Improvement on the Quaternion-based models: extension to larger datasets and Batch Normalization

Aritra Mukhopadhyay & Adhilsha A

Recap

Goal To improve some **quaternion models, implement models on larger datasets**, and **implement batch normalization.**

Datasets Used MNIST, Cifar-10 and Cifar-100.
(more datasets as we develop the models better)

Baseline Models Lenet-300-100, conv2, conv4, conv6
later more complex models like mobilenet and resnet.

What Are Quaternions?

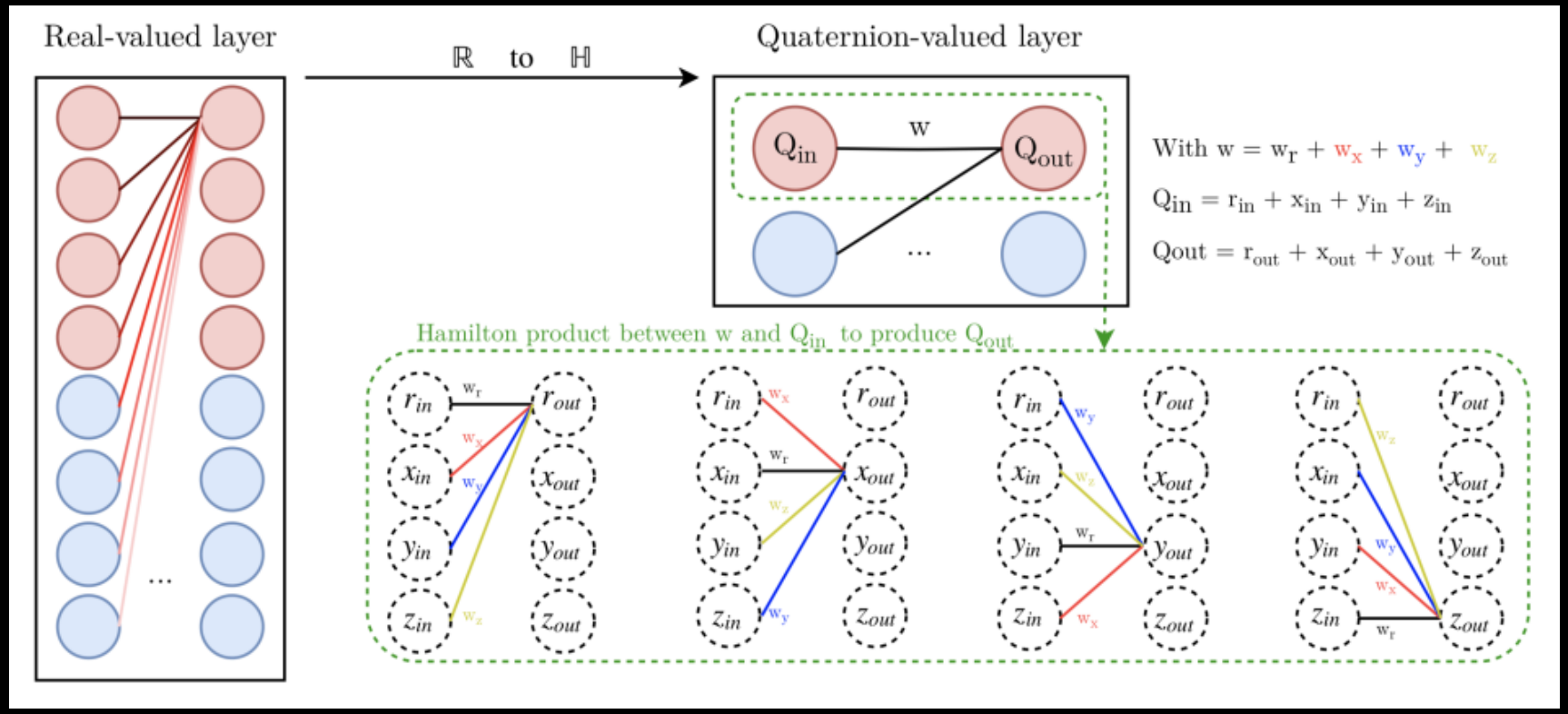
Quaternion is a four-dimensional extension of complex numbers, represented by a vector of the form $q = r + xi + yj + zk$. Given two quaternions q_1 and q_2, their product (known as the Hamilton product) is given by:

$$\begin{aligned} q_1 \otimes q_2 = \; & (r_1 r_2 - x_1 x_2 - y_1 y_2 - z_1 z_2) \\ & (r_1 x_2 + x_1 r_2 + y_1 z_2 - z_1 y_2)\,\mathbf{i} \\ & (r_1 y_2 - x_1 z_2 + y_1 r_2 - z_1 x_2)\,\mathbf{j} \\ & (r_1 z_2 + x_1 y_2 - y_1 x_2 - z_1 r_2)\,\mathbf{k} \end{aligned}$$

Quaternions multiplications not being commutative, they can be written in terms of 4*4 real matrices such that the matrix multiplication between such representations are consistent with the Hamilton product:

$$q = \begin{bmatrix} r & -x & -y & -z \\ x & r & -z & y \\ y & z & r & -x \\ z & -y & x & r \end{bmatrix}$$

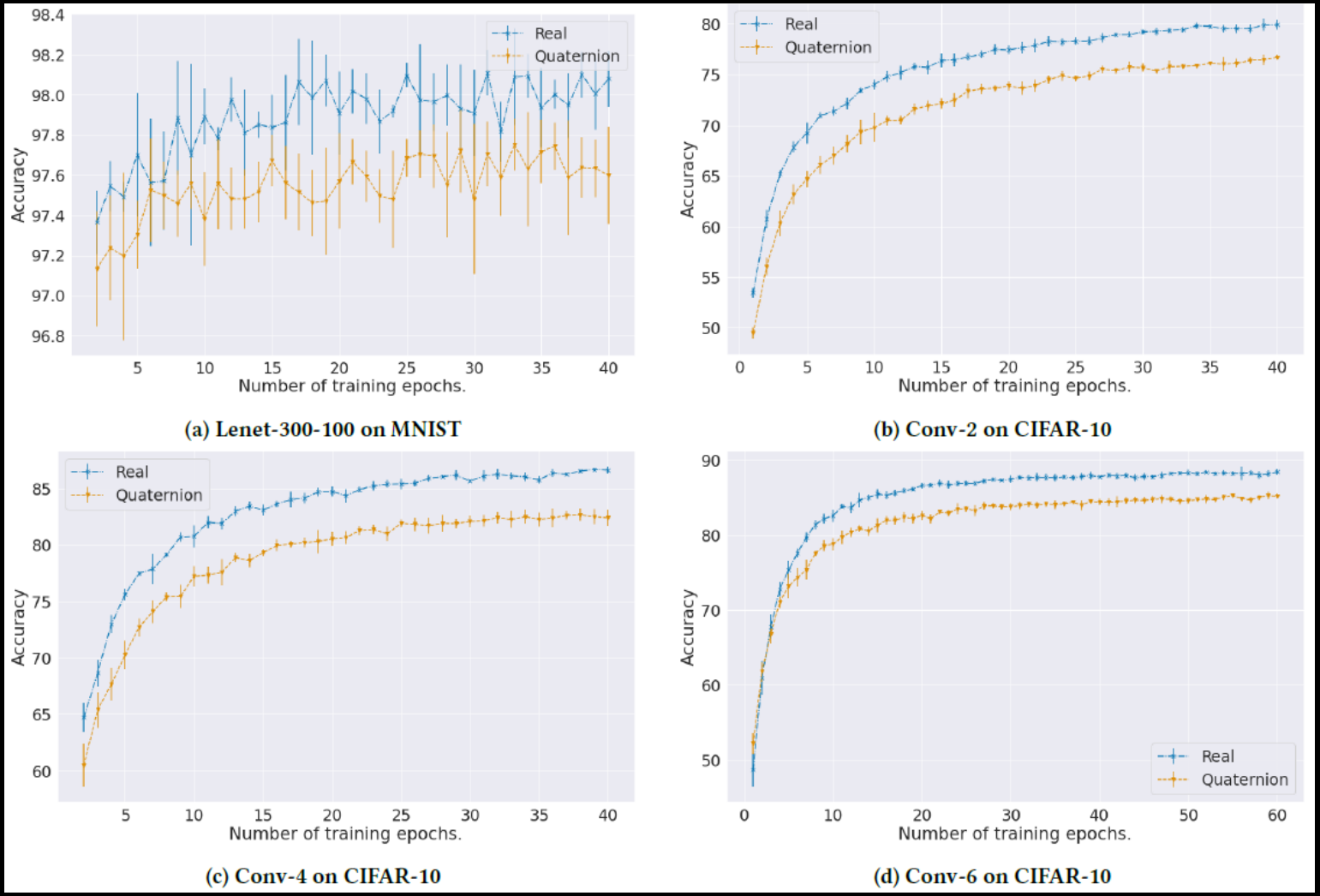
Why are Quaternions used in building Neural Networks?



This image has been taken from this paper.
Titouan Parcollet, Mirco Ravanelli, Mohamed Morchid, Georges Linarès, and Renato De Mori. 2018. *Speech recognition with quaternion neural networks.*

They can be used to make almost equally complex models with only 25% of the weights of the real version. This makes prediction whole lot more easier in low end devices.

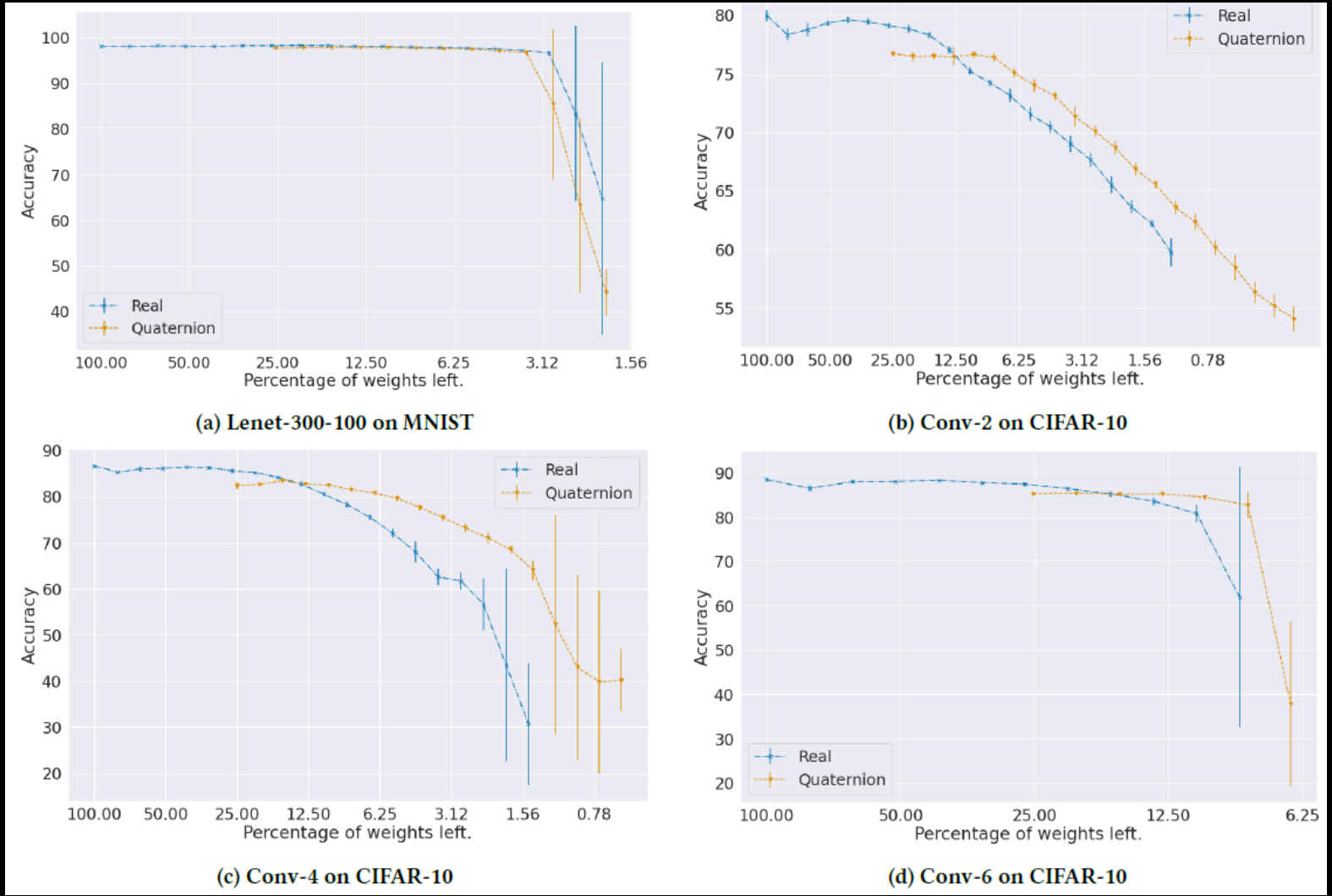
Previous work



- Graphs on training both real and quaternion models .
- For the MNIST dataset, the input images are grayscale with only one channel. In Lenet-300-100, each set of four pixels are fed to a quaternion neuron after flattening the image.
- For color vision tasks such as CIFAR-10, the RGB channels of each pixel are treated as belonging to a single quaternion neuron by adding one more channel to the input images, which is grayscale data.

This image has been taken from this paper. Sahel Mohammad Iqbal and Subhankar Mishra. 2023. *Neural Networks at a Fraction with Pruned Quaternions*..

Previous work



(a) Lenet-300-100 on MNIST

(b) Conv-2 on CIFAR-10

(c) Conv-4 on CIFAR-10

(d) Conv-6 on CIFAR-10

- "pruned quaternion models can be re-trained from scratch to match the original accuracy of the unpruned model, showing that lottery tickets exist for quaternion networks as well." (Sahel et al.)

- "when pruned to high levels of sparsities, quaternion implementations of certain models outperform their complementary real-valued models of equivalent architectures. "

This image has been taken from this paper. Sahel Mohammad Iqbal and Subhankar Mishra. 2023. *Neural Networks at a Fraction with Pruned Quaternions*..

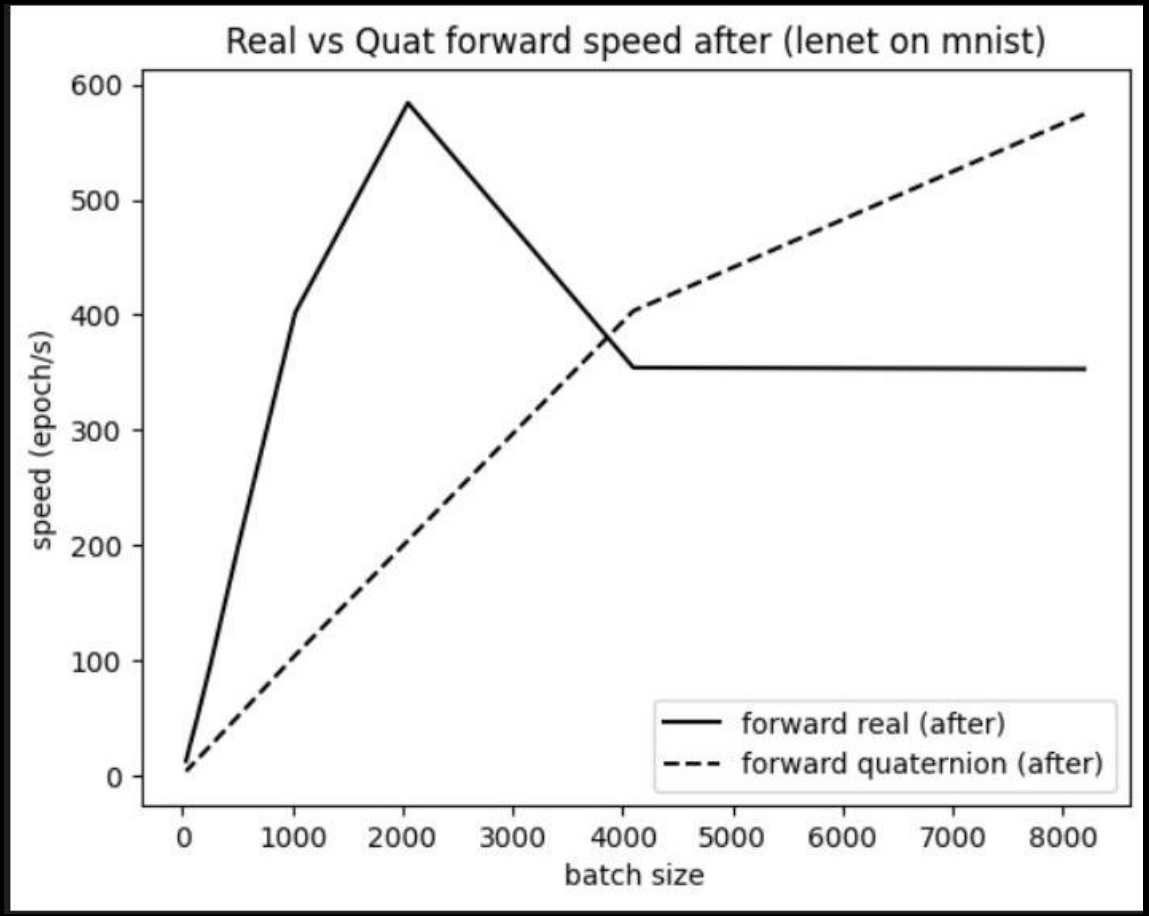
Our Approach...

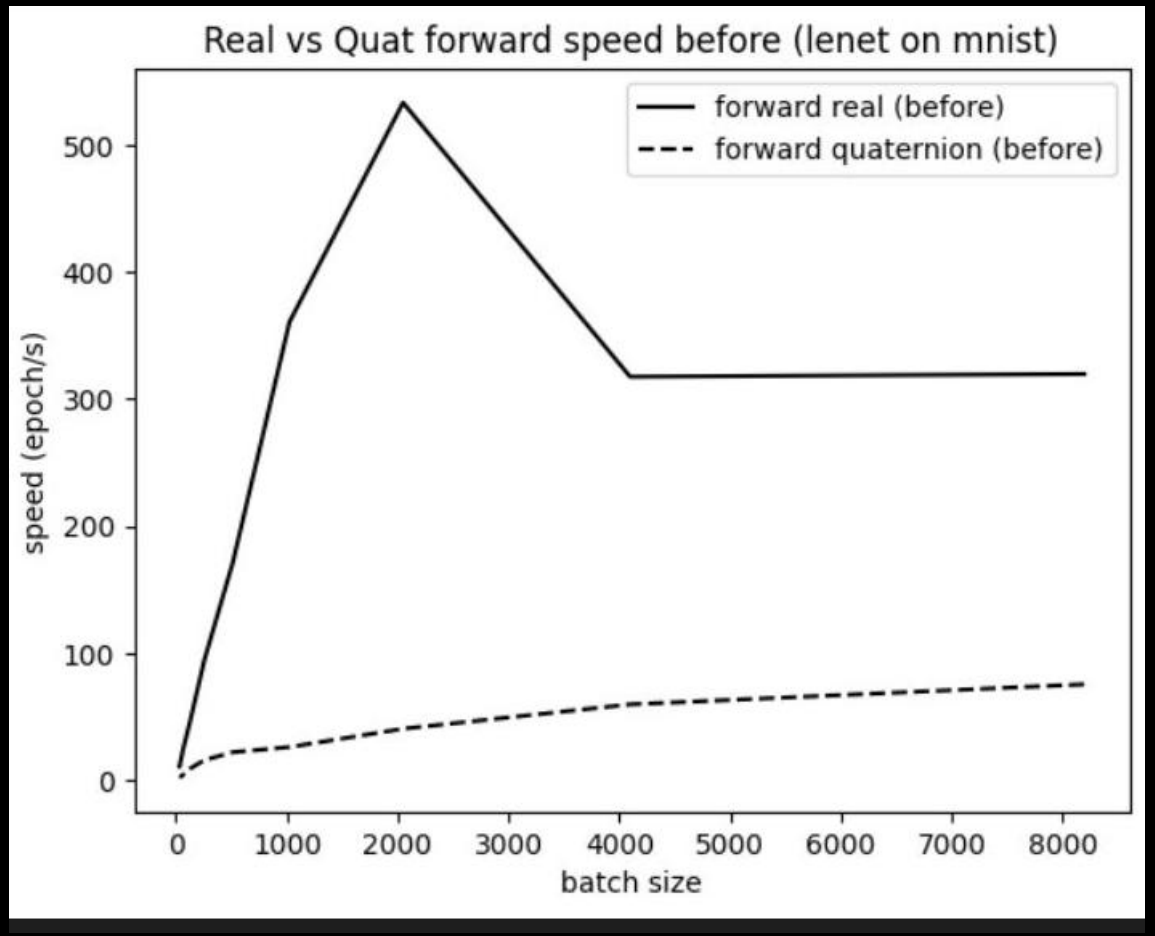
We were trying to speed up the quaternion layer calculations. For that we needed to see particularly why was it taking so long time. We found the three steps of forward propagation. They are:

1. building 4×4 Quaternion to real matrix (w)
2. Finding wx + b (applying linear function)
3. typecasting the output of step 2 to a Quaternion tensor.

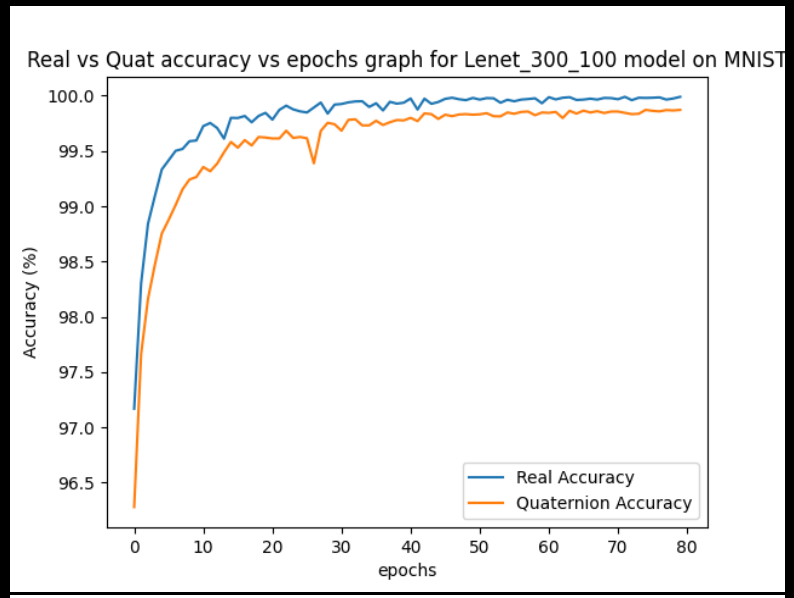
We found that step 3 was taking up 95% of the time. Further study revealed This was because of a line **q.cpu()** which was needlessly copying the x to the CPU memory (the RAM) after every layer. Being a highly experimental part of library, we changed it to **q.cuda()** and got rid of the redundant operations. This improved the speed by almost double (22.5 it/s to 57.5 it/s).

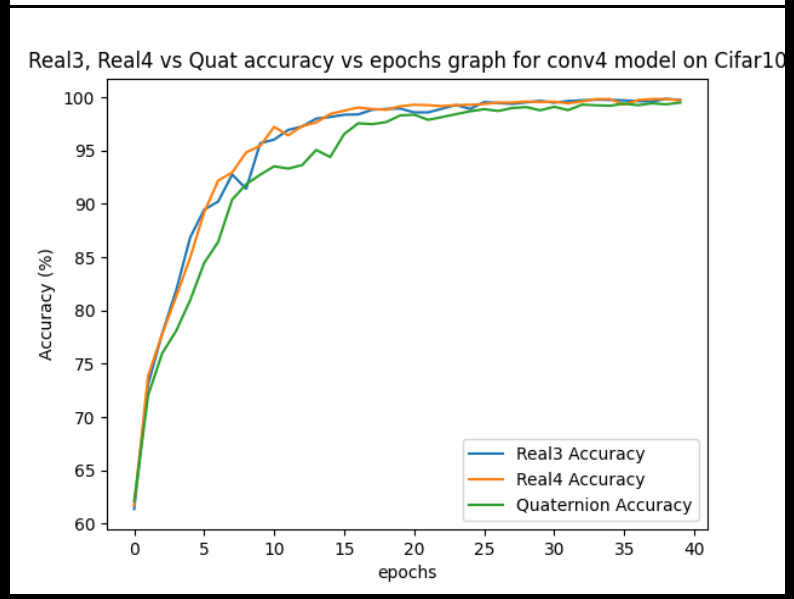
Relevant Graphs



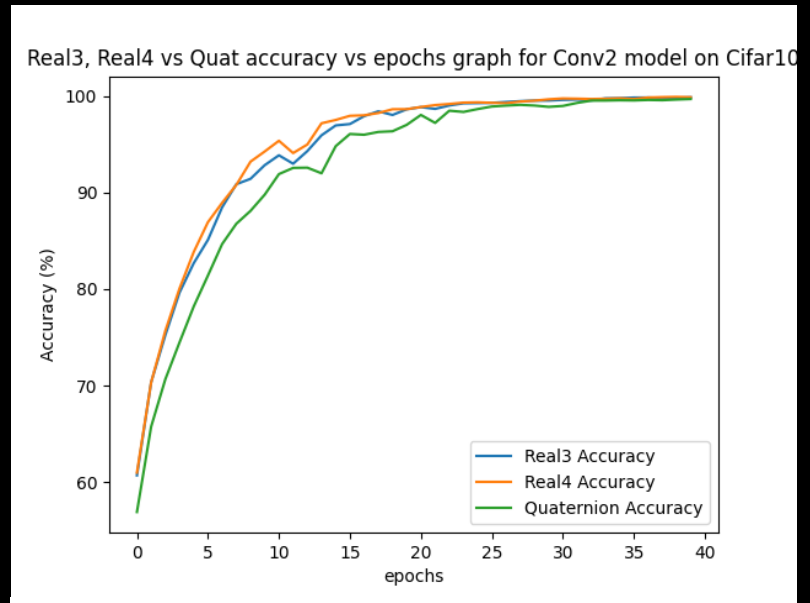


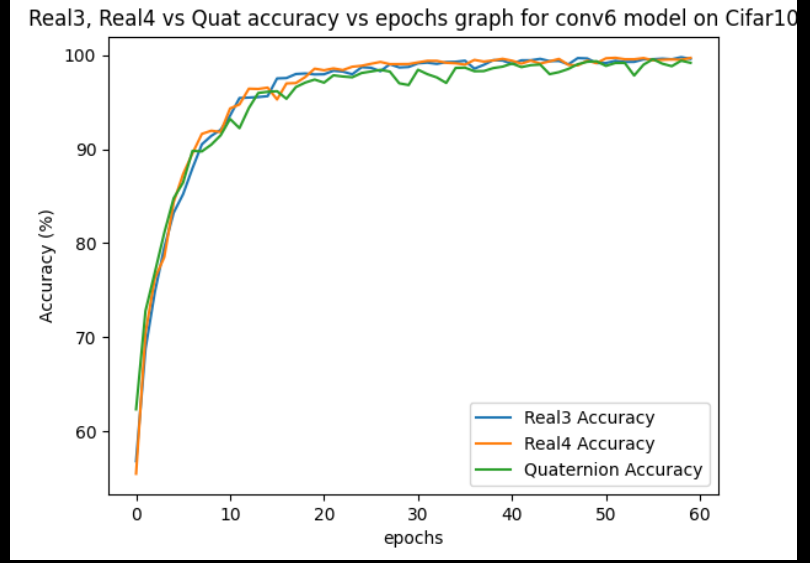
Interesting Results



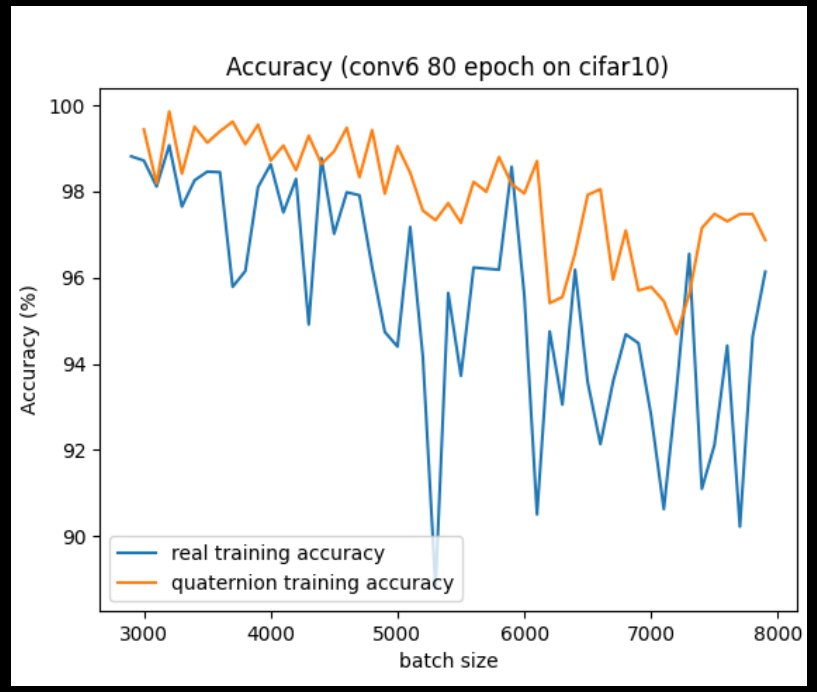


- Real3, Real4 and quaternion perform similar for larger datasets.
- In the original paper, the CNNs were trained on RGBdata as input (say **Real3**). But, as the quaternions needed 4 channels, RGBdata + grayscaledata of the same image was used together. The real and quaternion comparison is less fair so.
- We implemented the real model with RGBdata + grayscaledata for a fair analysis. This is **Real4**.

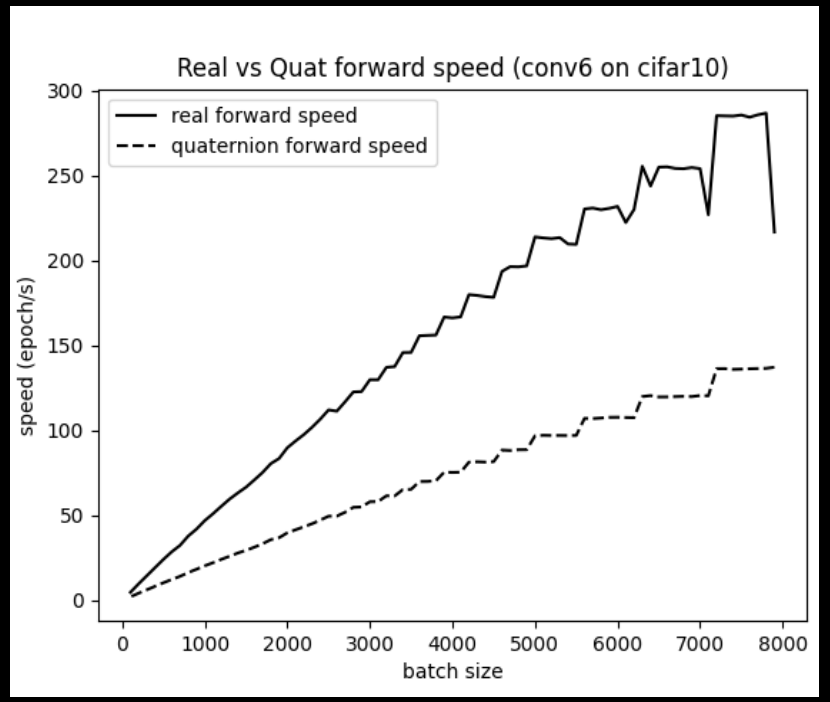




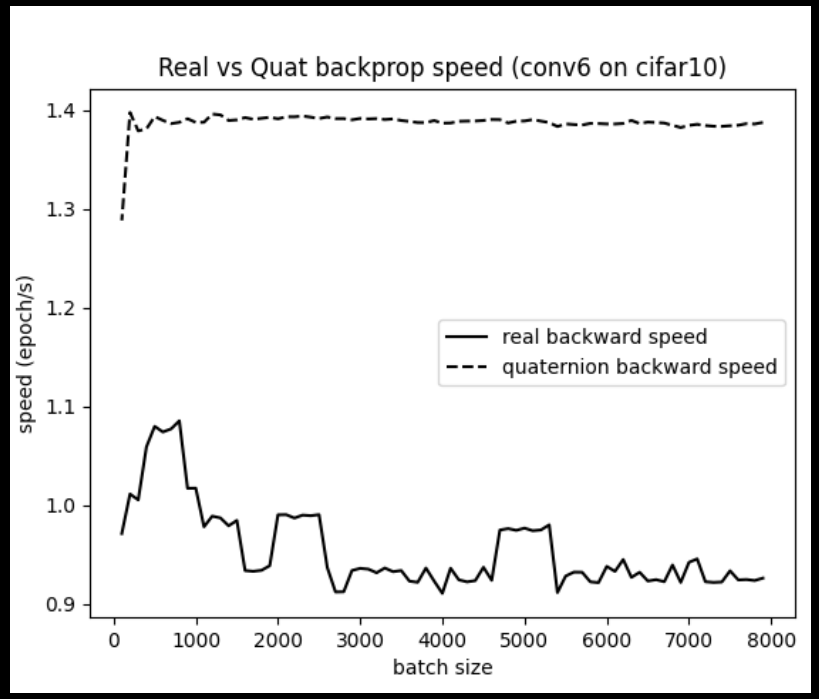
Can you explain these?



Accuracy depends on batch size



Quaternion backpropagation is almost 2x fast than real counterpart



Forward speed changes like a step function as a function of batch size

Further plans

- To implement batch normalization for quaternion models and to train and improve larger models with larger datasets.

- Continuing the pruning and analysis on these improved models.

- Do extensive experiments on different sorts of tasks (like vision, NLP etc.) with different large models to benchmark the performance of quaternions in those tasks.