing *n_bits_out*, which is directly related to the overall data rate.

Fig. 6 also shows that the numbers of epochs required to train the DL model till saturation increases as *n_bits_out* increases.

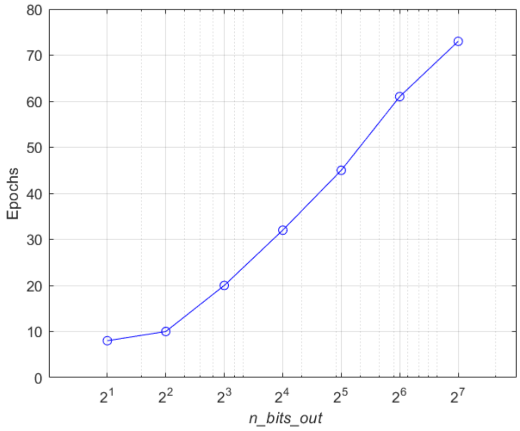

Figure 6: Epochs vs *n_bits_out*

Since our primary concern is improving the overall data rate, we will use *n_bits_out* = 64 for this section as an initial trade-off so that BER does not suffer too badly.

4.2 OFDM WITH CP

The DL pipeline is first compared with LS and LMMSE for OFDM waveforms with CP across Rician *K*-factors of -40 dB, 10 dB and 20 dB. As discussed in Section 3.2.2, the same model is used for both 3 and 10 channel taps.

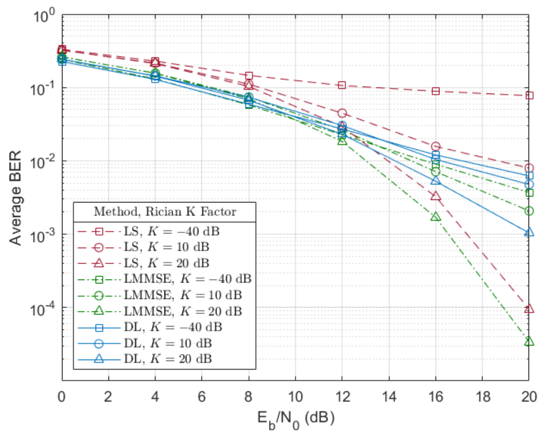

Figure 7: OFDM with CP, 3 channel taps

In Fig. 7 and Fig. 8, the DL method only performs better than the LS method for some scenarios, but consistently performs worse than the LMMSE method.

As train set diversity is accounted for earlier, the DL method does not need the knowledge of second order channel statistics, unlike the LMMSE method which used a channel tap estimate of length 3 and 6 to achieve the lowest BER for Fig. 7 and Fig. 8, respectively.

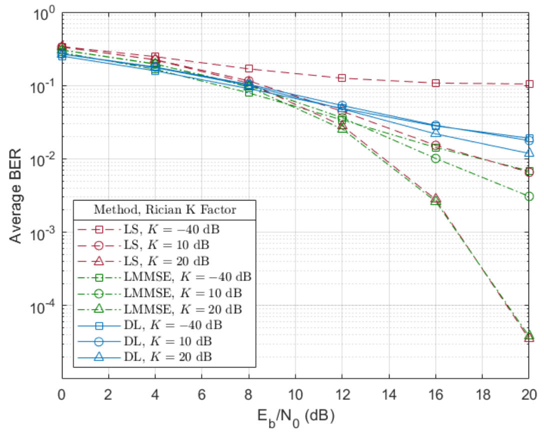

Figure 8: OFDM with CP, 10 channel taps

We will hence be using only 10 channel taps in subsequent simulation experiments as channel estimation is harder for a higher number of paths, as evident in Fig. 7 and Fig. 8. A channel tap estimate of length 6 will be used correspondingly for the LMMSE method.

4.3 OFDM WITHOUT CP

As indicated as our original motivation, we will now try to evaluate the DL pipeline performance after CP removal is performed. The DL model is also trained on OFDM without CP to prevent a mismatch between training and test sets.

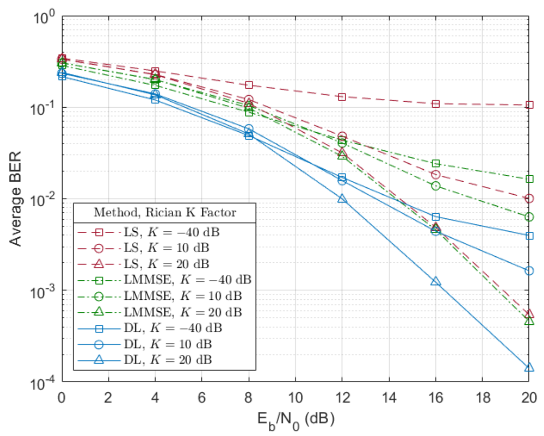

Figure 9: OFDM without CP, 10 channel taps

Fig. 9 shows the DL pipeline having the lowest BER across all scenarios. This shows that the DL method works well.

4.4 CONFLICT WITH MOTIVATION

The DL pipeline so far performs data recovery better from OFDM waveforms without CP than conventional channel estimation methods. However, it does not achieve our original goal of increasing the overall data rate since *n_bits_out* = 64 effectively caps our data rate to 50%. Hence, it is evident that a better DL pipeline is needed.

5. MODIFIED ARCHITECTURE

One important note is that the DL pipeline is not performing channel estimation explicitly but instead learning the complicated characteristics of the wireless channels and recovering data in an end-to-end manner.

It has been observed that there is an inverse relationship between the validation BER and *n_bits_out* seen in Fig. 5. Hence, the DL model was modified to Fig. 10, with a total of *n* stacks of smaller deep neural networks using different model weights and parameters.

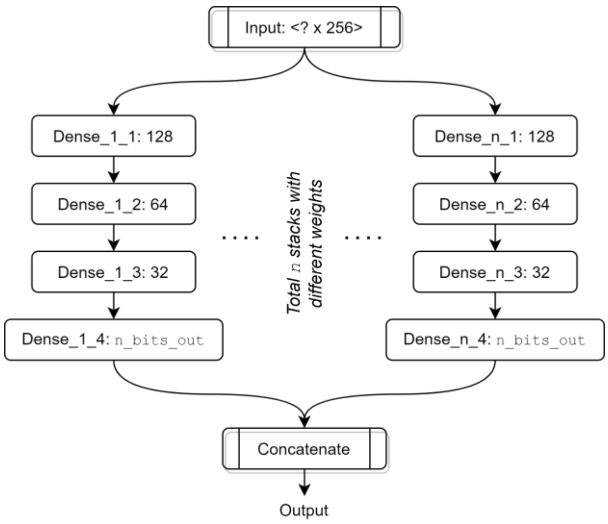

Figure 10: Modified DL model

Each stack predicts *n* bits. The outputs of each stack are then concatenated to form a final output which has a total of 128 or all the bits. *n* = 16 is chosen for this paper as a trade-off balance between the number of model parameters, training time and BER performance.

The change in model architecture also brought in an unforeseen benefit – it converges in 20 epochs compared to 60 epochs earlier. This is similar to the number of epochs needed for convergence in Fig. 6. As the stacks can be backpropagated in parallel, training on a GPU environment would be beneficial. As an example, training on the Google Collaboratory environment took at most 10 minutes, which is comparatively faster than the 30 minutes taken earlier.

As mentioned in Section 4.2, we will be using 10 channels taps for all the simulation experiments in this section.

5.1 IMPACT OF TRAIN SET NOISE

In this experiment, the training set SNR was varied to see how it would impact the BER during model evaluation. The test set channel statistic follows a Rician *K*-factor of 20 dB and 10 channel taps.

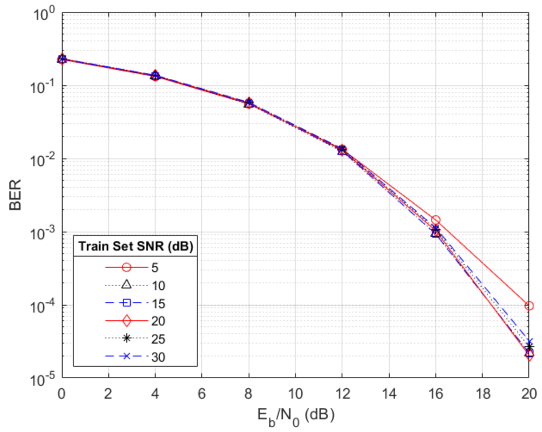

Figure 11: BER with varying noise on train set

The results in Fig. 11 show that the SNR in the training set does not really affect the BER, unless the noise is too large as in the case of 5 dB. Hence, we will use a training set SNR of 20 dB in subsequent simulation experiments to prevent overfitting of the DL model.

5.2 GENERALIZABILITY OF PIPELINE

The DL model is trained on OFDM signals with CP with the new architecture – we call this model DL1.

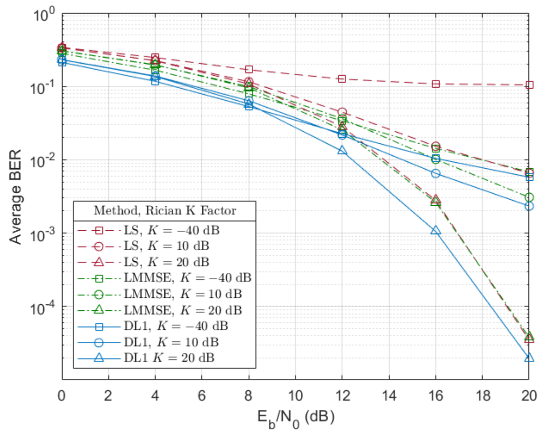

Figure 12: OFDM with CP, 10 channel taps

As seen in Fig. 12, our DL pipeline now yields the lowest BER compared to conventional channel estimation methods. We managed to preserve the BER range of *n_bits_out* = 8 in Fig. 5 while keeping data rate to the maximum. It is important to note that the DL method does not need require any channel statistics to recover the bits.

5.3 CP REMOVAL AND ROBUSTNESS

It is now known that the DL method yields the lowest BER. We now train the DL model on OFDM signals without CP – we call this model DL2. We then test this model to predict bits for OFDM signals with CP removed. As seen in Fig. 13, the BER range for each SNR remains in the same range as Fig. 12. This demonstrates that the DL pipeline can perform well if the training and test set have similar distributions – in this case both OFDM signals have CP removed.

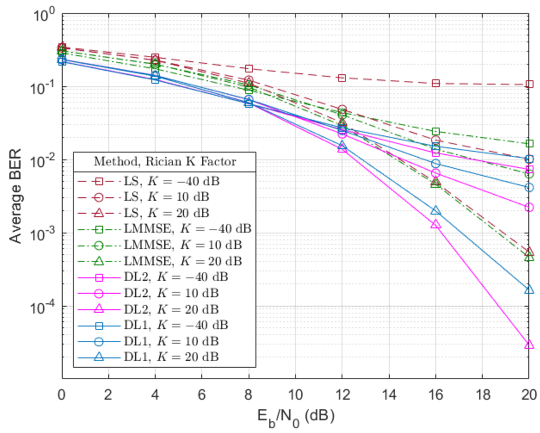

Figure 13: OFDM without CP, 10 channel taps

Consider now the case where the system assumes that the OFDM signals have CP, so we use model DL1. However, the actual test signal has the CP removed. It can be seen in Fig. 13 that the BER curves show robustness as the curve for DL1 method shifts up the least compared to LS and LMMSE. This highlights that the relationship learned by the DL method is complex and evidently robust by the change in train and test set distributions. This simulation is analogous to conventional methods having the assumption that OFDM signals have CP appended by default, but in actuality, CP has been removed.

5.4 IMPACT OF CLIPPING

There is one notable drawback of OFDM which is its high peak-to-average power ratio (PAPR) as indicated in [11], which decreases the efficiency of the power amplifier at the transmitter side.

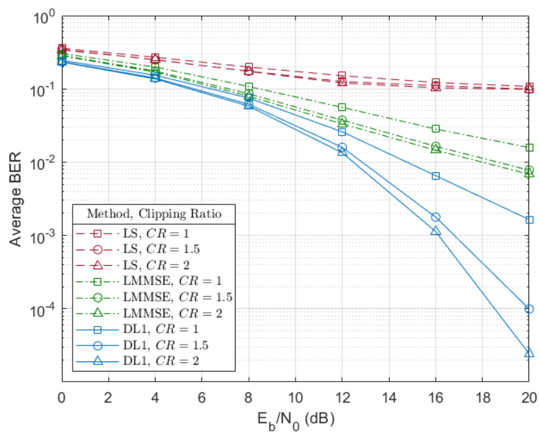

Figure 14: OFDM with CP and clipping, 10 channel taps, and K = -40 dB

As in [6], we investigate the impact of clipping to reduce PAPR on our method by experimenting on a Rician *K*-factor of -40 dB for the various methods. The time-domain signal is clipped as in equation (2):

$$\hat{x}(n) = \begin{cases} x(n) & \text{if } |x(n)| \leq A \\ Ae^{j\phi(n)} & \text{otherwise} \end{cases}, \text{ where } A = \text{CR} \cdot \sigma \quad (2)$$

where *A* is the threshold value determined by CR and σ which are the clipping ratio and root mean square value of the signal respectively, and *ϕ(n)* is the phase of *x(n)*.

Fig. 14 shows that the DL method is also robust to clipping effects and has a better performance than conventional channel estimation techniques.

5.5 EVALUATION TIME

With all the advantages that the DL pipeline brings, we now investigate whether there is a significant contribution in the evaluation time. The simulations were carried out using the *%%timeit* magic command in Google Collaboratory with no hardware accelerators.

Table 2: Evaluation time of various methods

Method	Evaluation time
LS	578 µs
LMMSE	32.8 ms
DL	43.6 ms
DL (tflite)	134 µs

The DL method has a comparable evaluation time to the LMMSE method as seen in Table 2. However, this evaluation time can be further improved by optimizing the model.

5.5.1 TENSORFLOW LITE

TensorFlow Lite is optimized for on-device inference such as mobile and embedded devices which have limited compute resources [12]. Many solutions like image classification and object detection have been successfully deployed using this framework.

One of the techniques used to reduce inference time is quantization in which the precision of the floating-point model weights is reduced, with little degradation in model accuracy. For our case, casting was done from float64 to float32 and other optimizations were done by the TensorFlow Lite library – we call this model DL3. We expect that the performance of DL3 will not change much relative to DL1.

As seen in Table 2, the optimized DL3 model has the lowest evaluation time. The performance of the model also does not deviate much from DL1 as seen in Fig. 15.

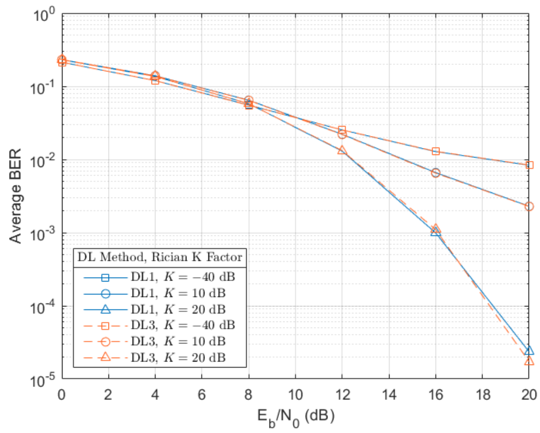

Figure 15: OFDM with CP, 10 channel taps

However, it is important to note that real-world implementations of conventional channel estimation techniques have been optimized [13] and the evaluation times seen in Table 2 do not reflect a fair comparison, but rather highlights that the DL pipeline can be optimized to further improve the evaluation time.

6. CONCLUSION

The proposed DL pipeline offers generalizability and robustness in various channel conditions, while yielding the lowest BER. The non-linear activation functions allow the DNN to learn the complex relationship between raw OFDM signals and data bits which does not follow the conventional theoretical analysis. However, the pipeline uses the block pilot scheme which might not be suitable for certain channel conditions.

For real world implementations, it is important that the channel used to train the DL model has the same distribution as the actual channel for the best performance of the DL pipeline. Faster implementations may be looked into using specialized ML-optimized chips and improved further as research progresses like in [14].

More theoretical work is needed perhaps to justify the workings of the pipeline, especially when FFT is also replaced in the DL pipeline.

ACKNOWLEDGMENT

I would like to express my appreciation and gratitude to my supervisor, Assoc Prof Teh Kah Chan, for giving me the opportunity to work with him on this project. His constant mentorship and dedication to our weekly updates helped to ensure that I am always on the right track with regards to the project. His insightful feedback pushed me to sharpen my thinking and methodologies, bringing my work to a higher potential that it was before.

I would also like to acknowledge [7], [8], [15]–[19], [20, pp. 377–395] for providing their source codes as a reference during the initial development phase.

Special thanks is given to Andrew Ng from deeplearning.ai [21], Prof. Aditya K. Jagannatham from the Electrical Engineering department at IIT Kanpur [22], and Prof. Iain Collings from the School of Engineering at Macquarie University [23] for their learning resources which helped in my understanding of wireless communication and deep learning concepts.

I would also like to acknowledge the funding support from Nanyang Technological University – URECA Undergraduate Research Programme for this research project.

REFERENCES

[1] "What is OFDM: Orthogonal Frequency Division Multiplexing » Electronics Notes." https://www.electronics-notes.com/articles/radio/multicarrier-modulation/ofdm-orthogonal-frequency-division-multiplexing-what-is-tutorial-basics.php (accessed Jun. 03, 2021).

[2] "lteencyclopedia." https://sites.google.com/site/lteencyclopedia/home (accessed Jun. 03, 2021).

[3] T. J. Rouphael, "Chapter 3 - Common Digital Modulation Methods," in *RF and Digital Signal Processing for Software-Defined Radio*, T. J. Rouphael, Ed. Burlington: Newnes, 2009, pp. 25–85. doi: 10.1016/B978-0-7506-8210-7.00003-5.

[4] Y. Li and G. L. Stüber, Eds., *Orthogonal frequency division multiplexing for wireless communications*. New York: Springer Science+Business Media, 2006.

[5] V.-H. Nguyen, M.-T. Nguyen, and Y.-H. Kim, "Deep neural network for orthogonal frequency division multiplexing systems without cyclic prefix transmission," *MATEC Web Conf.*, vol. 189, p. 04016, 2018, doi: 10.1051/matecconf/201818904016.

[6] H. Ye, G. Y. Li, and B.-H. Juang, "Power of Deep Learning for Channel Estimation and Signal Detection in OFDM Systems," *IEEE Wireless Communications Letters*, vol. 7, no. 1, pp. 114–117, Feb. 2018, doi: 10.1109/LWC.2017.2757490.

[7] haoyye, *haoyye/OFDM_DNN*. 2021. Accessed: Jan. 24, 2021. [Online]. Available: https://github.com/haoyye/OFDM_DNN

[8] L. Tian, *TianLin0509/DNN_detection_via_keras*. 2021. Accessed: Jan. 25, 2021. [Online]. Available: https://github.com/TianLin0509/DNN_detection_via_keras

[9] H. Liang, X. Sun, Y. Sun, and Y. Gao, "Text feature extraction based on deep learning: a review," *EURASIP Journal on Wireless Communications and Networking*, vol. 2017, no. 1, p. 211, Dec. 2017, doi: 10.1186/s13638-017-0993-1.

[10] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv:1412.6980 [cs]*, Jan. 2017, Accessed: Jun. 04, 2021. [Online]. Available: http://arxiv.org/abs/1412.6980

[11] X. Li and L. J. Cimini, "Effects of clipping and filtering on the performance of OFDM," *IEEE Communications Letters*, vol. 2, no. 5, pp. 131–133, May 1998, doi: 10.1109/4234.673657.

[12] "TensorFlow Lite | ML for Mobile and Edge Devices," *TensorFlow*. https://www.tensorflow.org/lite (accessed Jun. 15, 2021).

[13] J. Ketonen, M. Juntti, J. Ylioinas, and J. R. Cavallaro, "Implementation of LS, MMSE and SAGE channel estimators for mobile MIMO-OFDM," in *2012 Conference Record of the Forty Sixth Asilomar Conference on Signals, Systems and Computers (ASILOMAR)*, Pacific Grove, CA, USA, Nov. 2012, pp. 1092–1096. doi: 10.1109/ACSSC.2012.6489188.

[14] S. Oh *et al.*, "Energy-efficient Mott activation neuron for full-hardware implementation of neural networks," *Nat. Nanotechnol.*, pp. 1–8, Mar. 2021, doi: 10.1038/s41565-021-00874-8.

[15] "OFDM Channel Estimation in MATLAB." https://www.mathworks.com/matlabcentral/fileexchange/67156-ofdm-channel-estimation-in-matlab (accessed Jun. 10, 2021).

[16] "LS & MMSE channel estimators for OFDM." https://www.mathworks.com/matlabcentral/fileexchange/11033-ls-mmse-channel-estimators-for-ofdm (accessed Nov. 02, 2020).

[17] "Python OFDM Example - DSPIllustrations.com." https://dspillustrations.com/pages/posts/misc/python-ofdm-example.html (accessed Jan. 21, 2021).

[18] V. K. Veludandi, *vineel49/lmmse*. 2021. Accessed: Dec. 11, 2020. [Online]. Available: https://github.com/vineel49/lmmse

[19] Y. S. Cho, Ed., *MIMO-OFDM wireless communications with MATLAB*. Piscataway, NJ: IEEE Press, 2010.

[20] J. G. Proakis, M. Salehi, and G. Bauch, *Contemporary communication systems using MATLAB*, 3rd ed. Stamford, CT: Cengage Learning, 2013.

[21] "Deep Learning," *Coursera*. https://www.coursera.org/specializations/deep-learning (accessed Jun. 04, 2021).
[22] "Bayesian/ MMSE Estimation for Wireless Communications -MIMO/ OFDM - YouTube." https://www.youtube.com/playlist?list=PLvb0jK8jJn_Zv6tnuHyWdxhohB3NzhG6N (accessed Nov. 02, 2020).
[23] "Iain." http://www.iaincollings.com/ (accessed Mar. 04, 2021).