Full-Learning Rotational Quaternion Convolutional Neural Networks and Confluence of Differently Represented Data for PolSAR Land Classification

Yuya Matsumoto, Ryo Natsuaki, *Member, IEEE*, and Akira Hirose, *Fellow, IEEE*

***Abstract*—Quaternion convolutional neural networks (QCNNs) expand the range of their applications in processing optical and polarimetric synthetic aperture radar (PolSAR) images. Conventional real-valued convolutional neural networks (RVCNNs) compress a three-channel input image into a single-channel feature map and ignore the relationship among the channels. In contrast, QCNNs deal with the input image as a single quaternion matrix and perform quaternion operation without the reduction of the channels. They can learn the interrelationship among the channel components. Though there exist two types of QCNNs, they have problems, respectively. One type conducts physically unclear quaternion convolution by using simple quaternionic multiplications. The other employs quaternion rotations with fixed axes, resulting in impairment of expression ability. In this article, we propose full-learning rotational QCNNs, which perform quaternion rotation in convolution, and update all the four parameters of a quaternion weight by backpropagation. They realize quaternion rotational convolution with high expression ability. We also propose using two different kinds of features, namely PolSAR pseudocolor features and Stokes vectors normalized by their total power. These two features allow neural networks to learn totally different characteristics of land surface. We train two networks with these features independently. Then, we merge their two classification results to obtain final decision to compensate for the shortcomings of the respective features. Experiments demonstrate that our proposed QCNNs show better classification performance than that of RVCNNs and the two existing QCNNs. We also find that the combination of the two features improves final classification results measured by F-scores.**

***Index Terms*—Convolutional neural network (CNN), polarimetric synthetic aperture radar (PolSAR), quaternion neural network (QNN).**

I. Introduction

POLARIMETRIC synthetic aperture radar (PolSAR) systems obtain more information by using multiple polarizations than by using a single polarization [1]. Land classification is one of the most important PolSAR applications. There are various methods based on, e.g., entropy and alpha angle [2], matrix decomposition [3]–[5], wishart distribution [6], [7], and traditional machine learning methods, such as support vector machine [8], [9] and random forest [10], [11]. Recently, neural-network methods have been actively studied, and they have often shown better performance than these conventional classical machine learning methods [12]–[15].

There are unique difficulties for PolSAR image classification, unlike general optical image classification, such as selection of effective features for training, insufficient number of labeled data compared to ordinary RGB photo data, and speckle noise [16]. Hence, some unique neural networks for PolSAR land classification have been proposed.

Quaternion neural networks (QNNs) [17] have been employed for PolSAR land classification [18]–[20]. In QNN methods, a 3-D vector is regarded as a single quaternion. In other words, the three components of the vector are assigned to the three imaginary parts of a quaternion. QNNs restrict the linear transformation to quaternion rotation, that is, orthogonal transformation. Therefore, the input features do not degenerate. When mapping 3-D vectors to a 3-D space, real-valued neural networks (RVNNs) need a 3 × 3 matrix of weights, i.e., nine parameters, while QNNs need only four parameters of a single quaternion weight. Hence, QNNs can map input vectors with fewer parameters than RVNNs. Then, the QNNs have shown higher performance (generalization ability) than conventional PolSAR classification methods.

Quaternion convolutional neural networks (QCNNs) perform convolution by using quaternion operation. The QCNNs have kernels consisting of quaternion neurons to learn spatial texture. In general, real-valued convolutional neural networks (RVCNNs) compress a three-channel input image to a single channel feature map, which loses the information of the relationship among the channels. On the other hand, the QCNNs deal with a general RGB image as a single quaternion matrix by assigning its red, green, and blue channels to three quaternion imaginary components. Then, the QCNNs perform quaternion operations on the quaternion matrix to learn the relationship between the channels of the image instead of simply summing them up.

There are two existing types of conventional QCNNs. The first QCNNs realize quaternion convolution by replacing real-valued multiplication in general convolution with quaternionic

Manuscript received November 9, 2021; revised February 28, 2022 and March 28, 2022; accepted March 29, 2022. Date of publication April 11, 2022; date of current version April 25, 2022. This work was supported in part by the Japan Society for the Promotion of Science KAKENHI under Grant 18H04105 and in part by the Cooperative Research Project Program of the Research Institute of Electrical Communication (RIEC), Tohoku University. *(Corresponding author: Yuya Matsumoto.)*

The authors are with the Department of Electrical Engineering and Information Systems, The University of Tokyo, Bunkyo-ku 113-8656, Japan (e-mail: matsumoto@eis.t.u-tokyo.ac.jp; natsuaki@eis.t.u-tokyo.ac.jp; ahirose@ee.t.u-tokyo.ac.jp).

Digital Object Identifier 10.1109/JSTARS.2022.3164431

multiplication [21]. However, it is unclear that what the quaternion multiplication for an input image means. We call this type of QCNNs simple product-type QCNNs in this article. They are employed for RGB image analysis [22], [23] and speech recognition [24]. Second are QCNNs, which perform quaternion rotation with fixed axis, not simple quaternion multiplication, on the channels of an input image in convolution. We refer to this type of QCNNs as fixed-axis QCNNs. They are employed for RGB images [25] and PolSAR images [26]. The quaternion rotation is physically clear, and prevents the channels degenerating in orthogonal transformation. However, their fixed rotation axes restrict the degree of freedom (DoF) of a quaternion weight, which is intrinsically four, to two. Therefore, the fixed-axis QCNNs lose their expression ability.

We have two proposals in this article. First, this article proposes a novel QCNN, which performs quaternion rotation on an input image and updates all the four parameters of its quaternion weights by backpropagation. Our proposed QCNNs realize physically clear quaternion convolution with which the channels do not degenerate, compared to simple product-type QCNNs. In addition, they have higher expression ability by updating all the four quaternion parameters than that of fixed-axis QCNNs.

Second proposal is to use two differently represented features: 1) PolSAR pseudocolor features; and 2) Stokes vectors normalized by their power. PolSAR pseudocolor features represent the intensities of different scattering mechanisms. Intensity features include fine textures of a PolSAR image. By learning the spatial textures, neural networks can distinguish between classes, such as town and forest, which have so similar scattering characteristics in pixels that they often misclassified each other. Another set of features, the Stokes vectors normalized by their total power, hold the degree of polarization (DoP) representing the state of a partially polarized wave. Most scattered waves are partially polarized, and the DoP is significant information for land classification. It is very effective to learn the DoP in discriminating between classes with low backscattering, such as sea and grass, on which we focus in this article. We train two neural networks with these respective features independently. Finally, we combine the two results for the two features by adopting classification result with higher values of network's softmax output pixel-by-pixel.

In the experiments, in this article, we compare four neural networks: 1) RVCNNs; 2) simple product-type QCNN; 3) fixed-axis QCNNs; and 4) our proposed QCNN. As experimental results, our proposed QCNNs show the best classification performance by learning the relationship between the components of input vectors with arbitrary quaternion rotations. The neural networks learning the PolSAR pseudocolor images succeed in discriminating town and forest, while the neural networks learning the normalized Stokes vectors succeed in sea and grass discrimination. Merging the two classification results for different features compensates for the shortcomings of respective features to improve the final classification results measured by F-scores.

The rest of this article is organized as follows. Section II presents the theory of the full-learning rotational QCNN. Section III describes the details of the experiments and results. Finally, Section IV concludes this article.

II. Full-Learning Rotational QCNN and Its Detailed Processing

Our proposed QCNNs conduct quaternion rotational convolution, which is physically clear compared to simple product-type QCNNs. In addition, quaternion weights of our proposed QCNNs have four DoFs by updating all the four parameters by backpropagation, while those of fixed-axis QCNNs have only two. Therefore, our proposed QCNNs have higher expression ability than that of fixed-axis QCNNs. This section presents the detailed processing of our proposed QCNNs.

A. Quaternion Arithmetic

First, we describe the expressions of various operations of quaternions. A quaternion is expressed as

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

$$= (x^e, x^i, x^j, x^k) \tag{2}$$

$$= \begin{bmatrix} x^e \\ x^i \\ x^j \\ x^k \end{bmatrix} \tag{3}$$

where $\boldsymbol{i}$, $\boldsymbol{j}$, and $\boldsymbol{k}$ stand for the three imaginary units of a quaternion. They satisfy the following Hamilton rules:

$$\boldsymbol{i}^2 = \boldsymbol{j}^2 = \boldsymbol{k}^2 = -1 \tag{4}$$

$$\boldsymbol{ij} = -\boldsymbol{ij} = \boldsymbol{k},\ \boldsymbol{jk} = -\boldsymbol{kj} = \boldsymbol{i},\ \boldsymbol{ki} = -\boldsymbol{ik} = \boldsymbol{j}. \tag{5}$$

The operators between quaternions $\boldsymbol{p} = (p^e, p^i, p^j, p^k) = (p^e, \boldsymbol{v}_p)$ and $\boldsymbol{q} = (q^e, q^i, q^j, q^k) = (q^e, \boldsymbol{v}_q)$ are defined as follows.

1) *Addition and Subtraction:*

$$\boldsymbol{p} \pm \boldsymbol{q} = (p^e \pm q^e, \boldsymbol{v_p} \pm \boldsymbol{v_q}). \tag{6}$$

2) *Outer Product:*

$$\boldsymbol{p} \otimes \boldsymbol{q} = (p^e q^e - \boldsymbol{v_p} \cdot \boldsymbol{v_q}, p^e \boldsymbol{v_q} + q^e \boldsymbol{v_p} + \boldsymbol{v_p} \times \boldsymbol{v_q}). \tag{7}$$

3) *Hadamard Product:*

$$\boldsymbol{p} \odot \boldsymbol{q} = (p^e q^e, p^i q^i, p^j q^j, p^k q^k). \tag{8}$$

4) *Inner Product:*

$$\boldsymbol{p} \cdot \boldsymbol{q} = p^e q^e + p^i q^i + p^j q^j + p^k q^k. \tag{9}$$

5) *Conjugate:*

$$\boldsymbol{p}^* = (p^e, -p^i, -p^j, -p^k). \tag{10}$$

6) *Norm:*

$$|\boldsymbol{p}| = \sqrt{(p^e)^2 + (p^i)^2 + (p^j)^2 + (p^k)^2}. \tag{11}$$

7) *Scalar:*

$$\lambda\boldsymbol{p} = (\lambda p^e, \lambda p^i, \lambda p^j, \lambda p^k). \tag{12}$$

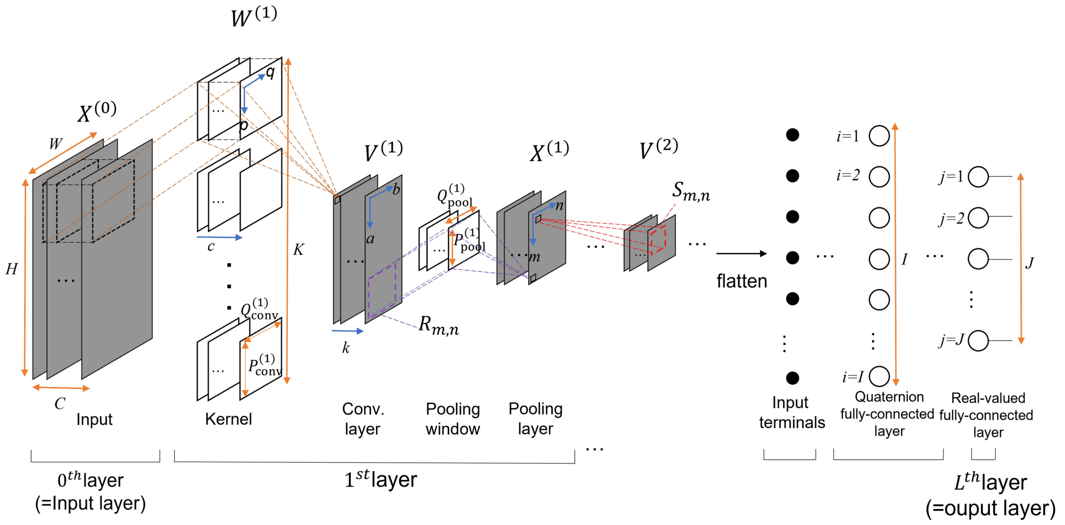


Fig. 1. Whole architecture of the QCNN.

B. Construction of the QCNN

The whole structure of the proposed QCNN is shown in Fig. 1. The basic architecture is the same as that of the conventional convolutional neural networks (CNNs). The QCNN has L layers. The 0th layer is the input-terminal layer. The last Lth layer is the output layer. All the signals, weights, and neurons are in the quaternion domain except for the real-valued Lth layer. The output values of the network are compared with the output teacher signals for learning, which is explained in Section II-C and -D. After the completion of the learning, classification is conducted.

In Fig. 1, c is the channel index for the lth layer's input $\boldsymbol{X}^{(l)}$ in the convolutional layer, k is the index for kernels, $(p,\ q)$, $(a,\ b)$, and $(m,\ n)$ are the positions in the kernels, the output of the convolutional layer, and the output of the pooling layer, respectively, for each channel.

C. Forward Processing

1) Convolutional Layer: Our proposed QCNNs perform quaternion rotational convolution, not simple quaternion multiplication. Therefore, their convolutional operation is physically clear, and prevents the channels of an input image degenerating in orthogonal transformation, compared to simple product-type QCNNs.

The inputs and outputs of the lth layer are denoted as $\boldsymbol{X}^{(l-1)} = [\boldsymbol{x}^{(l-1)}_{a+p,b+q,c}]$ and $\boldsymbol{V}^{(l)}_k = [\boldsymbol{v}^{(l)}_{a,b,k}]$, respectively, and the connection weights of convolutional kernels are represented as $\boldsymbol{W}^{(l)} = [\boldsymbol{w}^{(l)}_{p,q,c,k}]$, whose height and width are denoted as $P^{(l)}_{\text{conv}}$ and $Q^{(l)}_{\text{conv}}$, respectively. The quaternion rotational convolution is defined as

$$\boldsymbol{V}^{(l)} = \boldsymbol{f}(\boldsymbol{U}^{(l)}) \tag{13}$$

$$\boldsymbol{u}^{(l)}_{a,b,k} = \sum_{p,q,c} \frac{\boldsymbol{w}^{(l)}_{p,q,c,k} \otimes \boldsymbol{x}^{(l-1)}_{a+p,b+q,c} \otimes \boldsymbol{w}^{*(l)}_{p,q,c,k}}{|\boldsymbol{w}^{(l)}_{p,q,c,k}|} + \boldsymbol{b}^{(l)}_k \tag{14}$$

where $\boldsymbol{U}^{(l)} = [\boldsymbol{u}_{a,b,k}]$ and $\boldsymbol{b}^{(l)}_k$ are the neural inner state and bias, respectively. The function $\boldsymbol{f}$ is an activation function, defined as

$$\boldsymbol{f}(\boldsymbol{p}) = f(p^i)\boldsymbol{i} + f(p^j)\boldsymbol{j} + f(p^k)\boldsymbol{k} \tag{15}$$

where $\boldsymbol{p}$ is a purely imaginary quaternion and f is the rectified linear unit in our experiments.

This quaternion rotational convolution has already been employed by fixed-axis QCNNs [25], [26]. However, they fix their quaternion rotational axes, so that their trainable parameters of a single quaternion weight are only two: 1) scaling; and 2) rotational angle. Therefore, they lose expression ability. On the other hand, our proposed QCNNs realize the convolution by updating all the four parameters of their quaternion neural weights without fixing rotation axes. The learning processing in backpropagation is to be described later in Section II-D.

2) Pooling Layer: In our experiments, the pooling processing averages the inputs as

$$\boldsymbol{x}^{(l)}_{m,n,k} = \frac{1}{P^{(l)}_{\text{pool}} \cdot Q^{(l)}_{\text{pool}}} \sum_{(a,b)\in R_{m,n}} \boldsymbol{v}^{(l)}_{a,b,k} \tag{16}$$

where $P^{(l)}_{\text{pool}}$ and $Q^{(l)}_{\text{pool}}$ are the height and width of the pooling window $R_{m,n}$ to calculate $\boldsymbol{x}^{(l)}_{m,n,k}$, respectively. After the quaternion signals pass through convolutional layers and pooling layers, they are flattened and fed into quaternion fully-connected layers, whose processing follows [18].

3) Connection Between a Quaternion Fully-Connected Layer and a Real-Valued Output Layer: The output layer is a real-valued fully-connected layer directly following the last quaternion hidden layer. Before the quaternion signals are fed into the real-valued output layer, they are flattened to real-valued signals as

$$\begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \end{bmatrix} \rightarrow \begin{bmatrix} x_1^i \\ x_1^j \\ x_1^k \\ x_2^i \\ x_2^j \\ x_2^k \\ \vdots \end{bmatrix} \tag{17}$$

where $\boldsymbol{x}_d \ (d = 1, 2, \ldots)$ are purely imaginary quaternions whose imaginary components are x_d^i, x_d^j, and x_d^k.

Finally, the output of the QCNN goes through a softmax function, and the loss E is calculated with the output teacher signal $\boldsymbol{t}$, which is a one-hot vector. In our experiments, the loss E is the following cross entropy:

$$E = \sum_{j=1}^{J} -t_j \log y_j \tag{18}$$

where t_j denotes the jth component of the teacher signal $\boldsymbol{t}$ and y_j presents the output of the jth neuron of the output layer.

D. Backward Propagation in Learning

1) Connection Between a Quaternion Fully-Connected Layer and a Real-Valued Output Layer: Before real-valued gradients of the loss E in terms of the real-valued signals are passed to the previous quaternion fully-connected layer, they are converted to quaternion gradients as

$$\begin{bmatrix} \partial E/\partial x_1^i \\ \partial E/\partial x_1^j \\ \partial E/\partial x_1^k \\ \partial E/\partial x_2^i \\ \partial E/\partial x_2^j \\ \partial E/\partial x_2^k \\ \vdots \end{bmatrix} \rightarrow \begin{bmatrix} \partial E/\partial \boldsymbol{x}_1 \\ \partial E/\partial \boldsymbol{x}_2 \\ \vdots \end{bmatrix} \tag{19}$$

where $\partial E/\partial x_d^i$, $\partial E/\partial x_d^j$, and $\partial E/\partial x_d^k \ (d = 1, 2, \ldots)$ are converted to a single quaternion $\partial E/\partial \boldsymbol{x}_d$. After this conversion, the backpropagation dynamics in quaternion fully-connected layers follows [18].

2) Pooling Layer: In the lth pooling layer, $\partial E/\partial \boldsymbol{x}_{a,b,k}^{(l)}$ propagates from the $(l+1)$th layer. There are no trainable parameters in the pooling layer. However, we have to transfer the gradient given by the next layer as

$$\frac{\partial E}{\partial \boldsymbol{v}_{a,b,k}^{(l)}} = \frac{1}{P_{\text{pool}}^{(l)} \cdot Q_{\text{pool}}^{(l)}} \frac{\partial E}{\partial \boldsymbol{x}_{m,n,k}^{(l)}} \qquad ((a,b) \in R_{m,n}). \tag{20}$$

3) Convolutional Layer: Our proposed QCNNs update all the four quaternion parameters by backpropagation to have higher expression ability than that of fixed-axis QCNNs. Based on the chain rule in [17], the gradient of a loss function E in terms of lth-layer-kernel parameters is calculated as

$$\begin{aligned} \frac{\partial E}{\partial \boldsymbol{w}_{p,q,c,k}^{(l)}} &= -\sum_{a,b} \frac{1}{|\boldsymbol{w}_{p,q,c,k}^{(l)}|} \\ &\cdot \left\{ \frac{\partial E/\partial \boldsymbol{u}_{a,b,k}^{(l)} \cdot (\boldsymbol{w}_{p,q,c,k}^{(l)} \otimes \boldsymbol{x}_{a+p,b+q,c}^{(l-1)} \otimes \boldsymbol{w}_{p,q,c,k}^{*(l)})}{|\boldsymbol{w}_{p,q,c,k}^{(l)}|^2} \boldsymbol{w}_{p,q,c,k}^{(l)} \right. \\ &\left. - 2\partial E/\partial \boldsymbol{u}_{a,b,k}^{(l)} \otimes \boldsymbol{w}_{p,q,c,k}^{(l)} \otimes \boldsymbol{x}_{a+p,b+q,c}^{*(l-1)} \right\} \end{aligned} \tag{21}$$

and

$$\frac{\partial E}{\partial \boldsymbol{b}_k^{(l)}} = \sum_{a,b} \frac{\partial E}{\partial \boldsymbol{u}_{a,b,k}^{(l)}} \tag{22}$$

where

$$\frac{\partial E}{\partial \boldsymbol{u}_{a,b,k}^{(l)}} = \boldsymbol{f}'(\boldsymbol{u}_{a,b,k}) \odot \frac{\partial E}{\partial \boldsymbol{v}_{a,b,k}^{(l)}}. \tag{23}$$

The gradient, which propagates to the previous layer, is calculated as (24) shown at the bottom of this page, where $S_{m,n}$ is a set of points (a,b) connected to $\boldsymbol{x}_{m,n,k}^{(l-1)}$.

III. Experiments and Results

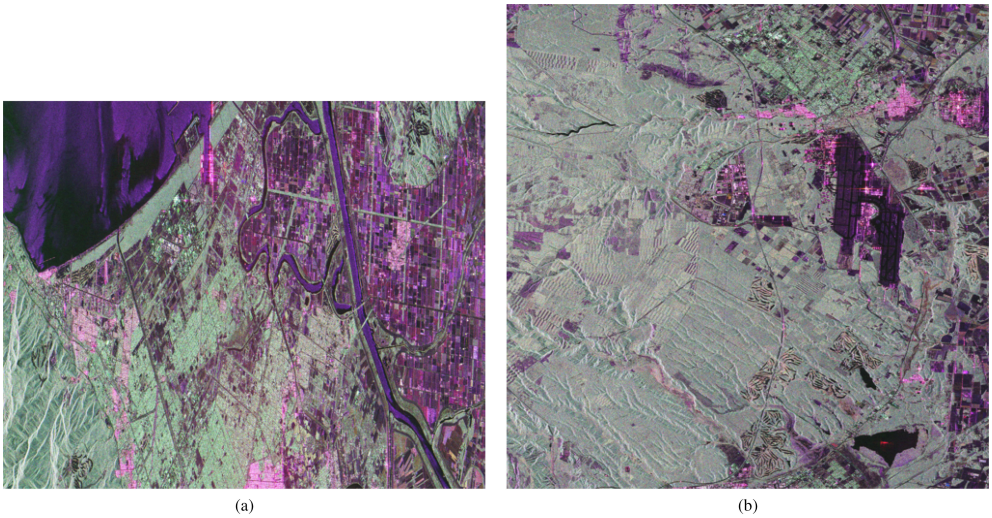
In this section, we explain the details of our experiments, including the dataset, two types of our proposed features for training, and learning and classification processing. Then, we present classification results of the four neural networks: 1) RVCNNs; 2) simple product-type QCNNs; 3) fixed-axis QCNNs; and 4) our proposed QCNNs, which learn each of our proposed features. Finally, the combination results are shown, including two more conventional neural networks.

A. Experimental Setup

1) Datasets and the Areas Where we Obtained Training Data and Test Data: PolSAR images in our experiments are a 4000 × 5000 pixel image of Ishikari Bay and a 8000 × 3000 pixel image of Tomakomai, Hokkaido, Japan, obtained by the Japan Aerospace Exploration Agency Advanced Land Observing Satellite-2, observed on August 2015 [offnadir angle: 25.4°, ascending, level 1.1 (no averaging)]. The RGB visualization of these two PolSAR images is shown in Fig. 2. We classify their pixels into four classes: 1) water; 2) town; 3) grass; and 4) forest.

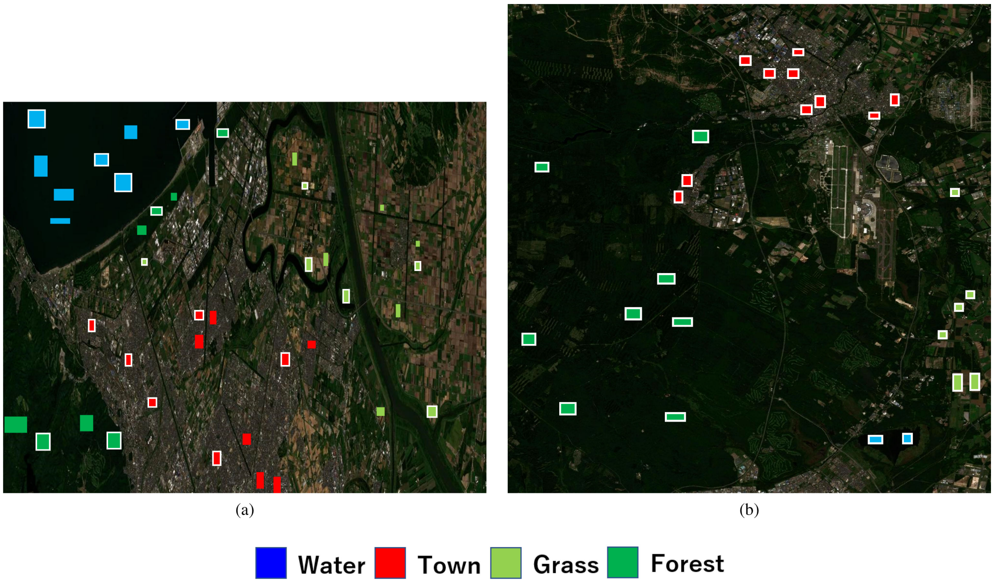
Fig. 3 shows the corresponding optical images obtained by Sentinel-2 on September 2017. The colored rectangles in Fig. 3 present the areas where we obtained training data and test data. We place a data window with a height and a width of both 15 pixels. We pick up 600-window training data for each class (total 2400) from the colored rectangular areas without white frames in the Ishikari image in Fig. 3(a). The 2400 test data

$$\frac{\partial E}{\partial \boldsymbol{x}_{m,n,c}^{(l-1)}} = \sum_{k} \sum_{(a,b) \in S_{m,n}} \frac{\boldsymbol{w}_{P_{\text{conv}}^{(l)}+m-a-1, Q_{\text{conv}}^{(l)}+n-b-1,c,k}^{*(l)} \otimes (\partial E/\partial \boldsymbol{u}_{a,b,k}^{(l)}) \otimes \boldsymbol{w}_{P_{\text{conv}}^{(l)}+m-a-1, Q_{\text{conv}}^{(l)}+n-b-1,c,k}^{(l)}}{\left|\boldsymbol{w}_{P_{\text{conv}}^{(l)}+m-a-1, Q_{\text{conv}}^{(l)}+n-b-1,c,k}^{(l)}\right|} \tag{24}$$


(a)

(b)

Fig. 2. Amplitude of scattering-matrix components of (a) Ishikari and (b) Tomakomai PolSAR images with 4000 × 5000 and 8000 × 3000 pixels, respectively. Note that the 90 percentile value of each components of scattering matrix in this image is set as the upper limits for color assignment, respectively, (R:$|S_{HH}|$, G:$|S_{HV}|$, B:$|S_{VV}|$).


(a)

(b)

Fig. 3. (a) Ishikari Bay and (b) Tomakomai observed by Sentinel-2 optical images, including the areas from which we obtained training data (unframed) and test data (framed).

(600 for each class) are obtained from the white-framed rectangles in each of the two PolSAR images, so that we obtain 4800 test data in total. The four colors of the rectangular areas correspond to the four classes of the legends below the images.

2) Features Used in the Experiments: The experimental features are derived from the complex scattering matrix expressed as

$$\boldsymbol{S} = \begin{bmatrix} S_{\mathrm{HH}} & S_{\mathrm{HV}} \\ S_{\mathrm{VH}} & S_{\mathrm{VV}} \end{bmatrix}. \tag{25}$$

We propose using two differently represented features: 1) PolSAR pseudocolor features; and 2) Stokes vectors normalized by their total power. They have totally different physical characteristics. We combine two classification results for these respective features to improve the final classification complementarily.

a) PolSAR Pseudocolor Image: The scattering matrix can be decomposed under the Pauli decomposition as

$$a = \frac{S_{\mathrm{HH}} + S_{\mathrm{VV}}}{\sqrt{2}}, b = \frac{S_{\mathrm{HH}} - S_{\mathrm{VV}}}{\sqrt{2}}, c = \frac{S_{\mathrm{HV}} + S_{\mathrm{VH}}}{\sqrt{2}}. \tag{26}$$

The PolSAR pseudocolor features are constructed by assigning $|b|^2$, $|c|^2$, and $|a|^2$ to red, green, and blue components, respectively [27]. Fixed-axis QCNNs employ these features as input RGB color images [26]. Intensity features, such as PolSAR pseudocolor features, have so fine texture of land surface that neural networks are able to extract the spatial texture by learning them. SAR images are often spatially averaged to mitigate speckle noise. However, since averaging hazes fine textures,

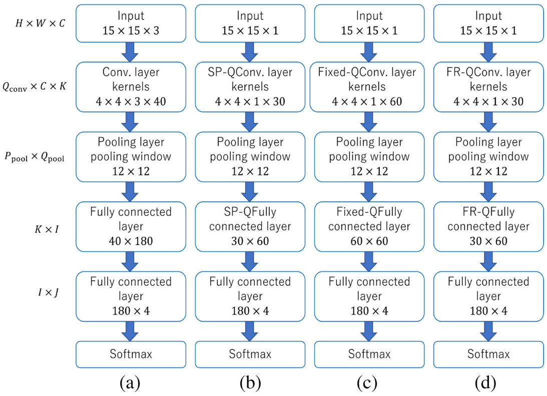

H × W × C
P_conv × Q_conv × C × K
P_pool × Q_pool
K × I
I × J
Input 15 × 15 × 3
Conv. layer kernels 4 × 4 × 3 × 40
Pooling layer pooling window 12 × 12
Fully connected layer 40 × 180
Fully connected layer 180 × 4
Softmax
(a)
Input 15 × 15 × 1
SP-QConv. layer kernels 4 × 4 × 1 × 30
Pooling layer pooling window 12 × 12
SP-QFully connected layer 30 × 60
Fully connected layer 180 × 4
Softmax
(b)
Input 15 × 15 × 1
Fixed-QConv. layer kernels 4 × 4 × 1 × 60
Pooling layer pooling window 12 × 12
Fixed-QFully connected layer 60 × 60
Fully connected layer 180 × 4
Softmax
(c)
Input 15 × 15 × 1
FR-QConv. layer kernels 4 × 4 × 1 × 30
Pooling layer pooling window 12 × 12
FR-QFully connected layer 30 × 60
Fully connected layer 180 × 4
Softmax
(d)

Fig. 4. Architectures of the three neural networks learning the PolSAR pseudocolor features. (a) RVCNN. (b) Simple product-type QCNN. (c) Fixed-axis QCNN. (d) Our proposed QCNN. The symbols in the left-hand side of this figure correspond to parameters in Fig. 1, and SP and FR mean simple-product and full-learning rotational, respectively.

we do not apply any spatial averaging process on the PolSAR images, so that we can utilize fine textures.

When the PolSAR pseudocolor features are fed into the networks, they should be normalized to a certain range. In general, some pixels of a PolSAR image have very large amplitude values compared to others. If we try to normalize the feature values in such a manner that their maximum value becomes unity, almost all the other amplitude values become zero. Therefore, we normalize them in another way that their 90 percentile values in an image become unity. After the normalization, the values larger than unity are set as unity, so that all the values fall in the range 0–1.

The architectures of the neural networks learning the PolSAR pseudocolor features are shown in Fig. 4. The values in Fig. 4 represent the number of real-valued neurons and quaternion neurons. Since a single quaternion neuron has multiple parameters, the number of parameters of simple product quaternion layers and full-learning rotational layers is four times the values in Fig. 4, while that of fixed-axis quaternion layers is twice. We set the number of parameters of each neural network in Fig. 4 to be the same as others. All the neural networks consist of one convolutional layer, one pooling layer, one hidden fully-connected layer, and the output layer. The pooling layer conducts average pooling to deemphasize pixels that could be contaminated with speckle noise in case of max pooling. Since in the sliding window classification, the positional relationship on the feature map is not important for classification, the pooling layer averages the entire feature map.

b) Stokes Vectors Normalized by Their Total Power: In order to derive Stokes vectors, we need to prepare incidence polarization $\boldsymbol{E}^{\mathrm{i}} = [E^{\mathrm{i}}_{\mathrm{H}}, E^{\mathrm{i}}_{\mathrm{V}}]^T$. We employed six incidence polarizations: 1) horizontal polarization; 2) vertical polarization; 3) +45° linear polarization; 4) −45° linear polarization; 5) left-handed circular polarization; and 6) right-handed circular polarization in our experiments. Then, the received waves $\boldsymbol{E}^{\mathrm{r}} = [E^{\mathrm{r}}_{\mathrm{H}}, E^{\mathrm{r}}_{\mathrm{V}}]^T$ are expressed as

$$\begin{bmatrix} E^{\mathrm{r}}_{\mathrm{H}} \\ E^{\mathrm{r}}_{\mathrm{V}} \end{bmatrix} = \begin{bmatrix} S_{\mathrm{HH}} & S_{\mathrm{HV}} \\ S_{\mathrm{VH}} & S_{\mathrm{VV}} \end{bmatrix} \begin{bmatrix} E^{\mathrm{i}}_{\mathrm{H}} \\ E^{\mathrm{i}}_{\mathrm{V}} \end{bmatrix}. \tag{27}$$

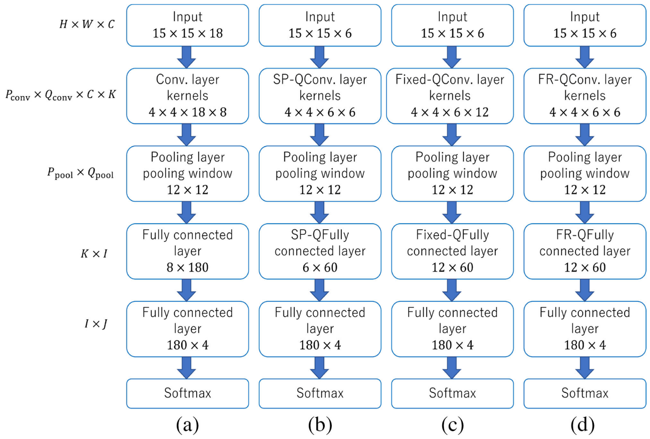

H × W × C
P_conv × Q_conv × C × K
P_pool × Q_pool
K × I
I × J
Input 15 × 15 × 18
Conv. layer kernels 4 × 4 × 18 × 8
Pooling layer pooling window 12 × 12
Fully connected layer 8 × 180
Fully connected layer 180 × 4
Softmax
(a)
Input 15 × 15 × 6
SP-QConv. layer kernels 4 × 4 × 6 × 6
Pooling layer pooling window 12 × 12
SP-QFully connected layer 6 × 60
Fully connected layer 180 × 4
Softmax
(b)
Input 15 × 15 × 6
Fixed-QConv. layer kernels 4 × 4 × 6 × 12
Pooling layer pooling window 12 × 12
Fixed-QFully connected layer 12 × 60
Fully connected layer 180 × 4
Softmax
(c)
Input 15 × 15 × 6
FR-QConv. layer kernels 4 × 4 × 6 × 6
Pooling layer pooling window 12 × 12
FR-QFully connected layer 12 × 60
Fully connected layer 180 × 4
Softmax
(d)

Fig. 5. Architectures of the three neural networks learning the Stokes vectors normalized by their total power. (a) RVCNN. (b) Simple product-type QCNN. (c) Fixed-axis QCNN. (d) Our proposed QCNN. The symbols in the left-hand side of this figure correspond to parameters in Fig. 1, and SP and FR mean simple-product and full-learning rotational, respectively.

Then, the corresponding Stokes vectors $\boldsymbol{g}$ are obtained as

$$\boldsymbol{g} = \begin{bmatrix} \langle g_0 \rangle \\ \langle g_1 \rangle \\ \langle g_2 \rangle \\ \langle g_3 \rangle \end{bmatrix} = \begin{bmatrix} \langle E^{\mathrm{r}}_{\mathrm{H}} E^{\mathrm{r}*}_{\mathrm{H}} + E^{\mathrm{r}}_{\mathrm{V}} E^{\mathrm{r}*}_{\mathrm{V}} \rangle \\ \langle E^{\mathrm{r}}_{\mathrm{V}} E^{\mathrm{r}*}_{\mathrm{H}} - E^{\mathrm{r}}_{\mathrm{V}} E^{\mathrm{r}*}_{\mathrm{V}} \rangle \\ \langle E^{\mathrm{r}}_{\mathrm{H}} E^{\mathrm{r}*}_{\mathrm{V}} - E^{\mathrm{r}}_{\mathrm{V}} E^{\mathrm{r}*}_{\mathrm{H}} \rangle \\ \langle j(E^{\mathrm{r}}_{\mathrm{H}} E^{\mathrm{r}*}_{\mathrm{V}} - E^{\mathrm{r}}_{\mathrm{V}} E^{\mathrm{r}*}_{\mathrm{H}}) \rangle \end{bmatrix} \tag{28}$$

where $\langle \cdot \rangle$ denotes the spatial averaging process. Finally, we normalize the Stokes vectors by their total power $\langle g_0 \rangle$ as

$$\boldsymbol{P} = \begin{bmatrix} \langle g_1 \rangle / \langle g_0 \rangle \\ \langle g_2 \rangle / \langle g_0 \rangle \\ \langle g_3 \rangle / \langle g_0 \rangle \end{bmatrix}. \tag{29}$$

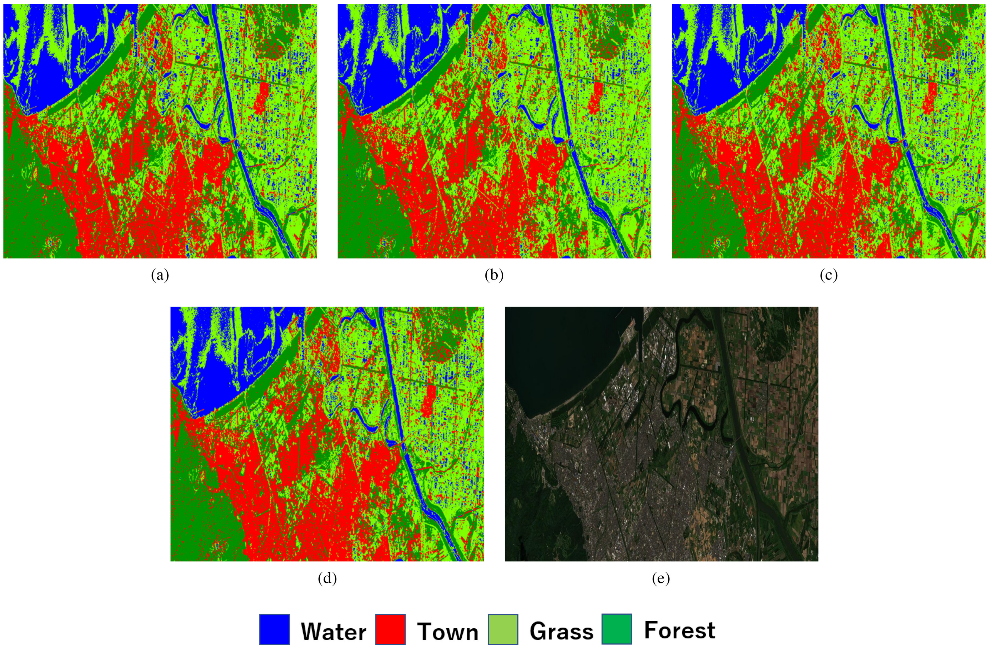
The normalized Stokes vectors have the DoP information in their norm as

$$\mathrm{DoP} = \sqrt{\frac{\langle g_1 \rangle^2 + \langle g_2 \rangle^2 + \langle g_3 \rangle^2}{\langle g_0 \rangle^2}}. \tag{30}$$

In general, scattered waves are partially polarized, and it is very important to learn the DoP information, which is totally different from intensity. The architectures of the neural networks learning the normalized Stokes vectors are shown in Fig. 5.

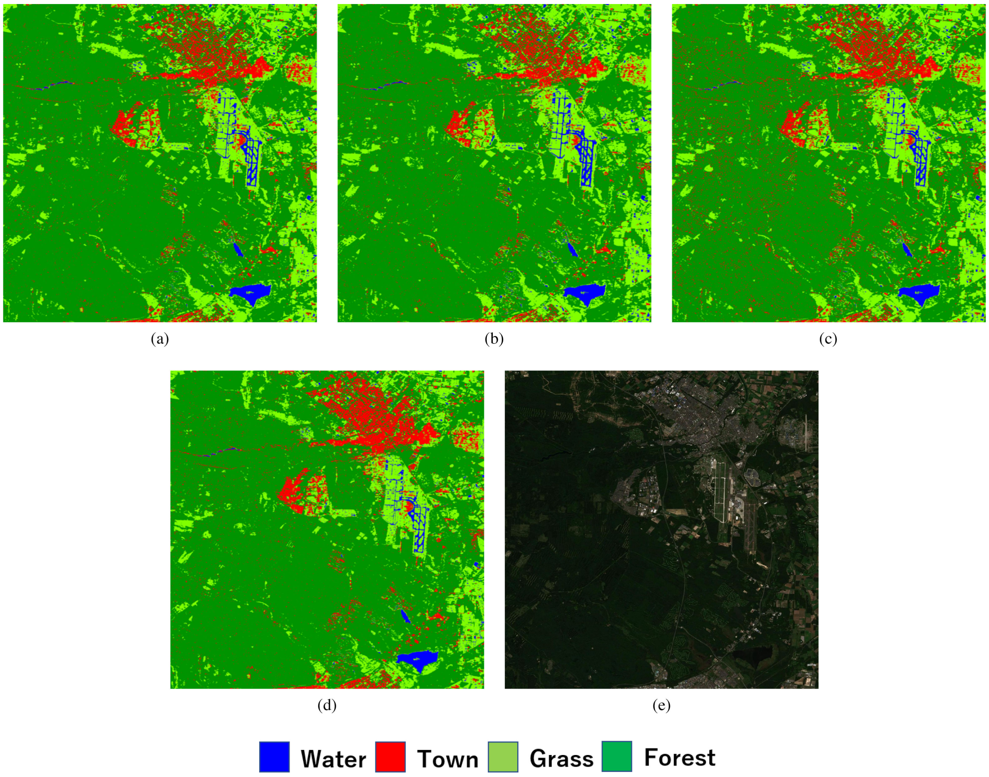
3) Learning and Classification Processing: We train two neural networks with the abovementioned two features, respectively, and independently. The neural networks are trained with a minibatch having 40 data. One epoch has $2400/40 = 60$ times weight updates. We train the neural networks learning the PolSAR pseudocolor features for 500 epochs and those learning the normalized Stokes vectors for 1000 epochs. The loss function is calculated as the cross-entropy loss in (18). The weight updating is conducted by stochastic gradient descent with momentum whose learning rate is 0.05 in learning PolSAR pseudocolor feature and 0.001 in learning normalized Stokes vectors, and momentum parameter is 0.9 in our experiments.

In classification processing, a 15 × 15 pixel-window image cropped from the PolSAR image is fed into the neural networks one-by-one. Then, the loss is calculated by the outputs and the


(a) (b) (c)

(d) (e)

Fig. 6. Classification results of Ishikari scene for PolSAR pseudocolor features by the following. (a) RVCNN. (b) Simple product-type QCNN. (c) Fixed-axis QCNN. (d) Proposed QCNN. (e) Sentinel-2 optical image for reference.


(a) (b) (c)

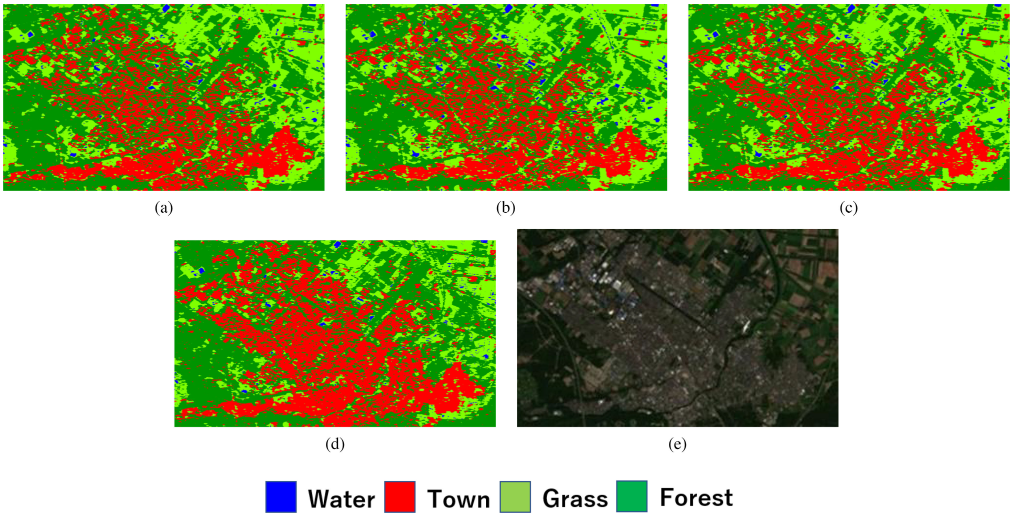
(d) (e)

Fig. 7. Classification results of Ishikari scene for PolSAR pseudocolor features by the following. (a) RVCNN. (b) Simple product-type QCNN. (c) Fixed-axis QCNN. (d) Proposed QCNN. (e) Sentinel-2 optical image for reference.

output teacher signals for each class, and the class having minimum loss is the class, which the pixel belongs to. This process is conducted until all the pixels are classified. Finally, we adopt the classification result having a higher value of network's softmax output pixel-by-pixel to combine these two classification results for the respective features.

B. Results

1) Classification Results for the PolSAR Pseudocolor Features: Fig. 6 shows the classification results of the PolSAR pseudocolor features obtained for the Ishikari scene. It is found that the sea area in the upper left is significantly misclassified into grass. Since the backscattering in the sea and grass is


(a) (b) (c)

(d) (e)

Fig. 8. Classification results of Ishikari town area for PolSAR pseudocolor features by the following. (a) RVCNN. (b) Simple product-type QCNN. (c) Fixed-axis QCNN. (d) Proposed QCNN. (e) Sentinel-2 optical image for reference.

TABLE I
CLASSIFICATION RESULTS OF RVCNN, SIMPLE PRODUCT-TYPE QCNN, FIXED-AXIS QCNN, AND PROPOSED QCNN FOR POLSAR PSEUDOCOLOR FEATURES

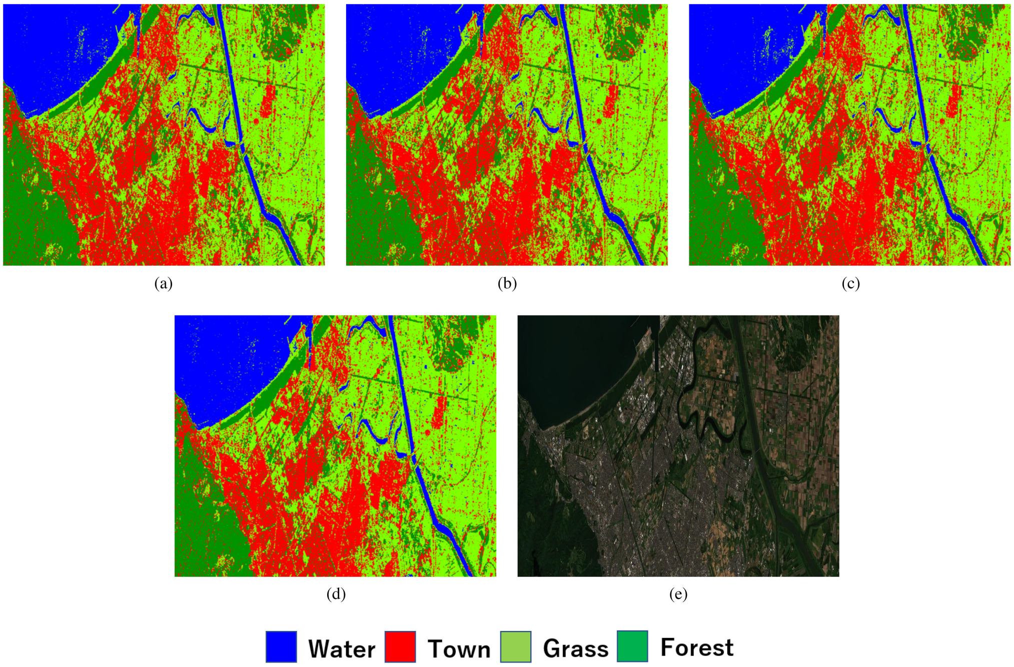
Neural network	Class	water	town	grass	forest	F-score	avg F-score
RVCNN	water	943	0	257	0	0.759	0.838
	town	0	1065	6	129	0.931	
	grass	342	5	836	17	0.727	
	forest	0	17	0	1183	0.936	
Simple Product-type QCNN	water	962	0	238	0	0.764	0.838
	town	0	1067	21	112	0.924	
	grass	355	7	830	8	0.725	
	forest	0	35	0	1165	0.938	
Fixed-axis QCNN	water	954	0	246	0	0.761	0.831
	town	0	1084	18	98	0.918	
	grass	355	5	828	12	0.723	
	forest	0	74	0	1126	0.925	
Proposed QCNN	water	967	0	233	0	0.777	0.857
	town	0	1134	4	62	0.953	
	grass	322	9	854	15	0.746	
	forest	0	37	0	1163	0.953	

very small, the neural networks learning intensity features, such as PolSAR pseudocolor features, cannot differentiate between them. In addition, there are some misclassifications between forest and town in the lower left of the results. In general, town and forest are misclassified each other because of their similar scattering characteristics.

Fig. 7 shows the classification results of Tomakomai by training data obtained from the Ishikari image. The town area in the upper right is misclassified into forest, while the forest area in the lower left into town. The buildings in this town area have such a large angle to the direction of the radar beam that the cross-polarization components of scattering in this town area, namely S_{HV} and S_{VH}, become relatively large, such as forest. Therefore, it is very difficult to discriminate between town and forest generally. Fig. 8 shows the enlarged views of this town area. Our proposed QCNN classifies it more accurately than the RVCNN, simple product-type QCNN, and fixed-axis QCNN, because it can learn the relationship between the channels of the input images by arbitrary quaternion rotations.

Table I presents the confusion matrices and F-scores of respective neural networks measured by test data obtained from both of the two PolSAR images. From the confusion matrices, it can be seen that the water and grass, and the town and forest are misclassified each other, respectively. Note that, for the PolSAR pseudocolor features, we find much misclassification between the water and the grass. The F-scores show that our proposed QCNN especially succeeded in discriminating between town and forest compared to the other two networks. Our proposed QCNN shows the best average F-score.

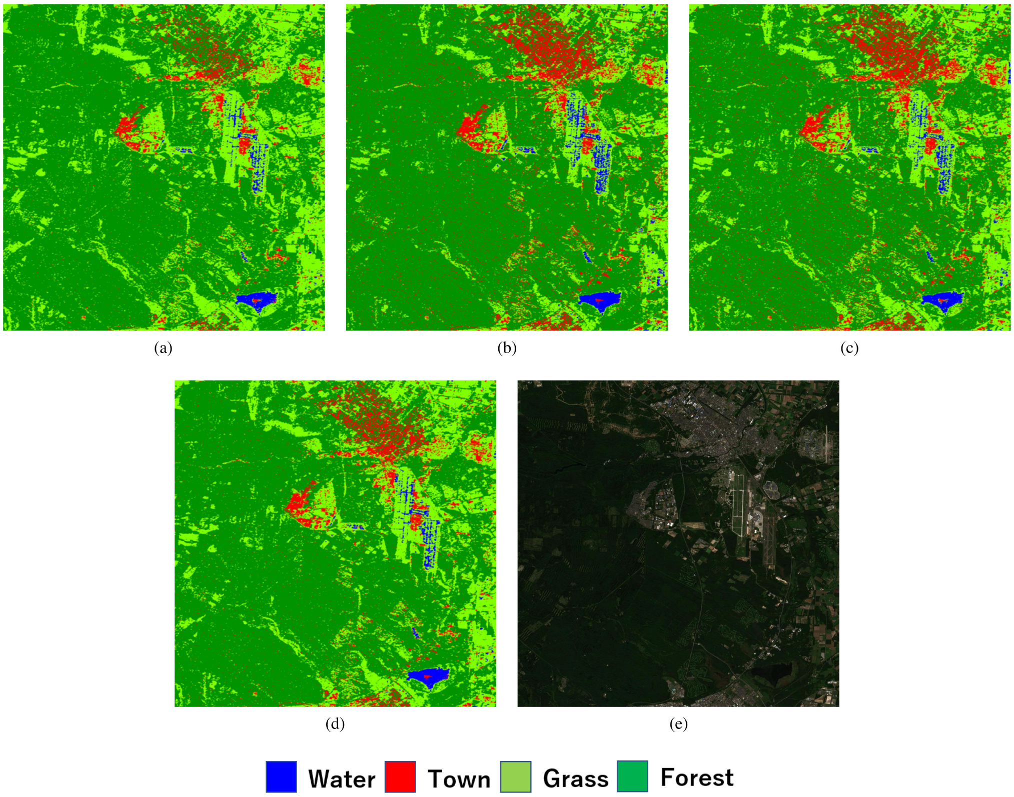
2) Classification Results for the Normalized Stokes Vectors: Fig. 9 shows the classification results of Ishikari for the normalized Stokes vectors. The sea area in the upper left of the classification results is accurately classified compared to that



(a) (b) (c)

(d) (e)

Fig. 9. Classification results of Ishikari scene for normalized Stokes vectors by the following. (a) RVCNN. (b) Simple product-type QCNN. (c) Fixed-axis QCNN. (d) Proposed QCNN. (e) Sentinel-2 optical image.



(a) (b) (c)

(d) (e)

Fig. 10. Classification results of Tomakomai scene for normalized Stokes vectors by the following. (a) RVCNN. (b) Simple product-type QCNN. (c) Fixed-axis QCNN. (d) Proposed QCNN. (e) Sentinel-2 optical image.

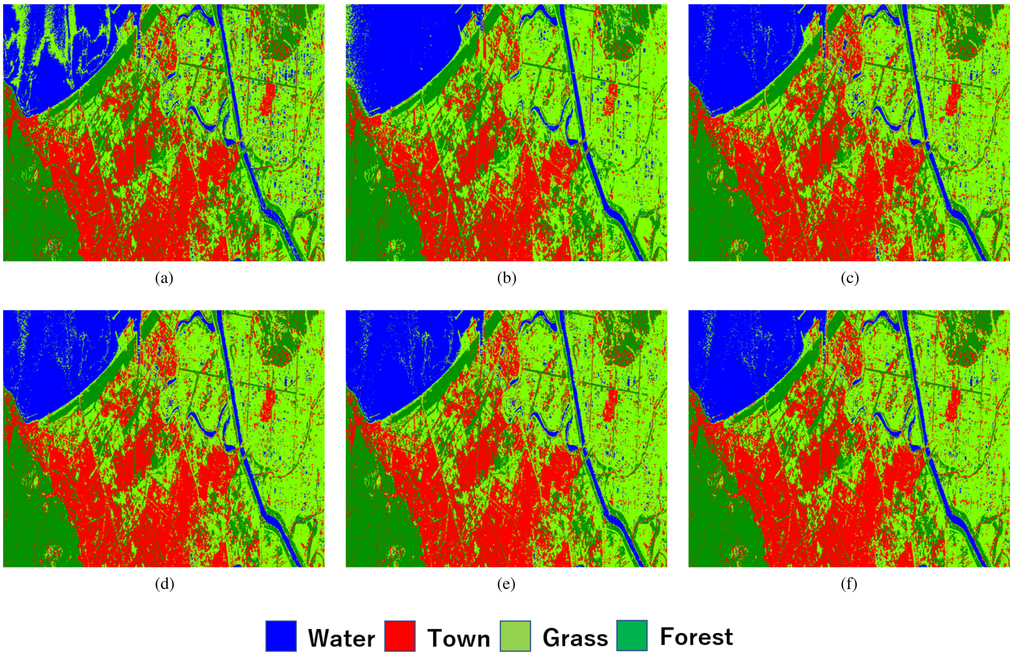


Fig. 11. Classification results of the Ishikari scene by the following. (a) Dual-CNN. (b) CVCNN. And, merged classification results of the Ishikari scene by the following. (c) RVCNN. (d) Simple product-type QCNN. (e) Fixed-axis QCNN. (f) Proposed QCNN.

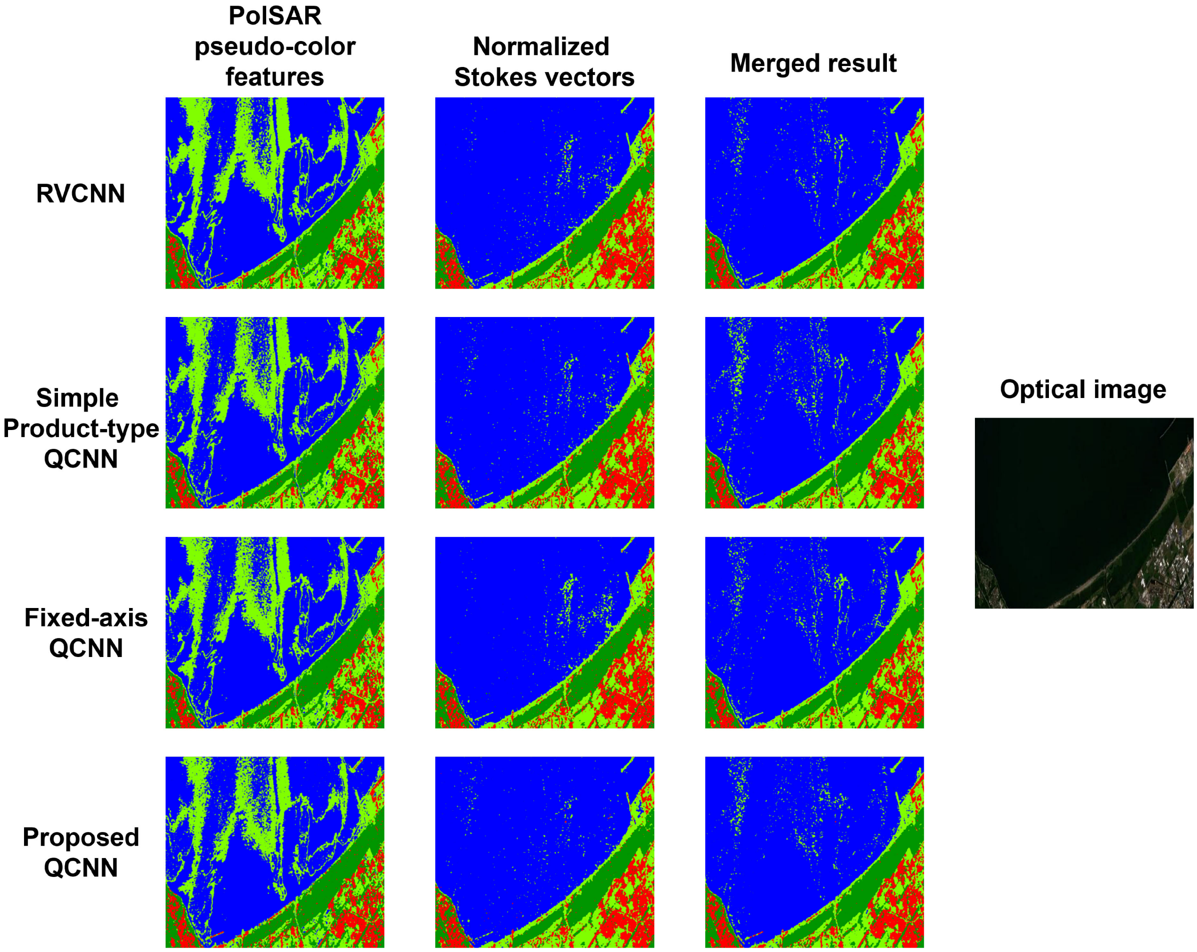


Fig. 12. Classification results of sea area in Ishikari by respective network and features as well as the corresponding area of a Sentinel-2 optical image.

TABLE II
CLASSIFICATION RESULTS OF RVCNN, SIMPLE PRODUCT-TYPE QCNN, FIXED-AXIS QCNN, AND PROPOSED QCNN FOR NORMALIZED STOKES VECTORS

Neural network	Class	water	town	grass	forest	F-score	avg F-score
RVCNN	water	1170	0	30	0	0.963	0.890
	town	0	852	136	212	0.808	
	grass	59	31	1104	6	0.890	
	forest	0	26	10	1164	0.902	
Simple Product-type QCNN	water	1172	1	27	0	0.930	0.906
	town	0	1004	85	111	0.885	
	grass	149	24	1023	4	0.875	
	forest	0	40	4	1156	0.936	
Fixed-axis QCNN	water	1156	0	44	0	0.941	0.915
	town	0	1049	48	103	0.900	
	grass	100	47	1039	14	0.887	
	forest	0	36	12	1152	0.933	
Proposed QCNN	water	1191	0	9	0	0.969	0.919
	town	0	948	103	149	0.874	
	grass	69	15	1112	4	0.927	
	forest	0	24	9	1167	0.915	

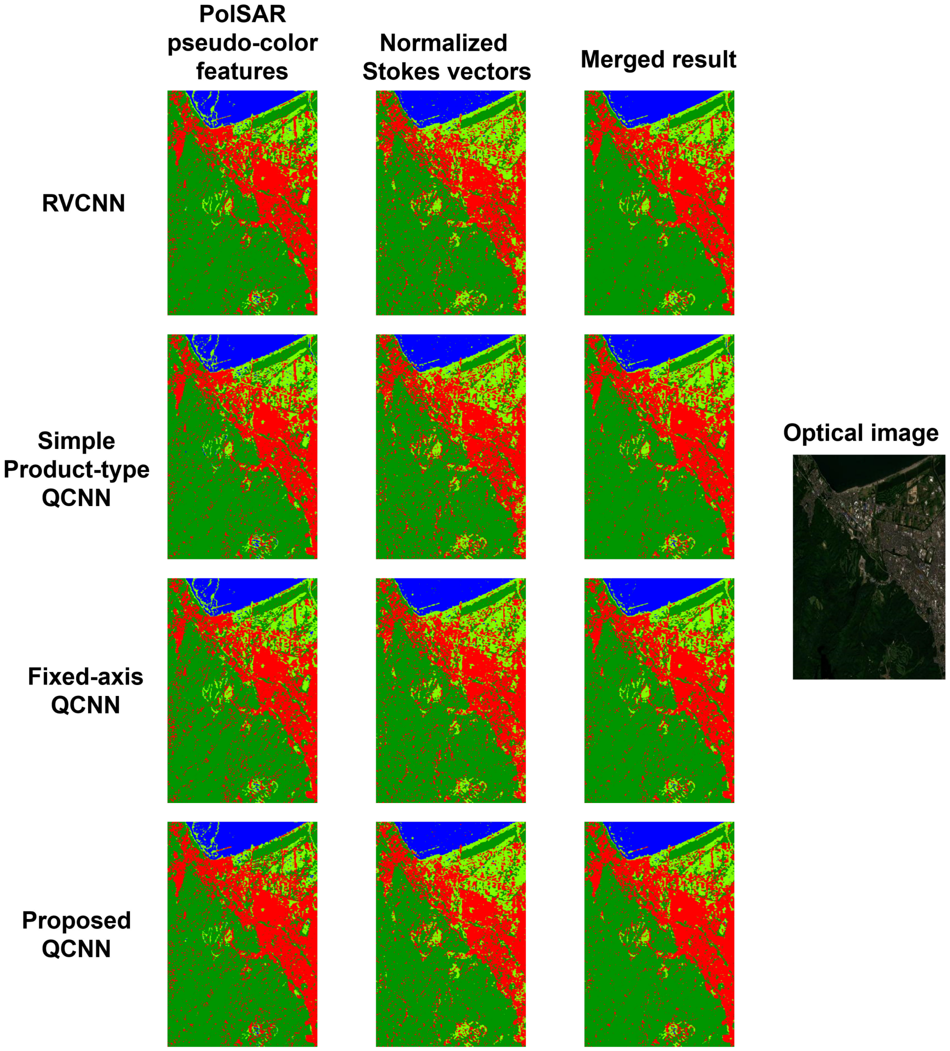
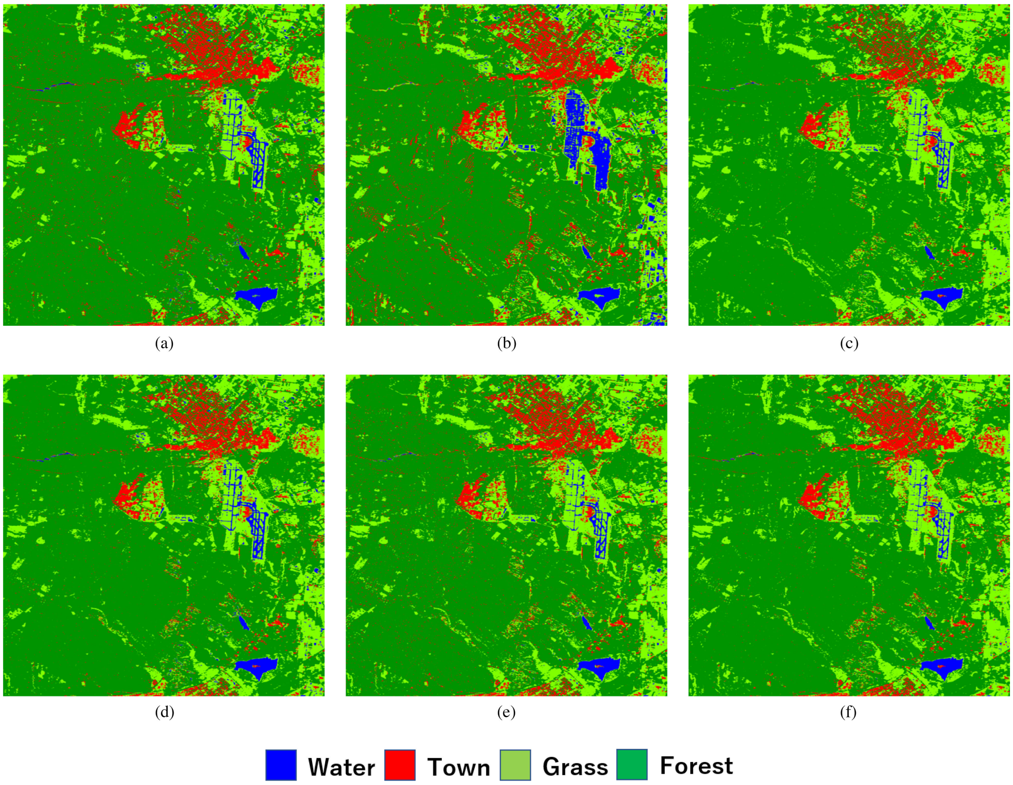


Fig. 13. Classification results of forest area in Ishikari by respective networks and features as well as the corresponding area of a Sentinel-2 optical image.


(a) (b) (c)

(d) (e) (f)

Fig. 14. Classification results of the Tomakomai scene by the following. (a) Dual-CNN. (b) CVCNN. And, merged classification results of the Tomakomai scene by the following. (c) RVCNN. (d) Simple product-type QCNN. (e) Fixed-axis QCNN. (f) Proposed QCNN.

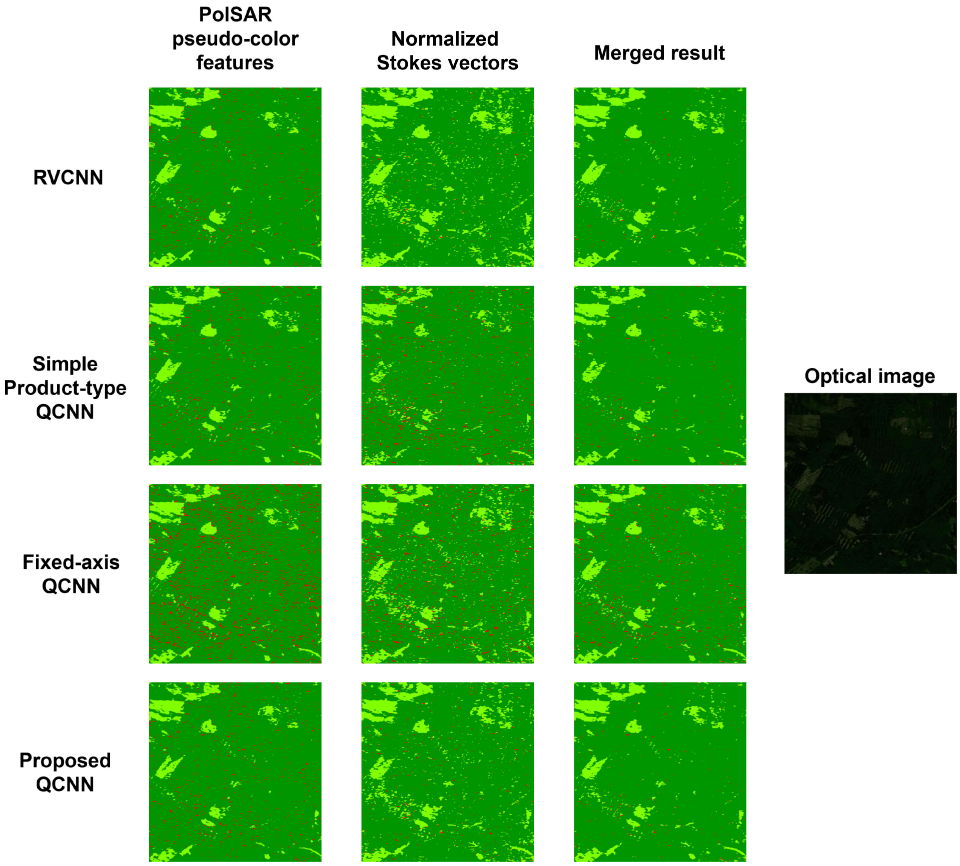


Fig. 15. Classification results of forest area in Tomakomai by respective networks and features as well as the corresponding area of a Sentinel-2 optical image.

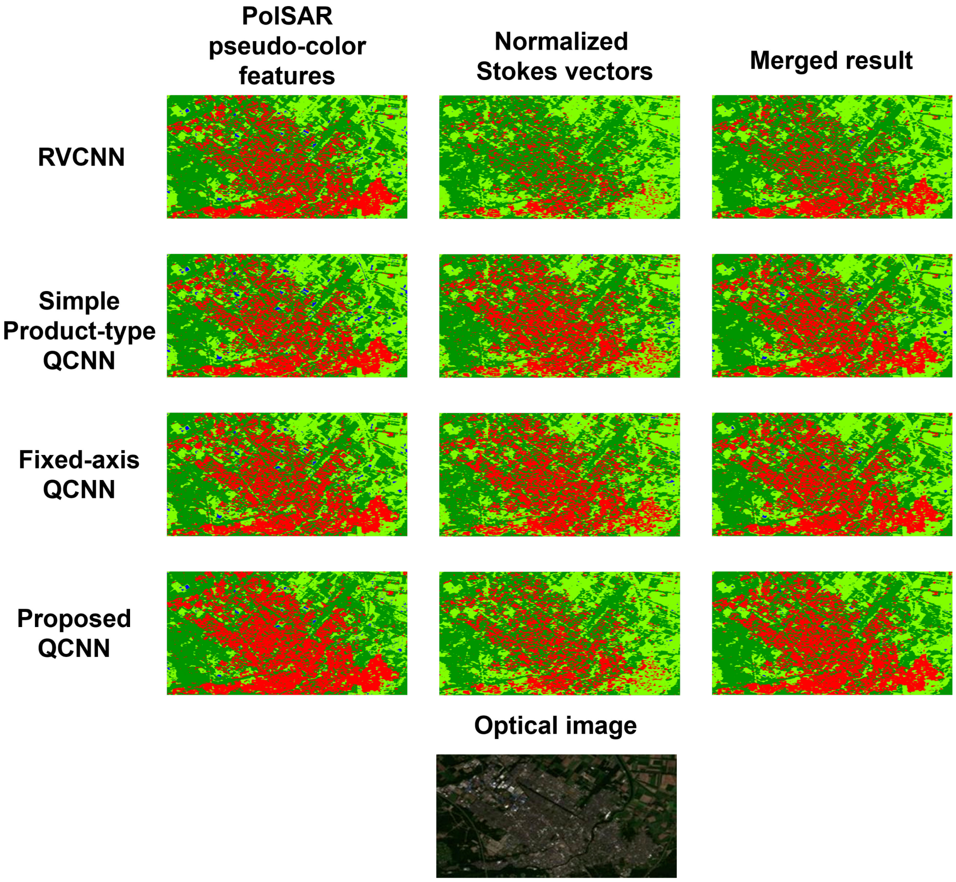


Fig. 16. Classification results of town area in Tomakomai by respective networks and features as well as the corresponding area of a Sentinel-2 optical image.

TABLE III
MERGED RESULTS OF RVCNN, SIMPLE PRODUCT-TYPE QCNN, FIXED-AXIS QCNN, AND QCNN

Neural network	Class	water	town	grass	forest	F-score	avg F-score
Dual-CNN	water	995	0	205	0	0.784	0.855
	town	0	1113	6	81	0.944	
	grass	345	6	839	10	0.746	
	forest	0	40	0	1160	0.947	
CVCNN	water	1191	0	9	0	0.848	0.874
	town	0	1080	26	94	0.938	
	grass	418	2	762	18	0.763	
	forest	0	22	0	1178	0.946	
RVCNN	water	1175	0	25	0	0.915	0.914
	town	0	1058	22	120	0.928	
	grass	193	14	966	27	0.873	
	forest	0	9	0	1191	0.939	
Simple Product-type QCNN	water	1150	0	50	0	0.894	0.914
	town	0	1114	18	68	0.950	
	grass	224	19	940	17	0.851	
	forest	0	12	0	1188	0.961	
Fixed-axis QCNN	water	1168	0	32	0	0.901	0.916
	town	0	1122	10	68	0.947	
	grass	224	21	938	17	0.861	
	forest	0	27	0	1173	0.954	
Proposed QCNN	water	1156	0	44	0	0.916	0.930
	town	0	1135	14	51	0.962	
	grass	183	15	986	16	0.879	
	forest	0	10	0	1190	0.969	

TABLE IV
SUMMARY OF THE AVERAGE F-SCORES FOR EACH NEURAL NETWORK AND EACH FEATURE

Feature	RVCNN	Simple product-type QCNN	Fixed-axis QCNN	Proposed QCNN
PolSAR pseudo-color features	0.838	0.838	0.831	0.857
Normalized Stokes vectors	0.890	0.906	0.915	0.919
Merged results	0.914	0.914	0.916	0.930

of the results for the PolSAR pseudocolor features. Scattering waves at water surface have low coherence due to ripples, while possible wind-swaying grass is invisible by an *L*-band electromagnetic wave. Therefore, the DoP values for these two areas become different. The neural networks are able to differentiate between the sea and grass by learning partially polarized state from the normalized Stokes vectors.

Fig. 10 shows the classification results of Tomakomai for the normalized Stokes vectors. The town area in the upper right is significantly misclassified into forest. The town and forest have so similar scattering characteristics in pixels that they need to be discriminated by spatial texture. The fine textures of PolSAR images are included mainly in intensity features. However, the Stokes vectors are normalized by their intensity, so that the textures are lost, coupled with the impact of spatial averaging. Therefore, the neural networks cannot learn spatial textures, and fail to discriminate between town and forest.

Table II presents the confusion matrices and F-scores of respective neural networks. Compared to Table I for the PolSAR pseudocolor features, all the neural networks succeeded in discriminating between water and grass, while they fail to distinguish between town and forest. From these two experiments for the two different features, it is found that the classification results largely depend on learned features.

3) Merged Results: The previous results show that the PolSAR pseudocolor features are suitable for discrimination between town and forest, but not between sea and grass, while the normalized Stokes vectors are opposite. Fig. 11 shows the merged results of Ishikari using the two features, and other two more conventional neural networks: 1) dual-CNN [28]; and 2) CVCNN [13]. The sea area seems accurately classified. Fig. 12 shows the classification results of the sea area presented so far for respective neural networks and features. Compared to the classification results for the PolSAR pseudocolor features, the sea area is accurately classified. In addition, as shown in Fig. 13, the forest area in the lower left is more accurately classified compared to the classification results for each feature.

The merged results of Tomakomai area are shown in Fig. 14. The forest area in the lower left and the town area in the upper right are accurately classified. Fig. 15 shows the classification results of the forest area in Tomakomai presented so far for respective neural networks and features. The forest area quality of the merged results is improved compared to the result for each feature, respectively. Fig. 16 shows the classification results of the town area in the upper right in Tomakomai. The merged results of this town area are worse than the results for the PolSAR pseudocolor features, while they are better than those for the normalized Stokes vectors.

Table III presents the confusion matrices and F-scores of respective neural networks. Compared to Tables I and II, the overall classification results are improved by the merging method. Table IV summarizes the average F-scores of respective neural networks and features presented so far. The results of our proposed QCNN show the best classification performance for respective features. In addition, the merging method significantly improved the classification results, demonstrating its effectiveness. Our proposed QCNNs are very effective for the selected-type of PolSAR data in this article.

IV. CONCLUSION

We proposed full-learning rotational QCNNs, which learn data by updating all four parameters of a quaternion. The proposed QCNNs showed better classification performance than the conventional RVCNNs, simple product-type QCNN, and fixed-axis QCNNs. This is because the full-learning QCNN can learn the relationship among the channels of an input image with arbitrary quaternion rotations. In addition, we showed that the classification results largely depend on input features. Specifically, the PolSAR pseudocolor features are suitable for discrimination between town and forest, but not between water and grass, while the normalized Stokes vectors are opposite. We merged the classification results for respective features, simply based on network's softmax outputs to highly improve the final classification by compensating for the shortcomings of respective features successfully.

ACKNOWLEDGMENT

The Advanced Land Observing Satellite (ALOS) original data are copyrighted by the Japan Aerospace Exploration Agency (JAXA) and provided under JAXA Fourth ALOS Research Announcement PI No. 1154 (AH).

REFERENCES

[1] J.-S. Lee, M. R. Grunes, and E. Pottier, "Quantitative comparison of classification capability: Fully polarimetric versus dual and single-polarization SAR," *IEEE Trans. Geosci. Remote Sens.*, vol. 39, no. 11, pp. 2343–2351, Nov. 2001.

[2] S. R. Cloude and E. Pottier, "An entropy based classification scheme for land applications of polarimetric SAR," *IEEE Trans. Geosci. Remote Sens.*, vol. 35, no. 1, pp. 68–78, Jan. 1997.

[3] A. Freeman and S. L. Durden, "A three-component scattering model for polarimetric SAR data," *IEEE Trans. Geosci. Remote Sens.*, vol. 36, no. 3, pp. 963–973, May 1998.

[4] Y. Yamaguchi, T. Moriyama, M. Ishido, and H. Yamada, "Four-component scattering model for polarimetric SAR image decomposition," *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 8, pp. 1699–1706, Aug. 2005.

[5] Y. Yamaguchi, A. Sato, W.-M. Boerner, R. Sato, and H. Yamada, "Four-component scattering power decomposition with rotation of coherency matrix," *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 6, pp. 2251–2258, Jun. 2011.

[6] J. S. Lee, M. R. Grunes, and R. Kwok, "Classification of multi-look polarimetric SAR imagery based on complex Wishart distribution," *Int. J. Remote Sens.*, vol. 15, no. 11, pp. 2299–2311, 1994, doi: 10.1080/01431169408954244.

[7] J.-S. Lee, M. R. Grunes, T. L. Ainsworth, L.-J. Du, D. L. Schuler, and S. R. Cloude, “Unsupervised classification using polarimetric decomposition and the complex Wishart classifier,” *IEEE Trans. Geosci. Remote Sens.*, vol. 37, no. 5, pp. 2249–2258, Sep. 1999.

[8] C. Lardeux *et al.*, “Support vector machine for multifrequency SAR polarimetric data classification,” *IEEE Trans. Geosci. Remote Sens.*, vol. 47, no. 12, pp. 4143–4152, Dec. 2009.

[9] X. Niu and Y. Ban, “Multi-temporal RADARSAT-2 polarimetric SAR data for urban land-cover classification using an object-based support vector machine and a rule-based approach,” *Int. J. Remote Sens.*, vol. 34, no. 1, pp. 1–26, 2013, doi: 10.1080/01431161.2012.700133.

[10] L. Loosvelt, J. Peters, H. Skriver, B. De Baets, and N. E. C. Verhoest, “Impact of reducing polarimetric SAR input on the uncertainty of crop classifications based on the random forests algorithm,” *IEEE Trans. Geosci. Remote Sens.*, vol. 50, no. 10, pp. 4185–4200, Oct. 2012.

[11] P. Du, A. Samat, B. Waske, S. Liu, and Z. Li, “Random forest and rotation forest for fully polarized SAR image classification using polarimetric and spatial features,” *ISPRS J. Photogramm. Remote Sens.*, vol. 105, pp. 38–53, 2015.

[12] Y. Zhou, H. Wang, F. Xu, and Y.-Q. Jin, “Polarimetric SAR image classification using deep convolutional neural networks,” *IEEE Geosci. Remote Sens. Lett.*, vol. 13, no. 12, pp. 1935–1939, Dec. 2016.

[13] Z. Zhang, H. Wang, F. Xu, and Y-Q. Jin, “Complex-valued convolutional neural network and its application in polarimetric SAR image classification,” *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 12, pp. 7177–7188, Dec. 2017.

[14] S. De, L. Bruzzone, A. Bhattacharya, F. Bovolo, and S. Chaudhuri, “A novel technique based on deep learning and a synthetic target database for classification of urban areas in PolSAR data,” *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 11, no. 1, pp. 154–170, Jan. 2018.

[15] S.-W. Chen and C.-S. Tao, “PolSAR image classification using polarimetric-feature-driven deep convolutional neural network,” *IEEE Geosci. Remote Sens. Lett.*, vol. 15, no. 4, pp. 627–631, Apr. 2018.

[16] H. Parikh, S. Patel, and V. Patel, “Classification of SAR and PolSAR images using deep learning: A review,” *Int. J. Image Data Fusion*, vol. 11, no. 1, pp. 1–32, 2020, doi: 10.1080/19479832.2019.1655489.

[17] N. Matsui, T. Isokawa, H. Kusamichi, F. Peper, and H. Nishimura, “Quaternion neural network with geometrical operators,” *J. Intell. Fuzzy Syst.*, vol. 15, no. 3, pp. 149–164, 2004.

[18] F. Shang and A. Hirose, “Quaternion neural-network-based PolSAR land classification in poincare-sphere-parameter space,” *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 9, pp. 5693–5703, Sep. 2014.

[19] H. Kim and A. Hirose, “Unsupervised fine land classification using quaternion autoencoder-based polarization feature extraction and self-organizing mapping,” *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 3, pp. 1839–1851, Mar. 2018.

[20] K. Kinugawa, F. Shang, N. Usami, and A. Hirose, “Isotropization of quaternion-neural-network-based PolSAR adaptive land classification in poincare-sphere parameter space,” *IEEE Geosci. Remote Sens. Lett.*, vol. 15, no. 8, pp. 1234–1238, Aug. 2018.

[21] C. J. Gaudet and A. S. Maida, “Deep quaternion networks,” in *Proc. Int. Joint Conf. Neural Netw.*, 2018, pp. 1–8.

[22] Q. Yin, J. Wang, X. Luo, J. Zhai, S. K. Jha, and Y.-Q. Shi, “Quaternion convolutional neural network for color image classification and forensics,” *IEEE Access*, vol. 7, pp. 20293–20301, 2019.

[23] T. Parcollet, M. Morchid, and G. Linarès, “Quaternion convolutional neural networks for heterogeneous image processing,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2019, pp. 8514–8518.

[24] T. Parcollet *et al.*, “Quaternion convolutional neural networks for end-to-end automatic speech recognition,” 2018, *arXiv:1806.07789*.

[25] X. Zhu, Y. Xu, H. Xu, and C. Chen, “Quaternion convolutional neural networks,” in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 631–647.

[26] X. Zhang, J. Xia, X. Tan, X. Zhou, and T. Wang, “PolSAR image classification via learned superpixels and QCNN integrating color features,” *Remote Sens.*, vol. 11, no. 15, 2019, Art. no. 1831.

[27] S. R. Cloude and E. Pottier, “A review of target decomposition theorems in radar polarimetry,” *IEEE Trans. Geosci. Remote Sens.*, vol. 34, no. 2, pp. 498–518, Mar. 1996.

[28] F. Gao, T. Huang, J. Wang, J. Sun, A. Hussain, and E. Yang, “Dual-branch deep convolution neural network for polarimetric SAR image classification,” *Appl. Sci.*, vol. 7, no. 5, 2017, Art. no. 447.

Yuya Matsumoto received the B.S. degree in electrical and electronic engineering from The University of Tokyo, Bunkyo-ku, Japan, in 2020, where he is currently working toward the master’s degree in electrical engineering and information systems with the Department of Electrical Engineering and Information Systems.

His research interests include polarimetric synthetic aperture radar and neural networks.

Ryo Natsuaki (Member, IEEE) received the B.S., M.S., and Ph.D. degrees in electrical engineering from The University of Tokyo, Bunkyo-ku, Japan, in 2009, 2011, and 2014, respectively.

From 2014 to 2017, he was an Aerospace Project Research Associate with Japan Aerospace Exploration Agency, Tsukuba, Japan. From 2018 to 2020, he was a Guest Scientist with Microwaves and Radar Institute, German Aerospace Center, Munich, Germany, under JSPS Overseas Research Fellowships. He is currently an Associate Professor with the Department of Electrical Engineering and Information Systems, The University of Tokyo. His research focuses on active remote sensing with synthetic aperture radar.

Dr. Natsuaki is a member of the IEEE Geoscience and Remote Sensing Society (GRSS) and the Institute of Electronics, Information and Communication Engineers (IEICE). He is an Associate Editor for the IEEE Transactions on Geoscience and Remote Sensing, a Technical Committee Member of the Frequency Allocations in Remote Sensing (FARS-TC) and the Electromagnetic Theory (EMT), and a Secretary of the IEICE technical committee on Space, Aeronautical and Navigational Electronics (SANE). He was a Publicity Chair of the IEEE International Geoscience and Remote Sensing Symposium (IGARSS) 2019 and serves for multiple international conferences.

Akira Hirose (Fellow, IEEE) received the Ph.D. degree in electronic engineering from The University of Tokyo, Bunkyo-ku, Japan, in 1991.

In 1987, he joined the Research Center for Advanced Science and Technology (RCAST), The University of Tokyo, as a Research Associate. In 1991, he was appointed as an Instructor with RCAST. From 1993 to 1995, on leave of absence from The University of Tokyo, he joined the Institute for Neuroinformatics, University of Bonn, Bonn, Germany. He is currently a Professor with the Department of Electrical Engineering and Information Systems, The University of Tokyo. He has authored or coauthored several books, such as Complex-Valued Neural Networks, 2nd Edition (2012), in the area of his research interests, including wireless electronics and neural networks.

Dr. Hirose was the President of the Asia-Pacific Neural Network Society (APNNS) (2016) and the Japanese Neural Network Society (JNNS) (2013–2015), a Vice President of the IEICE Electronics Society (ES) (2013–2015), an Editor-in-Chief for the *IEICE Transactions on Electronics* (2011–2012), an Associate Editor for journals, such as the IEEE Transactions on Neural Networks (2009–2011), *IEEE Geoscience and Remote Sensing Newsletter* (2009–2012), the IEEE GRSS All Japan Chapter Chair (2013–2015), the IEEE CIS All Japan Chapter Chair (2017–2018), and the General Chair of the Asia–Pacific Conference on Synthetic Aperture Radar (APSAR) 2013 Tsukuba, the International Conference on Neural Information Processing (ICONIP) 2016 Kyoto, and the International Geoscience and Remote Sensing Symposium (IGARSS) 2019 Yokohama. He has been a member of the IEEE Computational Intelligence Society (CIS) Neural Networks Technical Committee (NNTC) since 2009, the Founding Chair of the NNTC Complex-Valued Neural Network Task Force since 2010, the Governing Board Member of APNNA/APNNS since 2006, an Associate Editor for the IEEE Transactions on Neural Networks and Learning Systems since 2020, as well as the General Co-Chair of the IEEE World Congress on Computational Intelligence (WCCI) 2024 Yokohama. He is a Fellow of the IEICE and a member of the JNNS and the APNNS.