Quaternion Neural Network and Its Application

Teijiro Isokawa[1], Tomoaki Kusakabe[1],
Nobuyuki Matsui[1], and Ferdinand Peper[2]

[1] Division of Computer Engineering, Himeji Institute of Technology, Japan
{isokawa,tomoaki,matsui}@comp.eng.himeji-tech.ac.jp
[2] Communications Research Laboratory, Nanotechnology Group, Japan
peper@crl.go.jp

Abstract. Quaternion neural networks are models of which computations in the neurons is based on quaternions, the four-dimensional equivalents of imaginary numbers. This paper shows by experiments that the quaternion-version of the Back Propagation (BP) algorithm achieves correct geometrical transformations in color space for an image compression problem, whereas real-valued BP algorithms fail.

1 Introduction

Though most real-valued neural network models are able to learn arbitrary non-linear functions, they perform less well when it comes to geometrical transformations, like affine transformations in 2 or 3 dimensional space. Some researchers[1] have found that the use of complex-valued neural networks results in improved performance on such transformation problems. These results inspired the Quaternion Neural Network Model in [2], which is trained by a BP learning algorithm. Such models employ neurons in which all computations are based on quaternions, a four-dimensional extension of imaginary numbers discovered by Sir W.R. Hamilton, which have found extensive use in modern mathematics and physics[3]. It turns out that such a quaternion neural model learns 3D affine transformations very well[2]. Using this as the starting point in this paper, we apply a quaternion BP learning algorithm on a color image compression problem, which is a problem in which the fidelity of colors is only preserved when the affine transformations in color space are correct. Experiments show improved performance of our quaternion scheme as compared to real-valued BP.

2 Quaternion

Quaternions form a class of hypercomplex numbers that consist of a real number and three kinds of imaginary number, $\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k}$. Formally, a quaternion is defined as a vector $\boldsymbol{x}$ in a 4-dimensional vector space, i.e.,

$$\boldsymbol{x} = x^{(e)} + x^{(i)}\boldsymbol{i} + x^{(j)}\boldsymbol{j} + x^{(k)}\boldsymbol{k} \tag{1}$$

where $x^{(e)}$ and $x^{(i)}, x^{(j)}, x^{(k)}$ are real numbers. $\boldsymbol{K}^4$, the division ring of quaternions, thus constitutes the four-dimensional vector space over the real numbers with the bases $1, \boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k}$.

Quaternions satisfy the following identities, known as the Hamilton rules:

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

$$\boldsymbol{ij} = -\boldsymbol{ji} = \boldsymbol{k}, \quad \boldsymbol{jk} = -\boldsymbol{kj} = \boldsymbol{i}, \quad \boldsymbol{ki} = -\boldsymbol{ik} = \boldsymbol{j}. \tag{3}$$

From these rules it follows immediately that multiplication of quaternions is not commutative.

The quaternion conjugate is defined as

$$\bar{\boldsymbol{x}} = x^{(e)} - x^{(i)}\boldsymbol{i} - x^{(j)}\boldsymbol{j} - x^{(k)}\boldsymbol{k} \tag{4}$$

where $x^{(e)}$ is regarded as the real part and $x^{(i)}\boldsymbol{i} + x^{(j)}\boldsymbol{j} + x^{(k)}\boldsymbol{k}$ as the imaginary part of $\boldsymbol{x}$.

The quaternion norm of $\boldsymbol{x}$, $\mathrm{n}(\boldsymbol{x})$, is defined by

$$\begin{aligned} n(\boldsymbol{x}) &= \sqrt{\boldsymbol{x}\bar{\boldsymbol{x}}} = \sqrt{\bar{\boldsymbol{x}}\boldsymbol{x}} \\ &= \sqrt{{x^{(e)}}^2 + {x^{(j)}}^2 + {x^{(j)}}^2 + {x^{(k)}}^2}. \end{aligned} \tag{5}$$

For convenience in the following explanation, we define the purely imaginary quaternion as a quaternion with zero real part. A purely imaginary quaternion $\boldsymbol{x}$ can thus be expressed as

$$\boldsymbol{x} = x^{(i)}\boldsymbol{i} + x^{(j)}\boldsymbol{j} + x^{(k)}\boldsymbol{k}. \tag{6}$$

3 Geometric Description by Quaternion

A rotation $\boldsymbol{g}$ in 3D vector space $\boldsymbol{I} \in \boldsymbol{K}^3$ can be computed as

$$\boldsymbol{g} = \boldsymbol{av}\bar{\boldsymbol{a}} \tag{7}$$

where $\boldsymbol{a}$ is a quaternion with $n(\boldsymbol{a}) = 1$ and $\boldsymbol{v}$ is a purely imaginary quaternion with $n(\boldsymbol{v}) = 1$. This quaternion $\boldsymbol{a}$ is denoted by

$$\boldsymbol{a} = \cos\alpha + (\sin\alpha) \cdot \boldsymbol{u} \tag{8}$$

where α is an angle satisfying $|\alpha| \leq \pi$ and $\boldsymbol{u}$ is a purely imaginary quaternion with $n(\boldsymbol{u}) = 1$. Eq.(7) can always be applied whether $\boldsymbol{u}$ and $\boldsymbol{v}$ are orthogonal or not. In the case that $\boldsymbol{u}$ and $\boldsymbol{v}$ are orthogonal, rotation $\boldsymbol{g}$ is expressed from Eq.(8) as

$$\boldsymbol{g} = (\cos 2\alpha)\boldsymbol{v} + (\sin 2\alpha)\boldsymbol{u} \times \boldsymbol{v} \tag{9}$$

This shows the vector $\boldsymbol{v}$ rotated by the angle 2α around $\boldsymbol{u}$ (see Fig.1). In the case that $\boldsymbol{u}$ and $\boldsymbol{v}$ are not orthogonal, Eq.(7) is expressed as

$$\begin{aligned} \boldsymbol{g} &= \boldsymbol{a}(\boldsymbol{v}_1 + \boldsymbol{v}_2)\bar{\boldsymbol{a}} = \boldsymbol{av}_1\bar{\boldsymbol{a}} + \boldsymbol{av}_2\bar{\boldsymbol{a}} \\ &= \boldsymbol{v}_1 + (\sin 2\alpha)(\boldsymbol{u} \times \boldsymbol{v}_2) + (\cos 2\alpha)\boldsymbol{v}_2, \end{aligned} \tag{10}$$

where $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ are the components of vector $\boldsymbol{v}$ and satisfy $\boldsymbol{v}_1 // \boldsymbol{u}$ and $\boldsymbol{v}_2 \perp \boldsymbol{u}$. Thus Eq.(10) also shows the vector $\boldsymbol{v}$ rotated by the angle 2α around $\boldsymbol{u}$.

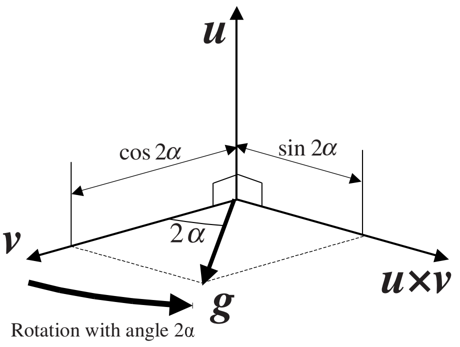


Fig. 1. Rotation $\boldsymbol{g}$ in the space $\boldsymbol{I}$

4 Quaternion Neural Network

In this section we propose a layered quaternion neural network model and a quaternion BP algorithm to train it. Our quaternion neuron model adopts purely imaginary quaternions as input and output signals. The output y_j of quaternion neuron j is expressed as

$$\boldsymbol{s}_j = \sum_i \frac{\boldsymbol{w}_{ji}\boldsymbol{x}_i\bar{\boldsymbol{w}}_{ji}}{|\boldsymbol{w}_{\boldsymbol{j}\boldsymbol{i}}|} - \boldsymbol{\theta}_j \tag{11}$$

$$\boldsymbol{y}_j = f(\boldsymbol{s}_{\boldsymbol{j}}) \tag{12}$$

where i denotes the indices of neurons in the previous layer, and $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\theta}, \boldsymbol{s} \in \boldsymbol{I}$, $\boldsymbol{w} \in \boldsymbol{K}^4$ respectively are the vector of inputs to the neurons, the vector of outputs from the neurons, the threshold, the internal potential, and the weights of the connections to the neurons in layer i. The activation function f is defined by

$$f(\boldsymbol{s}) = h(s^{(i)})\boldsymbol{i} + h(s^{(j)})\boldsymbol{j} + h(s^{(k)})\boldsymbol{k} \tag{13}$$

$$h(x) = \frac{1}{1+e^{-x}} \tag{14}$$

The quaternion neurons as defined above form the basis of a layered quaternion neural network. We adopt the quaternion equivalent of the BP algorithm for training the network, and define the error E between the output values of the network and the target training data as

$$E = \frac{1}{2}({e^{(i)}}^2 + {e^{(j)}}^2 + {e^{(k)}}^2) \tag{15}$$

$$e^{(\mu)} = y_k^{(\mu)} - d^{(\mu)}, \mu = \{i, j, k\}, \tag{16}$$

where $y_k^{(\mu)}$ is the output values of the neuron in the output layer and $d^{(\mu)}$ is the target value. The connection weights $\boldsymbol{w}$ are updated by the gradient descent

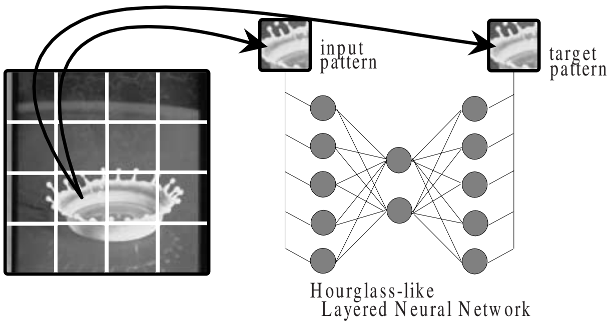


Fig. 2. Layered network for image compression

method:

$$\boldsymbol{w}^{new} = \boldsymbol{w}^{old} + \Delta \boldsymbol{w} \tag{17}$$

$$\Delta w^{(\nu)} = -\eta \cdot \frac{\partial E}{\partial w^{(\nu)}}, \quad \nu = \{e, i, j, k\}, \tag{18}$$

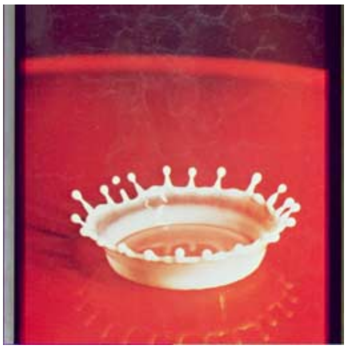
where η is a constant denoting the learning coefficient.

5 Experimental Results

5.1 Image Compression by Neural Networks

To show the improved performance of quaternion BP as compared to real-valued (conventional) BP, we conduct image compression using BP on the neural networks, a task first proposed in [4]. To conduct image compression, a neural network with three layers is used, such that the number of neurons in the input layer is the same as in the output layer, and the number of neurons in the hidden layer is less than the number of neurons in the input layer. The hidden layer thus acts as a bottleneck, through which the information from the input layer to the output layer must pass with as little information loss as possible. An image to be compressed is prepared and uniformly divided into many samples of small subimages. The network is then trained by inputting the samples to both the input as well as the output neurons of the network. As a result of the training the network will try to find values of its weights such that there is a strong association between the input and the output. Such types of networks are also called *autoassociators*. Figure 2 shows an example of a network for the image compression problem.

The following conditions are used for the computer simulations. The images used consist of 256×256(=65536) pixels, each of which has its color values represented by 24 bits. The images are divided into 4096 samples of 4×4(=16) pixels in size. The neural networks have three layers, and the neurons of the networks


(a) image for training

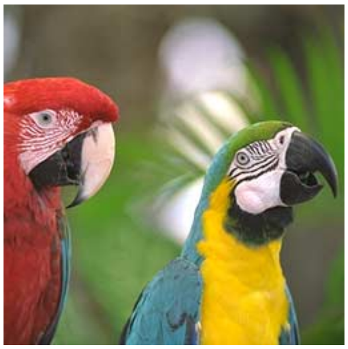

(b) image for testing

Fig. 3. Images for training and testing networks

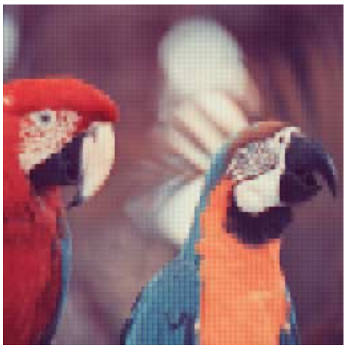
Table 1. PSNRs of output image (used Fig.3(a) as the input of network)

	Real-valued BP (dB)	Quaternion BP (dB)
B-Plane	26.92	27.55
G-Plane	25.60	25.36
R-Plane	27.79	27.49
Total	26.68	26.67

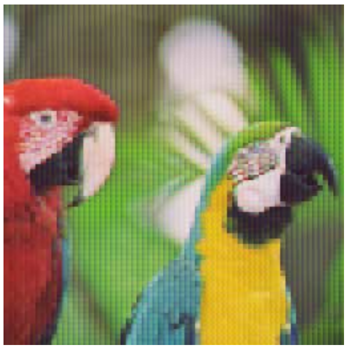
are distributed over the layers in a 48-12-48 configuration for the real-valued network and a 16-4-16 configuration for the quaternion network. These particular configurations ensure that both networks have approximately the same number of weight parameters, and thus can be used to compare performances. The image shown in Fig.3(a) is used for training the network and the image shown in Fig.3(b) is used after training for evaluating the generalizing ability of the network. The PSNR(peak-signal to noise ratio) between the original image and the output image is used as a measure to evaluate the performance of the networks.

5.2 Results

The networks are trained by using the image of Fig.3(a) up to the point that a certain mean square error (MSE) between the target data and the output of the network is achieved. The number of training iterations is equal to 10,000 and 9,421 for the real-valued BP and the quaternion BP respectively. Table 1 shows the PSNRs of the output images when the image of Fig.3(a) is imposed on the networks after training. Note that the expression "{B, G, R}-plane" in this table denotes the PSNRs calculated by using only the blue, green, and red components of the images respectively and "total" expresses PSNRs calculated



(a) image obtained from real-valued NN



(b) image obtained from quaternion NN

Fig. 4. Output images of real-valued and quaternion neural networks

Table 2. PSNRs of output image (used Fig.3(b) as the input of network)

	Real-valued BP (dB)	Quaternion BP (dB)
B-Plane	16.35	23.01
G-Plane	18.30	21.16
R-Plane	20.38	21.99
Total	18.04	21.99

by using all three components of the images. We see that the PSNRs are almost equal because the MSEs of the networks after training are virtually the same.

After training, Fig.3(b) is input to the networks to inspect the generalization abilities of the networks. Figure 4 shows the output images of the networks. The output image obtained by real-valued BP is reddish and its tone is different from the original image. Since large portions of the training image (Fig.3(a)) are red, the network is trained so as to project most colors onto red colors. The output image obtained by the quaternion BP, on the other hand, can catch the correct tone of the image. Table 2 also shows the PSNRs of the output images when the image of Fig.3(b) is input to the networks. The PSNRs calculated for the blue and green planes in the case of real-valued BP are less than those of the red plane, a tendency that can also be observed in Fig.4(a). On the other hand, in the case of the quaternion BP, the PSNRs of the blue plane(23.01(dB)) is larger than that of the green and the red plane, and it seems that learning in the quaternion BP is not affected by the distribution of the training image (Fig.3(a)).

6 Conclusion

We have investigated the performance of a quaternion neural network trained by the BP algorithm on a color image compression problem. This network contains quaternion neurons, whose computations and variables can be described in terms of a three-dimensional subspace of the four-dimensional space based on quaternions. We formulated a back propagation algorithm for training a quaternion neural network (quaternion BP), and compared its performance with BP on a real-valued neural network. Experimental results show that quaternion BP achieves correct transformations in color space of images, whereas real-valued BP failed.

The color image compression problem thus boils down to performing a transformation in 3-dimensional color space. The initial weights in the network have random values, so initially the network transforms the color space randomly. Training the network using BP results in the modification of the weights, such that eventually the transform is made as much invariant as possible. Quaternion BP utilizes the affine transformation of the vectors in its processing. This affine transformation is done in the whole color space, whereas the transformation using real-valued BP applies only to a part of the color space. Based on this, the difference between the output images, when using the test image, can be explained by the difference in the abilities of real-valued BP and quaternion BP to correctly transform the color space. A detailed analysis of this supposition remains future work.

References

[1] Arena,P., Fortuna,L., Muscato,G., Xibilia,M. G. : Neural Networks in Multidimensional Domains. Lecture Note in Control and Information Sciences **234** (1998) 318

[2] Kusakabe, T., Kouda, N., Isokawa, T., Matsui, N. : A Study of Neural Network Based on Quaternion. *Proceeding of SICE Annual Conference* (2002) 776–779 318

[3] Conway, A. W. : Quaternions and quantum mechanics. Pont. Acad. Sci. Acta **12** (1948) 204-277 318

[4] Cottrell, G. W., Munro, P., Zipser,D. : Image compression by back propagation: An example of extensional propagation. ICS report **8702** (1987) 321