Improving Axial-Attention Network via Cross-Channel Weight Sharing

Nazmul Shahadat
Truman State University
Missouri, USA

Anthony S. Maida
University of Louisiana at Lafayette
Lafayette, Louisiana, USA

Abstract

In recent years, Hypercomplex-inspired neural networks improved deep CNN architectures due to their ability to share weights across input channels and thus improve cohesiveness of representations within the layers. The work described herein studies the effect of replacing existing layers in an Axial Attention ResNet with their quaternion variants that use cross-channel weight sharing to assess the effect on image classification. We expect the quaternion enhancements to produce improved feature maps with more interlinked representations. We experiment with the stem of the network, the bottleneck layer, and the fully connected backend, by replacing them with quaternion versions. These modifications lead to novel architectures which yield improved accuracy performance on the ImageNet300k classification dataset. Our baseline networks for comparison were the original real-valued ResNet, the original quaternion-valued ResNet, and the Axial Attention ResNet. Since improvement was observed regardless of which part of the network was modified, there is a promise that this technique may be generally useful in improving classification accuracy for a large class of networks.

Introduction

This work studies the effect of adding representationally coherent layers to axial-attention networks to improve image classification accuracy. In this work, a representationally coherent layer means a layer with the ability to discover and represent cross-channel correlations in its inputs. For a CNN, this leads to feature maps with improved representational properties. This functionality is implemented using calculations inspired by hypercomplex number systems such as quaternions (Gaudet and Maida 2018), vectormaps (Gaudet and Maida 2021), and parameterized hypercomplex multiplication (PHM) (Zhang et al. 2021). In this work, we study whether hypercomplex augmentation can improve classification performance in axial-attention networks, which have been shown to outperform CNNs (Wang et al. 2020).

A core example illustrating the utility of representational coherence comes from CNN auto-encoders, which can be trained to reconstruct color from grayscale input. Parcollet et al. showed that the channel-based weight-sharing property of a quaternion-valued auto-encoder allowed the discovery of input correlations that supported the reconstruction of color from grayscale images, and this was not possible using a real-valued auto-encoder (Parcollet, Morchid, and Linarés 2019). This result was subsequently confirmed using a vectormap auto-encoder that used a similar form of weight sharing (Gaudet and Maida 2021).

We hypothesize that this improved ability to capture relationships between input channels may apply to other multidimensional input modalities besides color. It may depend on whether the input data contains interwoven cross-channel relationships to capture. We will say that layers that use this form of hypercomplex-inspired weight sharing are *representationally coherent*.

Our experiments studied novel axial-attention networks (Ho et al. 2019; Wang et al. 2020) by replacing existing layers with representationally coherent layers in three parts of the network. These modifications included:

1. Enhance axial bottleneck blocks to use quaternion modules to increase their representational capacity.
2. Replace the fully connected, real-valued backend with a 4D parameterized hypercomplex multiplication layer.
3. Replace the real-valued convolutional stem with a quaternion-valued convolutional stem.

We varied the depths of the original axial attention networks by using 26, 35, and 50 layers. In addition to the axial attention networks, we compared our results to baseline real-valued, and quaternion-valued ResNets. We found that adding representationally coherent layers improved performance over the standard ResNets, quaternion ResNets, and standard axial-attention ResNets. The most notable improvement occurred by adding representational coherence to the backend in the form of a 4D PHM layer.

Rationale for the Proposed Method

The main hypothesis of this paper is that multichannel feature map representations that are used as input to attention modules can be modified to improve their effectiveness. Our approach uses the output of quaternion modules to provide improved input representations to the attention modules. The rationale follows.

Parcollet et al. showed that a quaternion-valued, autoencoder can be trained to reconstruct color from grayscale input images, whereas a real-valued autoencoder cannot (Parcollet, Morchid, and Linarés 2019). Thus, a trained quaternion-valued layer generates a richer representation because it allows implicit relationships among the color channels of data and these relationships cannot be captured in a comparable real-valued auto-encoder. Parcollet et al., attribute this functionality to a weight-sharing property found in the Hamiltonian product (quaternion multiplication) that is used to implement convolution (Parcollet, Morchid, and Linarés 2019). This does not occur in real-valued networks. The representation produced by the quaternion network captures more information about the interrelationships between the color input channels, and in this sense, we can say that it is a richer *interwoven* or *interlinked representation* of the information contained in the input channels.

We hypothesize that this improved ability to capture relationships between input channels is likely to apply to other multidimensional inputs besides color (Gaudet and Maida 2020). It may depend on whether the input data happens to contain *interwoven cross-channel relationships* of this kind to capture. In the remainder of this paper, we use the term *interwoven/interlinked representation* or *feature map* to refer to the output of a quaternion layer.

Background and Related Work

Quaternion Convolution

A useful introduction to quaternion algebra and neural networks appears in (Parcollet, Morchid, and Linarés 2020). Trabelsi et al. extended the principles of convolution, batch normalization, and weight initialization from real-valued to complex-valued networks (Trabelsi et al. 2018). Gaudet and Maida, in turn, extended these principles to QCNNs (Gaudet and Maida 2018). For implementation purposes, quaternion calculations can be decomposed into operations on 4-tuples of real numbers. The take-home message from this is that a quaternion convolution module must accept four input channels. A quaternion layer can accept more than four input channels, say m, as long as m is a multiple of four. In this case, the layer must hold $m/4$ separate QCNN modules, each with its own weight sets.

Parcollet et al. analyzed the Hamilton product, where the Hamilton product of I and Q is $O_q = I_q \circledast W_q$, where the subscripts q indicate that I, W, and Q are quaternion numbers. When expanded into real-valued 4-tuples, this can be viewed as a linear mapping from a neural layer of four units, representing I_q, to another layer of four units representing O_q. However, instead of using 16 independent weights to connect the layers, the four weights in W_q are repeatedly substituted within the 16 weights, so the mapping only uses four independent weights. This weight-sharing forces the model to learn cross-channel interrelationships in the data. More recently, Gaudet and Maida showed the properties of the quaternion convolution were due entirely to weight sharing and had no dependence on the quaternion algebra (Gaudet and Maida 2020; 2021). The quaternion convolutions in our model are 1x1, so they can be interpreted as fully connected layers with shared weights.

PHM Layers

Parameterized hypercomplex multiplication (Zhang et al. 2021) is used for FC layers. The 4D PHM layer works similarly to the quaternion layers. The Kronecker product is used to construct the parameter matrix. The PHM-based fully connected hypercomplex transformation, which transforms the input $\mathbf{x} \in \mathbb{R}^d$ into an output $\mathbf{y} \in \mathbb{R}^k$, is defined as $y = Hx + b$, where $H \in \mathbb{R}^{k \times d}$ represents the PHM layer. H is calculated as $H = \sum_{i=1}^{n} \mathbf{I}_i \otimes \mathbf{A}_i$ where $\mathbf{I}_i \in \mathbb{R}^{n \times n}$ and $\mathbf{A}_i \in \mathbb{R}^{k/n \times d/n}$ are parameter matrices and $i = 1 \ldots n$ ($n = 4$). Parameter reduction comes from reusing matrices $\mathbf{I}$ and $\mathbf{A}$ in the PHM layer. The $\otimes$ is the Kronecker product. For 4D, the inputs are split as $Q_{in} = Q_r + Q_x + Q_y + Q_z$ and the outputs are merged into Q_{out} as $Q_{out} = Q_{ro} + Q_{xo} + Q_{yo} + Q_{zo}$ as seen in Figure 2. The 4D hypercomplex parameter matrix (Zhang et al. 2021) expresses the Hamiltonian product for 4D by preserving all PHM layer properties.

Axial-Attention Networks

As stated earlier, attention networks consisting of stacked attention modules can learn to outperform deep CNNs and hybrid CNN/attention models for image classification. This stems from their ability to detect affinities between pixels having large spatial separation in the image. The drawback of using attention is that it is impractically computationally expensive, consuming $O(N^2)$ resources for an image of length N (using a flattened pixel set) and using a global window to compare any pair of pixels in the image. For a 2D image of height, h, and width, w, where $N = hw$, the cost is $O((hw)^2) = O(h^2w^2)$ to detect similarities for any pair of pixels in the image (Ho et al. 2019; Wang et al. 2020).

Axial-attention networks reduce the cost of computing attention by decomposing the problem into consecutive 1D operations. They were introduced in (Ho et al. 2019) for generative modeling in auto-regressive models and use the assumption that images are approximately square so that h and w are both much less than the total pixel count, hw. For simplicity, assume a square 2D image where $h = w$, so $w^2 = N$. Axial attention only operates on one dimension at a time. It is first applied to, say, the h axis and then to the w axis. When applied to the h axis, then $h = w$ attention calculations are applied to a 1D region of length h. Axial attention applied to the columns, h, performs w self-attention operations on each column whose total cost is $O(h \ h^2) = O(\sqrt{N} \ N)$. This bound is the same for using an axial column module followed by an axial row module.

Wang et al. incorporated axial attention into a ResNet architecture (He et al. 2016) to develop the AANet, which reduced the computational requirements (described above) of the original attention networks (Ramachandran et al. 2019; Hu et al. 2019) used for image classification (Wang et al. 2020). The conversion from ResNet to Axial-ResNet was based entirely on modifying the bottleneck blocks within ResNet. Figure 1(a) shows the bottleneck block used in the

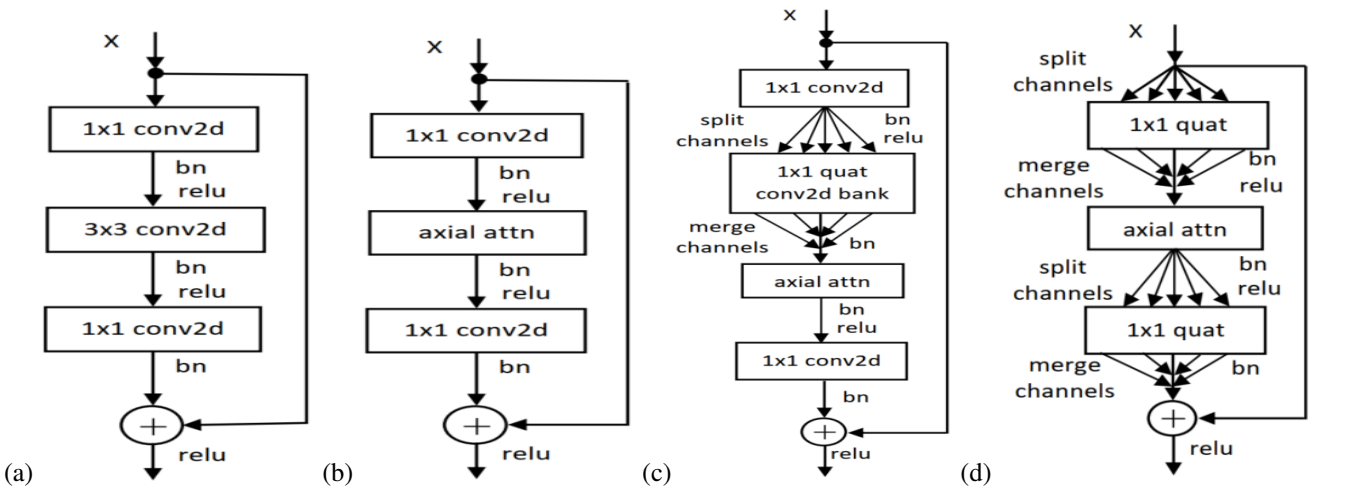

Figure 1: Bottleneck types. “bn”, “attn”, and “quat” stand for batch normalization, attention, and quaternion, respectively. (a and b) Original bottleneck modules found in ResNet (He et al. 2016), and Axial-ResNet (Wang et al. 2020), respectively. (c) Our quaternion-enhanced (QuatE) axial-ResNet bottleneck block. (d) Our representational Axial-ResNet block (RepAA).

original convolution-based ResNet. Figure 1(b) shows the axial-bottleneck block used in Axial-ResNet (Wang et al. 2020). The 3x3 2D convolution used in the original ResNet is replaced by an axial attention module.

Our Models: Axial-Attention with Quaternion

Our models are based on the axial-attention model used in (Wang et al. 2020) described above, but modified in the three ways described in the introduction. Our first modification enhances the bottleneck portion of AANet in two alternative ways, seen in Figure 1(c) and (d). The original is shown in Figure 1(b).

Figure 1(c) shows our first modification to the axial-bottleneck blocks. The original is shown in Figure 1(b). It inserts a bank of 1 1 quaternion conv2d modules in front of the axial-attention module. In addition to this, it is possible to add a quaternion front end to the axial attention module. The purpose of the front end is to generate potentially more useful interlinked input representations for the bottleneck portion of the network. The number of input channels to the quaternion bank must be a multiple of four. Hence, the output channels of the top 1 1 conv2d module are split into groups of four. One quaternion 2D convolution is applied to each group of four channels. Each quaternion convolution accepts four input channels and produces four output channels. Thus, the weight-sharing is compartmentalized into groups of four input channels. The set of output channels is merged (stacked) so that the number of input channels to the axial-attention module is unchanged from the original axial-attention block. In all other ways, the structure of the bottleneck portion of our quaternion-modified model is identical to the AANet model.

Figure 1(d) shows our second and final modification to the axial-bottleneck block. We redesign the axial-bottleneck block by removing the bank of 1 1 quaternion conv2d (QCNN) modules from our first proposal (Figure 1(c)) and replacing 1 1 convolutional down-sampling and up-sampling modules with a bank of 1 1 QCNN modules. The set of output channels of down-sampled 1 1 quaternion is merged into input to the AANet modules, and the output channels of AANet modules are split into groups of four again for the 1 1 up-sampled quaternion layer. One quaternion 2D convolution is applied to each group of four channels. Thus, the weight-sharing is compartmentalized into groups of four input channels.

For better representation, the quaternion layer is also applied in the stem layer (the first layer of the network) as a quaternion-based front-end layer and in the fully-connected dense layer as a quaternion (PHM layer with four dimensions) based back-end layer. To make a more interwoven output representation in the bottleneck block of the network, we also use quaternion layers in the axial-bottleneck architecture, which is described in Figure 1(d). We renamed this full representation based AANets as RepAA networks which are depicted in Figure 2.

Overview of Experiments

Our experiments studied novel architectures involving enhancements to the Axial Attention ResNet. Our first modification replaces the bottleneck blocks of the Axial ResNet with the quaternion bottleneck blocks depicted in Fig. 1(c). This converted 26 and 50-layer Axial Attention ResNets into 33 and 66-layer quaternion-enhanced Axial Attention ResNets. We refer to instances of this architecture as “QuatE-1”, short for “quaternion enhanced, version 1”. Since these QuatE-1 networks were deeper as a result of using quaternion bottleneck blocks, we also assessed the performance accuracy of a 35-layer unmodified Axial Attention network as a control (Experiment 2).

The second architecture tested was the same as QuatE-1 but with the real-valued, fully connected backend replaced by a 4D PHM layer (QPHM). We call this “QuatE-2”. Ex-

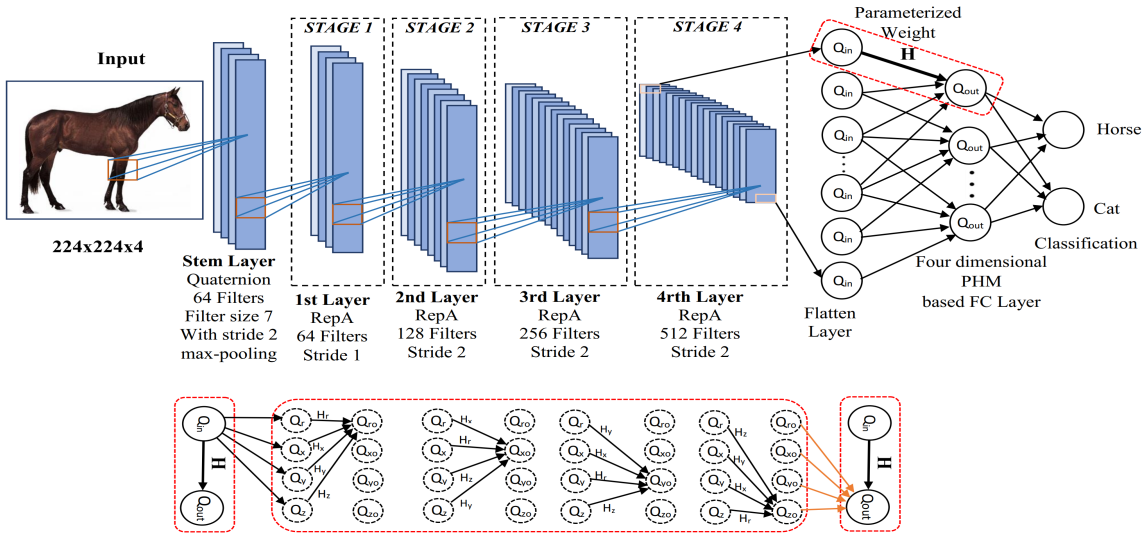

Figure 2: Proposed full representational axial-attention network with QPHM based FC layer. RepA stands for quaternion-based axial bottleneck block (shown in figure 1 (right)). Here, $Q_{in} = Q_r + Q_x + Q_y + Q_z$, $H = H_r + H_x + H_y + H_z$, and $Q_{out} = Q_{ro} + Q_{xo} + Q_{yo} + Q_{zo}$ are the input, hypercomplex parameterized weight, and output respectively.

periments with QuatE-2 give information about the effects of using a quaternion-enhanced backend.

The third tested architecture replaced the 7 7 real-valued, convolutional stem with a 7 7 quaternion-valued convolutional stem. We also used the bottleneck block shown in Fig. 1(d) instead of that in Fig. 1(c). That is, all possible layers were replaced with quaternion versions. We call this "RepAA" (representational axial attention).

Experiment 1

We compared our novel QuatE-1 model to three preexisting baseline models on a subset of the ImageNet dataset, called ImageNet300k, which we created (see below). ImageNet300k was used because we lacked the computing power to conduct simulations using the full ImageNet (Russakovsky et al. 2015).

Model	Tra. Acc.	Params	Vali. Acc.	Inf. time (ms)
ResNet-26	57.0	13.6M	45.48	8.86
QCNN-26	64.1	15.1M	50.09	25.32
Axial-ResNet-26	61.0	5.7M	54.79	27.94
QuatE-1 (ours)-33	78.2	6.0M	62.30	31.75
ResNet-50	65.8	25.5M	50.92	14.70
QCNN-50	73.4	27.6M	49.69	50.01
Axial-ResNet-50	63.6	11.5M	55.57	52.35
QuatE-1 (ours)-66	72.6	11.9M	59.71	58.41

Table 1: Image classification performance on the ImageNet300k dataset for 26- and 50-layer ResNet (He et al. 2016), Quaternion ResNet (QCNN) (Gaudet and Maida 2018), Axial-Attention (Wang et al. 2020), our proposed QuatE1 Axial-Attention architectures. We use Top-1 training (Tr. Acc.) and validation (Vali. Acc.) accuracies.

Method

The baseline models are: the standard convolution-based ResNet (He et al. 2016), the quaternion CNN (Gaudet and Maida 2018), the axial-ResNet (Wang et al. 2020). The main objective is to see if the representations generated by quaternion-enhanced bottleneck blocks improve the classification performance of axial-ResNet. We used a 26-layer version with the block multipliers "[1, 2, 4, 1]" and a 50-layer version with the block multipliers "[3, 4, 6, 3]" of the models. The bracketed expressions show the bottleneck blocks with the operations used and the number of output channels for each stage. If the quaternion modules are counted, then the layer counts for our model are 33 layers and 66 layers, respectively. We count the two-1D-layer axial-attention module as one layer because two 1D layers are equivalent to one 2D layer.

It took over two hours to train one of the axial-ResNet models for one epoch on the original ImageNet. The smaller ImageNet300k dataset uses lower computing resources and uses the same 1,000 image categories as the original ImageNet (Deng et al. 2009). The full ImageNet dataset has 1.28 million training images and 50,000 validation images. Our smaller dataset is sampled from the full ImageNet by taking the first 300 images for each category contained in the original dataset, yielding 300,000 training images. The smaller dataset uses the same 50,000 validation images with 50 images per category. Although the training dataset is smaller, it still allows us to train our 50-layer networks without overfitting. Overfitting was assessed by examining performance on the validation dataset compared to the training set. The validation dataset was the same as the original one.

All models (Experiments 1, 2, and 3) were trained using the same stochastic gradient descent optimizer and hyper-

Architecture	Training	Params	Validation
ResNet	63.8	18.5M	48.99
Quaternion ResNet	70.9	20.5M	48.11
Axial-ResNet	73.6	8.4M	60.49

Table 2: Image classification performance on the ImageNet300k dataset for 35 layer ResNet (He et al. 2016), quaternion, axial-attention (Wang et al. 2020) architectures. We use Top-1 training and validation accuracies.

parameters. All networks were trained for 150 epochs except our proposed models, which were run for 90, and using stochastic gradient descent optimization with momentum set to 0.9 and a learning rate that was warmed up linearly from ϵ near zero to 0.1 for 10 epochs. The learning rate was then cut by 10 at epochs 20, 40, and 70. We adopt the same training protocol as (Wang et al. 2020) except for batch size. The batch size was limited to ten because of memory limitations. For attention models, the number of attention heads was set to eight in all attention layers (Wang et al. 2020).

Architecture	Training Top-1 Acc.	Params	Validation Top-1 Acc.
ResNet-26	57.0	13.6M	45.48
QCNN-26	64.1	15.1M	50.09
Axial-ResNet-26	61.0	5.7M	54.79
RepAA (ours)-26	73.0	3.7M	60.70
ResNet-35	63.8	18.5M	48.99
QCNN-35	70.9	20.5M	48.11
Axial-ResNet-35	73.6	8.4M	60.49
QuatE-1(ours)-33	78.2	6.0M	62.30
QuatE-2(ours)-33	75.5	5.3M	61.27
RepAA (ours)-35	72.0	4.6M	62.03
ResNet-50	65.8	25.5M	50.92
QCNN-50	73.4	27.6M	49.69
Axial-ResNet-50	63.6	11.5M	55.7
QuatE-1(ours)-66	72.6	11.9M	59.71
QuatE-2(ours)-66	77.8	11.1M	62.46
RepAA (ours)-50	73.5	6.7M	62.49

Table 3: Image classification performance on ImageNet300k for 26, 35 and 50-layers ResNet (He et al. 2016), Quaternion ResNet (QCNN) (Gaudet and Maida 2018), Axial-Attention (Wang et al. 2020), our first modification (QuatE1), our second modification (QuatE2) architectures. We include Top-1 training and validation accuracies and parameter count. "QuatE" and "RepAA" stand for quaternion enhanced and representational axial-attention.

Results

The main results are in Table 1. The top half of the table shows the results for the 26-layer models (33-layers for the QuatE-1 model). The bottom half shows the results for the 50-layer models (66 layers for QuatE-1). Reported are the parameter count, inference time, and percent accuracy for a single simulation run for each model. Both axial-ResNets use far fewer parameters than the convolution-based ResNets.

The most important comparison is between the axial-ResNet and our QuatE-1, as this directly shows the effect of the quaternion-generated interlinked representations. There are two such comparisons, one for the 26/33 layer models and the other for the 50/66 layer models. In both cases, the quaternion-enhanced versions produced higher classification accuracy on both the training and validation data. This supports our main hypothesis that quaternion modules can produce more usable interlinked representations. There is an alternative explanation that the quaternion-enhanced axial-ResNet had more layers, and this caused better performance. This is addressed in Experiment 2.

Another result is that both axial-ResNet architectures provide higher performance for the validation set than either of the convolution-based ResNets. This is true for both the 26-layer (33-layer) and 50-layer (66-layer) versions.

Finally, it is surprising that the 33-layer QuatE-1 axial-ResNet gives better classification accuracy than the 66-layer version. This is true for both the training and validation data. We have no explanation for this. If this were only true for the validation data, then overfitting would be a possible explanation. This is addressed in Experiment 3.

Experiment 2

In Experiment 1, adding the quaternion frontend to the axial attention module increased the number of layers from 26 to 33 for the small network and from 50 to 66 for the large one. This offered an alternative explanation for better classification accuracy. The improvement might have been caused by the increased number of layers and not the added quaternion bank. Experiment 2 tests this explanation.

Method

This experiment increases the layer count of the baseline networks from 26 to 35 layers. This was done by using a block multiplier of "[2, 3, 4, 2]" for these models. Although this did not give us exactly 33 layers to exactly match the QuatE-1 axial-attention model, it preserved the bottleneck structure of the original design and enhanced comparability.

Results

Table 2 shows the parameter count and accuracy for these 35-layer models. The most important comparison is the 35-layer Axial-Attention Network in Table 2 with the 33-layer QuatE-1 Network in Table 1. Now the layer count for the non-quaternion version is slightly larger than that for the quaternion version, but the quaternion version still gives better performance on both the training and validation accuracy. The increased layer count did improve the performance of the 35-layer Axial Attention Network but not enough to surpass the quaternion version. Thus, we conclude that the quaternion blocks have more impact than the layer count.

Experiment 3

Table 1 shows that the 33-layer QuatE-1 model gave better classification accuracy than the 66-layer version. To address

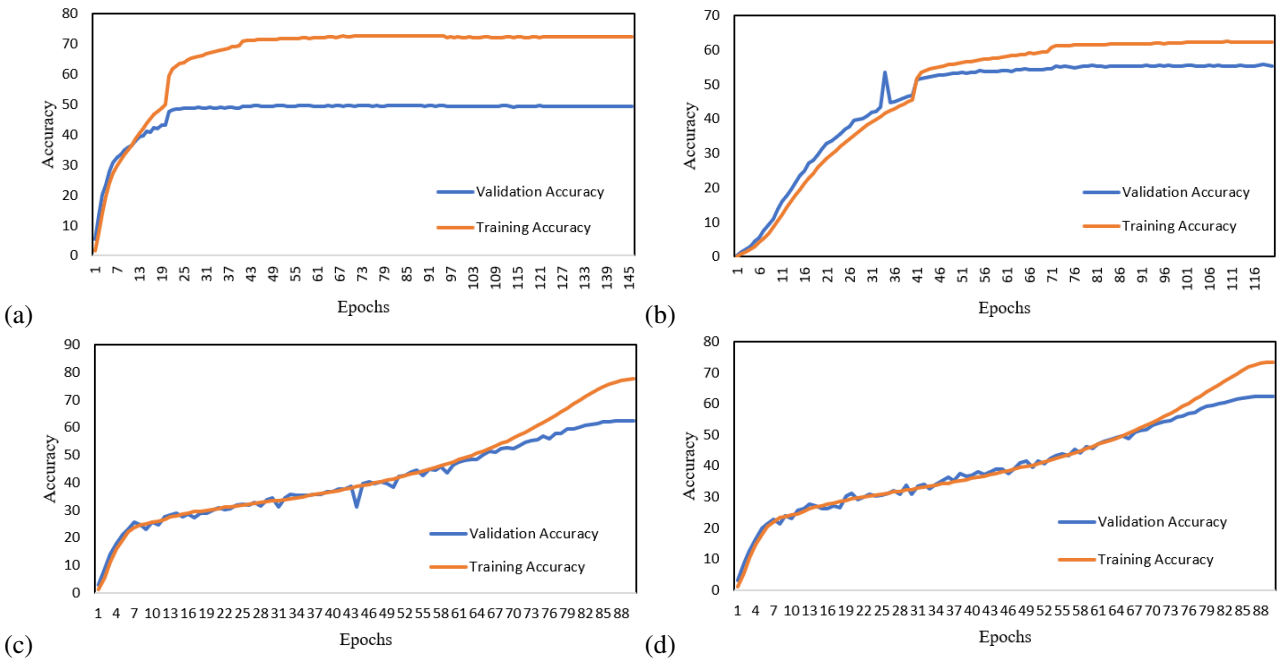

Figure 3: Training and validation performance of original quaternion ResNet, original axial-attention network, our QuatE-2 axial ResNet with QPHM, and our RepAA method (50-layers version) to assess possible overfitting for ImageNet300k dataset.

this, we performed another experiment using the QuatE-2 and RepAA networks on the ImageNet300k dataset.

Method

This experiment used 26-layer, 35-layer, and 50-layer architectures with the same multipliers and we trained the networks using the same hyperparameters as in Experiment 1.

Results

Table 3 collects the overall results for the 26, 35, and 50-layer architectures. The performances of QuatE axial-ResNets with QPHM in the back-end, and RepAA networks are compared in Tables 1, 2, and 3. Among them, our final proposed method performs best in validation accuracy and parameter count.

Figure 3 assesses the overfitting problem by examining the accuracy of the validation dataset in comparison to the training dataset and shows there is no overfitting for the ImageNet300k dataset. The most important comparisons are between the QuatE-2 axial-ResNet and our RepAA model, and axial-ResNet and RepAA model as these directly show the effect of removing extra 1 1 quaternion layer from QuatE axial-ResNet and applying cross-channel weight sharing throughout the attention network. In both cases, the RepAA networks outperformed in terms of accuracy and parameters.

Although we ran our model for only 90 epochs, which is less than the other models, its performance was unchanged beyond 90 up through 150 epochs. Unlike the QuatE-1 axial-ResNets, the validation performance of QuatE-2 for 66 layers is higher than the 33 layers (shown in Table 3). Also, the validation performance of RepAA for 35 layers is higher than the 26 layers, and 50 layers outperformed the others (shown in Table 3). This supports our main hypothesis; quaternion modules can produce more usable interlinked representations.

Conclusions and Future Work

We replaced traditional modules with representationally coherent modules in the stem, the bottleneck blocks, and the fully connected backend. All of these novel modifications improved accuracy to varying degrees when trained and tested on the ImageNet300k dataset. Our baseline networks for comparison were the real-valued ResNet, the quaternion CNN, and the Axial Attention ResNet.

Our results are significant because the improvement was observed when any part of the network was modified to use representational coherence. This suggests that this technique may be generally useful in improving classification accuracy for a large class of networks. This work was limited to the ImageNet300k dataset due to machine limitations and was not able to evaluate low-resolution datasets like CIFAR benchmarks as attention models require a large number of high-resolution data to handle the overfitting. More work may be directed at testing other architectures to assess this claim. Further work should also be directed toward checking if these results hold up on other datasets.

Finally, vectormaps and PHM layers offer more fine-grained control over weight sharing than quaternions. For quaternions, four weights are distributed over 16 slots, diminishing the weight count ratio to 25%. Since vectormap and PHM operations are not constrained to four dimensions, other weight count ratios can be tested.

References

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. ImageNet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, 248–255. IEEE.

Gaudet, C., and Maida, A. S. 2018. Deep quaternion networks. In *Intl. Joint Conf. on Neural Networks (IJCNN)*. IEEE.

Gaudet, C. J., and Maida, A. S. 2020. Generalizing complex/hyper-complex convolutions to vector map convolutions. *CoRR* abs/2009.04083.

Gaudet, C. J., and Maida, A. S. 2021. Removing dimensional restrictions on complex/hypercomplex networks. In *2021 IEEE International Conference on Image Processing*. IEEE Signal Processing Society.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *The Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ho, J.; Kalchbrenner, N.; Weissenborn, D.; and Salimans, T. 2019. Axial attention in multidimensional transformers.

Hu, H.; Zhang, Z.; Xie, Z.; and Lin, S. 2019. Local relation networks for image recognition. In *International Conference on Computer Vision (ICCV)*.

Parcollet, T.; Morchid, M.; and Linarés, G. 2019. Quaternion convolutional networks for heterogenous image processing. In *IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 8514–8518.

Parcollet, T.; Morchid, M.; and Linarés, G. 2020. A survey of quaternion neural networks. *Artificial Intelligence Review* 53(2957-2982).

Ramachandran, P.; Parmar, N.; Vaswani, A.; Bello, I.; Levskaya, A.; and Shlens, J. 2019. Stand-alone self-attention in vision models. *CoRR* abs/1906.05909.

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. 2015. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision* 115(3):211–252.

Trabelsi, C.; Bilaniuk, O.; Zhang, Y.; Serdyuk, D.; Subramanian, S.; Santos, J. F.; Mehri, S.; Rostamzadeh, N.; Bengio, Y.; and Pal, C. J. 2018. Deep complex networks.

Wang, H.; Zhu, Y.; Green, B.; Adam, H.; Yuille, A.; and Chen, L.-C. 2020. Axial-deeplab: Stand-alone axial-attention for panoptic segmentation.

Zhang, A.; Tay, Y.; Zhang, S.; Chan, A.; Luu, A. T.; Hui, S. C.; and Fu, J. 2021. Beyond fully-connected layers with quaternions: Parameterization of hypercomplex multiplications with $1/n$ parameters. *arXiv preprint arXiv:2102.08597*.