Split-Complex Convolutional Neural Networks

Timothy Anderson
Department of Electrical Engineering
Stanford University
Stanford, CA 94305
timothy.anderson@stanford.edu

1 Introduction

Beginning with the seminal work of [1], the last half-decade of artificial intelligence and computer vision has been dominated by the stunning success of convolutional neural networks (CNNs). In visual recognition, a robust classifier must be able to recognize objects under deformation. One solution that has been proposed for improving invariance under rotation is complex-valued CNNs [2, 3]. While the results for complex-valued CNNs have been intriguing, questions remain open about whether the successes of complex neural networks arises from the link between complex numbers and rotational transformations, the increased complexity of the complex-valued computational graph, or simply the inclusion of additional network parameters.

Meanwhile, split-complex numbers have proven useful in describing systems with hyperbolic structure, the most common application being special relativity and Lorentz transforms. Following the same logic behind complex-valued networks, this suggests that split-complex numbers—with their implicit hyperbolic structure—could potentially account for shear deformations or other distortions present in visual data.

Here, we extend the work in [2] to split-complex-valued CNNs. Doing so provides insight into the connection between the computational graph, algebraic system, and expressiveness of a neural network. In turn, we hope this work will help answer some of the open questions surrounding complex-valued neural networks and applications of other Clifford algebras to neural networks.

2 Background and Related Work

[4] proposed a variant of the LMS algorithm for training complex-valued adaptive filters, which was a precursor to complex-valued networks. Complex neural networks themselves were proposed by [5]. [6] and [7] separately presented a complex-valued version of backpropagation and suitable activation functions, [8] soon after presented a more general framework for complex backpropagation, and [9] further generalized complex backpropagation and showed the suitability of eight transcendental functions for use as activation functions.

Complex-valued networks are of interest due to their generalization properties [10], ability to capture phase in signals [11, 12, 13], and elegant mathematical properties [14, 15]. [11] asserts that the power of complex-valued networks lies in their ability to handle phase-amplitude transformations to signals. This interpretation is at least partially validated by results in [16], which demonstrates that a phase-amplitude representation better captured neural dynamics in biological neural networks. Most recently, [2] used complex-valued CNNs to achieve competitive performance on several visual recognition datasets, and state-of-the-art performance on the MusicNet dataset [17].

Complex numbers themselves belong to a broader family of algebraic systems known as Clifford algebras. Split-complex numbers are a counterpart to complex numbers, and together they form the two-dimensional cases of Clifford algebras. We can define a split-complex number as an analogue to a complex number. A complex number $x \in \mathbb{C}$ has the form $x = a + bi$ where $a, b \in \mathbb{R}$ and i is the imaginary component with the property $i^2 = -1$. A split-complex number $x \in \mathbb{S}$ assumes the same form, except the imaginary component has the property $i^2 = +1$ (with $i \neq 1$). Complex numbers are associated with rotational transformations due to the structure of their associated transformation matrices [10], while split-complex numbers are similarly associated with hyperbolic transformations [18].

The application of complex numbers to neural networks has been motivated by the natural connection between complex numbers and periodic structure in signals [5]. The more general hypothesis is that natural structure in data is most efficiently captured by algebraic systems with analogous mathematical structure, which in turn has motivated the application of other Clifford algebras to neural networks.

[19] was the first to propose Clifford algebras for neural networks and generalizes backpropagation to Clifford algebras. The "Clifford neuron" was introduced in [20], which provided a wider mathematical framework for Clifford algebras in neural networks. [21] presents further results from this same line of work. The only study which seems to explicitly treat split-complex-valued neural networks is [18]. Results from this suggested that in some contexts, split-complex networks may have improved generalization properties over real- and complex-valued networks, but results were only presented for a small synthetic dataset.

Besides these few studies, split-complex-valued networks have received little attention, and to the best of our knowledge have never been applied to CNNs. In the intervening years since Clifford algebras in neural networks were an active research interest, CNNs have taken over the field of neural networks due to their state-of-the-art performance on tasks ranging from visual recognition [1, 22, 23] to audio generation [24]. For this reason, we turn our attention here to evaluating split-complex CNNs for visual recognition tasks to provide a baseline against their complex-valued counterparts.

3 Mathematical Framework

For the remainder of this paper, we closely follow the methods in [2] and extend their approach to split-complex CNNs to provide a comparison between both number systems. The mathematical framework below is based on that in [2], and its extension to split-complex numbers is our original work.

Representation: We represent complex-valued parameters as two real-valued sets of parameters that are connected in the network computational graph according to the algebraic rules induced by complex or split-complex arithmetic. A convenient consequence of this is that we can rewrite each algebraic operation between complex numbers as a combination of real-valued algebraic operations between the real and imaginary parts. For example, if we have a split-complex-valued filter bank $\mathbf{W} = \mathbf{W}_R + \mathbf{W}_I i$ which convolve with inputs $\mathbf{X}_i = \mathbf{X}_{i,R} + \mathbf{X}_{i,I} i$, then

$$\mathbf{W} * \mathbf{X} = \mathbf{W}_R * \mathbf{X}_{i,R} + \mathbf{W}_I * \mathbf{X}_{i,I} + (\mathbf{W}_R * \mathbf{X}_{i,I} + \mathbf{W}_I * \mathbf{X}_{i,R})i$$

Activation: The same approach can be followed for the activation functions. Suppose we use nonlinearity $\sigma(\cdot)$ and treat σ as a complex-valued i.e. $\sigma : \mathbb{C} \to \mathbb{C}$. Then, we can rewrite σ as $\sigma(X) = \sigma_R(X_R, X_I) + \sigma_I(X_R, X_I)i$.

For ReLU activation, [3] proposes that $ReLU(x) = x$ if $\Re(x), \Im(x) \geq 0$. However, we use:

$$ReLU(x) = \begin{cases} x & \Re(x) \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

Our argument is that a real-valued ReLU activation is zero for half of the input space, and therefore should also be zero for half of the complex-valued input space.

Pooling: We use mean-pooling in the experiments below, although we should note that [3] presents a framework for max-pooling in complex-valued networks.

Weight Initialization: While [2] presents some results for weight-initialization in line with the work presented by [25] and [23], we choose here to initialize the real and imaginary parts of the network parameters to uncorrelated Gaussian random weights with variance 10^{-2}, since in practice this worked well.

Layer	Baseline network	Wide network	Deep network
Input	3x32x32 (1x32x32)	3x32x32 (1x32x32)	3x32x32 (1x32x32)
C1: feature maps	6@5x5	9@5x5	6@5x5
C1(b): feature maps	–	–	6@5x5
S2: feature maps	–	–	–
C3: feature maps	16@5x5	23@5x5	16@5x5
C3(b): feature maps	–	–	16@5x5
S4: feature maps	–	–	–
C5: layer	120@5x5	170@5x5	120@5x5
C5(b): layer	–	–	120@5x5
F6: layer	84	119	84
F6(b): layer	–	–	84
Output	10	10	10

Table 1: Outline of network topologies. Input is a 32x32 image (either 1 or 3 channels). Network topology is based on the LeNet-5 architecture [27]. *Note:* in complex- and split-complex-valued networks, a complex weight counts as a single parameter.

Training: Complex arithmetic can be viewed as inducing a "computational multi-graph" on the network, where each node in the computational graph of the complex network contains a computational sub-graph of operations between the real and imaginary parts. Since the complex and split-complex numbers are commutative fields, this means that backpropagation can simply flow through this computational multi-graph, instead of necessitating generalized derivatives in Clifford algebras (e.g. [26]).

To implement the complex network, we build the computational multi-graph using wrapper functions that call real-valued functions from an off-the-shelf neural network toolbox. This is equivalent to function overloading, but we instead use wrapper functions to implement the computational subgraphs in order to preserve GPU acceleration made possible with standard neural network libraries.

4 Experiments

4.1 Setup

The specific questions we seek to answer is whether the apparent increased expressivity in complex-valued networks is due to the algebraic properties of complex numbers, the increased number of network parameters, or locally increasing the complexity of the computational graph. To this end, we present results for three different network sizes: a baseline network based off LeNet-5 from [27], a "wide" network where the number of filters or neurons at each layer is increase by $\sim \sqrt{2}$, and a "deep" network where each layer in the baseline is repeated. The network topologies are described in table 1. We use ReLU activations following each C and F layer, and compute S layers via 2x2 mean-pooling.

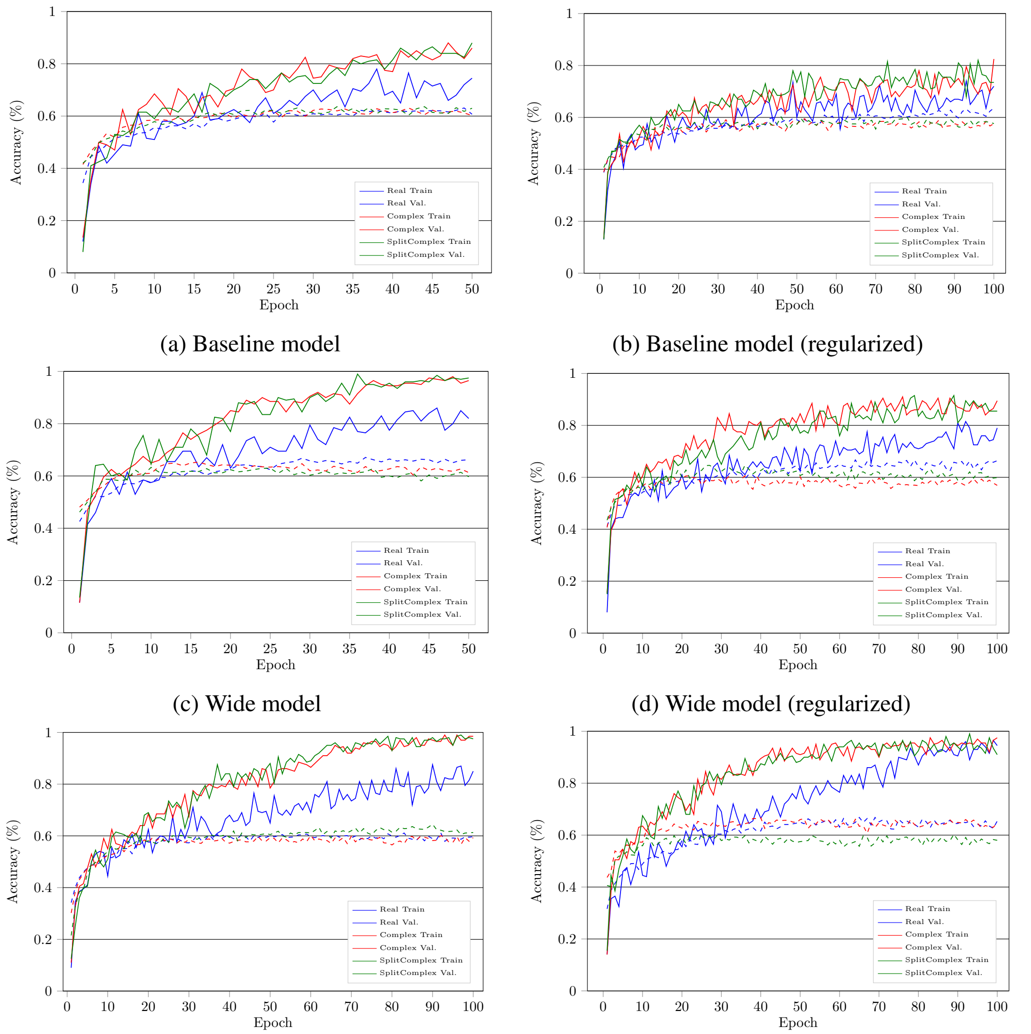
The wide and deep networks contain approximately 2x more parameters than the baseline. While the connection between network topology and expressivity is still poorly understood, providing results for both smaller and larger networks can control for the additional expressivity that may be induced simply by doubling the number of network parameters. This is the same motivation behind the "small" and "large" networks explored in [2], but note that the authors there doubled only the depth of the network rather than the width. (Specifically, [2] employs a ResNet-like architecture from [23], which naturally lends itself to increasing depth.)

4.2 Results

We tested the proposed network architectures on the MNIST [27] and CIFAR-10 [28] datasets. Networks were implemented in Tensorflow [29] and trained using the Adam optimizer [30]. [3] theorized that complex networks are self-regularizing, so to test the regularization properties of complex-valued and split-complex networks, we also added L_2 regularization for all parameters and trained on CIFAR-10. Results are summarized in table 2.

Architecture	MNIST	CIFAR-10	CIFAR-10 (+L_2 reg)
Real	1.1	38.3	39.0
Complex	1.1	40.6	41.4
Split-Complex	1.1	38.7	43.3
Real (Wide)	0.9	**35.1**	**35.9**
Complex (Wide)	1.0	38.7	43.6
Split-Complex (Wide)	**0.7**	38.9	41.2
Real (Deep)	**0.7**	42.2	37.9
Complex (Deep)	1.3	40.5	36.3
Split-Complex (Deep)	1.0	38.9	42.6

Table 2: Test set error (%) results from visual recognition experiments.



(a) Baseline model (b) Baseline model (regularized)

(c) Wide model (d) Wide model (regularized)

(e) Deep model (f) Deep model (regularized)

Figure 1: Training curves for CIFAR-10 data. Training set accuracy shown in solid lines, validation set accuracy in dashed lines.

4.3 Discussion

The most striking result is the regularization properties of the complex and split-complex models. Our results suggest that contrary to the claim in [3], complex-valued networks are not self-regularizing. If this property were true, we would expect the complex and split-complex models to consistently have a higher training set accuracy than the real-valued unregularized network. However, the training curves in fig. 1 show that all three networks have comparable validation accuracy in the unregularized case. This shows that complex-arithmetic (or split-complex) likely does not create the self-regularization that we expected. Indeed, the real-valued networks tend to overfit less, which is likely due to having fewer trainable parameters.

Furthermore, we observe that complex and split-complex networks are more difficult to regularize than real-valued networks. We see that it reduces overfitting during training in the real-valued networks, but has little effect on the complex and split-complex networks. Even when the hyperparameters were tuned, the complex and split-complex networks had little difference in performance under regularization. This suggests that the increased complexity of the computational graph in complex-valued networks may make these more sensitive to hyperparameter tuning.

Beyond regularization results, we also see that complex and split-complex networks do not have noticeably increased expressivity over their real-valued counterparts. Across all experiments, performance for networks were similar: changing the arithmetic did not dramatically increase the performance. This is in line with the results in [2], where using a deeper network architecture had a similar or even greater effect on accuracy than using complex-valued weights. Indeed, for our experiments, the real-valued wide network had the lowest error on CIFAR-10 for both regularized and unregularized models.

5 Conclusion

Here we have presented a simple neural network and tested its performance on visual recognition tasks using real, complex, and split-complex weights. Our results show that varying the macro-scale topology of the computational graph has a greater effect on the accuracy than small changes in the arithmetic rules of the algebraic system. While this does not definitively disprove the potential of complex networks, it suggests that in cases with real-valued input data, complex networks may perform comparably or even worse compared to real-valued networks.

The most promising future application of this work is in domains with naturally-arising complex-valued data (e.g. MRI or electromagnetics). While this work suggests that complex-valued networks are likely not superior to real-valued networks for real-valued data, it is possible that they offer distinct advantages for complex-valued data.

Another future step in this area is characterizing and improving regularization techniques for complex-valued neural networks. While the results presented here suggest that complex-valued networks are more prone to overfitting than real-valued networks, it may be possible to combat this by employing other regularization techniques. or more thoroughly exploring the hyperparameter space.

Finally, long-term work in this line of research would be applying other Clifford algebras to neural networks. While many neural network architectures originating in the 1990s have experienced an explosive resurgence in popularity over the last 5 years, complex-valued and neural networks based in other Clifford algebras have yet to follow suit. Consequently, there is ample opportunity to apply modern computing power to these networks.

6 Acknowledgements

Thank you to Dr. Monica Martinez-Canales (Intel Corporation) for her guidance on this project, and for introducing me to the beautiful subject of multi-dimensional algebraic systems.

7 Code

Code for this project is available at `https://github.com/timanderson1994/split-complex-net`.

References

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. *Advances In Neural Information Processing Systems*, pages 1–9, 2012.

[2] Chiheb Trabelsi, Olexa Bilaniuk, Dmitriy Serdyuk, Sandeep Subramanian, João Felipe Santos, Soroush Mehri, Negar Rostamzadeh, Yoshua Bengio, and Christopher J Pal. Deep Complex Networks. (Nips), 2017.

[3] Nitzan Guberman. On Complex Valued Convolutional Neural Networks. *arXiv*, 2016.

[4] Bernard Widrow, John McCool, and Michael Ball. The Complex LMS Algorithm. *Proceedings of the IEEE*, 63(4):719–720, 1975.

[5] T L Clarke. Generalization of Neural Networks to the Complex-Plane. *Ijcnn International Joint Conference on Neural Networks, Vols 1-3*, pages B435–B440, 1990.

[6] H. Leung and S. Haykin. The complex backpropagation algorithm. *IEEE Transactions on Signal Processing*, 39(9):2101–2104, 1991.

[7] A Hirose. Continuous Complex-Valued Backpropagation Learning. *Electronics Letters*, 28(20):1854–1855, 1992.

[8] George M. Georgiou and Cris Koutsougeras. Complex Domain Backpropagation. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, 39(5):330–334, 1992.

[9] T Kim, T Adali, Ieee, and Ieee. Complex backpropagation neural network using elementary transcendental activation functions. *2001 Ieee International Conference on Acoustics, Speech, and Signal Processing, Vols I-Vi, Proceedings: Vol I: Speech Processing 1; Vol Ii: Speech Processing 2 Ind Technol Track Design & Implementation of Signal Processing Systems Neuralnetworks for Signa*, pages 1281–1284, 2001.

[10] Akira Hirose and Shotaro Yoshida. Generalization characteristics of complex-valued feedforward neural networks in relation to signal coherence. *IEEE Transactions on Neural Networks and Learning Systems*, 23(4):541–551, 2012.

[11] Akira Hirose. Nature of complex number and complex-valued neural networks. *Frontiers of Electrical and Electronic Engineering in China*, 6(1):171–180, 2011.

[12] Guangji Shi, Maryam Modir Shanechi, and Parham Aarabi. Human Speech Recognition. *Language*, 14(5):1867–1874, 2006.

[13] Alan V. Oppenheim and Jae S. Lim. The Importance of Phase in Signals. *Proceedings of the IEEE*, 69(5):529–541, 1981.

[14] Joan Bruna and Stephane Mallat. Invariant Scattering Networks. 35(8):1872–1886, 2012.

[15] Joan Bruna, Soumith Chintala, Yann LeCun, Serkan Piantino, Arthur Szlam, and Mark Tygert. A mathematical motivation for complex-valued convolutional networks. (2014):1–11, 2015.

[16] David P. Reichert and Thomas Serre. Neuronal Synchrony in Complex-Valued Deep Networks. 2013.

[17] John Thickstun, Zaid Harchaoui, and Sham Kakade. Learning Features of Music from Scratch. pages 1–14, 2016.

[18] S. Buchholz and G Sommer. A hyperbolic multilayer perceptron. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium*, volume 2, pages 129–133 vol.2. IEEE, 2000.

[19] J.K. Pearson, D.L. Bisset, and IEEE. Neural networks in the Clifford domain. 3:1465–1469, 1994.

[20] S. Buchholz, G. Sommer, Zeevi Y.Y., and Sommer G. Learning geometric transformations with clifford neurons. 1888:144, 2000.

[21] S Buchholz. *A Theory of Neural Computation with Clifford Algebras*. PhD thesis, 2005.

[22] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. *Iclr*, pages 1–, 2014.

[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *Proceedings of the IEEE International Conference on Computer Vision*, 2015 Inter:1026–1034, 2015.

[24] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. WaveNet: A Generative Model for Raw Audio. pages 1–15, 2016.

[25] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 9:249–256, 2010.

[26] Dongpo Xu and Danilo P Mandic. Quaternion Gradient and Hessian. page 23, 2014.

[27] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[28] Alex Krizhevsky. Learning Multiple Layers of Features from Tiny Images. *... Science Department, University of Toronto, Tech. ...*, pages 1–60, 2009.

[29] GoogleResearch. TensorFlow: Large-scale machine learning on heterogeneous systems. 2015.

[30] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *Iclr*, pages 1–15, 2015.