Artificial Intelligence Review
https://doi.org/10.1007/s10462-019-09752-1

A survey of quaternion neural networks

Titouan Parcollet[1,2] · Mohamed Morchid[1] · Georges Linarès[1]

Abstract

Quaternion neural networks have recently received an increasing interest due to noticeable improvements over real-valued neural networks on real world tasks such as image, speech and signal processing. The extension of quaternion numbers to neural architectures reached state-of-the-art performances with a reduction of the number of neural parameters. This survey provides a review of past and recent research on quaternion neural networks and their applications in different domains. The paper details methods, algorithms and applications for each quaternion-valued neural networks proposed.

Keywords Hypercomplex numbers · Quaternion neural networks · Deep Learning

Abbreviations

ML	Machine learning
AI	Artificial intelligence
(R, G, B)	Red, green, blue
Q{Model}	Quaternion{Model}
CVNN	Complex-valued neural network
NN	Neural network
MLP	Multilayer perceptron
DNN	Deep neural network
RNN	Recurrent neural network
CNN	Convolutional neural network
DAE	Denoising autoencoder
CAE	Convolutional autoencoder
HNN	Hopfield neural network
SVM	Support vector machine

✉ Titouan Parcollet
titouan.parcollet@alumni.univ-avignon.fr

Mohamed Morchid
mohamed.morchid@univ-avignon.fr

Georges Linarès
georges.linares@univ-avignon.fr

[1] Laboratoire Informatique d'Avignon (LIA), Université d'Avignon, Avignon, France

[2] ORKIS, Aix-en-Provence, France

Published online: 16 August 2019

PCA	Principal component analysis
LDA	Latent Dirichlet allocation
ReLU	Rectified linear unit
tanh	Hyperbolic tangent
eLU	Exponential linear unit
CRF	Cauthy–Riemann–Fueter
MSE	Mean squared error
GAN	Gaussian angular noise
PSNR	Peak signal to noise ratio
ABr	Average brightness
HOG	Histograms oriented gradient
PolSAR	Polarimetric synthetic aperture radar
CCS	Customer care service

1 Introduction

Machine learning (ML) has strongly impacted a broad spectrum of realistic applications, such as pattern recognition and natural language processing. In the last few years, machine learning has influenced the design of many systems that are massively employed to solve a variety of real-world related problems. With the concurrent growth of the amount of available data and computation power, artificial neural networks (NN) have reached state-of-the-art performances in numerous real-world applications (Mikolov et al. 2010; Simonyan and Zisserman 2014; Graves et al. 2013). In particular, deep neural networks (DNN) started to outperform other traditional ML algorithms with the introduction of a fast and efficient model training procedure (Hinton et al. 2006). Relevant research progress has been made on novel architectures, theories, and applications of deep neural models making DNNs among the core topics of modern artificial intelligence (AI). Particular attention has been made in modeling relations between observations and latent representations of them (Huang and LeCun 2006). Convolutional neural networks (CNN) have been proposed for considering spatial dependencies (Krizhevsky et al. 2012), such as shapes and edges of an image, while recurrent neural networks (RNN) have been conceived to model time dependencies of basic sequence elements (Schuster and Paliwal 1997).

Real world data are often multidimensional, and therefore, require a specific approach to consider relations within the information. For example, in the case of image processing, pixels are manipulated based on their three visual basic features: the red, green, and blue (R, G, B) channels. For robotic and human-pose estimation, points in space characterized by 3D coordinates are employed as inputs of systems. The automatic speech recognition (ASR) process is commonly based on time-frames that are defined with groups of Mel-filter-bank energies with first and second order time derivatives. It has been shown that local relations exist within these components of multidimensional entities (Matsui et al. 2004). Indeed, the (R, G, B) components of a given pixel represent a more complex color such as pink or brown, with a three coordinates position in the color space. These relations need to be captured by neural networks to well generalize and represent the multidimensional view of the pixel (Kusamichi et al. 2004; Isokawa et al. 2009). Nonetheless, mere real-valued architectures process these composed entities as independent elements in a big real-valued input vector. The local relations are therefore considered at the same level as global dependencies, and the model has to figure out the right balance during learning between internal and external dependencies. For example, Parcollet et al. (2018a) demonstrated that a real-valued CNN fails to capture the color information when trained on a gray-scale image, making it unusable in heterogeneous conditions. Then, Matsui et al. (2004) proved that a real-valued NN is not able to preserve the 3D shape of an input object when transformed into the 3D space, thus forgetting the relations between the coordinates. To address this issue, neural networks based on multi-dimensional numbers have been proposed.

From these set of novel high dimensional neural networks, complex-valued neural networks (CVNN) are nowadays known as an effective solution for tasks requiring two dimensional input vectors. Most of the CVNN applications are related to signal processing, where the magnitude and the phase are embedded in a complex number. The specific complex algebra allows CVNNs to preserve the relation between the magnitude and the phase information during the learning. It has been shown that CVNNs mostly outperformed or at least matched real-valued ones (Trabelsi et al. 2017; Hirose 2012; Mandic and Goh 2009; Aizenberg and Gonzalez 2018; Aizenberg et al. 2011) since the data processed are two dimensional and related. Nonetheless, complex representations are limited to two dimensions and by the nature of human spaced features, a three dimensional representation has to be investigated.

Quaternion neural networks (QNNs) recently became an active field of research (Gaudet and Maida 2018; Parcollet et al. 2018c; Takahashi et al. 2017; Ogawa 2016; Bayro-Corrochano et al. 2018). Quaternions are hyper-complex numbers that contain a real and three separate imaginary components, fitting perfectly to three and four dimensional feature vectors, such as for the (R, G, B) channels in image processing or three dimensional features for robotic. As for CVNNs, QNNs are able to learn the local relations that exist within its components through the *Hamilton product* (Eq. 14). QNNs were first introduced by Arena et al. (1994), with a specific backpropagation algorithm to efficiently learn QNNs in the same manner as real-valued neural networks. Following this proposal, many works investigated the basics properties of QNNs, focusing on each core components, such as activation functions (De Leo and Rotelli 1997), loss functions (Nitta 1995), parameters initialization (Gaudet and Maida 2018), or developing new architectures to better tackle real world multidimensional data as detailed in Sect. 5. A recent and increasing interest for QNNs is visible throughout a special session in the last ICASSP conference on *Learning Methods in Complex and Hypercomplex Domains*, the multiple *complex-valued and quaternionic Neural Networks* special sessions of IJCNN conferences, or even special issues on the IEEE TNNLS journal with *Complex- and Hypercomplex-Valued Neural Networks*, and *Hypercomplex Signal Processing* of Signal Processing from Elsevier.

Concurrently to CVNNs that have became a natural alternative to real-valued representations, this survey proposes to introduce, motivate and list all the major and milestones works on quaternion-valued neural networks in a comprehensive way. Thereby, the reader will easily find out modern QNNs to further investigate their properties in novel realistic conditions. Motivations of the use of QNNs are described in the tale below, while the quaternion basics such as the algebra, the rotation principle and examples of applications are reported in Sect. 3. Then, a formal definition of a straightforward quaternion-valued neural network, including the definition of activation functions, the forward- and the back-propagation phases, the basic properties of a quaternion-valued neuron alongside with the recent advances in quaternion-valued learning algorithms are introduced in Sect. 4. Finally, QNN architectures in various application domains are summarized in Sect. 5.

2 Motivations for quaternion-valued neural networks

A major challenge of machine learning is to compute appropriate representations of large amounts of data in a robust latent space. For this purpose, a good model has to efficiently encode local relations within the input features (Tokuda et al. 2003; Matsui et al. 2004), such as between the Red, Green, and Blue (R, G, B) channels of a single pixel (Parcollet et al. 2018a), as well as structural relations, such as those describing edges or shapes composed by groups of pixels. An adequate representation also offers a positive side effect, by reducing the number of neural parameters needed to well-learn the input features, leading to a natural solution to the overfitting phenomenon. In the following, we detail the motivations to employ a quaternion-valued NN instead of a real-valued one to code inter and intra features dependencies with less parameters.

As a first step, a suitable representation of multidimensional data has to be found for encoding the relations between the properties of observable entities. For example, an efficient way to represent the information composing an image is to consider each pixel 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. Also, in a NN with real-valued parameters, the latent relations between the RGB components of a given pixel may not be suitably encoded in the latent space since the parameter estimation may fail to associate all the appropriate relations to all the pixels composing an image. This problem is effectively solved by replacing real numbers by quaternions. Indeed, quaternions are fourth dimensional and allow one to build and process entities made of up to four elements. The quaternion algebra and more precisely the *Hamilton product* (Eq. 14) allows quaternion neural networks to capture these internal latent relations within the features of a quaternion. It has been shown that QNN are able to restore the spatial relations within 3D coordinates (Matsui et al. 2004), and within color pixels (Isokawa et al. 2003), while real-valued NN failed. This is easily explained by the fact that the quaternion-weight components are shared through multiple quaternion-input parts during the *Hamilton product*, creating relations within the elements, as depicted in Fig. 1. Indeed, Fig. 1 shows that the multiple weights required to code latent relations within a 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*. In particular, Fig. 1 demonstrates that a slight change in any of the four components of the quaternion input signal will result in a completely different encoding in the output quaternion space, describing the ability to learn internal relations within the quaternion input features.

Then, while bigger neural networks allow better performances, quaternion neural networks make it possible to obtain comparable or better results on the same task, but with four times less neural model parameters. Indeed, a 4-number quaternion weight linking two 4-number quaternion units only has four degrees of freedom, whereas a standard neural net parametrization has $4 \times 4 = 16$, i.e., a fourfold 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.

3 Quaternion definition

Quaternion numbers $\mathbb{H}$ are part of the hyper-complex numbers as a non commutative extension of complex numbers. They were discovered by the Irish mathematician William Rowan

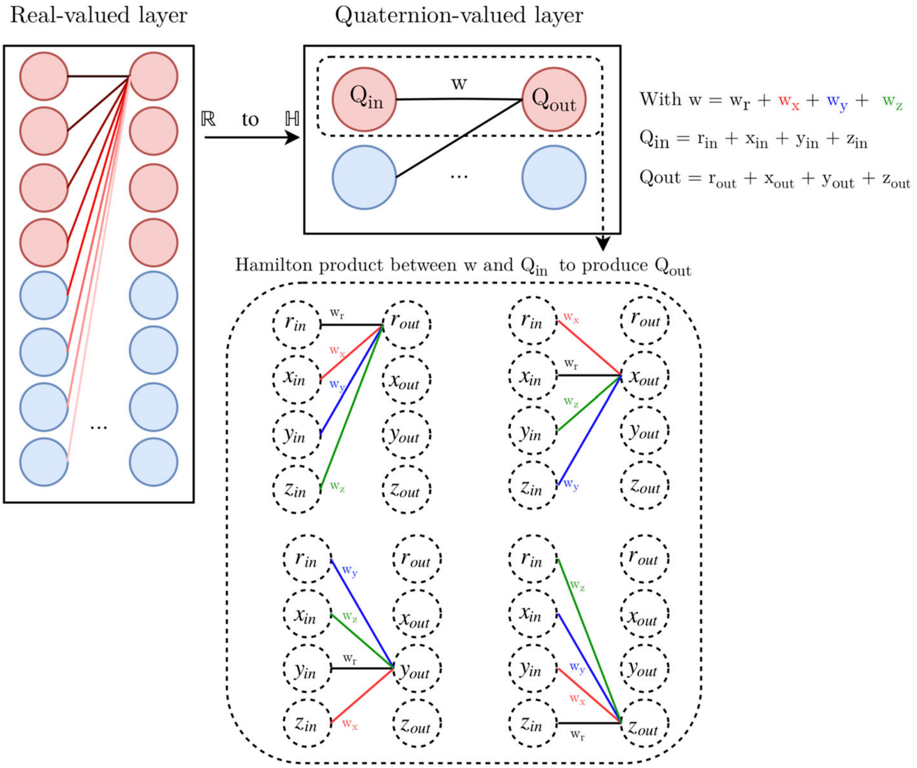
Real-valued layer
Quaternion-valued layer
ℝ to ℍ
w
Q_in
Q_out
...
With w = w_r + w_x + w_y + w_z
Q_in = r_in + x_in + y_in + z_in
Qout = r_out + x_out + y_out + z_out
Hamilton product between w and Q_in to produce Q_out
r_in
x_in
y_in
z_in
r_out
x_out
y_out
z_out
w_r
w_x
w_y
w_z

Fig. 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. 14), compared to a standard real-valued layer (left) (Parcollet et al. 2018b)

(Hamilton 1844). Strong of a well-designed algebra, quaternions numbers provide a suitable solution to the gimbal lock issue of Euler angles (Diebel 2006). The gimbal lock phenomenon is characterized by the loss of one degree of freedom in a three-dimensional space, when two axis of rotations are aligned. Such alignment degenerates the system to a two-dimensional space, preventing the use of 3D rotations. The division ring of quaternions constitutes the four-dimensional vector space with bases 1, **i**, **j**, and **k**. In fact, while complex numbers have been introduced to represent two-dimensional transformations, quaternions have been invented to extend these manipulations to the three-dimensional space. Therefore, a quaternion Q is written as:

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

As for complex number the quaternion algebra is ensured by mathematical relations between the imaginary components:

$$\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1. \tag{2}$$

Such relations are at the heart of quaternions, and make the $\mathbb{H}$ algebra an efficient tool in $\mathbb{R}^4$ to represent spatial rotations.

3.1 Quaternion algebra

In a quaternion, r is the real part or scalar part named s while $x\mathbf{i}$, $y\mathbf{j}$, and $z\mathbf{k}$ are the components of the imaginary part $x\mathbf{i} + y\mathbf{j} + z\mathbf{k}$, or vector part also noted $\overrightarrow{v}$. Therefore, Q can be summarized as:

$$Q = (s, \overrightarrow{v}). \tag{3}$$

Considering $\mathbb{R}$ and the complex plane $\mathbb{C}$ as subsets of the hyper-complex algebra, Q can be projected back to either of these spaces following:

$$Q_{mat} = \begin{bmatrix} r & -x & -y & -z \\ x & r & -z & y \\ y & z & r & -x \\ z & -y & x & r \end{bmatrix}, \tag{4}$$

for the real field and:

$$Q_{mat} = \begin{bmatrix} (r + x\mathbf{i}) & (y + z\mathbf{i}) \\ (-y + z\mathbf{i}) & (r - y\mathbf{i}) \end{bmatrix}, \tag{5}$$

for the complex one. The real-valued matrix representation turns out to be particularly efficient for computations (Fig. 3), due to the parallelization capabilities of modern graphic processing units (GPU). The conjugate Q^* is an involution of Q and is noted:

$$Q^* = r1 - x\mathbf{i} - y\mathbf{j} - z\mathbf{k}. \tag{6}$$

The norm of Q denoted $||Q||$ is the same as Euclidean norm in $\mathbb{R}$, and complex norm in $\mathbb{C}$ but in the four dimensional space $\mathbb{H}$:

$$||Q|| = \sqrt{r^2 + x^2 + y^2 + z^2}. \tag{7}$$

Therefore, a normalized quaternion or unit quaternion $Q^{\triangleleft}$ can be easily expressed as:

$$Q^{\triangleleft} = \frac{Q}{||Q||}. \tag{8}$$

In the same manner as for complex numbers, a unit quaternion lies on a unit hyper-sphere and is used to precisely describe a rotation. The inverse Q^{-1} of Q is written as:

$$Q^{-1} = \frac{Q^*}{||Q||^2}. \tag{9}$$

It is important to note that $Q^* = Q^{-1}$ if Q is a unit quaternion. Finally, the polar form of a quaternion can also derived as:

$$Q = |Q|(cos\theta + nsin\theta) = |Q|e^{n\theta}, \tag{10}$$

with

$$cos\theta = \frac{s}{||Q||}, \quad sin\theta = \frac{||\overrightarrow{v}||}{||Q||}, \quad n = \frac{\overrightarrow{v}}{||\overrightarrow{v}||}, \tag{11}$$

and $||Q||$ the norm of Q defined in Eq. 7.

3.2 Hamilton product and rotations

The quaternion algebra have been invented to represent and manipulate rotations and orientations on the three-dimensional space. These properties are used by quaternion-valued neural networks (Nitta 1995; Fortuna et al. 2001). More precisely, quaternion rotations lie on an infinite group of unit quaternions describing the unit hyper-sphere (four-dimensional). It is also interesting to note that pure unit quaternions (unit quaternions whose real part is equal to 0) lie on the unit 3D sphere. Consequently, a unit quaternion $Q^{\triangleleft}$ represents a rotation by an angle θ around a unit axis vector (of length 1) $\overrightarrow{v}$ as:

$$Q^{\triangleleft} = \left(cos\frac{\theta}{2} + \overrightarrow{v}\, sin\frac{\theta}{2} \right). \tag{12}$$

Let us define $p = (x, y, z)$, a point or a pure quaternion ($p = 0 + x\mathbf{i} + y\mathbf{j} + z\mathbf{k}$) of the three-dimensional space. p is rotated to p' following $Q^{\triangleleft}$ with:

$$p' = Q^{\triangleleft} \otimes p \otimes Q^{\triangleleft -1}, \tag{13}$$

with $\otimes$ the *Hamilton product* between two quaternions Q_1 and Q_2 defined as:

$$\begin{aligned} Q_1 \otimes Q_2 = {} & (r_1 r_2 - x_1 x_2 - y_1 y_2 - z_1 z_2) \\ & + (r_1 x_2 + x_1 r_2 + y_1 z_2 - z_1 y_2)\boldsymbol{i} \\ & + (r_1 y_2 - x_1 z_2 + y_1 r_2 + z_1 x_2)\boldsymbol{j} \\ & + (r_1 z_2 + x_1 y_2 - y_1 x_2 + z_1 r_2)\boldsymbol{k}. \end{aligned} \tag{14}$$

The *Hamilton product* is used to performs composition of rotations in $\mathbb{R}^3$. Consequently, the quaternion $Q_3 = Q_1 \otimes Q_2$ is equivalent to the rotation Q_1 followed by the rotation Q_2. It is also worth underlying that the commutativity does not hold in the quaternion space:

$$Q_1 \otimes Q_2 \neq Q_2 \otimes Q_1. \tag{15}$$

Indeed, an object rotated by Q_1 and Q_2 will not be in the same position as it was rotated by Q_2 and Q_1. Figure 2 shows the rotation angle and axis of $Q^{\triangleleft}$.

3.3 Applications of quaternion numbers

Quaternions are massively employed since the raise of computers and GPU cards, with direct applications in spacecraft modeling and computer graphics. Recently, a large number of real-world applications of quaternions have been proposed due to the wide variety of tasks that suit 3D-transformations such as rotations in the 3D space or image processing. Consequently, quaternions are used in rendering of 3D scenes in computer graphics (Shoemake 1985). Pletinckx (1989) presents a complete set of methods to use quaternions to transform object in space, such as linear interpolation of quaternions, splining quaternions or a comparison with Euler angles. More precisely, the authors report use cases for quaternions to render, model and animate simple objects using unit quaternions. Hyper-complex numbers are also naturally suitable for applications of motions tracking (Yun and Bachmann 2006). A well-adapted Fourier transform based on quaternion algebra has also been investigated for color image processing (Sangwine 1996). The later claims that a quaternion representation of color images alongside with this dedicated Fourier transform, allow the system to build more accurate filters, and better preserve the color information. Moreover, automated systems that evolve

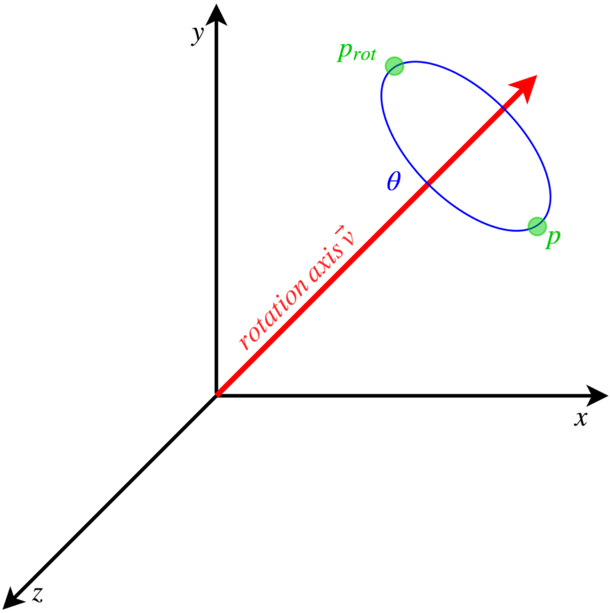

Fig. 2 The point $p = (x, y, z)$ is rotated by a unit quaternion describing a rotation axis, and a rotation angle

and interact in the three dimensional space such as robots, take advantage of the compact representation of quaternions for rotations as shown by Dornaika and Horaud (1998) and Chou (1992), efficiently solving the hand-eye calibration problem. Surprisingly, quaternions also provide an adequate representation to other tasks (Karney 2007) less related to 3D-spaces. Indeed, they were employed as a framework to represent the orientation of different molecules in realistic simulations for molecular modeling. Overall, modern applications of quaternions take advantage of the better representation in the quaternion space.

4 Fundamentals of quaternion-valued neural networks

Based on the motivations discussed in Sect. 2, novel neural architectures and methods modeling higher dimensional spaces have been introduced. Nitta and de Garis (1992) have proposed a backpropagation algorithm for neural networks based on three dimensional vector product. Although 3D vectors are close to quaternions, they are not processed with the same algorithms. Arena et al. (1994) have first proposed a quaternion-valued multilayer perceptron (QMLP), with an adapted backpropagation algorithm that takes into account the specificities of the *Hamilton product*. In particular, the experiments on quaternion-valued functions approximation have shown that the QMLP outperforms the real-valued MLP with a smaller number of connections leading to a smaller processing time required for learning the model. Concurrently, a complete derivation of the quaternion backpropagation algorithm for QMLP has also been developed by Nitta (1995). This section introduces the basic building blocks of a backropagation based QNN, including activation functions (Sect. 4.1), the forward (Sect. 4.2), backward and update (Sect. 4.3) phases. A recent breakthrough on the quaternion-valued signal processing field with the $\mathbb{GHR}$ calculus and well-designed learning algorithms (Sect. 4.4) is also presented. Finally, the basic properties of a quaternion neuron are discussed (Sect. 4.5).

4.1 Activation functions

Activation functions are mandatory to solve non-linear tasks. Commons activation functions are composed with the sigmoid, the hyperbolic tangent (tanh), or the more recent rectified linear unit (ReLU) (Glorot and Bengio 2010; Nair and Hinton 2010). In spite of an astonishing number of previous researches on real-valued non-linear activation functions, very few quaternion studies on these functions have been investigated so far. In fact, all of the existing quaternion-valued neural networks activation functions are based on complex extensions (Arena et al. 1997, 1993). This difficulty mainly comes from the Cauthy–Riemann–Fueter (CRF) equation that defines the analytic condition for the quaternionic functions. Indeed, functions that satisfy the CRF equation are only linear functions or constants. In regard to these principles, it is difficult to introduce non-linearity to the update phase of the quaternion-valued neurons. Nonetheless, it has been demonstrated (De Leo and Rotelli 1997; Isokawa et al. 2012) that locally analytic quaternion activation functions are adapted to train QNNs with the standard backpropagation algorithm. Thereby, two classes of quaternion-valued activation functions have emerged and are presented thereafter.

4.1.1 Fully quaternion-valued functions

Fully quaternionic activation functions are extensions to the hypercomplex domain of real-valued functions, such as sigmoid or tanh functions. These quaternion-valued activation functions reach better results (Ujang et al. 2010), but need a careful training phase due to an important number of singularities that can drastically alter the QNN performances. As an example, the hyperbolic tangent of a quaternion Q is defined as:

$$tanh(Q) = \frac{e^{2Q} - 1}{e^{2Q} + 1}, \tag{16}$$

with

$$e^{Q} = e^{s} \left(cos|\overrightarrow{v}| + \frac{\overrightarrow{v}}{|\overrightarrow{v}|} sin|\overrightarrow{v}| \right) \tag{17}$$

where s is the real component while $\overrightarrow{v}$ is the vectorial part (or complex part) of the quaternion Q.

4.1.2 Split functions

A more frequent and simpler solution is the so called *split activation* (Arena et al. 1994; Ujang et al. 2011). In the latter, a conventional, real-valued function is applied component wise to a quaternion, alleviating the singularities. In particular, Arena et al. (1993) demonstrated that the universal approximation theorem holds true when applying *split activation* to complex- and quaternion- valued neural networks, enabling the learning of various QNN architectures. Despite a lot of applications and a straightforward training phase, *split activation* functions are reported to obtain worst results compared to fully quaternionic functions (Mandic et al. 2011; Ujang et al. 2011). Indeed, *split activation* functions map quaternion numbers back to the real-valued space by ignoring the nature of the relation that exists between the components. The impact on the multi-view learning ability of the QNN is reduced by the fact that this relation is captured by the *Hamilton product*. In particular, with *split activation* functions, each quaternion-valued layer learns an internal relation scheme independently of the other layers. Let $\alpha(Q)$ be a *split activation* function applied to the quaternion Q:

$$\alpha(Q) = f(r) + f(x)\mathbf{i} + f(y)\mathbf{j} + f(z)\mathbf{k}, \tag{18}$$

with f corresponding to any standard and real-valued activation function (e.g, sigmoid, tanh, ReLU, eLU,…,). The split approach is adopted in the rest of the QMLP equations considering its usage in the first proposition of the QMLP (Arena et al. 1994) alongside with its higher number of applications.

4.2 Forward phase

In a quaternion-valued feed-forward neural network layer, all parameters are quaternions, including inputs, weights, biases and outputs. Let us introduce the traditional QMLP first described by Arena et al. (1997). The model is made of M layers of N nodes or neurons, whose number depends on the layer. Let x be the input to a node. Let N_l be the number of neurons contained in the layer l $(1 \leq l \leq M)$ and M be the number of layers of the QMLP. b_n^l is the bias of the neuron n $(1 \leq n \leq N_l)$ from the layer l. Given a set of P quaternion input patterns x_p $(1 \leq p \leq P)$ and a set of labels t_p associated to each x_p, the output γ_n^l $(\gamma_n^0 = x_p^n)$ of the neuron n of the layer l is given by:

$$\gamma_n^l = \alpha(S_n^l), \tag{19}$$

with

$$S_n^l = \sum_{m=0}^{N_{l-1}} w_{nm}^l \otimes \gamma_m^{l-1} + b_n^l, \tag{20}$$

and α any *split activation* function. It is interesting to note that few variations of the forward equation have been proposed, to fully expose the capabilities of the quaternion algebra. Indeed, and as proposed by Yoshida et al. (2005), the weights and the inputs can be inverted as:

$$S_n^l = \sum_{m=0}^{N_{l-1}} \gamma_m^{l-1} \otimes w_{nm}^l + b_n^l, \tag{21}$$

to create an inverted quaternion-valued neural network. Then, the rotation capability of quaternions can be exploited by replacing the forward equation with:

$$S_n^l = \sum_{m=0}^{N_{l-1}} w_{nm}^l \otimes \gamma_m^{l-1} \otimes (w_{nm}^l)^{-1} - b_n^l, \tag{22}$$

as suggested by Buchholz and Sommer (2000), denoting a spinor QNN. Empirical differences between these equations have been investigated and are detailed in Sect. 5. For convenience, the forward equation introduced by Arena et al. (1997) (Eq. 20) is used as a reference for the rest of the demonstrations.

Finally, the output layer of a QMLP is based on a *split activation* function with respect to the target task. Consequently, the sigmoid or tanh functions are applied component-wise to the output of the QMLP for a quaternion approximation task (Arena et al. 1997), and a split softmax is used to obtain a posterior distribution for classification purposes (Parcollet et al. 2016). Indeed, target classes are often defined as real-valued binary vectors, and it is therefore convenient to decorrelate the components of the quaternion output with a split softmax.

4.3 Backward and update phases

The loss function E is a crucial element that quantifies how far the inferred predictions S^M are from the target distribution t_p. Numerous real-valued loss functions have been defined and used for different tasks. Indeed, the mean squared error and absolute error (Willmott and Matsuura 2005) are used for approximation, while the cross-entropy and binary cross-entropy (De Boer et al. 2005), negative log-likelihood (Platt et al. 1999), and margin classifier fit better to classification problems. In its original paper, Arena et al. (1994) proposed an extension of the mean squared error (MSE) to quaternions (QMSE) by solely replacing real-numbers by hyper-complex numbers:

$$E = \frac{1}{N}\sum_{n=1}^{N}[(t_{rpn} - S_{rn}^M)^2 + (t_{ipn} - S_{in}^M)^2 + (t_{jpn} - S_{jn}^M)^2 + (t_{kpn} - S_{kn}^M)^2] \tag{23}$$

One may notice that any traditional and real-valued loss function can be applied to a real-valued QMLP output layer. Starting from the QMSE, the backpropagation of QNNs is an extension of the standard backpropagation for real-valued NNs. Its full version is derived by Nitta (1995) and Parcollet et al. (2018b), while a brief summary is introduced bellow. The gradient with respect to the loss E is expressed for each components of the quaternion weights w^l that compose the matrix W^l:

$$\Delta^l = \frac{\partial E}{\partial W^l} = \frac{\partial E}{\partial W_r^l} + \mathbf{i}\frac{\partial E}{\partial W_i^l} + \mathbf{j}\frac{\partial E}{\partial W_j^l} + \mathbf{k}\frac{\partial E}{\partial W_k^l}. \tag{24}$$

The gradient at the output layer ($l = M$), quantifying the error with respect to the target vector t_p for each neuron n, is obtained by deriving each term of Eq. 24 following the chain rule:

$$\begin{aligned}\Delta_n^{l=M} &= (t_{rpn} - S_{rn}^{l=M}) + \mathbf{i}(t_{ipn} - S_{in}^{l=M}) + \mathbf{j}(t_{jpn} - S_{jn}^{l=M}) + \mathbf{k}(t_{kpn} - S_{kn}^{l=M}) \\ &= (t_{pn} - S_n^{l=M}),\end{aligned} \tag{25}$$

with $S_n^{l=M}$ the predicted output. In the same manner, the gradients of the hidden layers parameters are derived following the chain rule:

$$\Delta_n^l = \sum_{n=1}^{N^{l+1}} w_{h,n}^{*l+1} \otimes (\Delta_h^{l+1} \odot \alpha'(S_n^{l+1})), \tag{26}$$

with $w_{h,n}^{*l+1}$ the conjugate of $w_{h,n}^{l+1}$, Δ_h^{l+1} the gradient observed at the next layer, and $\alpha'(S_n^{l+1})$ the derivative of the *split activation* function. It is important to note that all the parameters are quaternions, and that the product between the gradient and the derivative is a component-wise product. Finally, biases and each weight w that connects the neuron n to m of the layer l are updated based on the previously computed gradients Δ_n^l:

$$w_{nm}^l = w_{nm}^l + \lambda\Delta_n^l \otimes S_m^{*l-1}, \quad b_n^l = b_n^l + \lambda\Delta_n^l, \tag{27}$$

with λ the learning rate.

4.4 Alternative GHR based learning methods for quaternion neural networks

Most of the current quaternion-valued neural networks implementations and experimentations are based on the usual backpropagation algorithm described above. Nonetheless, a recent breaktrough in the quaternion-valued signal processing field derives other learning algorithms that are more suitable to quaternion signal. The $\mathbb{HR}$ calculus first proposed by Mandic et al. (2011) calculates derivatives of both analytic and non-analytic functions of quaternions to compute gradients that are needed to learning algorithms. Nonetheless, the $\mathbb{HR}$ suffers from the fact that the traditional product is not valid (Xu et al. 2015). Therefore, Xu et al. (2015) have proposed a generalized version of the $\mathbb{HR}$ calculus called G$\mathbb{HR}$ calculus. Based on the G$\mathbb{HR}$ framework, new learning algorithms have been developed, such as the resilient backpropagation, the conjugate and the scaled conjugate gradient, or the Gauss-Newton algorithms (Popa 2018; Xu et al. 2017). G$\mathbb{HR}$ based algorithms have been compared to the traditional bacpropagation algorithm in a toy task of time-prediction forecasting by Popa (2018), and have reached higher accuracies. Unfortunately, and despite an increasing interest on such algorithms, the training methods have not still been investigated on real world and scalable tasks. Consequently, the rest of the paper focuses on the traditional backpropagation algorithm and its applications.

4.5 Properties of quaternion-valued neurons

The promising results observed on the various tasks described in Sect. 5.1 are mostly due to the basic properties of each unit that composes a QNN. Nitta (2004) has shown that the fundamental properties of quaternion-valued neurons differ from real-valued ones. Indeed, the decision boundary of a single quaternionic neuron consists of four hyperplanes which intersect each other orthogonally. Such characteristic can intuitively be understood by considering the *Hamilton product*. In Eq. 14, it is clear that each part of the resulting quaternion can be transformed to an hyperplane equation composed of mutual coefficients. An experimental evidence is given by Nitta (2004) with the 4-bits parity problem. Such a task can be solved with a single quaternion-valued neuron compared to four real-valued ones, resulting in a higher generalization capability. Indeed, a 4-number quaternion weight linking two 4-number quaternion units only has four degrees of freedom, whereas a standard neural net parametrization have $4 \times 4 = 16$, i.e., a fourfold saving in memory. Nonetheless, ones have to take heed when computing the quaternion product. Indeed, the commutativity does not hold in the quaternion space, and the product between weights and inputs differs from the product between inputs and weights (partially commutative).

5 Architectures and applications

Numerous applications and extensions of the vanilla QMLP have been successfully proposed, and new architectures of quaternion-valued neural networks have emerged. From the perspective of promoting and helping future researches on QNNs, this section aims to group and presents these methods.

5.1 Quaternion-valued multilayer perceptron (QMLP)

Alongside with strong theoretical proofs, the original QMLP received multiple applications due to the natural multidimensionality of realistic applications. Indeed, despite the limited abstraction capability of mere multilayer perceptrons, many simple task have been better solved based on the QMLP or its variants.

5.1.1 Image processing

The quaternion algebra offers a natural way to express and process multidimensional input features as single entities. Such definition fit particularly well to the image processing area. Indeed, pixels are the basic elements employed during the image processing, and these pixels are usually defined by three components: red, green and blue (R, G, B) values.

Quaternions represent an efficient way to code each pixel as a whole element by composing quaternion-valued pixels as $Q = 0 + R\mathbf{i} + G\mathbf{j} + B\mathbf{k}$. In particular, and based on a spinor QMLP first introduced by Buchholz and Sommer (2000), Isokawa et al. (2003) have proposed a modified spinor QMLP to learn a compact representation of the R, G, and B values of pixels that compose a color image. This study proposes to compare a real-valued MLP to a spinor QMLP in a compression of color images task. Both models are trained to encode an input image into a smaller subspace throughout a reduced hidden layer, before reconstructing the image at the output from this hidden space. Such process is also known as encoder–decoder or autoencoder (Hinton and Salakhutdinov 2006) models. The quality of the reconstruction of the model is evaluated visually as well as with the Peak Signal to Noise Ratio (PSNR). The proposed spinor QMLP is equivalent to a normalized spinor QMLP (Eq. 22):

$$S_n^l = \sum_{m=0}^{N_{l-1}} \frac{w_{nm}^l \otimes \gamma_m^{l-1} \otimes (w_{nm}^l)^{-1}}{\|w_{nm}^l\|} - b_n^l. \tag{28}$$

The spinor QMLP rotates the pure quaternion input vector based on its weights, while the tradional QMLP performs a composition of rotations. Indeed, a pure input quaternion feature x ($x = \gamma_m^0$ with $x_r = 0$) that goes trough a spinor QMLP will be rotated in the 3D space by the quaternion weights. Conversely, a quaternion input feature passing a standard QMLP will represent a new rotation, that is a composition of x and the weights, due to the unique *Hamilton product* linking both terms (Sect. 3.2). For a fair comparison on the experiments, both QMLP and MLP are composed of roughly the same number of parameters. The training image is voluntary strongly based on the red color, while the test image is made of multiple bright colors to highlight the capability of QMLPs to learn the internal relations of each pixel between R, G, B features. The experiments show that the spinor QMLP outperforms the MLP with both a better PSNR value and a better visual reconstruction of the test image. Indeed, the MLP tends to produce a reddish version of the test image while the QMLP produces an almost perfect one. In fact, the QMLP have learned the whole color space due to the *Hamilton product*, compared to solely the training image color space for the MLP.

The same architectures are then benchmarked on a colors extraction task of gloomy pictures by Kusamichi et al. (2004). In the latter, the authors have used the spinor QMLP to better learn the mapping from a gloomy version of an image to a brightest one. A new metric of average brightness (ABr) is introduced to quantify the brightness of an image. The results show again that ABr, PSNR and visual reconstruction are by far superior with the spinor QMLP due to the natural quaternion-valued pixel representation, alongside with the *Hamilton product*.

In an other images processing domain, Greenblatt et al. (2013) have proposed a complex system to diagnose the Gleason grade that measures the severity of prostate cancers. As motivated by Greenblatt et al. (2013), traditional computer-aided diagnosis (CAD) systems commonly process input images in their monochrome version, or consider the three channels of color images independently. In this context, quaternion numbers are used to manage color pixels. Therefore, the work introduced a model that benefits from the R, G, B representation of quaternions, to produce quaternion wavelets (Soulard and Carré 2011) that are then fed into a traditional QMLP. The use of quaternion wavelets is a key component of this work. Indeed, while the processing of color pixels with quaternion numbers is common with QNNs, it is new to associate a quaternion features extractor to the input features. In this particular context of disease diagnose, quaternion wavelets allow the extraction and the representation in the quaternion domain, of important characteristics of an image such as shapes, edges, depth or brightness. In fact, quaternion wavelets further take advantage of the quaternion pixels representation, by extracting high-level quaternion characteristics of an image before classification with a QNN. Finally, Greenblatt et al. (2013) proposes to couple QNN predictions with a real-valued support vector machine (SVM) (Hearst et al. 1998) to increase the performances of the system. The experiments show that the QMLP obtains a significantly higher accuracy over existing works.

Facial expression recognition also take advantages of QMLPs. Indeed, Takahashi et al. (2014) have merged the recent approach of Histograms of Oriented Gradients (HOG) for human detection (Dalal and Triggs 2005) with the quaternion representation to determine human facial expression. HOG are a feature descriptor that allows a representation of the shape or edges of an object within an image, based on the direction of the gradients. Since, HOG features are known to be high-dimensional, they are first reduced trough a Principal Component Analysis (PCA) (Jolliffe 2011) before composing a quaternion input vector. Then, a standard QMLP is used to recognize the facial expressions. The results show that the QMLP outperforms real-valued MLP due to the better high-dimensional representation of HOG in the latent space.

More recently, QMLP were applied to Polarimetric Synthetic Aperture Radar (PolSAR) land classification (Shang and Hirose 2014; Kinugawa et al. 2018). The PolSAR system extracts and measures the complex scattering matrix for each pixel of an observed area. In both works, the authors have mapped image pixels in a three-dimensional Poincaré sphere. More precisely, Shang and Hirose (2014) used a spinor QMLP (Eq. 28) to classify each pixel of an image with four classes corresponding to *forest*, *grass*, *town* or *lake*. It is shown that the quaternion-valued model outperformed common real-valued methods, while being more robust to the height variations of the map, due to the quaternion representation of the 3D Poincaré coordinates. Kinugawa et al. (2018) have proposed to further improve the performances of QNNs with PolSAR systems, by introducing a new activation function. Indeed, the authors have demonstrated that the quaternion *split activation* used by Shang and Hirose (2014) alters the mapping of the signal, by changing both the length and direction of the quaternion vector. This is easily explained by the decorrelation of the components induced by the *split activation*. To address this issue, the authors proposed an *isotropic activation function* based on the scaling of a quaternion by the hyperbolic tangent of its norm. Such activation preserves the representation of the 3D Poincaré sphere by only scaling each component of the pure quaternion, and also offers better performances in term of accuracy on the considered land classification task.

5.1.2 Text processing

The images processing tasks described in the previous section detail approaches based on quaternion neural networks in multidimensional input spaces. Text processing tasks differ but also involve multi-view of a same basic element composing the textual document. Text processing tasks commonly rely on very high-dimensional features that allow the model to express the document content in different level of abstraction throughout the successive hidden layers. Indeed, input feature sizes are often related to the text embedding or vocabulary sizes, and can easily contain more than thousands dimensions. The trick to benefits from QNN is therefore to find an appropriate task where the input is easily expressed with a vector of three or four dimensions. Parcollet et al. (2016) first proposed an adapted document segmentation for a theme classification task of telephone conversations from a customer care service (CCS) of a public transportation system. Indeed, CCS conversations can be split in three elements: The first and second parts are made of the customer and agent speech turns respectively, while the third part represents the whole conversation. Each element is then projected in a Latent Dirichlet Allocation LDA (Blei et al. 2003) subspace of N topics (groups of words thematically close), to obtain a topic vector of size $N \times 3$. Such vector is easily transformed into a $N \times 4$ purely imaginary quaternion vector by padding the real part with zeros in the same manner as for the image processing. Consequently, the authors obtain quaternion input features with each imaginary component equal to a part of the conversation. The initial assumption is that a latent relation between the agent, the customer and the whole document exists, and can be modeled on the hidden space of a QNN. A first set of experiments based on a vanilla QMLP shows better classification accuracies on the DECODA dataset (Bechet et al. 2012) than real-valued MLP.

However quaternion-based systems suffer from the facts of being novel and barely investigated. Indeed, many methods have been developed for real-valued neural networks to help the model to learn faster, and to generalize better, such as dropout noise (Srivastava et al. 2014), or the more recent batch normalization (Ioffe and Szegedy 2015). In this extend, Parcollet et al. (2017b) have proposed a well-adapted Gaussian Angular Noise (GAN) to reproduce the benefits of well-known noises during the training phase of a quaternion denoising autoencoder (QDAE) to extract a small and robust representation of the input signal. The QDAE purpose is equivalent to traditional real-valued denoising encoder–decoder (Vincent et al. 2008) but in the quaternion space. Therefore, a QDAE is a QMLP that aims to reconstruct a clean output of an artificially noised version of the input. The proposed GAN uses quaternion rotations to corrupt the initial inputs with a random angle that follows a Gaussian distribution. The GAN noise is expected to preserve the structure of the quaternion signal, since it performs a quaternion rotation that is well-defined in the quaternion field, while standard noises, such as dropout, would just decorrelate the components by being applied randomly to the quaternion signal. At the end of the learning, the hidden layer outputs of the QDAE are used as a new representation of the input features to feed a traditional QMLP. Indeed, it is known that hidden representations of autoencoders or denoising autoencoders are efficient at compressing the useful input information into a neural embedding (Vincent et al. 2008) and being robust to unseen documents from validation datasets. The QDAE and QMLP have obtained better observed results on the task of theme classification of telephone conversations from the DECODA dataset than real-valued equivalent models.

Although there is a consensus about better performances with deeper architectures, all the previous explored QMLP were one-hidden-layered neural networks, and do not benefit from the high abstraction and generalization capabilities of deep neural networks (Hinton et al. 2012). Therefore, Parcollet et al. (2017a) have combined previous researches (Parcollet et al. 2016, 2017b) to propose the first pre-trained quaternion deep neural network (QDNN) made to up to 5 hidden layers. As for real-valued NNs (Hinton et al. 2006), QDNNs are pre-trained based on quaternion-valued autoencoders to allow a faster convergence. The proposed model reached higher accuracies on the same DECODA task than any other method. It is also worth mentioning that quaternion-based models always outperform their real-valued counterpart. An important remark on this work is that quaternion-valued neural networks scale well with the number of neural parameters, making it possible to use QNNs in real-life tasks with bigger models.

5.1.3 3D space transformations

A natural context of employing QNNs is the representation of 3D transformations in space. Matsui et al. (2004) have compared a spinor QMLP to a real-valued MLP in three different tasks of dilatation, translation and rotation. In each benchmark, a set of input points with their corresponding transformed coordinates are fed to the neural network at training time. The training process is then performed point after point to learn the desired transformation. At validation time, each point composing a geometrical entity is processed to obtain the same entity, but dilated, rotated or translated. Therefore the input and output size of the models are one with a purely imaginary quaternion for the QMLP, and three distinct coordinates for the real-valued MLP. As a result, the real-valued MLP failed to learn the transformations. Indeed, the MLP failed to preserve the volumetric shape of the entity, and just produced a flat variation of the input. Conversaly, the spinor QMLP perfectly learned each transformation by producing either a dilated, rotated or translated version of the input, and always with less parameters due to the quaternion algebra. Such results are easily explained by the nature of the spinor QMLP. Indeed, the model is designed to learn a mapping of a 3D input vector (or a pure quaternion), to an other point which has been transformed in the same three-dimensional space.

5.1.4 Robot control

An early comparison of hypercomplex-valued neural networks to real-valued ones for the attitude control of rigid bodies can be found in Fortuna et al. (2001). The compared models were first introduced by Fortuna et al. (1996). Results show that the quaternion alternative offers a reduced complexity and an higher accuracy over traditional real-valued approaches. An interesting remarks on the observed results is the extrapolation ability of the quaternion-based approach. Indeed, both tested models are trained on a set of trajectories with some disturbances. The test-set however, is composed of unseen trajectories. The QMLP is able to maintain such unseen trajectories while the real-valued model obtains very poor results. As for other application tasks, the QMLP learns the relation that exists within the x, y, z coordinates of the spacecraft, making it possible to extrapolate properly the trajectory from a starting point. Then, Cui et al. (2013) have detailed an application of QNNs to the robot control and inverse kinematic. The inverse kinematic can be easily understood as a method that allows a robot to know which movements it has to follow to reach a certain final position. Since traditional methods are based on real-valued neural networks, the authors proposed to adapt the multi-dimensionality view of quaternions to the existing solutions by using a QMLP. The experiments show that a QMLP converges faster and with less parameters than a real-valued MLP. More recently, Takahashi et al. (2017) have extended this research to a comparison with a spinor QMLP (Eq. 28). The results highlighted a smaller final loss function for the spinor QMLP compared to a common QMLP and a real-valued MLP. Interestingly, the QMLP also performed better than complex-valued neural networks. This is due to the initial quaternion representation. Indeed, as depicted in Eq. 5, two complex numbers are necessary to represent a quaternion, and these complex numbers are processed independently by the neural network model, making it harder to keep the internal relations unaltered. Nonetheless, spinor QMLP are more computational intensive. Other very similar applications of QMLP on robotic can be found in Ogawa (2016) and Bayro-Corrochano et al. (2018).

5.1.5 Polarized signal processing

Such for robotic, realistic signals that come from seismology, optics or even communications are often multidimensional and polarized. The polarization of a signal express a vector that characterizes the direction or the orientation of an observed wave. One of the biggest challenge of polarized signal processing is to keep and take advantage of the polarization information trough the process. Therefore, one can distinguish two tasks when dealing with polarized signals. First, the polarization information must be captured and preserved. Then, the information carried out by the signal himself has to be used efficiently. Quaternion numbers perfectly solve these problematics by encoding the polarization with the quaternion algebra, and by processing the signal with an adequate quaternion-valued model. A good example of this architecture has been proposed by Buchholz and Le Bihan (2006). In the latter, an adapted polarized representation based on the quaternion algebra alongside with a vanilla QMLP, obtain better results than conventional real-valued systems on a signal separation task.

Discussions on QMLP Mere quaternion-valued multilayer perceptrons represent the first step of the introduction of quaternion numbers to the neural network domain. Despite their simplicity and their small learning capabilities, both spinor and standard QMLP efficiently solved numerous multidimensional tasks. In the last decade, it has been shown that QMLPs always outperform real-valued MLPs when dealing with three or four dimensional signals due to the quaternion algebra and the *Hamilton product*, making them the perfect fit for realistic multidimensional applications. Nonetheless, multilayer perceptrons are limited by their architectures. Indeed, QMLPs and MLPs do not integrate any mechanisms that make it possible to capture complex dependencies between the features, such as the edges and shapes on an image, or a sequence of 3D movement in time, or even words composed of phonemes in an acoustic signal sequence. To address these issues, quaternion-valued convolutional (CNNs) and recurrent (RNNs) neural networks have been developed.

5.2 Quaternion-valued convolutional neural networks (QCNN)

Real-valued convolutional neural networks (CNN) have been proposed (Krizhevsky et al. 2012) to better capture dependencies that occur between close input features. However, in the case of a real-valued CNN, the model has to learn both internal and external dependencies at the same level. More precisely, the relations that exist between each element that composes a multidimensional input feature are considered equally with the relations that exist between this input feature and the others. To alleviate this problem, quaternion convolutional neural networks (QCNN) have been proposed by Gaudet and Maida (2018). Indeed, the quaternion algebra takes care of coding the internal relations while the convolution process learns the external ones. As for the QMLP all parameters of the QCNN are quaternion numbers. Therefore, let S_{ab}^{l} be the pre-activation quaternion output at layer l and at the indexes (a, b) of the new feature map, and w the weight filter map of size $f \times f$. A schematic description of the convolution process is depicted in Fig. 3, and a formal definition based on Eq. 20 can be derived as:

$$\gamma_{ab}^{l} = \alpha(S_{ab}^{l}), \tag{29}$$

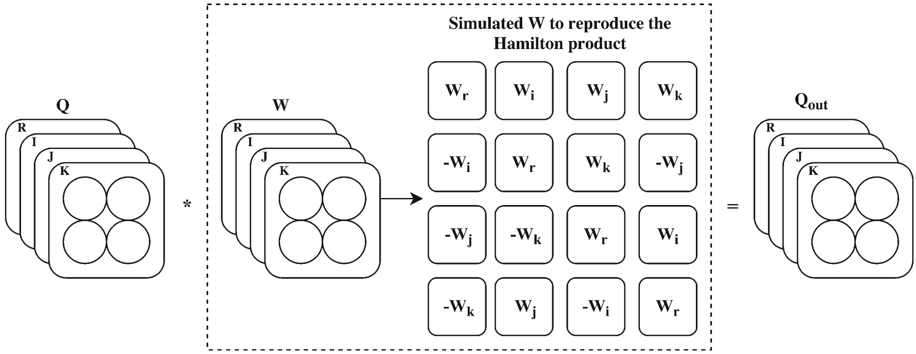
Q
R
I
J
K
*
W
R
I
J
K
Simulated W to reproduce the Hamilton product
W_r W_i W_j W_k
$-W_i$ W_r W_k $-W_j$
$-W_j$ $-W_k$ W_r W_i
$-W_k$ W_j $-W_i$ W_r
=
Q_{out}
R
I
J
K

Fig. 3 Illustration of the quaternion convolution (Parcollet et al. 2018c)

with

$$S^{l}_{ab} = \sum_{c=0}^{f-1} \sum_{d=0}^{f-1} w^{l} \otimes \gamma^{l-1}_{(a+c)(b+d)}. \tag{30}$$

As for QMLP, α is any quaternion activation function. Modern real-valued CNN are also deep, and Gaudet and Maida (2018) provided a batch-normalization for QNNs, extended from Trabelsi et al. (2017), and based on a Cholesky decomposition (Higham 1990). The quaternion batch-normalization procedure is equivalent to the real-valued one, but adapted to the quaternion algebra, and allows the build of deeper architectures by normalizing the outputs at each layer. Nonetheless, it is important to highlight the impact on the memory consumption and the computational complexity of the proposed quaternion batch-normalization. Indeed, such method requires the computation and the storage of numerous and bulky inverse matrices through the Cholesky decomposition. In practice, the use of the quaternion batch-normalization doubled the training time, and increased the memory consumption by a factor of four.

5.2.1 Image processing

Alongside with the basic structure of a QCNN, Gaudet and Maida (2018) have proposed three image processing tasks to evaluate the efficiency of the model. Therefore, the proposed deep QCNN have been evaluated on the CIFAR-10 and CIFAR-100 (Krizhevsky et al. 2014) image classification tasks, and on the KITTI image segmentation task (Geiger et al. 2013). In both experiments, 115 layers were stacked in a residual fashion (He et al. 2016) to enable fast training convergences. The R, G, B pixels values are used to represent the x, y, z components of quaternions and the grayscale value for the real part r. In either task, QCNNs outperform by far both real and complex valued CNNs in term of accuracy. The proposed QCNN demonstrates that quaternion-valued models can be compared, or beat state-of-the-art approaches by obtaining top of the line results in recent benchmarks.

More recently Parcollet et al. (2018a) proposed to investigate the impact of the *Hamilton product* on a QCNN with a gray-scale to color image task. In particular, the authors used a quaternion- and a real-valued convolutional autoencoder (QCAE, CAE) to compress a unique gray-scale image at training time. Both models are optimized to minimize the reconstruction error of the input image at the output, such as for any other autoencoder. At testing time, a color image is presented to the models, while the latter have only been trained on a gray-scale picture. Interestingly, the QCAE was able to perfectly reproduce the color image at the output. Conversely, the real-valued CAE produced a perfect reconstruction of the test image, but in gray-scale, failing to learn the color space. Such phenomenon highlights the ability of the *Hamilton product* to force QNNs to learn an internal relation within the input features.

5.2.2 3D acoustic signal processing

More recently, Comminiello et al. (2018) have applied the QCNN proposed by Gaudet and Maida (2018) to detection and localization of 3D sound events captured by first-order Ambisonics. The Ambisonics technique is able to capture 3D sounds and to create an immersive audio reconstruction. In the latter, the authors used a four dimensional representation of spatial sound field. Indeed, the B-Format Ambisonics is composed of an array of four coincident microphones, the first one is omnidirectional and the three others are orthogonal figure-of-eight microphones. Consequently, the real part of a quaternion is used to store the omnidirectional microphone signal, while the imaginary part is used for the three orthogonal ones. The tested network architecture is based on SELDnet (Adavanne et al. 2018). During the experiments, each acoustic input frame is mapped by the model into two parallel outputs describing first the sound event detection, and second the direction of arrival of the detected sound event. The proposed QCNN method obtained higher accuracies (F-score, error rate on event detection and custom metric for direction estimation) than the SELDNet with two different corpus.

5.2.3 Speech recognition

Quaternion representations fit particularly well to signals according to Sect. 5.1.5. Following Gaudet and Maida (2018), a QCNN has been applied to end-to-end speech recognition on the TIMIT (Garofolo et al. 1993) phoneme recognition task (Parcollet et al. 2018c). As for images processing, traditional speech recognition features can be naturally embedded into quaternions. Indeed, the speech signal is commonly split in fixed time-frames that are characterized by the Mel-filter banks energies, and their first and second order derivatives. It is crucial to consider that time derivatives of the spectral energy in a given frequency band at a specific time frame represent a special state of a time-frame, and are strongly correlated (Tokuda et al. 2003). These values can then be stacked together to obtain the x, y and z of a purely imaginary quaternion. Parcollet et al. (2018c) also provided a well designed algorithm to initialize the quaternion neural parameters with respect to well-known initialization criterions such as Glorot (Glorot and Bengio 2010) and He (He et al. 2015). The results show that QCNNs consistently outperform real-valued CNNs, in term of phoneme error rate (PER), with up to three times less parameters leading to a more efficient representation of the information. Evidence is provided that the better performance is due to a better representation and the ability of the QCNN to independently encode internal and global dependencies.

5.3 Quaternion-valued Hopfield neural networks

Hopfield neural networks (HNN) are a special case of recurrent neural networks. They are composed of binary threshold units with recurrent connections between them. It is important to note that in a real-valued HNN every single node is connected to all the neurons except itself. HNNs are often used for auto-association and optimization tasks (Hopfield and Tank

1985). Yoshida et al. (2005) provided the first extension of HNNs to a quaternion hopfield neural network (QHNN). In a QHNN, all parameters are quaternions, and the binary unit can be expressed as a binary quaternion with each component being either 0 or 1. Let us define a QHNN mades of P neurons, and let S_n^t, γ_n^t, γ_n^t, Θ_n^t be the action potential, the output state, the input, and the threshold of the neuron n at timestep t respectively. Also, w_{mn} is the weight that connects the neuron n to the neuron m. As demonstrated by Yoshida et al. (2005), and conversely to real-valued HNNs, the quaternion algebra allows QHNNs to contain self-connections. Based on Eq. 20 one can define the propagation of the signal in a QHNN as:

$$\gamma_n^t = \alpha(S_n^t), \tag{31}$$

with

$$S_n^t = \sum_{m=0}^{P} w_{nm}^t \otimes \gamma_m^{t-1} - \Theta_n^t. \tag{32}$$

To preserve the binary property of γ_n^t a split signum function α is applied to the state potential following:

$$\alpha(S_n^t) = f(r) + f(x)\mathbf{i} + f(y)\mathbf{j} + f(z)\mathbf{k}, \tag{33}$$

with,

$$f(a) = \begin{cases} 1, & \text{if } a \geq 0 \\ 0, & a < 0. \end{cases} \tag{34}$$

As for real-valued HNNs a quaternion version of the global and local energy functions are described and proven by Yoshida et al. (2005). Yoshida et al. (2005) provide a simple QHNN, a spinor QHNNs and an inverted QHNNs. It is proven that either QHNNs satisfy the same conditions than real-valued HNNs and are thus suitable to the same tasks. Isokawa et al. (2006) demonstrated that for a three and four neurons sized QHNN, the model works as an associative memory. Then, the number of neurons have been extended to 100 by Isokawa et al. (2008), and applied to an equivalent toy task. More recently, Kobayashi (2015) investigated an hybrid QHNN based on inverted QNNs (Kobayashi and Nakajima 2012) to be even more efficient. The dynamic of the QHNN have been studied by Valle and de Castro (2018). In spite of good theoretical performances, no realistic applications of QHNNs have been investigated yet. Nonetheless, QHNNs could have a positive impact on any task that involve multidimensional features such as image processing, where an HNN have been used for image segmentation (Lin et al. 1996).

5.4 Quaternion-valued recurrent neural networks

Although CNNs are efficient to detect and learn patterns in an input volume, recurrent neural networks (RNN) are more adapted to represent sequential data. Indeed, RNNs build a vector of activations at each timestep to code latent relations between input vectors or sequences. As for standard feed-forward and convolutional neural networks, a quaternary version of the RNN called QRNN have been proposed by Parcollet et al. (2018b). In the same manner as QCNNs, the authors claim that QRNNs are expected to code both sequential dependencies due to the recurrent architecture and internal relations between multi-dimensional input features that compose the sequence due to the *Hamilton product*. As for the others quaternion-valued models, all the parameters are quaternions. 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. Therefore, and with the same parameters as for a basic QMLP, the hidden state $h_n^{t,l}$ of the neuron n at timestep t and layer l can be computed as:

$$h_n^{t,l} = \alpha\left(\sum_{m=0}^{H} w_{nm,hh}^{l} \otimes h_m^{t-1,l} + \sum_{m=0}^{N_{l-1}} w_{nm,h\gamma}^{l} \otimes \gamma_m^{t,l-1} + b_n^l\right), \tag{35}$$

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

$$\gamma_n^{t,l} = \beta\left(\sum_{m=0}^{N_{l-1}} w_{nm,\gamma h}^{l} \otimes h_n^{t,l-1} + b_n^l\right), \tag{36}$$

with β any quaternion activation function. The full derivation of the backpropagation algorithm with the update phase can be found in the work of Parcollet et al. (2018b). The proposed QRNN is tested with the well-known TIMIT speech recognition task by variating the number of neurons in four hidden layers. As described in Sect. 5.2.3, time-frames of speech are encapsulated by quaternions. Therefore a quaternion Q is composed of the Mel filter bank energy, first order, second order, and third order derivatives. The results show that the QRNN obtains better phoneme error rates (PER) results compared to real-valued RNNs with a drastic reduction by a factor of more than three of the number of parameters. Parcollet et al. (2018b) claim that such performances are due to the combination of a better representation of the time-frames of speech as separate entities, with the higher generalization capability of quaternion valued models.

Nonetheless, non-gated RNNs are known to be prone to the exploding or vanishing gradient problems as demonstrated by Pascanu et al. (2013). In this extend, gated recurrent neural networks have been proposed. In particular, the long-short term memory neural network (LSTM) has been introduced by Hochreiter and Schmidhuber (1997) to establish state-of-the-art baselines in many applications. Parcollet et al. (2018b) proposed to extend the real-valued LSTM to a quaternionic version called QLSTM. Let us define $w_{f,hh}$, $w_{f,h\gamma}$, $w_{i,hh}$, $w_{i,h\gamma}$,$w_{o,hh}$, and $w_{o,h\gamma}$ the equivalent QRNN weights but for the forget $f_n^{t,l}$, input $i_n^{t,l}$ and output $o_n^{t,l}$ gates at time-step t, layer l and neuron n respectively:

$$f_n^{t,l} = \alpha\left(\sum_{m=0}^{H} w_{f,nm,hh}^{l} \otimes h_m^{t-1,l} + \sum_{m=0}^{N_{l-1}} w_{f,nm,h\gamma}^{l} \otimes \gamma_m^{t,l-1} + b_{f,n}^l\right), \tag{37}$$

$$i_n^{t,l} = \alpha\left(\sum_{m=0}^{H} w_{i,nm,hh}^{l} \otimes h_m^{t-1,l} + \sum_{m=0}^{N_{l-1}} w_{i,nm,h\gamma}^{l} \otimes \gamma_m^{t,l-1} + b_{i,n}^l\right), \tag{38}$$

$$o_n^{t,l} = \alpha\left(\sum_{m=0}^{H} w_{o,nm,hh}^{l} \otimes h_m^{t-1,l} + \sum_{m=0}^{N_{l-1}} w_{o,nm,h\gamma}^{l} \otimes \gamma_m^{t,l-1} + b_{o,n}^l\right), \tag{39}$$

with α any quaternion activation function. Then, the cell state $c_n^{t,l}$ is computed following:

$$c_n^{t,l} = f_n^{t,l} \odot c_n^{t-1,l} + i_n^{t,l} \odot tanh\left(\sum_{m=0}^{H} w_{c,nm,hh}^{l} \otimes h_m^{t-1,l} + \sum_{m=0}^{N_{l-1}} w_{c,nm,h\gamma}^{l} \otimes \gamma_m^{t,l-1} + b_{c,n}^{l}\right), \tag{40}$$

with $w_{c,hh}$, $w_{c,h\gamma}$ the cell-state weights matrices, and '$\odot$' the component-wise product. Note that $tanh()$ is the split $tanh()$ quaternion activation function. Finally, the hidden state $h_n^{t,l}$ is defined as:

$$h_n^{t,l} = o_n^{t,l} \odot tanh(c_n^{t,l}), \tag{41}$$

with $w_{h,hh}$, $w_{h,h\gamma}$ the hidden-state weights matrices. QLSTMs were compared to real-valued LSTMs on the speech recognition task of the Wall Street Journal, and offered a small word error rate gain alongside with an important reduction of the number of neural parameters, proving the suitability of quaternion representations to achieve top of the line results.

6 Conclusions and prospects

Quaternion-valued neural networks were introduced to enable neural networks to learn the internal and spatial relations that exist between multidimensional input features. The specific four dimensional algebra of quaternion numbers, including the *Hamilton product* allows quaternion-valued models to consistently outperform equivalent real-valued neural networks. In particular, QNNs are the perfect fit for three and four dimensional input features and should be preferred over traditional real-valued NNs for such tasks. Indeed, recent advances on the field make it possible to build state-of-the-art architectures such as quaternion convolutional and residual or quaternion recurrent neural networks, to efficiently solve modern and realistic tasks such as image processing, speech recognition, 3D modeling, robotic, or even signal processing. Nonetheless QNNs suffer from the fact of being a resurgent field, and crucial investigations are still missing. Promising directions for future are:

- *Quaternion-valued features* Most of the features that are currently used in quaternion-valued models are straightforward or commons real-valued features. New data preprocessing methods have to be investigated to naturally project the features into the quaternion space, such as the quaternion Fourier transform (Hitzer 2007).
- *New architectures* Despite a recent QCNN and QRNN, new neural networks architectures are still missing. For example, capsule networks, or generative adversarial neural networks could benefits from the introduction of quaternion numbers.
- *New learning algorithms* Real world applications of current QNN architectures are based on the straighforward extension of the real-valued backpropagation to the quaternion domain. The recent $\mathbb{GHR}$ calculus makes it possible to propose well-adapted learning algorithms that can speed-up the training, and increase the performances due to a better consideration of the quaternion algebra. Such learning algorithms must be improved and deployed in state-of-the-art QNN architectures to fully expose the potential of the $\mathbb{GHR}$ calculus.
- *Engineering* The *Hamilton product* is a powerful but expensive operation. In current implementations, this operation is not fully parallelized on GPUs, implying a longer

training time for quaternion neural networks. A proper CUDA implementation of this product would drastically reduce this computation time and makes of QNNs a mandatory alternative to real-valued NNs.

Finally, it is worth noticing that recent and promising works on generalized algebras such as geometric and Clifford algebras have shown interesting results when applied to neural networks, and could offer other alternatives to high dimensional problems.

References

Adavanne S, Politis A, Nikunen J, Virtanen T (2018) Sound event localization and detection of overlapping sources using convolutional recurrent neural networks. IEEE J Sel Top Signal Process 13:34–48

Aizenberg IN, Gonzalez A (2018) Image recognition using MLMVN and frequency domain features. In: 2018 International joint conference on neural networks (IJCNN), pp 1–8

Aizenberg I, Alexander S, Jackson J (2011) Recognition of blurred images using multilayer neural network based on multi-valued neurons. In: 2011 41st IEEE International symposium on multiple-valued logic. IEEE, pp 282–287

Arena P, Fortuna L, Re R, Xibilia MG (1993) On the capability of neural networks with complex neurons in complex valued functions approximation. In: 1993 IEEE International symposium on circuits and systems, ISCAS'93. IEEE, pp 2168–2171

Arena P, Fortuna L, Occhipinti L, Xibilia MG (1994) Neural networks for quaternion-valued function approximation. In: 1994 IEEE International symposium on circuits and systems, ISCAS'94, vol 6. IEEE, pp 307–310

Arena P, Fortuna L, Muscato G, Xibilia MG (1997) Multilayer perceptrons to approximate quaternion valued functions. Neural Netw 10(2):335–342

Bayro-Corrochano E, Lechuga-Gutiérrez L, Garza-Burgos M (2018) Geometric techniques for robotics and hmi: Interpolation and haptics in conformal geometric algebra and control using quaternion spike neural networks. Robot Auton Syst 104:72–84

Bechet F, Maza B, Bigouroux N, Bazillon T, El-Beze M, De Mori R, Arbillot E (2012) Decoda: a call-centre human–human spoken conversation corpus. In: LREC, pp 1343–1347

Blei DM, Ng AY, Jordan MI (2003) Latent dirichlet allocation. J Mach Learn Res 3(Jan):993–1022

Buchholz S, Sommer G (2000) Quaternionic spinor MLP. CiteSeer, Princeton

Buchholz S, Le Bihan N (2006) Optimal separation of polarized signals by quaternionic neural networks. In: 2006 14th European signal processing conference. IEEE, pp 1–5

Chou JC (1992) Quaternion kinematic and dynamic differential equations. IEEE Trans Robot Autom 8(1):53–64

Comminiello D, Lella M, Scardapane S, Uncini A (2018) Quaternion convolutional neural networks for detection and localization of 3D sound events. arXiv:181206811

Cui Y, Takahashi K, Hashimoto M (2013) Design of control systems using quaternion neural network and its application to inverse kinematics of robot manipulator. In: 2013 IEEE/SICE International symposium on system integration (SII). IEEE, pp 527–532

Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: IEEE Computer society conference on computer vision and pattern recognition, CVPR 2005, vol 1. IEEE, pp 886–893

De Boer PT, Kroese DP, Mannor S, Rubinstein RY (2005) A tutorial on the cross-entropy method. Ann Oper Res 134(1):19–67

De Leo S, Rotelli P (1997) Local hypercomplex analyticity. arXiv preprint arXiv:9703002 [funct-an]

Diebel J (2006) Representing attitude: Euler angles, unit quaternions, and rotation vectors. Matrix 58(15–16):1–35

Dornaika F, Horaud R (1998) Simultaneous robot-world and hand-eye calibration. IEEE Trans Robot Autom 14(4):617–622

Fortuna L, Muscato G, Xibilia M (1996) An hypercomplex neural network platform for robot positioning. In: 1996 IEEE International symposium on circuits and systems, ISCAS'96. Connecting the World, vol 3. IEEE, pp 609–612

Fortuna L, Muscato G, Xibilia MG (2001) A comparison between hmlp and hrbf for attitude control. IEEE Trans Neural Netw 12(2):318–328

Garofolo JS, Lamel LF, Fisher WM, Fiscus JG, Pallett DS (1993) Darpa timit acoustic-phonetic continous speech corpus CD-ROM. NIST speech disc 1-1.1. NASA STI/Recon technical report no. 93

Gaudet CJ, Maida AS (2018) Deep quaternion networks. In: 2018 International joint conference on neural networks (IJCNN). IEEE, pp 1–8

Geiger A, Lenz P, Stiller C, Urtasun R (2013) Vision meets robotics: the KITTI dataset. Int J Robot Res (IJRR) 32:1231–1237

Glorot X, Bengio Y (2010) Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the thirteenth international conference on artificial intelligence and statistics, pp 249–256

Graves A, Mohamed Ar, Hinton G (2013) Speech recognition with deep recurrent neural networks. In: 2013 IEEE International conference on acoustics, speech and signal processing. IEEE, pp 6645–6649

Greenblatt A, Mosquera-Lopez C, Agaian S (2013) Quaternion neural networks applied to prostate cancer gleason grading. In: 2013 IEEE International conference on systems, man, and cybernetics (SMC). IEEE, pp 1144–1149

Hamilton WR (1844) Ii. on quaternions; or on a new system of imaginaries in algebra. Lond Edinb Dublin Philos Mag J Sci 25(163):10–13

Hearst MA, Dumais ST, Osuna E, Platt J, Scholkopf B (1998) Support vector machines. IEEE Intell Syst Appl 13(4):18–28

He K, Zhang X, Ren S, Sun J (2015) Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE international conference on computer vision, pp 1026–1034

He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778

Higham NJ (1990) Analysis of the Cholesky decomposition of a semi-definite matrix. Oxford University Press, Oxford

Hinton GE, Salakhutdinov RR (2006) Reducing the dimensionality of data with neural networks. Science 313(5786):504–507

Hinton GE, Osindero S, Teh YW (2006) A fast learning algorithm for deep belief nets. Neural Comput 18(7):1527–1554

Hinton G, Deng L, Yu D, Dahl GE, Mohamed A, Jaitly N, Senior A, Vanhoucke V, Nguyen P, Sainath TN et al (2012) Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. IEEE Signal Process Mag 29(6):82–97

Hirose A (2012) Complex-valued neural networks, vol 400. Springer, Berlin

Hitzer EM (2007) Quaternion fourier transform on quaternion fields and generalizations. Adv Appl Clifford Algebras 17(3):497–517

Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780

Hopfield JJ, Tank DW (1985) “Neural” computation of decisions in optimization problems. Biol Cybern 52(3):141–152

Huang FJ, LeCun Y (2006) Large-scale learning with SVM and convolutional for generic object categorization. In: null. IEEE, pp 284–291

Ioffe S, Szegedy C (2015) Batch normalization: accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:150203167

Isokawa T, Kusakabe T, Matsui N, Peper F (2003) Quaternion neural network and its application. In: International conference on knowledge-based and intelligent information and engineering systems. Springer, pp 318–324

Isokawa T, Nishimura H, Kamiura N, Matsui N (2006) Fundamental properties of quaternionic hopfield neural network. In: 2006 International joint conference on neural networks, IJCNN’06. IEEE, pp 218–223

Isokawa T, Nishimura H, Kamiura N, Matsui N (2008) Associative memory in quaternionic hopfield neural network. Int J Neural Syst 18(02):135–145

Isokawa T, Matsui N, Nishimura H (2009) Quaternionic neural networks: fundamental properties and applications. In: Complex-valued neural networks: utilizing high-dimensional parameters. IGI global. pp 411–439

Isokawa T, Nishimura H, Matsui N (2012) Quaternionic multilayer perceptron with local analyticity. Information 3(4):756–770

Jolliffe I (2011) Principal component analysis. In: Lovric M (ed) International encyclopedia of statistical science. Springer, Berlin, pp 1094–1096

Karney CF (2007) Quaternions in molecular modeling. J Mol Graph Model 25(5):595–604

Kinugawa K, Shang F, Usami N, Hirose A (2018) Isotropization of quaternion-neural-network-based PolSAR adaptive land classification in Poincare-sphere parameter space. IEEE Geosci Remote Sens Lett 15:1234–1238

Kobayashi M (2015) Hybrid quaternionic hopfield neural network. IEICE Trans Fundam Electron Commun Comput Sci 98(7):1512–1518

Kobayashi M, Nakajima A (2012) Twisted quaternary neural networks. IEEJ Trans Electr Electron Eng 7(4):397–401

Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems, pp 1097–1105

Krizhevsky A, Nair V, Hinton G (2014) The cifar-10 dataset. http://www.cs.toronto.edu/kriz/cifarhtml

Kusamichi H, Isokawa T, Matsui N, Ogawa Y, Maeda K (2004) A new scheme for color night vision by quaternion neural network. In: Proceedings of the 2nd international conference on autonomous robots and agents, vol 1315. Citeseer

Lin JS, Cheng KS, Mao CW (1996) A fuzzy hopfield neural network for medical image segmentation. IEEE Trans Nucl Sci 43(4):2389–2398

Mandic DP, Goh VSL (2009) Complex valued nonlinear adaptive filters: noncircularity, widely linear and neural models, vol 59. Wiley, New York

Mandic DP, Jahanchahi C, Took CC (2011) A quaternion gradient operator and its applications. IEEE Signal Process Lett 18(1):47–50

Matsui N, Isokawa T, Kusamichi H, Peper F, Nishimura H (2004) Quaternion neural network with geometrical operators. J Intell Fuzzy Syst 15(3, 4):149–164

Mikolov T, Karafiát M, Burget L, Černockỳ J, Khudanpur S (2010) Recurrent neural network based language model. In: Eleventh annual conference of the international speech communication association

Nair V, Hinton GE (2010) Rectified linear units improve restricted Boltzmann machines. In: Proceedings of the 27th international conference on machine learning (ICML-10), pp 807–814

Nitta T (1995) A quaternary version of the back-propagation algorithm. In: IEEE International conference on neural networks, 1995. Proceedings, vol 5. IEEE, pp 2753–2756

Nitta T (2004) A solution to the 4-bit parity problem with a single quaternary neuron. Neural Inf Process Lett Rev 5(2):33–39

Nitta T, de Garis H (1992) A 3D vector version of the back-propagation algorithm. In: Proceedings of international joint conference on neural networks, pp 511–516

Ogawa T (2016) Neural network inversion for multilayer quaternion neural networks. Comput Technol Appl 7:73–82

Parcollet T, Morchid M, Bousquet PM, Dufour R, Linarès G, De Mori R (2016) Quaternion neural networks for spoken language understanding. In: 2016 IEEE Spoken language technology workshop (SLT). IEEE, pp 362–368

Parcollet T, Morchid M, Linares G (2017a) Deep quaternion neural networks for spoken language understanding. In: 2017 IEEE Automatic speech recognition and understanding workshop (ASRU). IEEE, pp 504–511

Parcollet T, Morchid M, Linares G (2017b) Quaternion denoising encoder–decoder for theme identification of telephone conversations. Proceedings of Interspeech 2017, pp 3325–3328

Parcollet T, Morchid M, Linarès G (2018a) Quaternion convolutional neural networks for heterogeneous image processing. arXiv preprint arXiv:181102656

Parcollet T, Ravanelli M, Morchid M, Linarès G, Trabelsi C, Mori RD, Bengio Y (2018b) Quaternion recurrent neural networks. arXiv preprint arXiv:1806.04418

Parcollet T, Zhang Y, Morchid M, Trabelsi C, Linarès G, de Mori R, Bengio Y (2018c) Quaternion convolutional neural networks for end-to-end automatic speech recognition. In: Interspeech 2018, 19th Annual conference of the international speech communication association, Hyderabad, India, 2–6 September 2018, pp 22–26. https://doi.org/10.21437/Interspeech.2018-1898

Pascanu R, Mikolov T, Bengio Y (2013) On the difficulty of training recurrent neural networks. In: International conference on machine learning, pp 1310–1318

Platt J, et al. (1999) Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In: Advances in large margin classifiers. MIT Press, pp 61–74

Pletinckx D (1989) Quaternion calculus as a basic tool in computer graphics. Vis Comput 5(1–2):2–13

Popa CA (2018) Learning algorithms for quaternion-valued neural networks. Neural Process Lett 47(3):949–973

Sangwine SJ (1996) Fourier transforms of colour images using quaternion or hypercomplex, numbers. Electron Lett 32(21):1979–1980

Schuster M, Paliwal KK (1997) Bidirectional recurrent neural networks. IEEE Trans Signal Process 45(11):2673–2681

Shang F, Hirose A (2014) Quaternion neural-network-based PolSAR land classification in poincare-sphere-parameter space. IEEE Trans Geosci Remote Sensing 52(9):5693–5703

Shoemake K (1985) Animating rotation with quaternion curves. In: ACM SIGGRAPH computer graphics, vol 19. ACM, pp 245–254

Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:14091556

Soulard R, Carré P (2011) Quaternionic wavelets for texture classification. Pattern Recognit Lett 32(13):1669–1678

Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res 15(1):1929–1958

Takahashi K, Takahashi S, Cui Y, Hashimoto M (2014) Remarks on computational facial expression recognition from HOG features using quaternion multi-layer neural network. In: International conference on engineering applications of neural networks. Springer, pp 15–24

Takahashi K, Isaka A, Fudaba T, Hashimoto M (2017) Remarks on quaternion neural network-based controller trained by feedback error learning. In: 2017 IEEE/SICE International symposium on system integration (SII), pp 875–880

Tokuda, K., Zen, H., Kitamura, T. (2003) Trajectory modeling based on HMMs with the explicit relationship between static and dynamic features. In Eighth European conference on speech communication and technology

Trabelsi C, Bilaniuk O, Zhang Y, Serdyuk D, Subramanian S, Santos JF, Mehri S, Rostamzadeh N, Bengio Y, Pal CJ (2017) Deep complex networks. arXiv preprint arXiv:170509792

Ujang BC, Jahanchahi C, Took CC, Mandic D (2010) Quaternion valued neural networks and nonlinear adaptive filters.

Ujang BC, Took CC, Mandic DP (2011) Quaternion-valued nonlinear adaptive filtering. IEEE Trans Neural Netw 22(8):1193–1206

Valle ME, de Castro FZ (2018) On the dynamics of hopfield neural networks on unit quaternions. IEEE Trans Neural Netw Learn Syst 29(6):2464–2471

Vincent P, Larochelle H, Bengio Y, Manzagol PA (2008) Extracting and composing robust features with denoising autoencoders. In: Proceedings of the 25th international conference on machine learning. ACM, pp 1096–1103

Willmott CJ, Matsuura K (2005) Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. Clim Res 30(1):79–82

Xu D, Jahanchahi C, Took CC, Mandic DP (2015) Enabling quaternion derivatives: the generalized HR calculus. R Soc Open Sci 2(8):150255

Xu D, Zhang L, Zhang H (2017) Learning algorithms in quaternion neural networks using GHR calculus. Neural Netw World 27(3):271

Yoshida M, Kuroe Y, Mori T (2005) Models of hopfield-type quaternion neural networks and their energy functions. Int J Neural Syst 15:129–135

Yun X, Bachmann ER (2006) Design, implementation, and experimental results of a quaternion-based kalman filter for human body motion tracking. IEEE Trans Robot 22(6):1216–1227

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.