Complex-Valued Convolutions for Modulation Recognition using Deep Learning

Jakob Krzyston
School of Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, USA
jakobk@gatech.edu

Rajib Bhattacharjea
Electro-Optical Systems Laboratory
Georgia Tech Research Institute
Atlanta, USA
Rajib.Bhattacharjea@gtri.gatech.edu

Andrew Stark
Electro-Optical Systems Laboratory
Georgia Tech Research Institute
Atlanta, USA
andy.stark@gtri.gatech.edu

***Abstract*—Natural signals are inherently comprised of two components, real and imaginary components. Due to recent successes and progress in Deep Learning, specifically Convolutional Neural Networks (CNNs), this field of machine learning has become extremely popular when handling a wide variety of data, including natural signals. However, deep learning frameworks have been developed to deal with exclusively real-valued data and are unable to compute convolutions for complex-valued data. In this work, we present a linear combination that enables deep learning architectures to compute complex convolutions and learn features across the real and imaginary components of natural signals. When implemented into existing I/Q modulation classification architectures, this small change increases classification accuracy across a range of SNR levels by up to 35%.**

***Index Terms*—Complex Convolution, Deep Learning, I/Q Modulation**

I. Introduction

Complex numbers are fundamental for signal processing. In fields such as telecommunications, speech and audio processing, and medical image processing, the data is comprised of real and imaginary components. For example, in digital communications the real and imaginary components of complex numbers are the in-phase (I) and quadrature (Q) components of a signal, representing orthogonal dimensions. Utilizing the relationship between the real and imaginary parts of a signal, not simply the structure within the real and imaginary channels individually, is critical for properly executing convolutions.

Within the fields of signal processing and machine learning, deep learning has gained a great deal of popularity in recent years due to remarkable performance on traditional machine learning tasks [1]–[3]. Deep learning is also being used to tackle problems in biology [4], [5], transportation [6], [7], and finance [8], [9]. Deep learning scripting frameworks have been developed such as TensorFlow [10] and Keras [11] in order to lower the level of programming expertise needed to deploy such models. However, the deep learning architectures these scripting languages create are optimized for real-valued inputs and cannot execute convolutions for complex-valued data. Real-valued optimization presents challenges when looking to apply these deep learning techniques to datasets comprised of real and imaginary components, such as I/Q modulated data from a communication system.

In this work we present a methodology for handling complex data as two-columned real arrays combined with real-valued 2D convolution followed by a linear combination kernel that enables neural networks to compute complex convolutions within real-valued deep learning paradigms. Easy to implement, we show this modification can improve performance over traditional approaches to complex-valued data by learning features that account for the relationship between the real and imaginary components in addition to structure within the respective channels.

II. Current Methods for Handling Complex Numbers in Deep Learning

There are a few different approaches for performing computations in the complex domain. Simple ways to handle complex data streams include keeping only the real values or taking the magnitudes of the complex numbers. Although simplifying the dataset, these approaches are detrimental as they discard important information and structure comprised in the dataset. In deep learning, specifically Convolutional Neural Networks (CNN's) [12], there are a few common approaches to learn features on multi-channeled data (complex-valued data, RGB images, etc.)

1. Reduce the input dimension down to one channel via a 1 x 1 convolution, then learn features [13]
2. Learn a set of features per channel
3. Learn one set of features for all channels (i.e. both the I and Q components as demonstrated in [14])

However, these approaches do not account for relationships between the channels of the data.

Recent work aiming to allow complex numbers in neural networks includes [15] where the authors explored the use of complex values to parameterize real-valued orthogonal matrices in order to stabilize the training of Recurrent Neural Networks. They introduce an extension of the ReLU activation which only effects the absolute value of a complex number called modReLU. Further, [16] developed another complex-valued activation function that preserves the phase of a signal while taking the sigmoid of the magnitude called the complex cardiod. Trabelsi et al. [17] developed Complex Neural Networks which were comprised of various components to handle complex values in deep learning paradigms such as

978-1-7281-7440-2/20/$31.00 ©2020 IEEE

convolutions, batch normalizations, and activation functions. The convolutions presented aimed to create complex-valued representations or weights for a neural network but did not enable a network to seamlessly compute complex convolutions on complex-valued data. Additionally, [18], developed Fourier Convolutional Neural Networks which leverage converting the inputs into the Fourier domain in order to speed up computation by using the Hadamard product in place of computing a convolution. The reduction in computational complexity enabled larger images to be input into the network as well as being able learn features over larger kernel sizes, but does not address the inability to compute a convolution with complex-valued inputs.

III. Deep Learning for Modulation Classification

Before 2016, works such as [19] and [20] were the latest developments in the field of modulation classification, before [14] began to use deep learning for this problem space.

Since [14], [21] performed a thorough exercise varying methods for radio signal classification. The results show deep learning architectures such as ResNets [22] significantly outperform advanced statistical machine learning methods such as gradient boosted trees with hand crafted high-order statistical features. In [23], researchers furthered this work by trying many different deep learning approaches to classify modulated radio signals and investigated how to best reduce the training time of the approaches. Deep learning architectures investigated include their own CNN, DenseNet [24], and Convolutional Long Short-term Deep Neural Network (CLDNN) [25] architectures, as well as hyperparameter tuned versions of the ResNet and LSTM architectures. Their results indicated tuned versions of the ResNet and LSTM performed better at different SNRs. Others [26] tested modern convolutional arhitectures such as AlexNet [1] and GoogleNet [13] on the classification of signal constellation plots. Their results show these deep learning approaches perform significantly better than traditional cumulant and statistical machine learning methods.

However, [23] concluded that the use of more sophisticated architectures compared to that of [14] results in better performance as these approaches use far more parameters and introduce new non-linear relationships in the learned mapping. However, it is not clear if their superior performance was due solely to the increase in parameterization, or because the approach was able to learn feature representations that leveraged inherent structure in the data.

The focus of this work is to use simpler convolutional models, like those in [14], to enhance performance by enabling the architecture to directly compute complex convolutions. Future work will investigate more complicated architectures, alternative activation functions, batch-normalizations, etc.

IV. Complex Convolution Using 2D Real Convolution

In this section, we detail how to compute a complex convolution within real-valued deep learning frameworks. Consider a two dimensional I/Q data stream $(Z_n)_{n=1}^{N}$. The elements of Z_n are defined by

$$Z_n = I_n + jQ_n,\ I_n, Q_n \in \mathbb{R}$$

where I_n and Q_n represent the n^{th} in-phase and quadrature components of Z and j is the imaginary unit with value equal to $\sqrt{-1}$. I and Q are defined to contain all N elements in the respective channels.

We introduce another I/Q data stream, h, that contains M complex filter coefficients,

$$h_m = h'_m + jh''_m$$

where h'_m and h''_m are the m^{th} in-phase and quadrature components of h. The sequence of coefficients h can also be called taps (from signal processing parlance), or weights (from deep learning parlance).

The two dimensional $(N \times 2)$ depiction of Z and h are shown below.

$$Z = \begin{array}{|c|c|}\hline I_1 & Q_1 \\ \hline I_2 & Q_2 \\ \hline I_3 & Q_3 \\ \hline \vdots & \vdots \\ \hline I_N & Q_N \\ \hline \end{array} \quad , \quad h = \begin{array}{|c|c|}\hline h'_1 & h''_1 \\ \hline h'_2 & h''_2 \\ \hline h'_3 & h''_3 \\ \hline \vdots & \vdots \\ \hline h'_M & h''_M \\ \hline \end{array}$$

Convolving the signals Z and h in accordance with Deep Learning convolutions, as a naïve sliding window, yields X_{DL} which contains three columns as shown in Equations 1 and 2.

$$X_{\text{DL}} = \begin{array}{|c|c|}\hline I_1 & Q_1 \\ \hline I_2 & Q_2 \\ \hline I_3 & Q_3 \\ \hline \vdots & \vdots \\ \hline I_N & Q_N \\ \hline \end{array} \circledast \begin{array}{|c|c|}\hline h'_1 & h''_1 \\ \hline h'_2 & h''_2 \\ \hline h'_3 & h''_3 \\ \hline \vdots & \vdots \\ \hline h'_M & h''_M \\ \hline \end{array} \tag{1}$$

$$X_{\text{DL}} = \begin{array}{|c|c|c|}\hline \uparrow & \uparrow & \uparrow \\ I \circledast h' & I \circledast h'' + Q \circledast h' & Q \circledast h'' \\ \downarrow & \downarrow & \downarrow \\ \hline \end{array} \tag{2}$$

However, the one dimensional complex-valued convolution between sequences Z and h is Equation 3.

$$X = (I+jQ)\circledast(h'+jh'') = (I\circledast h' - Q\circledast h'') + j(I\circledast h'' + Q\circledast h') \tag{3}$$

This is rewritten into the real/imaginary two column and shown in 4

$$X = \begin{array}{|c|c|}\hline \uparrow & \uparrow \\ I \circledast h' - Q \circledast h'' & I \circledast h'' + Q \circledast h' \\ \downarrow & \downarrow \\ \hline \end{array} \tag{4}$$

Comparing Equation 2 with Equation 4, it can be readily seen that X is the result of a linear combination of the columns of X_{DL}, shown below.

$$X = X_{\text{DL}} \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 0 \end{bmatrix}$$

Therefore the operation of the deep learning 2D real-valued convolution followed by a multiplication with a constant matrix implements a 1D complex convolution, where the weights, inputs, and outputs are all in two-column complex number format.

For higher dimensional inputs, say $(Z_n^D)_{n=1}^N$, where $Z \in \mathbb{C}^D$, this linear combination extends to a $3 \times 2 \times D$ matrix.

V. Integrating into Existing Deep Learning Architectures

Since our proposed modification is a linear combination operation it does not necessitate extensive changes to existing convolutional architectures, it simply needs to be placed after a traditional convolutional layer. For example Fig. 1a is a Convolutional Neural Network that has been used to classify modulated I/Q data. Details for this CNN, originally named 'CNN2' [14], can be seen in Table I.

In Fig. 1b we show the CNN with the addition of the linear transformation, which we name the 'Complex' network, with architecture details in Table II. After zero-padding the input and performing a standard convolution without an activation function, the linear transformation changes the tensor with a dimension size of three, via linear combination, producing an output such that the I and Q structure is preserved. We understand this may not be the most efficient way to implement this architectural change, see Table III, but this is due to legacy code developed in Theano [27]. The linear transformation increases the strength of the representation learned as it allows the original I/Q structure to be passed along in the network, allows for increased filter size to better address the I and Q components, and learn the relationship between the real and imaginary components which is not done in [14].

As one would imagine, with larger kernel sizes there are more parameters to optimize and in turn boost the Complex network to outperform the CNN2 architecture. In fact, the Complex architecture has 1.54% more parameters. To address this difference, we created a modified CNN2 network with a dense layer having 260 nodes instead of 256, which we will refer to as CNN2-260. The Complex network has 0.003% more parameters than the CNN2-260 architecture, which we deem as equivalent to demonstrate the difference between simply adding more parameters and being able to compute a complex convolution. Further details about the architectures is shown in Table III.

VI. IQ Modulation Classification Task

These architectures were trained and tested on the RadioML 2016.10A open source dataset used in [14]. The dataset consists of 11 modulations (8 digital and 3 analog) at SNR levels from -20 to 20 dB with 2 dB steps while including variation in other properties such as center frequency, sample clock rate, sample clock offset, and initial phase, amongst other properties [14]. The models were trained to classify modulation for a given input signal. With a total of 220,000 data samples, the data shuffled, across modulation type and SNR, and was split into even 50/50 training/testing sets.

VII. Results

The addition of the linear combination increased accuracy at all SNR levels, Fig. 2(a). Figure 2(b) quantifies the improved classification accuracy at each SNR.

The greatest performance increases were at SNR levels lower than -5 dB, with improvements exceeding 30% relative to the CNN2 and CNN2-260 networks. The overall, average across all SNR's, confusion matrices and classification accuracies can be found in Fig. 3, the -20 dB SNR confusion matrices and classification accuracies can be found in Fig. 4, and the 20 dB SNR confusion matrices and classification accuracies can be found in Fig. 5. Figure 4(c) shows the complex convolution enables features to be learned that can classify with an accuracy over 30% greater than random chance (11.95% versus 9.09%) at -20 dB. In Fig.5 we see the complex convolutions allowed for improved classification at high SNR cases as well. When comparing the Complex network and the CNN2-260 performance in 5 (b) and (c), we see the Complex network having less off-diagonal noise, indicating better performance. However, the Complex network has more errors with differentiating between QAM16 and QAM64 modulations which is not surprising because these two modulations have very similar structure.

In Table III we compare the accuracy and the time required for training. We observe the complex convolutions require more than four times more computation time compared to CNN2 methods. As mentioned earlier, our current implementation, relies on legacy code, which was written in Theano, and there are likely computational differences to be found. The training and validation curves are shown in Fig. 6.

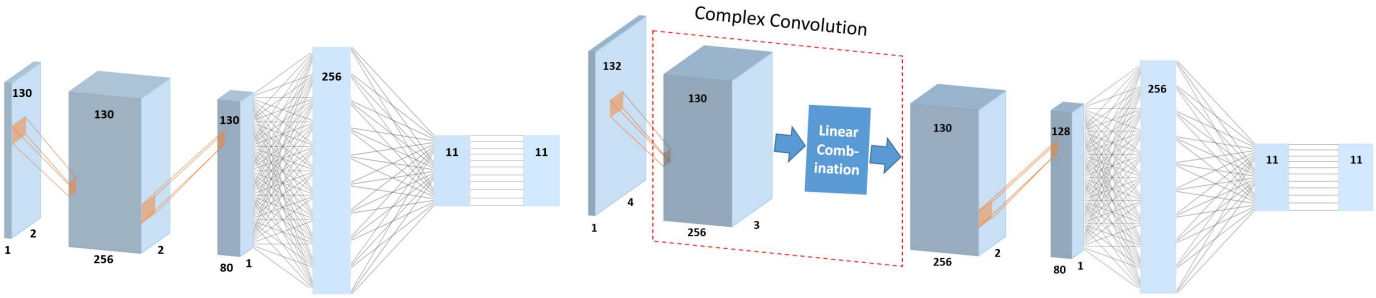
The overall accuracy achieved by the Complex network is much greater than that of the CNN2 and the CNN2-260 networks. This validates the Complex did not perform well simply because of having more parameters, rather leveraging the complex structure of the data and learning features that correlate the real and complex channels. Overall accuracy and the number parameters for each of the three architectures are presented in Table III.

VIII. Open Source Code

The GitHub repository where code and information about the dataset can be found is github.com/JakobKrzyston/Complex_Convolutions/.

Conclusion

By implementing a linear combination we enable neural networks to compute complex convolutions and extract features over the real and complex components with respect to the structure of the data. When implemented into an existing architecture, overall accuracy increases, especially in low SNR conditions. Our method outperformed a very similar architecture with nearly the same number of parameters. With this modification, all operations are done on real-valued arrays, but are interpreted as complex numbers. The proposed approach ensures that this complex interpretation is enforced throughout



(a) (b)

Fig. 1. (a) Schematic of the CNN architecture, CNN2 used to classify modulated I/Q signals in [14]. Please see Table I for the architecture details. (b) Schematic of the Complex architecture, which is CNN2 with modification to compute complex convolutions. For architecture details, see the Table II.

TABLE I
ARCHITECTURE DETAILS OF CNN ARCHITECTURE, USED IN [14], CNN2*

Layer Type	Input Size	Output Size	Details
Input	1 x 2 x 128	-	-
Zero-Padding	1 x 2 x 128	1 x 2 x 130	Padding: 0, 0, 2
Convolution	1 x 2 x 130	256 x 2 x 130	Activation: ReLU, Kernel: 1 x 3, Dropout: 0.5
Convolution	256 x 2 x 130	80 x 1 x 130	Activation: ReLU, Kernel: 2 x 1, Dropout: 0.5
Flatten	80 x 1 x 130	10400	-
Dense	10400	256	ReLU

*This Architecture has 2,707,547 parameters.

TABLE II
ARCHITECTURE DETAILS OF THE COMPLEX ARCHITECTURE*

Layer Type	Input Size	Output Size	Details
Input	1 x 2 x 128	-	-
Zero-Padding	1 x 2 x 128	1 x 4 x 132	Padding: 0, 2, 4
Convolution	1 x 4 x 132	256 x 3 x 130	Activation: None, Kernel: 2 x 3
Permute	256 x 3 x 130	256 x 130 x 3	-
Linear Transformation	256 x 130 x 3	256 x 130 x 2	-
Permute	256 x 130 x 2	256 x 2 x 130	-
Activation	256 x 2 x 130	256 x 2 x 130	Activation: ReLU
Dropout	256 x 2 x 130	256 x 2 x 130	Dropout = 0.5
Convolution	256 x 2 x 130	80 x 1 x 128	Activation: ReLU, Kernel: 2 x 3, Dropout = 0.5
Flatten	80 x 1 x 128	10240	-
Dense	10240	256	ReLU
Dense	256	11	ReLU
Softmax	11	11	One-Hot Output

*This Architecture has 2,749,275 parameters.

the processing chain of neural network layers, enabling greater performance from the network.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105. [Online]. Available: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[2] Y. Huang, Y. Cheng, A. Bapna, O. Firat, M. X. Chen, D. Chen, H. Lee, J. Ngiam, Q. V. Le, Y. Wu, and Z. Chen, "Gpipe: Efficient training of giant neural networks using pipeline parallelism," 2018.

[3] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap, K. Simonyan, and D. Hassabis, "Mastering chess and shogi by self-play with a general reinforcement learning algorithm," 2017.

[4] P. Mamoshina, A. Vieira, E. Putin, and A. Zhavoronkov, "Applications of deep learning in biomedicine," *Molecular pharmaceutics*, vol. 13, no. 5, pp. 1445–1454, 2016.

[5] G. Zhu, B. Jiang, L. Tong, Y. Xie, G. Zaharchuk, and M. Wintermark, "Applications of deep learning to neuro-imaging techniques," *Frontiers in Neurology*, vol. 10, p. 869, 2019. [Online]. Available: https://www.frontiersin.org/article/10.3389/fneur.2019.00869

[6] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," 2015.

[7] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," 2015.

[8] W. Bao, J. Yue, and Y. Rao, "A deep learning framework for financial time series using stacked autoencoders and long-short term memory," *PLOS ONE*, vol. 12, no. 7, pp. 1–24, 07 2017. [Online]. Available: https://doi.org/10.1371/journal.pone.0180944

[9] F. Bertoluzzo and M. Corazza, "Testing different reinforcement learning configurations for financial trading: Introduction and applications," *Procedia Economics and Finance*, vol. 3, p. 68–77, 12 2012.

[10] M. Abadi *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: http://tensorflow.org/

[11] F. Chollet *et al.*, "keras," 2015.

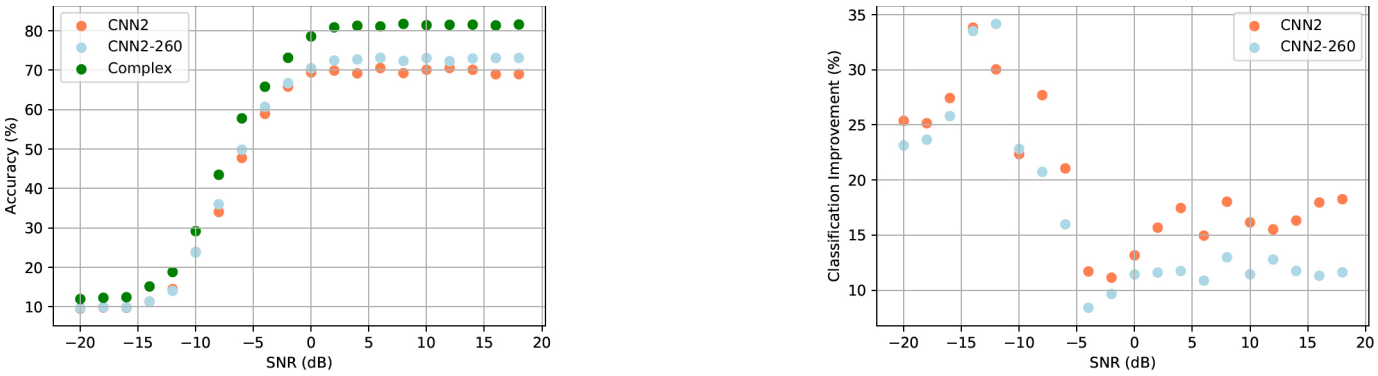


Fig. 2. (a) Classification accuracy and (b) classification improvement (%) of the Complex architecture compared to the CNN2 and CNN2-260 architectures, computed by $100 \times \frac{Accuracy_{Complex} - Accuracy_{model}}{Accuracy_{model}}$, as a function of SNR.

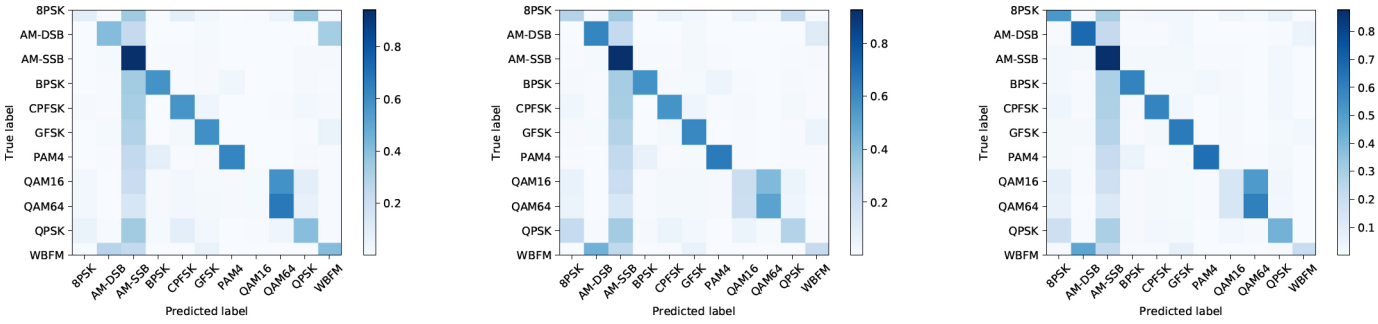


Fig. 3. Average, average across all SNR's, confusion matrices for the (a) CNN2, (b) CNN2-260, and (c) Complex architectures. The respective classification accuracies are 48.45%, 49.63%, and 53.79%.

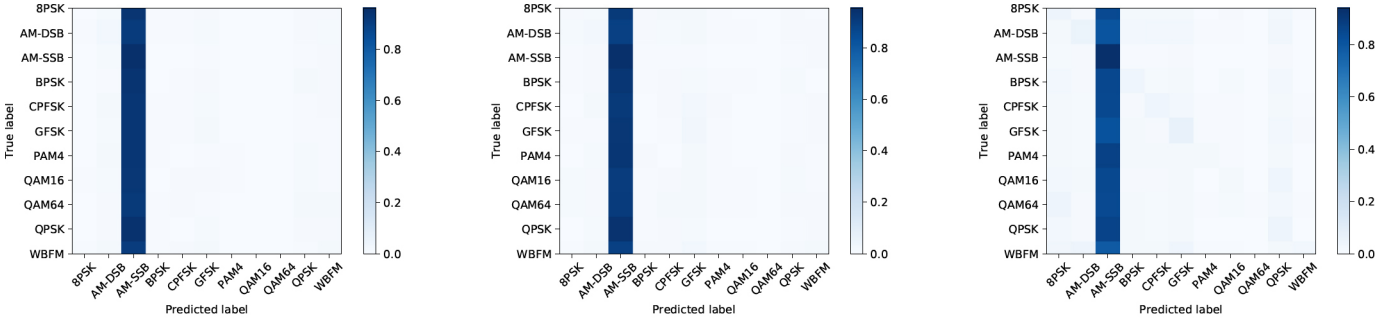


Fig. 4. Confusion matrices at -20 dB for the (a) CNN2, (b) CNN2-260, and (c) Complex architectures. The respective classification accuracies are 9.54%, 9.71%, and 11.95%. In addition to the Complex network outperforming random guess by over 30%, the corresponding confusion matrix shows a faint diagonal forming. Thus, the features learned from the use of complex convolutions are more robust in the presence of 20 dB noise than those learned by traditional CNN architectures.

[12] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," in *Proceedings of the IEEE*, 1998, pp. 2278–2324.

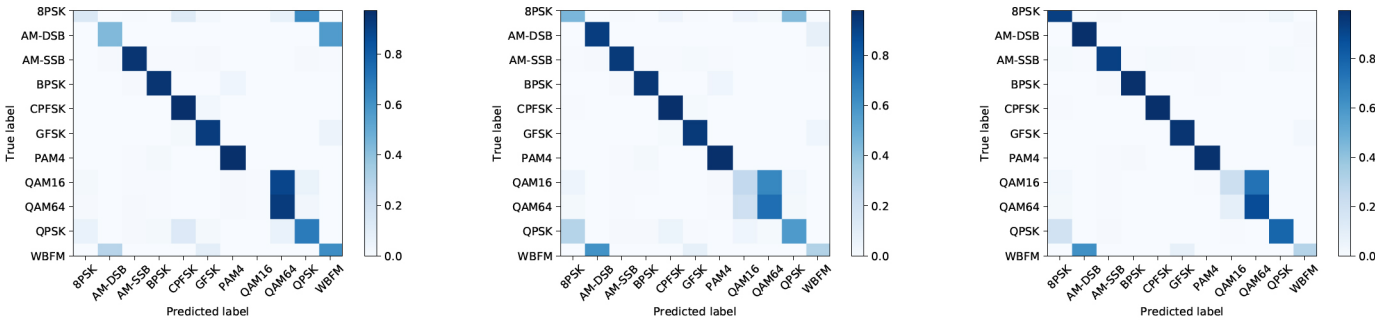
[13] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Computer Vision and Pattern Recognition (CVPR)*, 2015. [Online]. Available: http://arxiv.org/abs/1409.4842

[14] T. J. O'Shea, J. Corgan, and T. C. Clancy, "Convolutional radio modulation recognition networks," in *International conference on engineering applications of neural networks*. Springer, 2016, pp. 213–226.

[15] M. Arjovsky, A. Shah, and Y. Bengio, "Unitary evolution recurrent neural networks," in *International Conference on Machine Learning*, 2016, pp. 1120–1128.

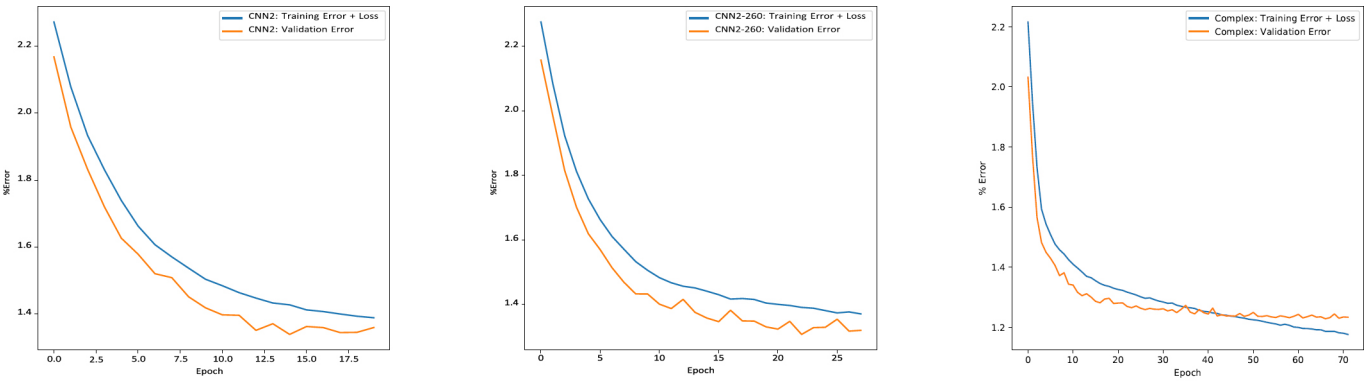
[16] P. Virtue, X. Y. Stella, and M. Lustig, "Better than real: Complex-valued neural nets for mri fingerprinting," in *2017 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2017, pp. 3953–3957.

[17] C. Trabelsi, O. Bilaniuk, Y. Zhang, D. Serdyuk, S. Subramanian, J. F. Santos, S. Mehri, N. Rostamzadeh, Y. Bengio, and C. J. Pal, "Deep



(a) (b) (c)

Fig. 5. Confusion matrices at 20 dB for the (a) CNN2, (b) CNN2-260, and (c) Complex architectures. The respective classification accuracies are 68.99%, 73.09%, and 81.58%. In the Complex confusion matrix, the off diagonal noise is greatly reduced. Thus, the features learned through the use of complex convolutions are able to better differentiate the modulations in the presence of high signal strength compared to the other CNN approaches used.



(a) (b) (c)

Fig. 6. The training and validation loss curves for the (a) CNN2, (b) CNN2-260, and (c) Complex architectures.

TABLE III
ARCHITECTURAL PERFORMANCE

Architecture	Overall Accuracy	Paramaters	Epochs	Avg Epoch Duration (s)
CNN2	48.45%	2,707,547	20	6.72
CNN2-260	49.63%	2,749,195	28	6.74
Complex	53.79%	2,749,275	72	13.62

complex networks," *arXiv preprint arXiv:1705.09792*, 2017.

[18] H. Pratt, B. Williams, F. Coenen, and Y. Zheng, "Fcnn: Fourier convolutional neural networks," in *Machine Learning and Knowledge Discovery in Databases*, M. Ceci, J. Hollmén, L. Todorovski, C. Vens, and S. Džeroski, Eds. Cham: Springer International Publishing, 2017, pp. 786–798.

[19] P. Isautier, J. Pan, R. DeSalvo, and S. E. Ralph, "Stokes space-based modulation format recognition for autonomous optical receivers," *Journal of Lightwave Technology*, vol. 33, no. 24, pp. 5157–5163, 2015.

[20] P. Isautier, J. Langston, J. Pan, and S. E. Ralph, "Agnostic software-defined coherent optical receiver performing time-domain hybrid modulation format recognition," in *Optical Fiber Communication Conference*. Optical Society of America, 2015, pp. Th2A–21.

[21] T. J. O'Shea, T. Roy, and T. C. Clancy, "Over-the-air deep learning based radio signal classification," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 168–179, 2018.

[22] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[23] S. Ramjee, S. Ju, D. Yang, X. Liu, A. E. Gamal, and Y. C. Eldar, "Fast deep learning for automatic modulation classification," *arXiv preprint arXiv:1901.05850*, 2019.

[24] G. Huang, Z. Liu, and K. Q. Weinberger, "Densely connected convolutional networks. corr abs/1608.06993 (2016)," *arXiv preprint arXiv:1608.06993*, 2016.

[25] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, "Convolutional, long short-term memory, fully connected deep neural networks," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 4580–4584.

[26] 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, 2018.

[27] Theano Development Team, "Theano: A Python framework for fast computation of mathematical expressions," *arXiv e-prints*, vol. abs/1605.02688, May 2016. [Online]. Available: http://arxiv.org/abs/1605.02688