Biologically Plausible Complex-Valued Neural Networks and Model Optimization

Ryan Yu[1], Andrew Wood[1], Sarel Cohen[3], Moshick Hershcovitch[2], Daniel Waddington[2], and Peter Chin[1]

[1] Boston University, Boston, MA 02215, USA
{ryu1, aewood, spchin}@bu.edu
[2] IBM Research
moshikh@il.ibm.com, daniel.waddington@ibm.com
[3] The Academic College of Tel Aviv-Yaffo, Israel
sarelco@mta.ac.il

Abstract. Artificial Neural Networks (ANNs) are thinly based on biological neural pathways. In an ANN, each node computes its activation by applying a non-linearity to a weighted sum of its inputs. While this formulation has been wildly successful for a variety of tasks, it is still a far cry from its biological counterpart, largely due to ANNs lack of phase information during computation. In this paper, we adapt ANNs to operate on complex values which naturally allows the inclusion of phase information during the forward pass. We demonstrate that our complex-valued architecture generally performs better compared to real-valued and other complex-valued networks in similar conditions. Additionally, we couple our model with a biologically inspired form of dimensionality reduction and present our findings on the MNIST and MusicNet data sets.

1 Introduction

Inspired by the brain, the first Artificial Neural Network (ANN), called the *Perceptron* was developed in 1961 [14]. The Perceptron, like a real neuron, computes its output as a function of its input. By studying real neurons, Rosenblatt designed the Perceptron to compute its output (called its *activation*) as a weighted sum of its inputs passed through a step function. Perceptrons were then trained by fitting the coefficients of the weighted sum as well as the bias to data. While Perceptrons performed well on simple tasks, they could not be stacked together which limited their usefulness for more complex tasks.

Through relaxing the step function with differentiable counterparts, the first modern ANN was created. The major benefit of using a differentiable nonlinearity was that ANN nodes were stack-able; solving the previous limitation of the Perceptrion. In general, ANNs are formulated as a graph of nodes with the weights and bias of each node as free parameters. The graph (often called a *computation graph*) is traditionally organized into layers: the nodes of a layer process the activations from nodes at the previous layer, while the first layer processes the input data.

ANNs have since risen in popularity and have produced state of the art results for several problems [2, 12, 18, 25, 26]. However, while new ANN architectures have been developed since the Perceptron, all architectures are still fundamentally based on the original 1960s biological approximation. While there has been work on updating ANNs with a more modern understanding of biology, one important observation from biological neurons that does not see explicit representation in their artificial counterparts is neural synchrony. Neural synchrony is a biological phenomenon where neurons learn to fire near-simultaneously. Different degrees of synchronization affect the output of the receiving neuron; highly synchronized input neurons will elicit a stronger response in their target compared to the same number of non-synchronized input neurons. Synchronization of artificial neurons can be represented and trained using complex-valued neural networks.

In this paper we extend the work of [13] and [20] and present a novel complex-valued ANN architecture inspired by neural synchrony. We demonstrate that our model performs better on two data sets, MNIST and MusicNet, compared to real-valued ANNs. Furthermore, we combine our model with a cortical-stem inspired preprocessing technique called Geometric Multi-Scale Resolution Analysis (GMRA) to improve performance by giving our model the underlying representation of data. We evaluate our performance on the MNIST [6] and MusicNet[19] data sets.

2 Background and Motivation

2.1 Neural Synchrony and Biological Neural Networks

Timing plays a crucial role in Biological Neural Networks (BNNs). A biological neuron accumulates positive charge in its body until it reaches an activation threshold, at which point it will produce an output. The amount of positive charge accumulated is a direct result of the input signals it receives from input neurons. The charge of a neuron will always approach its negative value resting potential; if weak positive stimuli is received, but not enough to cross the activation threshold, this positive charge is "leaked" out of the neuron over a short time period. Therefore the output of the neuron does not only depend on the intensity of the inputs, but also the degree of time-synchronization between the inputs. Different degrees of synchronization in the input will illicit different responses in the neuron [15]. An example can be seen in Figure 1.

The notion that groups of neurons fire together with respect to time, known as *neural synchrony*, is not a novel concept [15, 16, 21]. The phenomenon is posited to play a key role in biological information processing. Horn *et al* provides evidence that without neural synchrony, the brain creates less stable representation of audio stimulus. These fluctuations in representation may contribute to deficits in auditory brain-stem function by negatively impacting how neurons represent complex acoustic sounds. Furthermore, multiple studies found that the cells of certain brain regions were more likely to produce an output if the inputs to it were time synchronized rather than time dispersed [7, 17].

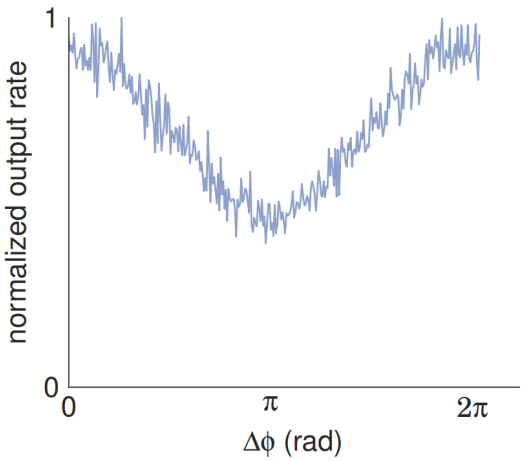

Fig. 1. Taken from [13] displays the normalized output rate of a biological neuron that is stimulated with two rhythmic spike trains as a function of the phase difference between the two stimuli.

2.2 Complex-Valued Neural Networks

Current ANNs operating on real values have no method of internally representing time. Time can be introduced to ANNs via complex numbers. A complex value z, can be defined in two forms. First, $F = x + iy$ where $x, y \in \mathbb{R}$ is the Cartesian form. In the Cartesian form, the real component of z is x, and the imaginary component of z is y. The polar form can be described as $F = (r, \phi)$ where $r, \phi \in \mathbb{R}$. In the polar form, r is known as the magnitude and ϕ is referred to as the phase offset. By treating r as the output strength of the activation and ϕ as the phase difference between neurons, Reichert *et al* created complex-valued artificial neurons that can process timing information of their inputs similar to biological cells. A visualization of the desired behavior which they were able to mimic via an ANN can be seen in Figure 1.

Complex-valued neural networks replace some or all of the (originally real-valued) model parameters with complex-valued ones. Fortunately, while there are special considerations that need to be taken with regards to initialization and normalization Trabelsi *et al* has shown that, with slight modifications, the same learning algorithms can be used to train and test complex-valued networks.

Several studies have already demonstrated the effectiveness of complex-valued neural networks [3, 4, 8, 20]. Gao *et al* demonstrate that complex-valued neural networks performed better than the rest of the strategies implemented for enhancing radar imaging [4]. Reichert *et al* proposed a biologically plausible deep network that constructs better data representations through complex values [13].

2.3 Geometric Multi-Scale Resolution Analysis (GMRA)

GMRA is a dimensionality reduction technique that is inspired by the cortex. At the microscale, neurons in the cell fire, which induces neural synchrony. Neural synchrony at the macroscale produces patterns, which, when combined with other firings, have been long believed to be the intermediary representation of data [15, 16, 21, 5]. These patterns are described by not only which neurons are firing (a subset of the population), but the activity of the firings as well, meaning that macroscale neural synchrony is a lower-dimensional representation of the data processed at the microscale. Therefore, it is believed that the cortex finds a lower-dimensional representation of data by producing increasingly abstract representations as a function of scale. GMRA mimics this behavior by processing a point cloud at different scales to produce increasingly fine-grained manifolds.

The GMRA algorithm contains three steps to compute manifolds at different scales:

1. It computes a leveled tree decomposition of the manifold into dyadic cells. Dyadic cells have the following properties:
   (a) Each dyadic cell contains a subset of the points that exist within a sphere of fixed radius.
   (b) The children of a dyadic cell contain disjoint subsets of the points contained inside the parent.
   (c) The children of a dyadic cell cover the points inside the parent.
2. It computes a d-dimensional affine approximation (i.e. linear approximations) for each dyadic cell. This approximation represents the basis of each dyadic cell and is a linear piecewise approximation (i.e. the SVD decomposition of the cell's covariance).
3. It computes a sequence of low-dimensonal affine difference operators that encode the difference between subsequent levels of the tree (i.e. scales). These difference operators allow efficient querying of the points by scale as well as projection of new points.

As seen in Figure 2, the linear approximation, called the *scaling function* fit to each group can be queried by starting at scale 0 (the roughest scale), getting the approximation for the query at that scale, and then applying the difference operator (*wavelet correction*) to get the approximation at the next scale (finer scale). Since each level of the tree represents the decomposition of the point cloud at a scale, by walking from the root to the child at the appropriate level, GMRA produces low-dimensional embeddings for each point at arbitrary scales.

By using GMRA as a preprocessing technique, we provide our complex-valued ANNs with low-dimensional representations of the data which are tailored to the embedded manifold.

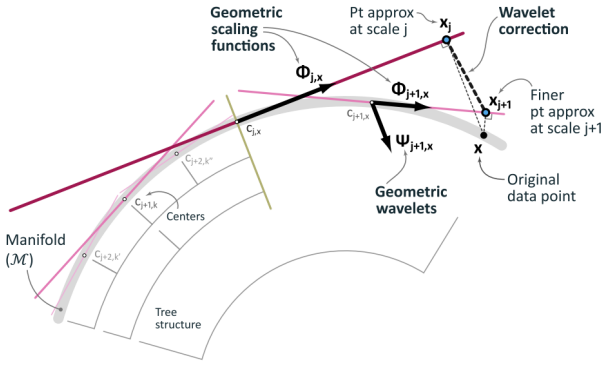

Fig. 2. A visualization of the linear approximation of a point using a basis function at scale j, the approximation at a finer scale $j+1$ and difference operator (geometric wavelet) from scale j to scale $j+1$. This image was taken from [1].

3 Methodology

3.1 Complex-Valued Model

Our model builds on the work from Reichert *et al*; they define the forward pass of their complex-valued ANN layer using the following equations [13]:

$$t_1 = (|X| \times W) \tag{1}$$

$$t_2 = |X \times W| \tag{2}$$

$$O = \mathcal{F}(r_1 * t_1 + r_2 * t_2) \tag{3}$$

Where $X \in \mathbb{C}^{N \times M}$, $W \in \mathbb{R}^{M \times P}$, N is the batch size, M is the dimensionality of the data, and P is the dimensionality of the output of the layer. O is a $N \times P$ output of the layer and is derived via a weighted combination (with coefficients r_1, r_2 typically set to 0.5 each) passed through non-linearity $\mathcal{F}$.

The order of the operations in Expression (1) produces values which are invariant to any phase offsets between inputs and their respective weights; the magnitude of values with opposing phase values are not cancelled, but summed. Expression (2) is the opposite: by performing $X \times W$, any vectors with opposing phases conflict with each other in the multiplication, meaning that t_2 contains the timing information between inputs while t_1 contains the magnitude of the transformed inputs. By sharing weight matrices between the two operations, Reichert *et al*'s model learns how to process the timing between input firings as well as the magnitude of the inputs firings. With this formulation, Reichert *et al* were able to reproduce the observations in Figure 1 using their ANN.

Our model attempts to improve this idea. While Reichert *et al* define their input to be complex-valued and their free parameters to be real-valued, we define the opposite: $X \in \mathbb{R}^{N \times M}, W \in \mathbb{C}^{M \times P}$, and a bias term $b \in \mathbb{R}^{P \times 1}$. Our forward pass, while different, follows the same spirit:

$$t_1 = (X \times |W|) \tag{4}$$
$$t_2 = |X \times W| \tag{5}$$
$$O = \mathcal{F}(r_1 * t_1 + r_2 * t_2 + b) \tag{6}$$

In Reichert *et al*'s model, each artificial neuron had a unique phase value which is shared amongst all its connections. This models a biological setting where a neuron sends a signal, and it is received by all recipients at the same time. However, a stimulus from a neuron sent to multiple destination neurons simultaneously may not trigger simultaneous responses from the destination neurons. This discrepancy can be caused by a variety of subtle differences, such as difference in distance the signal must traverse or differences in connection strength between sender and receiver. In our formulation, each edge is given the potential for a unique phase value. We believe that this model more accurately captures biology where synaptic processing is nontrivial. Our model is able to express these realistic discrepancies while Reichert *et al*'s mode cannot. The individual phase values for each artificial synapse is learnt separately via gradient descent.

Additionally, we incorporated layer regularization in our complex-valued models. Regularization is the synthetic counterpart to the biological trait of thresholding. After the neuron fires, it enters a refractory period, which makes it extremely difficult for the neuron to fire again for a short duration, thus limiting the maximum number of times a neuron is able to fire over a period of time. Both of these features can be replicated in an artificial setting by increasing the threshold value of the activation function, and by using bounded activation functions or activity regularizers. We tested with two forms of regularization: batch normalization, and layer regularization. We found that while layer normalization has a net positive effect on our complex ANN, batch normalization destabilizes the model by almost certainly pushing the ANN into a state of either vanishing or exploding gradient. Therefore, our models used layer normalization between every feed forward layer in both complex and real-valued variants.

Lastly, by using real-valued layer outputs and complex-valued weights, we can bypass some of the uncertainties when applying our model to naturally real-valued data. Most complex-valued network studies find success on naturally complex-valued data [2, 20], or mixed results on transforming naturally real-valued data into the complex plane [9, 10].

3.2 Data sets

We used two data sets, MNIST [6] and MusicNet [19]. A breakdown of the data sets (preprocessed and in original form) is shown Table 1.

Interestingly, we were able to reduce MNIST into a 11-dimensional representation using GMRA. This is encouraging since there are only 10 labels in MNIST. The 11-dimensional representation was chosen from the roughest scale, which we selected after comparing recovered images against the original representation, which we found to be subjectively adequate. MNIST served as our

Table 1. A breakdown of the data sets, preprocessed (GMRA-*) and no tag to represent the original data sets.

Name	# examples	dimensionality	# labels
MNIST	70k	784	10
GMRA-MNIST	70k	11	10
MusicNet	> 6000000	4096	84
GMRA-MusicNet	327887	163	84

first proof of concept data set for both high dimensional input (MNIST) and low dimensional input (GMRA-MNIST).

MusicNet is a data set created by [19]. The data set is composed of recordings of classical music from a variety of instruments, such as piano and violin, and a varying number of performers per recording (e.g. solo, duet, quartet, etc). There are 330 musical excerpts each ranging from one to three minutes in length. All audio files were re-sampled to 11khz. Three music files, with IDs ['2303', '2383', '1819'] were reserved as the test subset, and all other files were randomly split for training and validation during each experiment.

The task associated with this data set is automatic music transcription: the transcription of audio to musical score. To accomplish this, the learning model does not process the entire audio file at once, but instead processes overlapping windows of size 4096. The output of the model is a binary vector of length 84, where a 1 indicates at index k indicates that note k was present at the midpoint of the 4096 length audio clip.

To obtain the GMRA coefficients of MusicNet, we converted the MusicNet data set into a point cloud using the approach mentioned above; however we had trouble deciding an appropriate setting for the stride. With a stride of 1 we generated over 6 million 4096 floating point vectors and quickly ran into a hardware bottleneck. We therefore used the technique of Wood *et al* who implemented the GMRA algorithm with a new python package called PyMM [24, 23]. PyMM, short for Python Micro MCAS is a python wrapper for a larger library called MCAS (Memory Centric Active Storage) [22]; middleware that provides an interface between applications and Non-Volatile Memory. Non-Volatile Memory is new hardware which occupies memory slots (instead of RAM) and boasts orders of magnitude higher capacity than RAM while running at a third of the speed. As explained by Wood *et al*, by combining GMRA with PyMM, we can process point clouds significantly larger than with ordinary machines at the cost of run time. For this reason, we settled on a stride of 4096 and generated 327,887 non-overlapping points. These 327,887 points represent approximately a 5% subset of the original data.

In total, the GMRA algorithm took 8 days to process this point cloud and consumed in excess of 500GB of Non-Volatile Memory. GMRA produced representations at 36 different scales, of which we used the roughest scale as it produced an acceptable reconstruction error, and was also one of the only scales which all the points share the same dimensionality (i.e. different dyadic cells can

have different dimensionalities). At the roughest scale, the MusicNet data was reduced from 4096 dimensions to 163 dimensions, a 25x reduction.

3.3 Experiments

Our experiments consisted of training real-valued and complex-valued ANNs on each data set using a cross validation training scheme, and compared their performance averaged over the cross validation splitting. The data set is randomly shuffled and split between training, testing, and validation with percentages of 80%, 10%, and 10% respectively. For MNIST our uniform shuffling procedure roughly preserves the class balancing of the splits, we measured the performance of each model via its accuracy. For MusicNet there were predetermined data points deliberately set aside for validation and testing, as was done in [19, 20]. When training a model, we used the Early Stopping [11] mechanism. Our early stopping window was set to three.

MNIST In our experiments on MNIST, we varied the number of nodes inside a hidden layer. For the MNIST data set, we trained fully connected shallow real-valued ANNs in parallel with complex-valued network counterparts. Our MNIST experiments serve as a proof of concept for both high dimensional data (MNIST) and very low dimensional data (GMRA-MNIST). The data set was selected due to its accessibility, and training speed.

From these experiments we discovered the following limitation of our model: not all operations with complex values are yet supported on the GPU during back propagation. As a result, the training time increases drastically as you scale the network in terms of depth and size per layer. While the limits on the size of our network prevent it from being used in lieu of deep real-valued neural networks, we wanted to compare the performance of our architecture to real-valued architectures of the same size in the context of smaller networks and dimensionality reduction.

MusicNet There are three main goals for our experiments in MusicNet. First is to explore the performance of our complex-valued layers compared to real-valued layers in a setting that is not single class classification. In MusicNet, multiple notes may be played at the same time, compared to MNIST in which every image belongs only to one class. Second, to explore performance on an inputs other than images. Lastly, to assess our models in conjunction with a dimensionality reduction technique (GMRA) applied to a challenging data set.

To these ends, two sets of experiments were repeated. The first set of experiments used the original MusicNet data points with each input point as a length 4096 vector. Four networks were trained on the original MusicNet files. A complex-valued, shallow classifier with a single hidden layer of dimension 2048 and a real-valued shallow classifier of the same dimensions were trained in parallel and assessed. Then, a network with a single real-valued convolutional layer and complex-valued classifier was trained in parallel with a completely real-valued network of the same specifications.

Our second set of experiments looked at very small shallow networks in conjunction with GMRA reduction on MusicNet. Each network, real-valued and complex-valued, was comprised of a single hidden layer of only 200 nodes. These experiments used a 5% subset of the MusicNet that was converted into GMRA coefficients. As previously stated, the decision to use a small fraction of the original data set was made in order to accommodate for the time constraint associated with GMRA processing.

4 Results and Discussion

4.1 MNIST

Two conclusions can be drawn from our experiments on MNIST. First, our complex-valued model has the potential to out perform real-valued models of the same size. As can be seen in Figure 3, our complex ANN significantly outperformed its real-valued counterpart for every configuration of a three-layer ANN on both MNIST data sets.

Second, GMRA in conjunction with shallow models can lead to very strong performance with the added benefit of a significantly smaller number of parameters, and a significantly faster training speed. As was stated in 1, GMRA-MNIST reduced each point from a 784 length vector, to an 11 point vector. Despite this, both real and complex-valued networks are still able to perform at a high level, as shown in Figure 3.

Note that performance on GMRA-MNIST is around 10% worse than that of MNIST. We believe that this is an artifact of using the roughest scale embeddings from GMRA. While these embeddings, when recovered into the original image space, were satisfactory to our subjective evaluation, GMRA does induce signal degradation. By using the roughest scale, we invite the accompanying signal loss to influence model performance. Despite using the roughest scale, both models are able to achieve around 90% accuracy with a short early stopping window. On MNIST, our model is statistically better than the corresponding real-valued ANN with 99.9% confidence, and overlaps significantly with its real-valued counterpart on GMRA-MNIST.

4.2 MusicNet

Original MusicNet We began our experiments for the original MusicNet data set by first training two shallow classifiers using the same training methods (including data split) laid out by [20] and [19]. We found that for the original data set, our shallow complex-valued classifier achieved a significantly higher AP over repeated trials (Table 2.

In order to compare our method with previous shallow networks on MusicNet [19, 20], we added a single real-valued convolutional layer to our model. We found that the addition of a single convolutional layer improved Average Precision (AP) for both real and complex-valued classifiers. It remained consistent that the complex-valued classifier out performed the real-valued classifier with the addition of a single convolutional layer (Table 2).

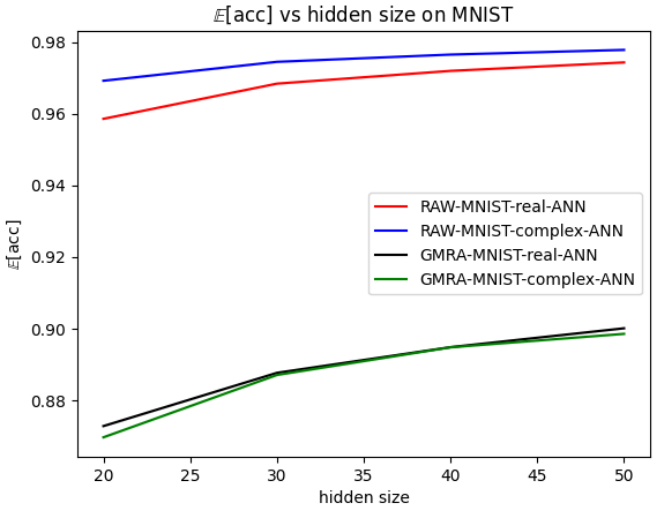

Fig. 3. Expected accuracy averaged over the trials for a given hidden layer size on both MNIST data sets. Note that this image actually includes error bars to show variances, however the variances are on the order of $1e^{-6}$; they cannot be seen without extreme magnification.

We also found that our shallow complex-valued network preformed significantly better than the shallow complex-valued and shallow real-valued network reported in [20]. Our shallow complex-valued classifier with a convolutional layer achieved a 67.5% AP compared to the 66.0 % AP from a complex-valued model of the same dimensions previously reported (Table 2).

A sample reconstruction from our best model ("Shallow Complex w Conv" on Table 2) is presented in Figure 4 as well as the original ground truth data. By comparing the two figures, we can visually match every note in the ground truth to a counterpart in the reconstruction. The main discrepancies appear to be that the presence of small note artifacts in the reconstruction, where there

Table 2. The breakdown of our experiments and testing Average Precision (AP) for each network architecture on both original and GMRA MusicNet data sets. All fully connected classifiers had two layers: a hidden layer of size 2048 and an output layer of size 84. The results cited from [20] use the same network dimensions as "Conv + Fully Connected". For GMRA MusicNet experiments, the hidden layer used 200 nodes instead of 2048.

	Real-Valued	Complex-Valued
Original Data, Fully Connected	56 AP %	**64.0 AP %**
Original Data, Conv + Fully Connected	65.8 AP %	**67.5 AP %**
Original Data, [20]	66.1 AP %	66.0 AP %
GMRA Data, Fully Connected	42.2 AP %	**46.1 AP %**

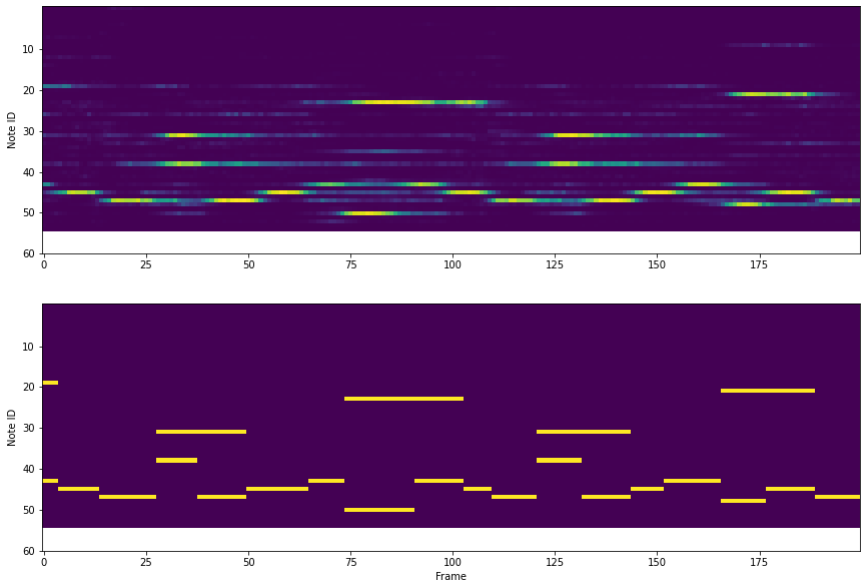

Fig. 4. A sample transcription from our complex-valued shallow model with a convolutional layer compared to the true transcription. (Above) The output of our model for frames 0 to 200, and note IDs 10 to 65. (Below) The actual transcript corresponding to the same frame and notes as above.

should be silence, and increased note duration in the reconstruction compared to the truth.

GMRA MusicNet Our second set of MusicNet experiments use the GMRA coefficients of the subset of MusicNet. For both real-valued and complex-valued classifiers, the AP of real-valued and complex-valued networks decreased by 14 and 18, respectively, when compared to the same architecture trained on the original data points. The decrease in performance can be attributed to two factors. First, using GMRA can result in some information loss; this is seen clearly by the experiments on GMRA-MNIST. The second factor is the difference in training set size. Due to the computational costs of GMRA, this study uses a subset of the MusicNet data set for both raw and GMRA experiments. The subset is comprised of over 327,000 data points and labels. It is approximately 5% of the total number of data points. Despite the large decrease in the number of data points and the dimension reduction of each point from a 4096 length vector to a 163 length vector, both real and complex-valued networks trained on GMRA MusicNet were able to reach over 40 AP%.

Comparing performance between real-valued and complex-valued networks, it remains consistent that complex-valued networks performed significantly better. An interesting observation was found during our experiments: there were multiple instances where the real-valued classifier failed to converge to a meaningful solution. The AP reported in Table 2 for real-valued networks trained on

GMRA coefficients was computed for instances where the real-valued network did not fail. Failure to converge was determined by observing the validation AP over several epochs. Instances where validation AP was significantly lower, less than a third, of other experiments, and the validation AP did not improve after 10 epochs, were deemed a failure. Because all validation AP that were categorized as normal did not express high variance, identifying failure experiments was straight-forward. Complex-valued networks on the same data set never failed to converge. This could suggest increased robustness for complex-valued models in terms of their application to dimension reduction problems.

5 Threats to Validity

While our experiments show an improvement between complex and real-valued neural networks, there are a few limiting factors. We recognize that our early stopping window is small, and that with a larger window, real-valued networks might improve. However, there are a few reasons as to why we do not expect a larger window to change the trend.

First, complex ANNs have twice the number of parameters as real-valued ANNs with the same number of nodes. Therefore, we expect that be increasing the window, complex ANNs would benefit more than real-valued networks due to saturating the larger number of parameters. We confirmed this behavior by performing preliminary experiments with a larger window. With an early stopping window size of 10, we observed complex ANNs achieve an expected boost of 5% accuracy on GMRA-MNIST, and 5% on MNIST. On the same data sets with the same increased window, we observed that real-valued ANNs did not improve on average.

Second, since our complex ANNs have twice the number of parameters as real-valued ANNs, our experiments might be viewed as an apples to oranges comparison. However, our complex ANN, despite having twice the number of parameters, first can only express half of the parameter values as its real-valued counterparts, since complex numbers take up the same number of bytes as real values but split those bytes between real and imaginary components. Second, we ran preliminary experiments where we doubled the precision in a real-valued ANN (from 32 to 64 bits per value) and found that this model still performed worse than our complex-valued ANN of the same precision.

6 Conclusion and Future Work

In conclusion, we introduce a novel complex-valued network architecture that has complex-valued weights but has real-valued layer outputs. In order to demonstrate how well our complex-valued models performed, we assessed its metrics on two data sets, MNIST and MusicNet. In addition, we tested the dimension reduction technique, GMRA, used in conjunction with real and complex-valued networks. GMRA was able to greatly reduce the input space, from 784 to 11 in MNIST and 4096 to 163 in MusicNet. This significantly reduced training time and model size. In both raw and GMRA MNIST and MusicNet, our shallow complex-valued network significantly outperformed a shallow real-valued network of the same parameter count.

Finally, we offer a few extensions to improve our proposed model. The first extension is the application our complex-valued dense architecture to any existing neural network and any data set by replacing the existing network's real-valued classification layer(s) . Currently this is not practical for all deep networks. Because certain operations in our implementation lack GPU support, training large networks with the proposed dense layer incurs a large time penalty. An implementation of our architecture which uses multiple real valued weight matrices and GPU supported functions to simulate the behaviour of a complex weight matrix is ideal.

The second extension is to study the results of making complex-valued architecture more similar to our current understanding of biological networks. Biological neurons function in an all-or-nothing manner. If it receives an input above its activation threshold, the neurons fire, otherwise it will not. Artificial neurons will output the sum of its inputs assuming the sum is positive. An implementation of activation thresholding such that a neuron would produce positive output only if it receives multiple strong positive inputs would bring artificial neurons more in line with our current understanding of biology. It would be interesting to observe such an implementation in conjunction with our proposed complex-valued architecture.

References

[1] William K Allard, Guangliang Chen, and Mauro Maggioni. "Multi-scale geometric methods for data sets II: Geometric multi-resolution analysis". In: *Applied and Computational Harmonic Analysis* 32.3 (2012), pp. 435–462.

[2] Elizabeth Cole et al. "Analysis of deep complex-valued convolutional neural networks for MRI reconstruction and phase-focused applications". In: *Magnetic Resonance in Medicine* 86 (2021). URL: https://onlinelibrary.wiley.com/doi/10.1002/mrm.28733.

[3] Jesper Sören Dramsch, Mikael Lüthje, and Anders Nymark Christensen. "Complex-valued neural networks for machine learning on non-stationary physical data". In: *Computers Geosciences* 146 (Jan. 2021), p. 104643. ISSN: 0098-3004. DOI: 10.1016/j.cageo.2020.104643. URL: http://dx.doi.org/10.1016/j.cageo.2020.104643.

[4] J. Gao et al. "Enhanced Radar Imaging Using a Complex-Valued Convolutional Neural Network". In: *IEEE Geoscience and Remote Sensing Letters* 16.1 (2019), pp. 35–39. DOI: 10.1109/LGRS.2018.2866567.

[5] Jane Hornickel and Nina Kraus. "Unstable Representation of Sound: A Biological Marker of Dyslexia". In: *Journal of Neuroscience* 33.8 (2013), pp. 3500–3504. ISSN: 0270-6474. DOI: 10.1523/JNEUROSCI.4205-12.2013. eprint: https://www.jneurosci.org/content/33/8/3500.full.pdf. URL: https://www.jneurosci.org/content/33/8/3500.

[6] Yann LeCun. "The MNIST database of handwritten digits". In: *http://yann.lecun. com/exdb/mnist/* (1998).

[7] Michikazu Matsumura et al. "Synaptic Interactions between Primate Precentral Cortex Neurons Revealed by Spike-Triggered Averaging of Intracellular Membrane Potentials In Vivo". In: *Journal of Neuroscience* 16.23 (1996), pp. 7757–7767. ISSN: 0270-6474. DOI: 10.1523/JNEUROSCI.16-23-07757.1996. eprint:

https://www.jneurosci.org/content/16/23/7757.full.pdf. URL: https://www.jneurosci.org/content/16/23/7757.

[8] Nils Mönning and Suresh Manandhar. "Evaluation of Complex-Valued Neural Networks on Real-Valued Classification Tasks". In: (2018). arXiv: 1811.12351 [cs.LG].

[9] Nils Mönning and Suresh Manandhar. "Evaluation of Complex-Valued Neural Networks on Real-Valued Classification Tasks". In: *CoRR* abs/1811.12351 (2018). arXiv: 1811.12351. URL: http://arxiv.org/abs/1811.12351.

[10] Călin-Adrian Popa. "Complex-valued convolutional neural networks for real-valued image classification". In: (2017), pp. 816–822. DOI: 10.1109/IJCNN.2017.7965936.

[11] Lutz Prechelt. "Early stopping-but when?" In: (1998), pp. 55–69.

[12] Hendrik Purwins et al. "Deep Learning for Audio Signal Processing". In: *CoRR* abs/1905.00078 (2019). arXiv: 1905.00078. URL: http://arxiv.org/abs/1905.00078.

[13] David P. Reichert and Thomas Serre. "Neuronal Synchrony in Complex-Valued Deep Networks". In: (2014). arXiv: 1312.6115 [stat.ML].

[14] Frank Rosenblatt. "Perceptron Simulation Experiments". In: *Proceedings of the IRE* 48.3 (1960), pp. 301–309. DOI: 10.1109/JRPROC.1960.287598.

[15] Wolf Singer. "Neural Synchrony: A Versatile Code for the Definition of Relations?" In: *Neuron* 24 (1 1999).

[16] Garrett B Stanley. "Reading and writing the neural code". In: *Nature Neuroscience* (2013). URL: https://www.nature.com/articles/nn.3330.

[17] Charles Stevens and Anthony Zador. "Novel Integrate-and-fire-like Model of Repetitive Firing in Cortical Neurons". In: (July 1998).

[18] Mingxing Tan and Quoc V. Le. "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks". In: *CoRR* abs/1905.11946 (2019). arXiv: 1905.11946. URL: http://arxiv.org/abs/1905.11946.

[19] John Thickstun, Zaid Harchaoui, and Sham Kakade. "Learning Features of Music from Scratch". In: (2017). arXiv: 1611.09827 [stat.ML].

[20] Chiheb Trabelsi et al. "Deep Complex Networks". In: *CoRR* abs/1705.09792 (2017). arXiv: 1705.09792. URL: http://arxiv.org/abs/1705.09792.

[21] Peter J. Uhlhaas et al. "Neural Synchrony in the Cortical Networks: History, Concept and Current Status". In: *Frontiers in Integrative Neuroscience* 3 (2009).

[22] Daniel Waddington et al. "An architecture for memory centric active storage (MCAS)". In: *arXiv preprint arXiv:2103.00007* (2021).

[23] Daniel G. Waddington, Moshik Hershcovitch, and Clem Dickey. "PyMM: Heterogeneous Memory Programming for Python Data Science". In: *PLOS '21: Proceedings of the 11th Workshop on Programming Languages and Operating Systems, Virtual Event, Germany, October 25, 2021*. ACM, 2021, pp. 31–37. DOI: 10.1145/3477113.3487266. URL: https://doi.org/10.1145/3477113.3487266.

[24] Andrew Wood et al. "Non-volatile memory accelerated geometric multi-scale resolution analysis". In: *2021 IEEE High Performance Extreme Computing Conference (HPEC)*. IEEE. 2021, pp. 1–7.

[25] Lonce Wyse. "Audio spectrogram representations for processing with Convolutional Neural Networks". In: *Workshop on Deep Learning for Music* (2017).

[26] Yong Xu et al. "Attention and Localization based on a Deep Convolutional Recurrent Model for Weakly Supervised Audio Tagging". In: *CoRR* abs/1703.06052 (2017). arXiv: 1703.06052. URL: http://arxiv.org/abs/1703.06052.