Speech Recognition with Quaternion Neural Networks

Titouan Parcollet[1,4] **Mirco Ravanelli**[2] **Mohamed Morchid**[1]
Georges Linarès[1] **Renato De Mori**[1,3]

[1]LIA, Université d'Avignon, France
[2]MILA, Université de Montréal, Québec, Canada
[3]McGill University, Québec, Canada
[4] Orkis, Aix-en-provence, France
`titouan.parcollet@alumni.univ-avignon.fr`
`mirco.ravanelli@gmail.com`
`firstname.lastname@univ-avignon.fr`
`rdemori@cs.mcgill.ca`

Abstract

Neural network architectures are at the core of powerful automatic speech recognition systems (ASR). However, while recent researches focus on novel model architectures, the acoustic input features remain almost unchanged. Traditional ASR systems rely on multidimensional acoustic features such as the Mel filter bank energies alongside with the first, and second order derivatives to characterize time-frames that compose the signal sequence. Considering that these components describe three different views of the same element, neural networks have to learn both the internal relations that exist within these features, and external or global dependencies that exist between the time-frames. Quaternion-valued neural networks (QNN), recently received an important interest from researchers to process and learn such relations in multidimensional spaces. Indeed, quaternion numbers and QNNs have shown their efficiency to process multidimensional inputs as entities, to encode internal dependencies, and to solve many tasks with up to four times less learning parameters than real-valued models. We propose to investigate modern quaternion-valued models such as convolutional and recurrent quaternion neural networks in the context of speech recognition with the TIMIT dataset. The experiments show that QNNs always outperform real-valued equivalent models with way less free parameters, leading to a more efficient, compact, and expressive representation of the relevant information.

1 Introduction

During the last decade, deep neural networks (DNN) have encountered a wide success in automatic speech recognition. Many architectures such as recurrent (RNN) [35, 15, 1, 32, 13], time-delay (TDNN) [40, 29], or convolutional neural networks (CNN) [43] have been proposed and achieved better performances than traditional hidden Markov models (HMM) combined with gaussian mixtures models (GMM) in different speech recognition tasks. However, despite such evolution of models and paradigms, the acoustic feature representation remains almost the same. The acoustic signal is commonly split into time-frames, for which Mel-filter banks energies, or Mel frequency scaled cepstral coefficients (MFCC) [8] are extracted, alongside with the first and second order derivatives. In fact, time-frames are characterized by 3-dimensional features that are related by representing three different views of the same basic element. Consequently an efficient neural networks-based model has to learn both external dependencies between time-frames, and internal relations within the features. Traditional real-valued architectures deal with both dependencies at the same level, due to the lack of a dedicated mechanism to learn the internal and external relations separately.

Quaternions are hypercomplex numbers that contain a real and three separate imaginary components, fitting perfectly to three and four dimensional feature vectors, such as for image processing and robot kinematics [36, 30, 4]. The idea of bundling groups of numbers into separate entities is also exploited by the recent capsule network [34]. Contrary to traditional homogeneous representations, capsule and quaternion neural networks bundle sets of features together. Thereby, quaternion numbers allow neural models to code latent inter-dependencies between groups of input features during the learning process with up to four times less parameters than real-valued neural networks, by taking advantage of the *Hamilton product* as the equivalent of the dot product between quaternions. Early applications of quaternion-valued backpropagation algorithms [3, 2] have efficiently solved quaternion functions approximation tasks. More recently, neural networks of complex and hypercomplex numbers have received an increasing attention [16, 39, 7, 41], and some efforts have shown promising results in different applications. In particular, a deep quaternion network [24, 25, 38], and a deep quaternion convolutional network [5, 28] have been successfully employed for challenging tasks such as images and language processing.

Contributions: This paper proposes to evaluate previously investigated quaternion-valued models in two different realistic conditions of speech recognition, to see whether the quaternion encoding of the signal, alongside with the quaternion algebra and the important parameter reduction help to better capture the acoustic signal nature, leading to a more expressive representation of the information. Based on the TIMIT [10] phoneme recognition task, a quaternion convolutional neural network (QCNN) is compared to a real-valued CNN in a end-to-end framework, and a quaternion recurrent neural network (QRNN) is compared to an RNN within a more traditional HMM-based system. In the end-to-end approach, the experiments show that the QCNN outperforms the CNN with a phoneme error rate (PER) of 19.5% against the 20.6% achieved for CNNs. Moreover, the QRNN outperforms the RNN with a PER of 18.5% against 19.0% for the RNN. Furthermore, such results are observed with a maximum reduction factor of the number of neural network parameters of 3.96 times.

2 Motivations

A major challenge of current machine learning models is to obtain efficient representations of relevant information for solving a specific task. Consequently, a good model has to efficiently code both the relations that occur at the feature level, such as between the Mel filter energies, the first, and second order derivatives values of a single time-frame, and at a global level, such as a phonemes or words described by a group of time-frames. Moreover, to avoid overfitting, better generalize, and to be more efficient, such models also have to be as small as possible. Nonetheless, real-valued neural networks usually require a huge set of parameters to well-perform on speech recognition tasks, and hardly code internal dependencies within the features, since they are considered at the same level as global dependencies during the learning. In the following, we detail the motivations to employ quaternion-valued neural networks instead of real-valued ones to code inter and intra features dependencies with less parameters.

First, a better representation of multidimensional data has to be explored to naturally capture internal relations within the input features. For example, an efficient way to represent the information composing an acoustic signal sequence is to consider each time-frame as being a whole entity of three strongly related elements, instead of a group of unidimensional elements that *could* be related to each others, as in traditional real-valued neural networks. Indeed, with a real-valued NN, the latent relations between the Mel filter banks energies, and the first and second order derivatives of a given time-frame are hardly coded in the latent space since the weight has to find out these relations among all the time-frames composing the sequence. Quaternions are fourth dimensional entities and allow one to build and process elements made of up to four elements, mitigating the above described problem. Indeed, the quaternion algebra and more precisely the *Hamilton product* allows quaternion neural network to capture these internal latent relations within the features of a quaternion. It has been shown that QNNs are able to restore the spatial relations within 3D coordinates [20], and within color pixels [17], while real-valued NNs failed. In fact, the quaternion-weight components are shared through multiple quaternion input parts during the *Hamilton product* , creating relations within the elements. Indeed, Figure 1 shows that the multiple weights required to code latent relations within a

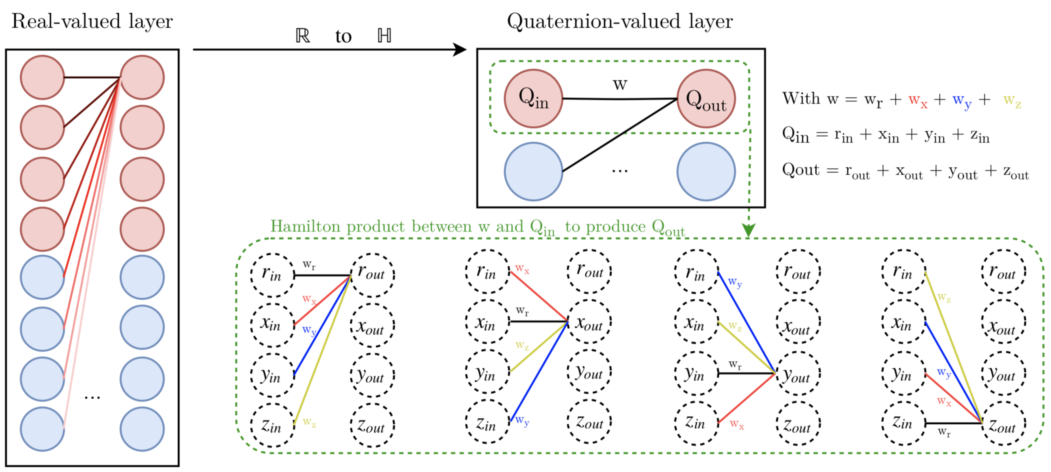
Real-valued layer
ℝ to ℍ
Quaternion-valued layer
Q_{in}
w
Q_{out}
...
With $w = w_r + w_x + w_y + w_z$
$Q_{in} = r_{in} + x_{in} + y_{in} + z_{in}$
$Qout = r_{out} + x_{out} + y_{out} + z_{out}$
Hamilton product between w and Q_{in} to produce Q_{out}

Figure 1: Illustration of the input features (Q_{in}) latent relations learning ability of a quaternion-valued layer (right) due to the quaternion weight sharing of the *Hamilton product* (Eq. 5), compared to a standard real-valued layer (left).

feature are considered at the same level as for learning global relations between different features, while the quaternion weight w codes these internal relations within a unique quaternion Q_{out} during the *Hamilton product* (right).

Second, quaternion neural networks make it possible to deal with the same signal dimension than real-valued NN, but with four times less neural parameters. Indeed, a 4-number quaternion weight linking two 4-number quaternion units only has 4 degrees of freedom, whereas a standard neural net parametrization have $4 \times 4 = 16$, i.e., a 4-fold saving in memory. Therefore, the natural multidimensional representation of quaternions alongside with their ability to drastically reduce the number of parameters indicate that hyper-complex numbers are a better fit than real numbers to create more efficient models in multidimensional spaces such as speech recognition.

Indeed, modern automatic speech recognition systems usually employ input sequences composed of multidimensional acoustic features, such as log Mel features, that are often enriched with their first, second and third time derivatives [8, 9], to integrate contextual information. In standard NNs, static features are simply concatenated with their derivatives to form a large input vector, without effectively considering that signal derivatives represent different views of the same input. Nonetheless, it is crucial to consider that these three descriptors represent a special state of a time-frame, and are thus correlated. Following the above motivations and the results observed on previous works about quaternion neural networks, we hypothesize that for acoustic data, quaternion NNs naturally provide a more suitable representation of the input sequence, since these multiple views can be directly embedded in the multiple dimensions space of the quaternion, leading to smaller and more accurate models.

3 Quaternion Neural Networks

Real-valued neural networks architectures are extended to the quaternion domain to benefit from its capacities. Therefore, this section proposes to introduce the quaternion algebra (Section 3.1), the quaternion internal representation (section 3.2), a quaternion convolutional neural networks (QCNN, Section 3.3) and a quaternion recurrent neural network (QRNN, Section 3.4).

3.1 Quaternion Algebra

The quaternion algebra ℍ defines operations between quaternion numbers. A quaternion Q is an extension of a complex number defined in a four dimensional space as:

$$Q = r1 + x\mathbf{i} + y\mathbf{j} + z\mathbf{k}, \tag{1}$$

where r, x, y, and z are real numbers, and 1, **i**, **j**, and **k** are the quaternion unit basis. In a quaternion, r is the real part, while $x\mathbf{i} + y\mathbf{j} + z\mathbf{k}$ with $\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1$ is the imaginary part, or the vector part. Such a definition can be used to describe spatial rotations. The information embedded in the quaterion Q can be summarized into the following matrix of real numbers, that turns out to be more suitable for computations:

$$Q_{mat} = \begin{bmatrix} r & -x & -y & -z \\ x & r & -z & y \\ y & z & r & -x \\ z & -y & x & r \end{bmatrix}. \tag{2}$$

The conjugate Q^* of Q is defined as:

$$Q^* = r1 - x\mathbf{i} - y\mathbf{j} - z\mathbf{k}. \tag{3}$$

Then, a normalized or unit quaternion $Q^{\triangleleft}$ is expressed as:

$$Q^{\triangleleft} = \frac{Q}{\sqrt{r^2 + x^2 + y^2 + z^2}}. \tag{4}$$

Finally, the Hamilton product $\otimes$ between two quaternions Q_1 and Q_2 is computed as follows:

$$\begin{aligned} Q_1 \otimes Q_2 = &(r_1 r_2 - x_1 x_2 - y_1 y_2 - z_1 z_2) + \\ &(r_1 x_2 + x_1 r_2 + y_1 z_2 - z_1 y_2)\boldsymbol{i} + \\ &(r_1 y_2 - x_1 z_2 + y_1 r_2 + z_1 x_2)\boldsymbol{j} + \\ &(r_1 z_2 + x_1 y_2 - y_1 x_2 + z_1 r_2)\boldsymbol{k}. \end{aligned} \tag{5}$$

The Hamilton product is used in QRNNs to perform transformations of vectors representing quaternions, as well as scaling and interpolation between two rotations following a geodesic over a sphere in the $\mathbb{R}^3$ space as shown in [21].

3.2 Quaternion internal representation

In a quaternion layer, all parameters are quaternions, including inputs, outputs, weights, and biases. The quaternion algebra is ensured by manipulating matrices of real numbers [28]. Consequently, for each input vector of size N, output vector of size M, dimensions are split into four parts: the first one equals to r, the second is $x\mathbf{i}$, the third one equals to $y\mathbf{j}$, and the last one to $z\mathbf{k}$ to compose a quaternion $Q = r1 + x\mathbf{i} + y\mathbf{j} + z\mathbf{k}$. The inference process is based in the real-valued space on the dot product between input features and weights matrices. In any quaternion-valued NN, this operation is replaced with the *Hamilton product* (eq. 5) with quaternion-valued matrices (i.e. each entry in the weight matrix is a quaternion).

3.3 Quaternion convolutional neural networks

Convolutional neural networks (CNN) [19] have been proposed to capture the high-level relations that occur between neighbours features, such as shape and edges on an image. However, internal dependencies within the features are considered at the same level than these high-level relations by real-valued CNNs, and it is thus not guaranteed that they are well-captured. In this extend, a quaternion convolutional neural network (QCNN) have been proposed by [5, 28][1]. Let γ^l_{ab} and S^l_{ab} , be the quaternion output and the pre-activation quaternion output at layer l and at the indexes (a, b) of the new feature map, and w the quaternion-valued weight filter map of size $K \times K$. A formal definition of the convolution process is:

$$\gamma^l_{ab} = \alpha(S^l_{ab}), \tag{6}$$

with

$$S^l_{ab} = \sum_{c=0}^{K-1} \sum_{d=0}^{K-1} w^l \otimes \gamma^{l-1}_{(a+c)(b+d)}, \tag{7}$$

[1]https://github.com/Orkis-Research/Pytorch-Quaternion-Neural-Networks

where α is a *quaternion split activation* function [42] defined as:

$$\alpha(Q) = f(r) + f(x)\mathbf{i} + f(y)\mathbf{j} + f(z)\mathbf{k}, \tag{8}$$

with f corresponding to any standard activation function. The output layer of a quaternion neural network is commonly either quaternion-valued such as for quaternion approximation [2], or real-valued to obtains a posterior distribution based on a softmax function following the split approach of Eq. 8. Indeed, target classes are often expressed as real numbers. Finally, the full derivation of the backpropagation algorithm for quaternion valued neural networks can be found in [23].

3.4 Quaternion recurrent neural networks

Despite the fact that CNNs are efficient to detect and learn patterns in an input volume, recurrent neural networks (RNN) are more adapted to represent sequential data. Indeed, recurrent neural networks obtained state-of-the-art results on many tasks related to speech recognition [33, 12]. Therefore, a quaternary version of the RNN called QRNN have been proposed by [27] [2]. Let us define a QRNN with an hidden state composed of H neurons. Then, let w_{hh}, $w_{h\gamma}$, and $w_{\gamma h}$ be the hidden to hidden, input to hidden, and hidden to output weight matrices respectively, and b_n^l be the bias at neuron n and layer l. Therefore, and with the same parameters as for the QCNN, the hidden state $h_n^{t,l}$ of the neuron n at timestep t and layer l can be computed as:

$$h_n^{t,l} = \alpha(\sum_{m=0}^{H} w_{nm,hh}^{t,l} \otimes h_m^{t-1,l} + \sum_{m=0}^{N_{l-1}} w_{nm,h\gamma}^{t,l} \otimes \gamma_m^{t,l-1} + b_n^l), \tag{9}$$

with α any *split activation* function. Finally, the output of the neuron n is computed following:

$$\gamma_n^{t,l} = \beta(\sum_{m=0}^{N_{l-1}} w_{nm,\gamma h}^{t,l} \otimes h_n^{t,l-1} + b_n^l), \tag{10}$$

with β any *split activation* function. The full derivation of the backpropagation trough time of the QRNN can be found in [27].

4 Experiments on TIMIT

Quaternion-valued models are compared to their real-valued equivalents on two different benchmarks with the TIMIT phoneme recognition task [10]. First, an end-to-end approach is investigated based on QCNNs compared to CNNs in Section 4.2. Then, a more traditional and powerful method based on QRNNs compared to RNNs alongside with HMM decoding is explored in Section 4.3. During both experiments, the training process is performed on the standard $3,696$ sentences uttered by 462 speakers, while testing is conducted on 192 sentences uttered by 24 speakers of the TIMIT dataset. A validation set composed of 400 sentences uttered by 50 speakers is used for hyper-parameter tuning. All the results are from an average of three runs (3-folds) to alleviate any variation due to the random initialization.

4.1 Acoustic quaternions

End-to-end and HMM based experiments share the same quaternion input vector extracted from the acoustic signal. The raw audio is first transformed into 40-dimensional log Mel-filter-bank coefficients using the *pytorch-kaldi*[3] [26] toolkit and the Kaldi s5 recipes [31]. Then, the first, second, and third order derivatives are extracted. Consequently, an acoustic quaternion $Q(f,t)$ associated with a frequency f and a time-frame t is formed as:

$$Q(f,t) = e(f,t) + \frac{\partial e(f,t)}{\partial t}\mathbf{i} + \frac{\partial^2 e(f,t)}{\partial^2 t}\mathbf{j} + \frac{\partial^3 e(f,t)}{\partial^3 t}\mathbf{k}. \tag{11}$$

[2]`https://github.com/Orkis-Research/Pytorch-Quaternion-Neural-Networks`

[3]`https://github.com/mravanelli/pytorch-kaldi`

$Q(f,t)$ represents multiple views of a frequency f at time frame t, consisting of the energy $e(f,t)$ in the filter band at frequency f, its first time derivative describing a slope view, its second time derivative describing a concavity view, and the third derivative describing the rate of change of the second derivative. Quaternions are used to learn the spatial relations that exist between the different views that characterize a same frequency. Thus, the quaternion input vector length is $160/4 = 40$.

4.2 Toward end-to-end phonemes recognition

End-to-end systems are at the heart of modern researches in the speech recognition domain [43]. The task is particularly difficult due to the differences that exists between the raw or pre-processed acoustic signal used as input features, and the word or phonemes expected at the output. Indeed, both features are not defined at the same time-scale, and an automatic alignment method has to be defined. This section proposes to evaluate the QCNN compared to traditional CNN in an end-to-end model based on the connectionist temporal classification (CTC) method, to see whether the quaternion encoding of the signal, alongside with the quaternion algebra, help to better capture the acoustic signal nature and therefore better generalize.

4.2.1 Connectionist Temporal Classification

In the acoustic modeling part of ASR systems, the task of sequence-to-sequence mapping from an input acoustic signal $X = [x_1, ..., x_n]$ to a sequence of symbols $T = [t_1, ..., t_m]$ is complex due to:

- X and T could be in arbitrary length.
- The alignment between X and T is unknown in most cases.

Specially, T is usually shorter than X in terms of phoneme symbols. To alleviate these problems, connectionist temporal classification (CTC) has been proposed [11]. First, a softmax is applied at each timestep, or frame, providing a probability of emitting each symbol X at that timestep. This probability results in a symbol sequences representation $P(O|X)$, with $O = [o_1, ..., o_n]$ in the latent space O. A blank symbol $'-'$ is introduced as an extra label to allow the classifier to deal with the unknown alignment. Then, O is transformed to the final output sequence with a many-to-one function $g(O)$ defined as follows:

$$\left.\begin{array}{l} g(z_1, z_2, -, z_3, -) \\ g(z_1, z_2, z_3, z_3, -) \\ g(z_1, -, z_2, z_3, z_3) \end{array}\right\} = (z_1, z_2, z_3). \tag{12}$$

Consequently, the output sequence is a summation over the probability of all possible alignments between X and T after applying the function $g(O)$. Accordingly to [11] the parameters of the models are learned based on the cross entropy loss function:

$$\sum\nolimits_{X,T \in train} -\log(P(O|X)). \tag{13}$$

During the inference, a best path decoding algorithm is performed. Therefore, the latent sequence with the highest probability is obtained by performing argmax of the softmax output at each timestep. The final sequence is obtained by applying the function $g(.)$ to the latent sequence.

4.2.2 Model Architectures

A first 2D convolutional layer is followed by a maxpooling layer along the frequency axis to reduce the internal dimension. Then, $n = 10$ 2D convolutional layers are included, together with three dense layers of sizes 1024 and 256 respectively for real- and quaternion-valued models. Indeed, the output of a dense quaternion-valued layer has $256 \times 4 = 1024$ nodes and is four times larger than the number of units. The filter size is rectangular $(3, 5)$, and a padding is applied to keep the sequence and signal sizes unaltered. The number of feature maps varies from 32 to 256 for the real-valued models and from 8 to 64 for quaternion-valued models. Indeed, the number of output feature maps is four times larger in the QCNN due to the quaternion convolution, meaning 32 quaternion-valued feature maps (FM) correspond to 128 real-valued ones. Therefore, for a fair comparison, the number of feature maps is represented in the real-valued space (e.g., a number of real-valued FM of 256 corresponds to $256/4 = 64$ quaternion-valued neurons). The PReLU activation function is employed for both models [14]. A dropout of 0.2 and a L_2 regularization of $1e^{-5}$ are used across all the layers, except the input and output ones. CNNs and QCNNs are trained with the RMSPROP learning rate optimizer and vanilla hyperparameters [18] during 100 epochs. The learning rate starts at $8 \cdot 10^{-4}$, and is decayed by a factor of 0.5 every time the results observed on the validation set do not improve. Quaternion parameters including weights and biases are initialized following the adapted quaternion initialization scheme provided in [27]. Finally, the standard CTC loss function defined in [11] and implemented in [6] is applied.

4.2.3 Results and discussions

End-to-end results of QCNN and CNN are reported in Table 1. In agreement with our hypothesis, one may notice an important difference in the amount of learning parameters between real and quaternion valued CNNs. An explanation comes from the quaternion algebra. A dense layer with $1,024$ input values and $1,024$ hidden units contains $1,024^2 \approx 1$M parameters, while the quaternion equivalent needs $256^2 \times 4 \approx 0.26$M parameters to deal with the same signal dimension. Such reduction in the number of parameters have multiple positive impact on the model. First, a smaller memory footprint for embedded and limited devices. Second, and as demonstrated in Table 1, a better generalization ability leading to better performances. Indeed, the best PER observed in realistic conditions (w.r.t to the development PER) is 19.5% for the QCNN compared to 20.6% for the CNN, giving an absolute improvement of 0.5% with QCNN. Such results are obtained with 32.1M parameters for CNN, and only $8.1M$ for QCNN, representing a reduction factor of 3.96x of the number of parameters.

Table 1: Phoneme error rate (PER%) of CNN and QCNN models on the development and test sets of the TIMIT dataset. "Params" stands for the total number of trainable parameters, and "FM" for the feature maps size.

Models	FM	Dev.	Test	Params
CNN	32	22.0	23.1	3.4M
	64	19.6	20.7	5.4M
	128	19.6	20.8	11.5M
	256	**19.0**	**20.6**	**32.1M**
QCNN	32	22.3	23.3	0.9M
	64	19.9	20.5	1.4M
	128	18.9	19.9	2.9M
	256	**18.2**	**19.5**	**8.1M**

It is worth noticing that with much fewer learning parameters for a given architecture, the QCNN always performs better than the real-valued one. Consequently, the quaternion-valued convolutional approach offers an alternative to traditional real-valued end-to-end models, that is more efficient, and more accurate. However, due to an higher number of computations involved during the *Hamilton product* and to the lack of proper engineered implementations, the QCNN is one time slower than the CNN to train. Nonetheless, such behavior can be alleviated with a dedicated implementation in CUDA of the *Hamilton product*. Indeed, this operation is a matrix product and can thus benefit from the parallel computation of GPUs. In fact, a proper implementation of the *Hamilton product* will leads to a higher and more efficient usage of GPUs.

4.3 HMM-based phonemes recognition

A conventional ASR pipeline based on a HMM decoding process alongside with recurrent neural networks is also investigated to reach state-of-the-art results on the TIMIT task. While input features remain the same as for end-to-end experiments, RNNs and QRNNs are trained to predict the HMM states that are then decoded in the standard Kaldi recipes [31]. As hypothetized for the end-to-end solution, QRNN models are expected to better generalize than RNNs due to their specific algebra.

4.3.1 Model Architectures

RNN and QRNN models are compared on a fixed number of layers $M = 4$ and by varying the number of neurons N from 256 to $2,048$, and 64 to 512 for the RNN and QRNN respectively. Indeed, as demonstrated on the previous experiments the number of hidden neurons in the quaternion and real spaces do not handle the same amount of real-number values. Tanh activations are used across all the layers except for the output layer that is based on a softmax function. Models are optimized with RMSPROP [18] with vanilla hyper-parameters and an initial learning rate of $8 \cdot 10^{-4}$. The learning rate is progressively annealed using an halving factor of 0.5 that is applied when no performance improvement on the validation set is observed. The models are trained during 25 epochs. A dropout rate of 0.2 is applied over all the hidden layers [37] except the output. The negative log-likelihood loss function is used as an objective function. As for QCNNs, quaternion parameters are initialized based on [27]. Finally, decoding is based on Kaldi [31] and weighted finite state transducers (WFST) [22] that integrate acoustic, lexicon and language model probabilities into a single HMM-based search graph.

4.3.2 Results and discussions

The results of both QRNNs and RNNs alongside with an HMM decoding phase are presented in Table 2. A best testing PER of 18.5% is reported for QRNN compared to 19.0% for RNN, with respect to the best development PER. Such results are obtained with 3.8M, and 9.4M parameters for the QRNN and RNN respectively, with an equal hidden dimension of $1,024$, leading to a reduction of the number of parameters by a factor of 2.47 times. As for previous experiments the QRNN always outperform equivalents architectures in term of PER, with significantly less learning parameters. It is also important to notice that both models tend to overfit with larger architectures. However, such phenomenom is lowered by the small number of free parameters of the QRNN. Indeed, a QRNN whose hidden dimension is $2,048$ only has 11.2M parameters compared to 33.4M for an equivalently sized RNN, leading to less degrees of freedom, and therefore less overfitting.

Table 2: Phoneme error rate (PER%) of both QRNN, RNN models on the development and test sets of the TIMIT dataset. “Params" stands for the total number of trainable parameters, and "Hidden dim" represents the dimension of the hidden layers. As an example, a QRNN layer of "Hidden dim." $= 512$ is equivalent to 128 hidden quaternion neurons.

Models	**Hidden dim.**	**Dev.**	**Test**	**Params**
RNN	256	22.4	23.4	1M
	512	19.6	20.4	2.8M
	1,024	**17.9**	**19.0**	**9.4M**
	2,048	20.0	20.7	33.4M
QRNN	256	23.6	23.9	0.6M
	512	19.2	20.1	1.4M
	1,024	**17.4**	**18.5**	**3.8M**
	2,048	17.5	18.7	11.2M

The reported results show that the QRNN is a better framework for ASR systems than real-valued RNN when dealing with conventional multidimensional acoustic features. Indeed, the QRNN performed better and with less parameters, leading to a more efficient representation of the information.

5 Conclusion

Summary. This paper proposes to investigate novel quaternion-valued architectures in two different conditions of speech recognition on the TIMIT phoneme recognition tasks. The experiments show that quaternion approaches always outperform real-valued equivalents in both benchmarks, with a maximum reduction factor of the number of learning parameters of 3.96 times. It has been shown that the appropriate multidimensional quaternion representation of acoustic features, alongside with the *Hamilton product*, help QCNN and QRNN to well-learn both internal and external relations that exists within the features, leading to a better generalization capability, and to a more efficient representation of the relevant information through significantly less free parameters than traditional real-valued neural networks.

Future Work. Future investigation will be to develop multi-view features that contribute to decrease ambiguities in representing phonemes in the quaternion space. In this extend, a recent approach based on a quaternion Fourrier transform to create quaternion-valued signal has to be investigated.

References

[1] Ossama Abdel-Hamid, Abdel-rahman Mohamed, Hui Jiang, and Gerald Penn. Applying convolutional neural networks concepts to hybrid nn-hmm model for speech recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pages 4277–4280. IEEE, 2012.

[2] Paolo Arena, Luigi Fortuna, Giovanni Muscato, and Maria Gabriella Xibilia. Multilayer perceptrons to approximate quaternion valued functions. *Neural Networks*, 10(2):335–342, 1997.

[3] Paolo Arena, Luigi Fortuna, Luigi Occhipinti, and Maria Gabriella Xibilia. Neural networks for quaternion-valued function approximation. In *Circuits and Systems, ISCAS'94., IEEE International Symposium on*, volume 6, pages 307–310. IEEE, 1994.

[4] Nicholas A Aspragathos and John K Dimitros. A comparative study of three methods for robot kinematics. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 28(2):135–145, 1998.

[5] Anthony Maida Chase Gaudet. Deep quaternion networks. *arXiv preprint arXiv:1712.04604v2*, 2017.

[6] François Chollet et al. Keras. `https://github.com/keras-team/keras`, 2015.

[7] Ivo Danihelka, Greg Wayne, Benigno Uria, Nal Kalchbrenner, and Alex Graves. Associative long short-term memory. *arXiv preprint arXiv:1602.03032*, 2016.

[8] Steven B Davis and Paul Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. In *Readings in speech recognition*, pages 65–74. Elsevier, 1990.

[9] Sadaoki Furui. Speaker-independent isolated word recognition based on emphasized spectral dynamics. In *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'86.*, volume 11, pages 1991–1994. IEEE, 1986.

[10] John S Garofolo, Lori F Lamel, William M Fisher, Jonathan G Fiscus, and David S Pallett. Darpa timit acoustic-phonetic continous speech corpus cd-rom. nist speech disc 1-1.1. *NASA STI/Recon technical report n*, 93, 1993.

[11] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376. ACM, 2006.

[12] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*, pages 6645–6649. IEEE, 2013.

[13] Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. Lstm: A search space odyssey. *IEEE transactions on neural networks and learning systems*, 28(10):2222–2232, 2017.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[15] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *Signal Processing Magazine, IEEE*, 29(6):82–97, 2012.

[16] 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.

[17] Teijiro Isokawa, Tomoaki Kusakabe, Nobuyuki Matsui, and Ferdinand Peper. Quaternion neural network and its application. In *International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*, pages 318–324. Springer, 2003.

[18] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[19] Yann LeCun, Patrick Haffner, Léon Bottou, and Yoshua Bengio. Object recognition with gradient-based learning. In *Shape, contour and grouping in computer vision*, pages 319–345. Springer, 1999.

[20] Nobuyuki Matsui, Teijiro Isokawa, Hiromi Kusamichi, Ferdinand Peper, and Haruhiko Nishimura. Quaternion neural network with geometrical operators. *Journal of Intelligent & Fuzzy Systems*, 15(3, 4):149–164, 2004.

[21] Toshifumi Minemoto, Teijiro Isokawa, Haruhiko Nishimura, and Nobuyuki Matsui. Feed forward neural network with random quaternionic neurons. *Signal Processing*, 136:59–68, 2017.

[22] Mehryar Mohri, Fernando Pereira, and Michael Riley. Weighted finite-state transducers in speech recognition. *Computer Speech and Language*, 16(1):69 – 88, 2002.

[23] Tohru Nitta. A quaternary version of the back-propagation algorithm. In *Neural Networks, 1995. Proceedings., IEEE International Conference on*, volume 5, pages 2753–2756. IEEE, 1995.

[24] Titouan Parcollet, Mohamed Morchid, Pierre-Michel Bousquet, Richard Dufour, Georges Linarès, and Renato De Mori. Quaternion neural networks for spoken language understanding. In *Spoken Language Technology Workshop (SLT), 2016 IEEE*, pages 362–368. IEEE, 2016.

[25] Titouan Parcollet, Mohamed Morchid, and Georges Linares. Deep quaternion neural networks for spoken language understanding. In *Automatic Speech Recognition and Understanding Workshop (ASRU), 2017 IEEE*, pages 504–511. IEEE, 2017.

[26] Mirco Ravanelli, Parcollet Titouan, and Yoshua Bengio The PyTorch-Kaldi Speech Recognition Toolkit, 2018.

[27] Titouan Parcollet, Mirco Ravanelli, Mohamed Morchid, Georges Linarès, Chiheb Trabelsi, Renato De Mori, and Yoshua Bengio. Quaternion recurrent neural networks, 2018.

[28] Titouan Parcollet, Ying Zhang, Mohamed Morchid, Chiheb Trabelsi, Georges Linarès, Renato De Mori, and Yoshua Bengio. Quaternion convolutional neural networks for end-to-end automatic speech recognition. *arXiv preprint arXiv:1806.07789*, 2018.

[29] Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur. A time delay neural network architecture for efficient modeling of long temporal contexts. In *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[30] Soo-Chang Pei and Ching-Min Cheng. Color image processing by using binary quaternion-moment-preserving thresholding technique. *IEEE Transactions on Image Processing*, 8(5):614–628, 1999.

[31] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, Jan Silovsky, Georg Stemmer, and Karel Vesely. The kaldi speech recognition toolkit. In *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society, December 2011. IEEE Catalog No.: CFP11SRW-USB, 2011.

[32] Mirco Ravanelli, Philemon Brakel, Maurizio Omologo, and Yoshua Bengio. Improving speech recognition by revising gated recurrent units. *Proc. Interspeech 2017*, 2017.

[33] Mirco Ravanelli, Philemon Brakel, Maurizio Omologo, and Yoshua Bengio. Light gated recurrent units for speech recognition. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2(2):92–102, 2018.

[34] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. *arXiv preprint arXiv:1710.09829v2*, 2017.

[35] Haşim Sak, Andrew Senior, and Françoise Beaufays. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In *Fifteenth annual conference of the international speech communication association*, 2014.

[36] Stephen John Sangwine. Fourier transforms of colour images using quaternion or hypercomplex, numbers. *Electronics letters*, 32(21):1979–1980, 1996.

[37] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[38] Parcollet Titouan, Mohamed Morchid, and Georges Linares. Quaternion denoising encoder-decoder for theme identification of telephone conversations. *Proc. Interspeech 2017*, pages 3325–3328, 2017.

[39] Mark Tygert, Joan Bruna, Soumith Chintala, Yann LeCun, Serkan Piantino, and Arthur Szlam. A mathematical motivation for complex-valued convolutional networks. *Neural computation*, 28(5):815–825, 2016.

[40] Alexander Waibel, Toshiyuki Hanazawa, Geoffrey Hinton, Kiyohiro Shikano, and Kevin J Lang. Phoneme recognition using time-delay neural networks. In *Readings in speech recognition*, pages 393–404. Elsevier, 1990.

[41] Scott Wisdom, Thomas Powers, John Hershey, Jonathan Le Roux, and Les Atlas. Full-capacity unitary recurrent neural networks. In *Advances in Neural Information Processing Systems*, pages 4880–4888, 2016.

[42] D Xu, L Zhang, and H Zhang. Learning alogrithms in quaternion neural networks using ghr calculus. *Neural Network World*, 27(3):271, 2017.

[43] Ying Zhang, Mohammad Pezeshki, Philémon Brakel, Saizheng Zhang, Cesar Laurent Yoshua Bengio, and Aaron Courville. Towards end-to-end speech recognition with deep convolutional neural networks. *arXiv preprint arXiv:1701.02720*, 2017.