Exploring CNN-Based Automatic Modulation Classification Using Small Modulation Sets

Andrew Oosthuizen*†, Marelie H. Davel*†, Albert Helberg*
*School of Electrical, Electronic and Computer Engineering, North-West University
†Centre for AI Research (CAIR), South Africa
aj.oosthuizen.ao@gmail.com
marelie.davel@nwu.ac.za
albert.helberg@nwu.ac.za

***Abstract*—We investigate the effect of a reduced modulation scheme pool on a CNN-based automatic modulation classifier. Similar classifiers in literature are typically used to classify sets of five or more different modulation types [1] [2], whereas our analysis is of a CNN classifier that classifies between two modulation types, 16-QAM and 8-PSK, only. While implementing the network, we observe that the network's classification accuracy improves for lower SNR instead of reducing as expected. This analysis exposes characteristics of such classifiers that can be used to improve CNN classifiers on larger sets of modulation types. We show that presenting the SNR data as an extra data point to the network can significantly increase classification accuracy.**

***Index Terms*—Automatic Modulation Classification, In-phase and Quadrature-phase (I/Q) symbols, Deep learning**

I. Introduction

In this paper we investigate a deep learning based approach to automatic modulation classification (AMC). AMC is used in the telecommunications field to identify transmission modulation schemes without this information being explicitly communicated between transmitters and receivers. AMC reduces overhead in communication and allows for effective switching between modulation schemes in cognitive radio applications. In the past, AMC has been implemented with statistical [3] and machine learning methods, such as clustering [4] and support vector machines [5]. In recent years deep learning architectures such as multilayer perceptrons (MLPs) and convolutional neural networks (CNNs) have been applied to the problem and have shown better performance over the more traditional approaches with regard to both accuracy and speed [6].

This paper investigates classification behaviour of deep neural networks on modulation types under additive white Gaussian noise (AWGN), by classifying between two modulation schemes, both varying in type and order. By using a reduced modulation pool for classification, we are able to better understand how a CNN interacts with an AMC task.

II. Related work

There exist several methods to approach AMC using deep learning models [7]. Most approaches supply some constellation data obtained from raw signal data to a neural network. How the constellation data is presented to the neural network does, however, vary depending on the method. Popular methods include presenting the quadrature and in-phase data points of constellation diagrams in a $2 \times N$ array, where N is the number of data points [2], or presenting the constellation plots as images [8]. The last method often contains several stages of feature extraction and pre-processing before the data is presented to the network.

CNNs are typically used for AMC problems [7] and function by presenting the input data to convolutional layers. The convolutional layers extract features from the data by making use of filters, also known as kernels. After feature extraction, the convolutional layers are flattened and passed to dense layers that make use of the previously extracted features to perform classification [9], [10].

For our study we use a CNN structure, similar to the network used by Yongshi et al. [2], that receives constellation data points rather than images as inputs. The reason for this is to simplify the investigation of the neural network, since pre-processing and presentation of image data to CNNs add extra levels of analysis to the process. We make use of 16th order quadrature amplitude modulation (16-QAM) and 8th order phase-shift keying (8-PSK) modulation schemes as input, as both the order and method of modulation differ between the two.

III. Experimental setup

A. Data

The data is presented as complex values of the signal constellation in the I/Q plane generated using Matlab version 2020b. A random bit stream source is modulated in baseband using one of two modulation types (8-PSK or 16-QAM). The modulated data is sent over an additive white Gaussian noise (AWGN) channel with varying normalised signal-to-noise (SNR) ratios (E_b/N_o) with an average signal power of 1W over 1Ω. The complex valued channel symbols are then grouped into samples containing 1024 constellation points of each symbol's in-phase and quadrature component. Thus, each sample consists of 1024 32-bit real and 1024 32-bit imaginary data points to create a 2×1024 sized data set.

The SNR, E_b/N_o, is discretely stepped over the range of -15 dB to 5 dB in 1 dB increments to create training, validation and evaluation sets respectively. The number of generated data samples per E_b/N_o is listed in Table I.

TABLE I
NUMBER OF DATA SAMPLES GENERATED PER E_B/N_O OF A MODULATION SCHEME, AS WELL AS THE TOTAL SET SIZE OVER 21 SNR RANGE.

Modulation	Train	Validation	Test
8-PSK	1 000	500	1 000
16-QAM	1 000	500	1 000
Total set size	42 000	21 000	42 000

The training and validation sets are used in the training process as described below, while the evaluation sets are kept separate to evaluate the performance per E_b/N_o level.

B. Baseline architecture

The classifier architecture is based on that of Yongshi et al. [2], but with fewer nodes in the hidden dense layer. The hidden dense layer is reduced to 100 nodes, as the number of modulation types to classify has been reduced. This change is made to improve the training time and throughput of the network. The network consists of two convolutional layers that are ReLu-activated [11], makes use of batch normalisation, has no padding and a stride of 1. A max pooling layer, with a stride of 2, is placed between the convolutional layers to reduce the complexity of the network. After the convolutional layers a linear layer with 100 nodes is placed, followed by the classification layer (also a linear layer) with 2 outputs [2].

C. Training protocol

The same training protocol is followed for all networks. Networks are trained with the Adam [12] optimiser using a cross-entropy loss function. Adam is selected for its ability to adapt the learning rate of different parameters, and cross-entropy loss is used for its good performance in classification problems [9], [10]. Since ReLU activation functions are used, the weights of the network are initialised using a uniform Kaiming initialisation [13].

The following hyperparameters are optimised: learning rate, batch size, and weight decay (L2 penalty). We selected these hyperparameters, as preliminary tests showed that they have noticeable effects on the model's performance. Hyperparameter tuning is performed using grid searches on the predefined CNN architecture, by comparing the networks' results on the validation data set. The grid search is performed over different learning rates $\{0.01, 0.001, 0.0001\}$, batch sizes $\{32, 512\}$, weight decay values $\{0, 0.001, 0.01\}$ and 3 random initialisation seeds. The initialisation seeds are used to ensure a particularly strong or weak network initialisation does not affect the results. We also make use of a grid search over the architecture by varying the convolution kernel width $\{4, 16, 32, 64\}$ and amount of dropout $\{0, 0.5\}$ hyperparameters [14].

To ensure the network trains until it convergences, the network is trained for a minimum of 50 epochs, after which the training is terminated when no improvements in validation accuracy is found in the last 20% of epochs. Early stopping is then implemented by selecting the epoch at which the model achieved its highest classification accuracy on the validation set.

IV. ANALYSIS AND RESULTS

A. Classification performance

The goal of this experiment is to analyse the behaviour of a CNN classifier on two modulation schemes of different types and orders that exposes fundamental characteristic when using a data point driven constellation diagram input.

From the classification accuracy and average class recall of the baseline architecture network in Figure 1, we can see that the model classifies well for SNRs above 0 dB. At lower SNRs the accuracy decreases as the signal falls below the noise floor. We also see an increase in accuracy when the number of kernels is increased to 36 kernels. Varying other hyperparameters revealed that adding dropout and using a weight decay value of 0.01 also increases accuracy and that the model generalises better on the validation set. The final hyperparameters for the baseline architecture can be found in Table II.

TABLE II
HYPERPARAMETERS OPTIMISED FOR THE BASELINE ARCHITECTURE.

Parameter	Value
Learning rate	0.0001
Batch size	32
Dropout	0.5
Weight decay	0.01
Kernel width	36

An interesting observation to make from Figure 1 is the unexpected improvement of the declining 8-PSK classification accuracy at very low SNR. The 16-QAM classification also increases slightly, but not as much as 8-PSK. This observation is strange, as we usually expect modulation classifiers to show reduced classification ability as noise increases until a reliable classification can no longer be made and the network shows 50% classification accuracy.

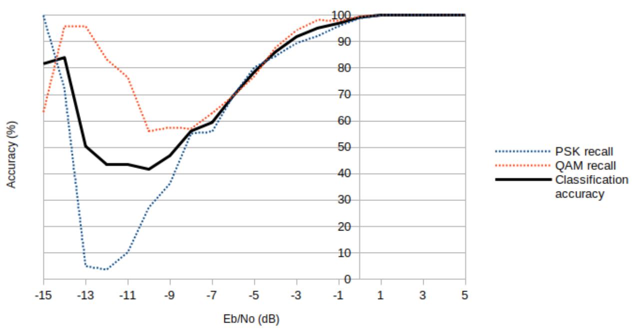

Fig. 1. Average classification accuracy (over 3 seeds) of the baseline architecture using optimised hyperparameters, for the evaluation data set with range of -15 dB to 5 dB. The average recall of 16-QAM and 8-PSK, respectively, is also shown.

B. Analysis of low SNR artefact

In order to find out why this increase in accuracy at lower E_b/N_o exists, we investigate whether this effect is due to the boundaries of the range of SNRs evaluated. Using the same training hyperparameters, the network was retrained for two ranges of the SNR, namely [-20;0] and [-10;10]. This investigation is also used to establish if the improvement in accuracy is tied to certain SNRs, or to a E_b/N_o position in the range. In addition, we investigate the effect of architectural changes to the neural network to ensure the artefact is not caused by lack of representational ability. The neural network will be adapted in the following ways:

- Increasing the dense layer node count from 100 to 1 000
- Changing the convolution layer kernels from 36 to 512
- Adding an extra convolution layer

To ensure well-defined results, each one of these changes is applied independently from the other.

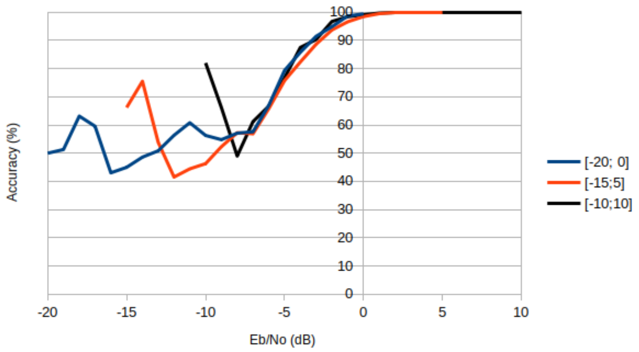

Fig. 2. Baseline network evaluation set performance when the network is trained on -20 dB to 0 dB, -15 dB to 5 dB and -10 dB to 10 dB SNR ranges, respectively.

When using the same baseline architecture as before but changing the E_b/N_o range, Figure 2 shows a similar trend to that observed before. Figure 2 indicates that the increase in accuracy occurs at lower E_b/N_o values, irrespective of the input data range. However, it is observed that the accuracy increase does not appear at a specific E_b/N_o. It should be noted that accuracy fluctuations only appear after the 0 dB accuracy descent and results near and above the noise floor increase gradually as expected.

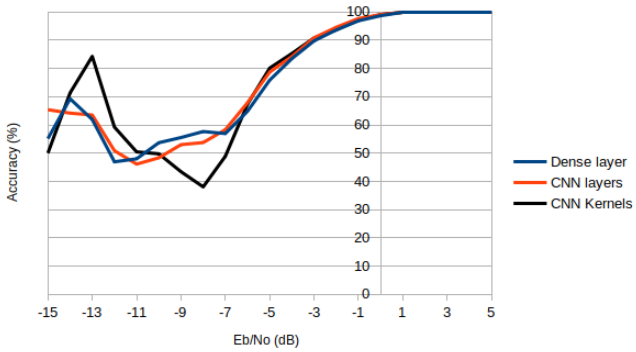

Fig. 3. Evaluation set performances when the dense layer of the baseline network is increased to 1 000 nodes ('Dense layer'), the baseline network's convolutional layers are increased from 2 to 3 ('CNN layers') and the baseline network's convolution kernels are increased to 512 ('CNN kernels').

When increasing the size and complexity of the network, Figure 3 shows a similar trend to that observed in the baseline architecture, except for variations in the average validation accuracy. Some of the methods change the shape and intensity of the accuracy increase in the E_b/N_o range, but all still exhibit the same artefact.

C. SNR-specific training

When observing Figure 1, we note that the increase in accuracy at lower E_b/N_o resembles models trained with SNR pairs selection [15]. With pairs selection, the network is only trained using two E_b/N_o data sets, instead of the entire range. In some instances this may cause an increase in accuracy surrounding the selected E_b/N_o pairs, especially in the low E_b/N_o range. To determine if this training method can give insight into the occurrence of the increase at low SNRs, we test how well the network can classify a single SNR's data by training baseline networks on only single SNR data sets. We also test the generalisation of each network over the entire SNR range to identify the classification abilities of our network on low SNR data.

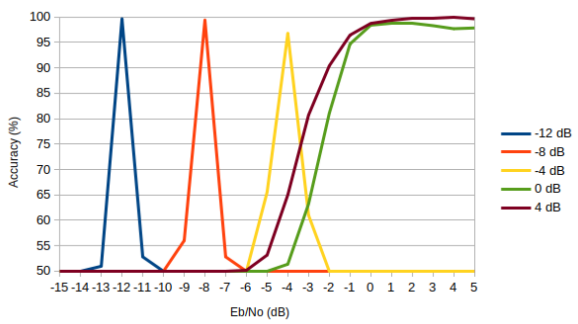

Fig. 4. The performance of five baseline networks, each trained on a specific dB value of SNR data and evaluated on the evaluation set.

From the classification accuracy of five baseline networks, in Figure 4, obtained from SNR-specific training, we see that each E_b/N_o could be classified above 90% accuracy over the -15dB to 5dB range, showing that the network can classify between the two modulation types, even when large amounts of noise is added, if the task is restricted to a narrower noise range. Furthermore, this shows that the network is indeed able to classify accurately at lower E_b/N_o levels.

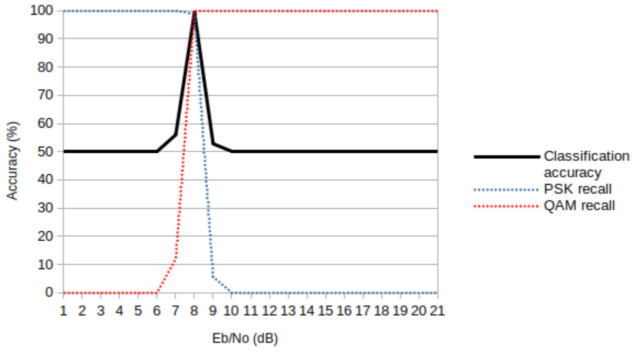

Fig. 5. Classification accuracy of a baseline network trained using only -8 dB data and evaluated on the evaluation set. The recall of 16-QAM and 8-PSK, respectively, is also shown.

From the classification accuracy and class recall of a baseline network trained on -8 dB data as shown in Figure 5, it is seen that networks trained on lower E_b/N_o data tend to generalise poorly to neighbouring E_b/N_o values and only show accuracy above 50% for one to two neighbouring E_b/N_o ranges before falling into a bias classification of 50%. Networks trained on E_b/N_ovalues above 0 dB E_b/N_o, however, show good generalisation, especially to higher E_b/N_o ranges than the E_b/N_o it is trained on. Accuracy of networks trained on high SNR data do however decrease when approaching and passing the noise floor at 0 dB E_b/N_o.

The knowledge that the network can accurately classify at any SNR level in our entire range, but then does not generalise well to other ranges, leads us to the observation that different classification criteria are being utilised for each SNR level, especially at lower SNRs. We can also see on which E_b/N_o level the network classifies accurately, not only by the increase in accuracy but also by the point where the network bias switches from 16-QAM to 8-PSK. This is also seen in our original network (Figure 1), where accuracy increases due to 16-QAM and 8-PSK classifications crossing over at lower E_b/N_o levels. These observations suggest the hypothesis that the network is prioritising certain E_b/N_o levels, or classification criteria, over others in the training process. By prioritising certain lower E_b/N_o levels the network increases the average validation accuracy.

D. Adding SNR as a feature

Knowing that the model can achieve an accuracy near 100% for any of the E_b/N_o levels within our range and the hypothesis that the network is selecting specific E_b/N_o ranges to optimise for, we turn to literature to find possible clarification of this behaviour. Several deep learning AMC networks [1] show increased accuracy when the E_b/N_o dB of the given constellation diagram is provided to the network to aid in classification. Providing the E_b/N_o level might allow the network to better optimise for the entire range, as it will not be blindly selecting an area in the E_b/N_o range to optimise for.

We provide the oracle E_b/N_o dB value to the network to test if this additional information aids the network in optimising better in the lower E_b/N_o range. The E_b/N_o value is provided to the linear layer of the network after the initial feature extraction has taken place in the convolutional layers, and prior to classification based on the extracted feature maps. By providing the E_b/N_o values, we test if the network will be able to adapt the classification criteria based on the E_b/N_o level.

From the classification accuracy of the network provided with SNR data in Figure 6, we observe a substantial increase in the validation accuracy across the entire E_b/N_o range. By providing the network with oracle E_b/N_o values, the network is able to adjust its classification criteria based on the amount of noise on the constellation data points.

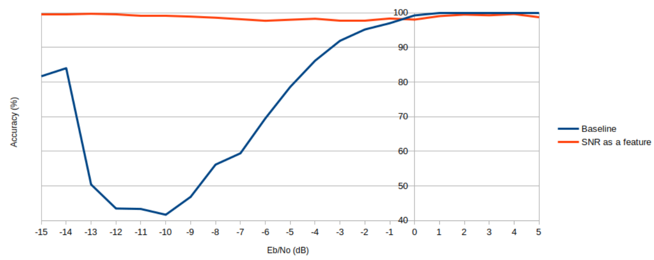

Fig. 6. Classification accuracy on the evaluation set, comparing the baseline network and a network where the SNR of the modulation scheme is presented to the network as a feature.

E. Effect of SNR estimation accuracy

By providing the oracle E_b/N_o data, we create a much-improved classifier for our binary classification problem. We do, however, know that the performance greatly relies on the provision of E_b/N_o values using SNR estimation at the receiver. This affects the implementation of this network in practical applications as noise level estimation accuracy tends to decrease at lower E_b/N_o levels [2]. To further understand the robustness and generalisation of the SNR-tagged network, we performed a sensitivity analysis.

The sensitivity analysis is conducted by generating statistical noise within a given range, as if an error was made when estimating the E_b/N_o value of the received signal, and adding that to the provided E_b/N_o data point when making a classification. The evaluation set and best performing baseline network is used for this analysis.

The results of the sensitivity analysis results for the baseline architecture is shown in Figure 7. We observe that variations within a 0.5 dB E_b/N_o range do not affect classification accuracy substantially, since all E_b/N_o still achieve above 95% accuracy. It is only after variation over 1 dB is introduced that the network's classification accuracy is noticeably reduced, especially at lower E_b/N_o levels.

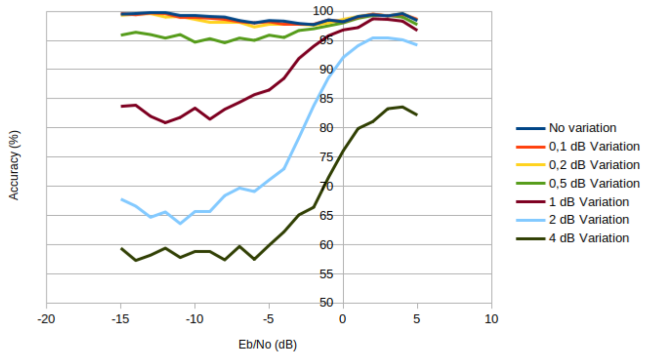

Fig. 7. A sensitivity analysis of SNR-tagged networks where the SNR input is corrupted with increased levels of stochastic variance. This mimics the effect of inaccurate SNR estimators.

This reduction in accuracy can be attributed to the network being trained with E_b/N_o step sizes of 1 dB, since as the provided SNR value moves closer to a neighbouring value, the generalisation of the classification criteria is reduced. The generalisation effect can also be observed at higher E_b/N_o levels as high accuracy levels are still achieved, even at high E_b/N_o variation. This observation once again highlights the specificity of the classification criteria needed to classify accurately at low E_b/N_o levels as opposed to above the noise floor.

V. Conclusion

This paper uncovers and investigates an artefact that occurs when implementing a modulation classifier for two modulation schemes, which provides constellation diagram data points as input to a CNN. It is found that the network can classify accurately at low SNR levels when only trained using specific E_b/N_o's data and that it generalises poorly to neighbouring E_b/N_o values. Knowing that the problem does not lie with the representational capacity of the network but rather with how the network models the task, the SNR values are provided as an additional input feature. This technique significantly increases accuracy at low E_b/N_o values as the network now has reference to the classification criteria to select when making a prediction. A sensitivity analysis of the effect when the additional input features are corrupted shows the weak generalisation of the classification criteria by highlighting the drop in performance when SNR estimation accuracy is low. This means that this network will only be of use if a SNR estimator that can accurately predict E_b/N_o at low SNR is used. Moving forward, this network and method could be further researched to improve AMC for low SNR environments.

Acknowledgement

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

References

[1] X. Xie, Y. Ni, S. Peng, and Y.-D. Yao, "Deep learning based automatic modulation classification for varying SNR environment," in *2019 28th Wireless and Optical Communications Conference (WOCC)*, 2019, pp. 1–5.

[2] W. Yongshi, G. Jie, L. Hao, L. Li, W. Zhigang, and W. Houjun, "CNN-based modulation classification in the complicated communication channel," in *2017 13th IEEE International Conference on Electronic Measurement Instruments (ICEMI)*, 2017, pp. 512–516.

[3] J. L. Xu, W. Su, and M. Zhou, "Likelihood-ratio approaches to automatic modulation classification," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 41, no. 4, pp. 455–469, 2011.

[4] J. P. Mouton, M. Ferreira, and A. S. Helberg, "A comparison of clustering algorithms for automatic modulation classification," *Expert Systems with Applications*, vol. 151, p. 113317, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417420301421

[5] Y. Wei, S. Fang, and X. Wang, "Automatic modulation classification of digital communication signals using SVM based on hybrid features, cyclostationary, and information entropy," *Entropy*, vol. 21, no. 8, 2019. [Online]. Available: https://www.mdpi.com/1099-4300/21/8/745

[6] F. Meng, P. Chen, L. Wu, and X. Wang, "Automatic modulation classification: A deep learning enabled approach," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 11, pp. 10 760–10 772, 2018.

[7] S. A. Ghunaim, Q. Nasir, and M. A. Talib, "Deep learning techniques for automatic modulation classification: A systematic literature review," in *2020 14th International Conference on Innovations in Information Technology (IIT)*, 2020, pp. 108–113.

[8] S. Peng, H. Jiang, H. Wang, H. Alwageed, Y. Zhou, M. M. Sebdani, and Y.-D. Yao, "Modulation classification based on signal constellation diagrams and deep learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 3, pp. 718–727, 2019.

[9] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[10] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, *Dive into Deep Learning*, 2020, https://d2l.ai.

[11] A. F. Agarap, "Deep learning using rectified linear units (ReLU)," 2019, preprint on webpage at https://arxiv.org/abs/1803.08375.

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

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," 2015.

[14] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," 2012, preprint on webpage at https://arxiv.org/abs/1207.0580.

[15] S. Ramjee, S. Ju, D. Yang, X. Liu, A. E. Gamal, and Y. C. Eldar, "Fast deep learning for automatic modulation classification," 2019, preprint on webpage at https://arxiv.org/abs/1901.05850.

Andrew Oosthuizen received his B.Eng Computer and Electronic Engineering degree in 2020 and is currently a first 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.