Hybrid Quantum-Classical Convolutional Neural Network Model for Image Classification

Fan Fan, Yilei Shi, *Member, IEEE*, Tobias Guggemos, and Xiao Xiang Zhu, *Fellow, IEEE*

***Abstract*— Image classification plays an important role in remote sensing. Earth observation (EO) has inevitably arrived in the big data era, but the high requirement on computation power has already become a bottleneck for analyzing large amounts of remote sensing data with sophisticated machine learning models. Exploiting quantum computing might contribute to a solution to tackle this challenge by leveraging quantum properties. This article introduces a hybrid quantum-classical convolutional neural network (QC-CNN) that applies quantum computing to effectively extract high-level critical features from EO data for classification purposes. Besides that, the adoption of the amplitude encoding technique reduces the required quantum bit resources. The complexity analysis indicates that the proposed model can accelerate the convolutional operation in comparison with its classical counterpart. The model's performance is evaluated with different EO benchmarks, including Overhead-MNIST, So2Sat LCZ42, PatternNet, RSI-CB256, and NaSC-TG2, through the TensorFlow Quantum platform, and it can achieve better performance than its classical counterpart and have higher generalizability, which verifies the validity of the QC-CNN model on EO data classification tasks.**

***Index Terms*— Image classification, quantum circuit, quantum machine learning (QML), remote sensing imagery.**

Nomenclature

qL	Qubits for the spatial information.
qC	Qubits for the color information.
qK	Qubits for the index of the kernels and feature maps.
qR	Qubits for the values of the feature maps.
$\lvert l_{x,y}\rangle$	Quantum state of the (x, y)-coordinate in the input; $\lvert l_x\rangle$ and $\lvert l_y\rangle$ are for the coordinates x and y, respectively, which can be written as $\lvert x_{n-1}, \ldots, x_0\rangle$ and $\lvert y_{n-1}, \ldots, y_0\rangle$.
$\lvert c_{x,y}\rangle$	Pixel value at the coordinate (x, y); $\lvert 0\rangle_C$ and $\lvert 1\rangle_C$ are the basis states of qC.
$\lvert r_{x,y}\rangle$	Weight of the kernel for the pixel at the coordinate (x, y); $\lvert 0\rangle_R$ and $\lvert 1\rangle_R$ are the basic states of qR.
$\lvert k\rangle$	Quantum state for the index of the kernels and feature maps.
$\lvert \Psi_i\rangle$	Quantum state of the encoded image.
$\lvert \Psi_{f_1}\rangle$	Result of the elementwise product in the quantum state.
$\lvert \Psi_{f_2}\rangle$	Generated feature map in the quantum state.
$\lvert \Psi_m\rangle$	Quantum state of the feature map for measurement.
$E(M)$	Expectation value with the measurement operator M.

Manuscript received 2 December 2022; revised 15 May 2023 and 2 August 2023; accepted 31 August 2023. Date of publication 18 September 2023; date of current version 3 December 2024. The work of Xiao Xiang Zhu was supported in part by the German Federal Ministry of Education and Research (BMBF) in the framework of the International Future AI Lab "AI4EO—Artificial Intelligence for Earth Observation: Reasoning, Uncertainties, Ethics and Beyond" under Grant 01DD20001; and in part by the Munich Center for Machine Learning (MCML). *(Corresponding author: Xiao Xiang Zhu.)*

Fan Fan is with the Chair of Data Science in Earth Observation (SiPEO), Technical University of Munich (TUM), 80333 München, Germany, and also with the Remote Sensing Technology Institute (IMF), German Aerospace Center (DLR), 82234 Weßling, Germany.

Yilei Shi is with the School of Engineering and Design, Technical University of Munich (TUM), 80333 München, Germany.

Tobias Guggemos is with the Remote Sensing Technology Institute (IMF), German Aerospace Center (DLR), 82234 Weßling, Germany.

Xiao Xiang Zhu is with the Chair of Data Science in Earth Observation (SiPEO), Technical University of Munich (TUM), 80333 München, Germany (e-mail: xiaoxiang.zhu.ieee@gmail.com).

Digital Object Identifier 10.1109/TNNLS.2023.3312170

I. Introduction

IN THE Earth observation (EO) domain, image classification is an active research field, contributing to deriving land use and land cover information from the remote sensing imagery [1]. However, due to the advances in remote sensing technologies, EO has irreversibly arrived in the big data era. Given the rapid growth of the data and the complexity of machine learning models for analysis, the required computation capacity has already been a primary barrier to the use of classical machine learning algorithms to automatically comprehend remote sensing images. Leveraging the power of quantum computing might overcome this challenge in the future since it is expected to efficiently solve problems that are prohibitively expensive for classical computers [2]. Investigating how to apply quantum computing in remote sensing is important and prospective.

Quantum machine learning (QML) is an interdisciplinary field that integrates machine learning and quantum computing. Numerous contributions from researchers attempting to exploit its potential for various tasks have been made to the field, such as data processing [3], [4], classification [5], [6], [7], segmentation [8], optimization [9], and quantum entanglement indication [10]. In the noisy intermediate-scale quantum (NISQ) era [11], applying QML for image classification is highly suitable. The reason is that quantum computing can speed up the complicated computation for image comprehension, and the classification output is simple but decisive, which is suitable for the probabilistic result from QML algorithms [12]. Several QML models have been proposed to classify images, using either quantum annealer [13] or parameterized quantum circuits (PQCs) [14]. However, whether quantum neural networks (QNNs) (a subfield of QML) can outperform their classical counterparts remains an open question. Moreover, only a few studies regarding applying QML to classify remote sensing images have been explored [15], [16]. Therefore, further investigations on QML for image classification are necessary and beneficial, especially in remote sensing.

However, current quantum machines are not fully fault-tolerant, and only a few qubits are supported, which restricts the applications of QML algorithms in practice. Most proposed quantum algorithms can only be verified via simulation on a small scale. Despite that, the significance and necessity of the research on QML algorithms should not be underestimated because it is fundamental for further practical applications when more advanced quantum devices are available.

Regarding remote sensing image classification, deep learning has been widely investigated [17], [18], [19]. Among them, the convolutional neural network (CNN) is essential for feature extraction due to its performance and adaptability. However, only a few proposals related to the quantum version of CNN have been made. We seek to investigate a quantum CNN model that can not only leverage quantum computing to extract critical features for classifying remote sensing data but also be qubit-efficient to meet the constraints of available qubits in quantum machines or simulators in the NISQ era.

Inspired by the study introduced in [20], a new hybrid quantum-classical CNN (QC-CNN) is developed, in which the quantum part is a parameterized circuit to extract essential features from images, and the classical part conducts the classification accordingly. In addition, our model exploits the amplitude encoding technique for image classification tasks, which requires relatively fewer qubits than using computation basis encoding.

To evaluate the effectiveness of our model, we used various EO benchmarks in our experiments with the TensorFlow Quantum (TFQ) platform [21], i.e., Overhead-MNIST [22], So2Sat LCZ42 [23], PatternNet [24], RSI-CB256 [25], and NaSC-TG2 [26]. The experimental results suggest that the QC-CNN model outperforms its classical counterpart and exhibits greater generalizability. Furthermore, in our experiments, we also studied our model's performance with different quantum gates, measurements, model structures, and noise effects to gain deeper insights into our model's properties and validity.

Regarding the efficiency, due to quantum parallelism, the proposed model can perform the elementwise product efficiently and speed up the feature extraction process by simultaneously transforming all desired quantum states.

The main contributions of this work are given as follows.

1) This work introduces a new hybrid QC-CNN for multicategory image classification, which can effectively extract critical features from images by using quantum circuits and achieve superior classification performance to classical CNN models.
2) This work presents a quantum convolution layer that can reduce the number of qubits and simplify the model's structure for classification by applying amplitude encoding.
3) This work investigates the impacts of quantum gates, measurement strategies, the structure of the model, and the noise effects on the classification performance.

This article is structured as follows. Section II introduces basic concepts of quantum computing, and Section III presents related work for image classification with quantum computing. Section IV details the structure of the QC-CNN model. The experimental evaluation of the model's performance is presented in Section V. Then, we analyze the scalability and efficiency of the model in Section VI. Finally, the conclusion and future work are discussed in Section VII.

II. Background

A quantum is the minimum discrete unit of any physical entity, and it has many special phenomena, such as superposition and entanglement, which can be utilized to perform computation tasks. This computation methodology refers to quantum computing.

Qubit is the elementary concept in quantum computing, an analogous concept of bits in classical computation. The specialty of a qubit lies in that the qubit state is a linear combination of the basis states (e.g., $|0\rangle$ and $|1\rangle$), which is called superposition, as shown in (1), where the coefficients α and β are complex numbers indicating the amplitudes of the quantum state, and the states $|0\rangle$ and $|1\rangle$ are computational basis states. When measuring a qubit in the computational basis, it will collapse either to the state $|0\rangle$ or $|1\rangle$ with the probability $|\alpha|^2$ and $|\beta|^2$, respectively,

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle; \quad |\alpha|^2 + |\beta|^2 = 1; \ \alpha, \beta \in \mathbb{C}. \tag{1}$$

Another quantum property is entanglement. The entangled qubits are correlated, and the state of one qubit affects the state of the other. The entangled quantum state cannot be separated into two states, which means that the entangled state $|\psi_{AB}\rangle$ cannot be written as a tensor product of its component state $|\psi_A\rangle$ and $|\psi_B\rangle$. Thus, $|\psi_{AB}\rangle$ is not equal to $|\psi_A\rangle \otimes |\psi_B\rangle$ (note that $|\psi_A\rangle \otimes |\psi_B\rangle$ can also be written as $|\psi_A\rangle|\psi_B\rangle$ or $|\psi_A\psi_B\rangle$).

To perform quantum computing, the quantum circuit model is one of the most popular models, which generally consists of three sequential parts.

A. Information Encoding

The input of any quantum algorithm is a quantum state. Classical data need to be encoded in such a state first. There are two basic encoding techniques: computation basis encoding and amplitude encoding. Specifically, the computation basis encoding maps the classical data into a basic quantum state of a quantum system, so it requires the classical data to be converted into the binary string form and uses the corresponding quantum basis state to represent it. As for the amplitude encoding, the classical data will be indicated by the amplitude of a quantum state. The classical data have to be normalized to guarantee that the sum of the squared amplitudes of the quantum state is equal to 1.

Based on these encoding techniques, various methods for encoding images to quantum states have been explored [27].

$$\begin{bmatrix}\cos(\frac{\theta}{2}) & -\sin(\frac{\theta}{2})\\ \sin(\frac{\theta}{2}) & \cos(\frac{\theta}{2})\end{bmatrix} \qquad \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} \qquad \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}$$

RY(θ) Gate **Hadamard Gate** **X Gate**

$$\begin{bmatrix}\cos(\frac{\theta}{2}) & -e^{i\lambda}\sin(\frac{\theta}{2})\\ e^{i\phi}\sin(\frac{\theta}{2}) & e^{i(\phi+\lambda)}\cos(\frac{\theta}{2})\end{bmatrix} \qquad \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & \multicolumn{2}{c}{\multirow{2}{*}{$U3$}}\\ 0 & 0 & & \end{bmatrix}$$

U3(θ,ϕ,λ) Gate **CU3(θ,ϕ,λ) Gate**

Fig. 1. Matrix representations for the involved elementary quantum gates.

B. Quantum State Transformation

Transforming quantum states is a critical step of quantum computing, playing a pivotal role in various computation tasks. After information encoding, a sequence of quantum gates, acting on one or more qubits, is utilized to manipulate the input state and convert it into another state through entanglement, rotation, and so on. The transformed quantum state is expected to be suitable for subsequent computation purposes.

The quantum gate is an analogy of logic gates in classical computers, but the difference is the gate for quantum computing must be unitary, which preserves the normalization and reversibility of the quantum system.

In this article, the following elementary quantum gates are involved: RY gate (rotation around the Y-axis), Hadamard gate (rotation around the $X + Z$-axis with π), X gate (rotation around the X-axis with π), U3 gate (rotation with three Euler angles), and CU3 gate (controlled version of the U3 gate). Their matrix representations are shown in Fig. 1.

C. Measurement

Quantum state measurement takes place at the end of the quantum circuit, which obtains the information from the quantum state to classical data. The obtained output, e.g., the expectation values, can be treated as the result of quantum computing. In this work, we use the expectation values as the extracted features from the input image for further classification processing.

Regarding QML, PQCs are commonly used as a hybrid approach, such as [28], [29], and [30]. Specifically, a PQC is composed of fixed quantum gates, but the parameters of these gates are trainable. They will be optimized during the training process in the classical machine, but the quantum state's transformation and measurement will be performed in the quantum machine.

III. Related Work

In recent years, there has been a growing interest in exploring how to incorporate quantum computing into image classification amidst the current limitations of quantum hardware. This section gives a brief overview of relevant studies, starting with those using either quantum machine learning or quantum deep learning for image classification. Then, recent contributions involving quantum computing in the EO domain are discussed.

A. Quantum Machine Learning

Quantum image classification has attracted great attention from researchers in the QML community, and various studies have been conducted.

Rebentrost et al. [31] and Havlíček et al. [32] proposed a quantum support vector machine (QSVM) for classification tasks. Ostaszewski et al. [33] introduced a quantum model using the principal component analysis (PCA) to classify images. Besides integrating SVM and PCA algorithms with quantum computing, Ruan et al. [34] focused on using the K-nearest neighbor (KNN) algorithm for image classification. They presented a method to compute the Hamming distance and utilized it in their proposed QKNN model to realize a good analog for the classical KNN algorithm. Dang et al. [35] employed the quantum minimum search algorithm in their quantum KNN model to speed up the similarity search processing without the negative influence on the classification accuracy.

B. Quantum Neural Networks

QNNs have also been investigated for image classification and recognition. CNN, which can automatically extract the high-level critical features from images by applying various filters in its sequential structure, plays an important role and shows promising performances. Researchers also attempted to implement the classical CNN with quantum computing.

Cong et al. [36] presented a QCNN containing successive convolution layers and pooling layers. Their proposed model is structured by combining multiscale entanglement renormalization ansatz and quantum error correction, and they have explicitly illustrated the potential of the proposed model in phase classification for quantum physical systems. Herrmann et al. [37] experimentally implemented the QCNN for recognizing topological quantum phases. Regarding employing this model to classify images, Chen et al. [38] proposed a fractal scaling down dimension reduction algorithm to reduce the image's features and then applied the QCNN model afterward for image classification. Lü et al. [39] used a quantum state preparation model to approximate the input image's quantum state and applied the QCNN model for the classification. The effectiveness of the introduced models was verified in their experiments.

Kerenidis et al. [40] proposed a quantum algorithm for applying deep CNNs, and it can achieve a similar accuracy on the MNIST dataset compared with the classical CNN. Wei et al. [41] demonstrated another basic framework of a quantum convolution neural network, which uses fewer parameters to achieve comparable classification performance as the classical CNN for digit recognition tasks. Hur et al. [42] presented fully parameterized QCNNs for classical data classification. When classifying images, they first apply classical algorithms, such as PCA and Autoencoder [43], to preprocess input images and extract preliminary global features. Afterward, their proposed model continuously extracts higher level features from these preliminary features by applying a sequence of convolutional circuits and pooling circuits for classification.

Henderson et al. [44] introduced a new type of transformation layer (quanvolution layers) using a random quantum circuit in their proposed quanvolutional neural networks to obtain meaningful local features for classification purposes. Their experiments show that the proposed QNN model can outperform purely classical CNN on the MNIST dataset in terms of accuracy and efficiency. However, the random quantum circuit results in an unrepeatable operation. More recently, Matic et al. [45] and Chen et al. [46] independently proposed novel hybrid CNNs, in which they both use PQCs instead of a random one as the convolutional kernel to get values of the feature maps for different classification tasks. Riaz et al. [47] used a strongly entangled circuit without any additional trainable parameters as a kernel to transform image features for classification.

Li et al. [20] proposed a quantum deep CNN based on a quantum parameterized circuit for image recognition. Their experiments on the MNIST and GTSRB datasets verified the model's validity. They use computation basis encoding to encode kernels' values and the image's color information. Thus, to accurately encode a pixel value from 0 to 255, eight qubits are needed. In addition, to achieve the convolutional operation, converting between amplitude encoding and computation basis encoding using quantum phase estimation [48] algorithms is necessary. Thus, their proposed model not only requires more qubits to encode images compared with utilizing the amplitude encoding technique but also needs extra quantum gates and computation costs.

However, note that most of the previous research focuses on binary classification tasks. To handle multicategories, Zeng et al. [49] presented a hybrid QNN that applies a ladder-like PQC to encode the input image and transform the features and a classical dense layer to handle multicategory classification tasks. However, their model extracts features without considering the spatial information of the input image. In addition, their model uses one qubit to encode only one feature. Thus, when classifying large images, the number of needed qubits in their model will increase significantly.

C. Applied QML and QNNs in the Remote Sensing Domain

As for applying QML/QNNs to classify remote sensing images, some researchers focused on using quantum annealers for classification [50], [51], [52]. Besides that, studies based on quantum circuits also have been conducted to analyze remote sensing data. Still, many of the related proposals exploit classical algorithms for feature extraction, and quantum circuits are basically applied for the final prediction.

For example, Gawron and Lewiński [53] presented a neural network based on a quantum circuit to classify multispectral images for land cover mapping, which uses the classical PCA algorithm to reduce the image's features. Zaidenberg et al. [15] introduced a hybrid model to classify the EO data, in which a CNN model is used to extract features from the images, and a quantum circuit is applied for the final classification task. Sebastianelli et al. [54] proposed a circuit-based hybrid QNN to analyze remote sensing images for land use and land cover classification. Their model uses a classical CNN derived from the LeNet-5 to extract high-level features from images and applies a quantum layer implemented with a quantum circuit for the final classification. Abdel-Khalek et al. [55] adopted an Inception ResNet to extract features from high-resolution imagery and a QNN for classification. The effectiveness of their model was proven in their experiments.

In addition, Otgonbaatar and Datcu [56] introduced a hybrid model to analyze polarimetric synthetic aperture radar images. Their model performs pixelwise classification, which extracts the features from the Stokes parameters of the pixel from the image using a quantum circuit for analysis.

IV. Methodology

In this article, a hybrid QC-CNN is proposed, aiming to classify images with the supervised deep learning method by exclusively exploiting the amplitude encoding technique. As shown in Fig. 2, the proposed QC-CNN model contains two sequential sections. The quantum section is expected to extract important features from images efficiently, and the classical section performs the final classification accordingly.

Our model consists of four types of layers sequentially: 1) encoding layer; 2) quantum convolution layer; 3) measurement layer; and 4) dense layer. The utilized qubits in the model can be categorized into four groups regarding their encoded information, represented by different colors in Fig. 2: qL qubits (white) for the input image's spatial information, qC qubits (gray) for the image's color information, qK qubits (green) for the kernels applied in the quantum convolution layer, and qR qubits (yellow) for the generated quantum feature maps. The functions of these layers will be explained in the following, and Nomenclature clarifies the definitions of the main notations involved in this section.

A. Encoding Layer

Encoding an image in a quantum state is a crucial step for our model. Various quantum image representation methods have been investigated and developed [27]. Note that the choice of the representation method will affect not only the number of needed qubits and quantum gates in the encoding layer but also the structure of the further quantum layers. The proposed model adopts the Flexible Representation of Quantum Images (FRQI) [57] to encode gray-scale images into quantum states, which can represent images using a small number of qubits, as it uses amplitude encoding to encode the color information of all pixels as follows.

Every pixel of an image with $2^n \times 2^n$ pixels is represented by its (x, y)-coordinate. By using a quantum state $|l_{x,y}\rangle$ for the respective coordinate, one requires 2n qubits (qLs) in superposition, and this is achieved by applying Hadamard gates. The respective quantum state is

$$\frac{1}{2^n}\sum_{x=0}^{2^n-1}\sum_{y=0}^{2^n-1}|l_{x,y}\rangle \tag{2}$$

where $|l_{x,y}\rangle$ indicates the (x, y)-coordinate of a pixel as a binary string form as $|x_{n-1},\ldots,x_0 y_{n-1},\ldots,y_0\rangle$.

For the color information of a gray-scale image, we use a single qubit (qC), of which the amplitude indicates the pixel values. In particular, for each pixel of the image, one RY (θ)-gate is used for the qC's amplitude transformation, where the gray value $[0, 255]$ of pixel (x, y) is mapped to $\theta_{x,y} = [0, \pi/2]$, such that the qubit state is

$$|c_{x,y}\rangle = \cos\theta_{x,y}|0\rangle_C + \sin\theta_{x,y}|1\rangle_C. \tag{3}$$

Entanglement allows binding the pixel's location with its color information stored in the two registers $|l_{x,y}\rangle$ and $|c_{x,y}\rangle$. This is achieved by Multicontrolled-RY(θ)-gates where the control-bits encode the pixel's (x, y)-coordinates, and the final quantum state for image representation is

$$|\Psi_i\rangle = \frac{1}{2^n}\sum_{x=0}^{2^n-1}\sum_{y=0}^{2^n-1}|l_{x,y}\rangle|c_{x,y}\rangle \tag{4}$$

which encodes the image with $2^n \times 2^n$ pixel values using 2n + 1 qubits. In contrast, with computation basis encoding, we need 2n + 8 qubits to accurately encode this image.

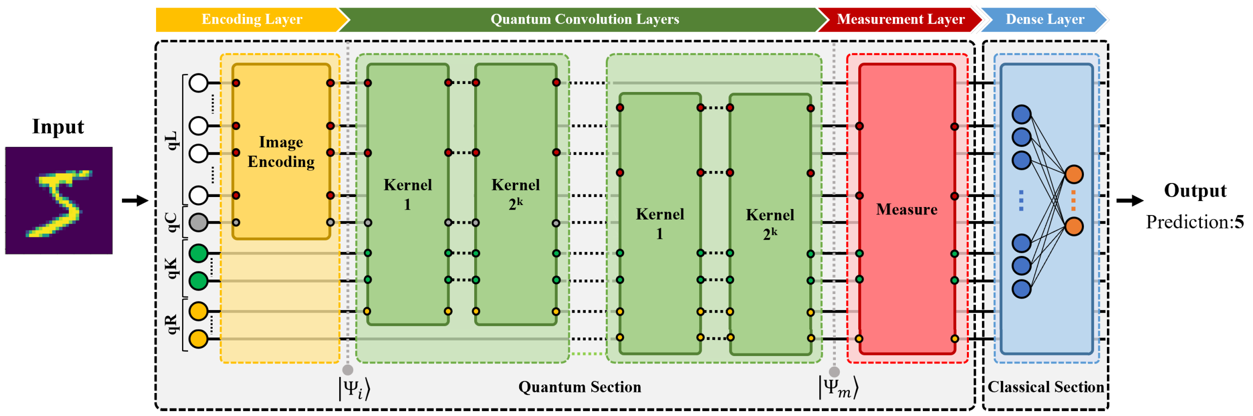


Fig. 2. Quantum circuit for QC-CNN model: 1) white for qL qubits (spatial information encoding), gray for qC qubit (color information encoding), green for qK qubits (kernel index encoding), and yellow for qR qubits (feature map information encoding); 2) dot markers in the circuit highlight the involved qubits in the applied quantum gates or the measured qubits in the specific layers; 3) the model contains m convolution layers and each layer involves 2^k kernels; and 4) $|\Psi_i\rangle$ and $|\Psi_m\rangle$ are the outputs from the encoding layer and quantum convolution layers, respectively.

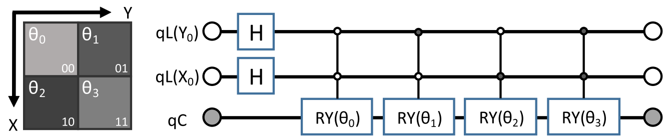


Fig. 3. Circuit example of the encoding layer for one 2 × 2 image: 1) dot markers indicate the controlled state: white dots for $|0\rangle$ and black dots for $|1\rangle$; 2) H represents the Hadamard gate and RY(θ) represents the RY gate with θ degrees; and 3) θ is the converted pixel value, and the subscripts here distinguish pixels in the image.

Fig. 3 illustrates one example circuit of the implementation of FRQI on a 2 × 2 image. The dot markers in the circuit represent the controlled state: white dots for $|0\rangle$ and black dots for $|1\rangle$. In this way, the encoding layer translates a classical image into a quantum state.

B. Quantum Convolution Layer

The quantum convolution layer in the QC-CNN model aims to achieve the convolutional operation, which plays a critical role in the proposed model to extract features and generate corresponding feature maps. In this layer, the kernel size and the convolutional stride are set to the same to perform the fast dimension reduction for the generated feature maps, and they should be modified according to the input image's size as classical CNN models. In this article, the QC-CNN model with 2 × 2-sized kernels and the convolutional stride of 2 is exampled for clarification.

Given one kernel to perform the convolutional operation for feature extraction, the kernel will be located at different places on the input image. At each location, a two-step process will happen: 1) perform the elementwise product between the patch of the input image and kernel and 2) sum the output of the products for each patch to obtain the value of the feature map.

To realize this transformation in the quantum circuit, we introduce a qR qubit and apply a series of controlled rotation gates to it. The respective quantum state after the elementwise product can be generally represented as (5), where $|r_{x,y}\rangle$ indicates the weight of the kernel for the corresponding pixel and $|\Psi_{f_1}\rangle$ represents the output of the elementwise product process

$$|\Psi_{f_1}\rangle = \frac{1}{2^n}\sum_{x=0}^{2^n-1}\sum_{y=0}^{2^n-1}|l_{x,y}\rangle|c_{x,y}\rangle|r_{x,y}\rangle. \qquad (5)$$

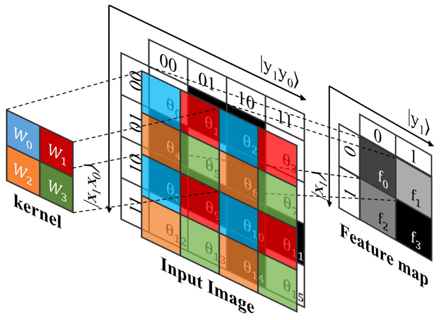


Fig. 4. Illustration of the convolutional computation with a 2 × 2 kernel procedure: 1) encoding a 4 × 4-sized input image's spatial information needs four qLs, $|x_1x_0\rangle$ and $|y_1y_0\rangle$ for the vertical and horizontal dimensions, respectively; 2) color indicates weight variables (i.e., W_0–W_3) of the kernel; 3) the quantum states $|x_1x_0y_1y_0\rangle$ with the specific $|x_0\rangle$ and $|y_0\rangle$ compute with the equivalent weight (the same color) for the elementwise product; and 4) the spatial information of the quantum feature map can be represented by $|x_1y_1\rangle$, and the values of the feature map are represented by $f_0 - f_3$.

Specifically, given the example illustrated in Fig. 4, when the convolutional stride is two, and the kernel size is 2 × 2, to apply this kernel (different colors indicate different weights) on the given image, the pixels at the specific locations should be transformed with the same weight. In Fig. 4, the location of the pixels marked with the same color should apply the same weight. Thus, to conduct this elementwise product over the input in the quantum convolution layer, we need to identify the pixels with the shared weight in the quantum domain.

As stated in [20], when setting the kernel size as 2 × 2 and the convolutional stride as two, the first qL in $|l_x\rangle$ and $|l_y\rangle$ can be employed to specify pixels for different weights. Since $|l_x\rangle$ and $|l_y\rangle$ can be written as $|x_{n-1},\ldots,x_0\rangle$ and $|y_{n-1},\ldots,y_0\rangle$, respectively, the states with the same $|x_0\rangle$ and $|y_0\rangle$ states should apply the equivalent weight. For the case shown in Fig. 4, this 4 × 4 image needs four qLs to encode its spatial information. The states $|x_1x_0y_1y_0\rangle$ with the specific $|x_0\rangle$ and $|y_0\rangle$ should have the equivalent weight (illustrated with the same color). Thus, to specify the location of the pixels with the shared weight for the convolutional computation between the image and a kernel, only $|x_0\rangle$ and $|y_0\rangle$ should be focused. As a result, to perform the convolutional operation over the entire input, only four gates are required due to the four possible states of $|x_0y_0\rangle$, regardless of the input's spatial size. In contrast, classical convolutional operations over the input rely on the sliding window technique, leading to a quadratic increase in computation with the spatial size of the input. A detailed discussion can be found in Section VI-A.

To realize the elementwise product for the convolution operation in the quantum circuit, besides considering the pixel's location, the pixel's value encoded in the qC ($|c\rangle$) is also important. For this reason, in our model, we apply the U3 gates with three controllers (i.e., $|x_0\rangle$, $|y_0\rangle$, and $|c\rangle$) on the qR qubit, which can rotate the qR with three Euler angles based on the pixel's value and its location. The rotated qR can be described by (6), in which $|0\rangle_R$ and $|1\rangle_R$ are basic quantum states for qR, and $k^0_{x,y}$ and $k^1_{x,y}$ are the values of one weight variable of the applied kernel. To identify a 2×2 kernel, four weight variables should be identified

$$|r_{x,y}\rangle = \cos k^0_{x,y} |0\rangle_R + e^{ik^1_{x,y}} \sin k^0_{x,y} |1\rangle_R. \tag{6}$$

After the elementwise product, we obtain the quantum state $|\Psi_{f_1}\rangle$. To obtain the feature map, we sum the resulting values of the elementwise product for each patch, as illustrated in Fig. 4. The generated feature map $|\Psi_{f_2}\rangle$ can be indicated as

$$|\Psi_{f_2}\rangle = \frac{2}{2^n} \sum_{x'=0}^{\frac{2^n}{2}-1} \sum_{y'=0}^{\frac{2^n}{2}-1} |l_{x',y'}\rangle |f_{x',y'}\rangle \tag{7}$$

in which $|l_{x',y'}\rangle$ indicates the spatial information of the feature map and $|f_{x',y'}\rangle$ encodes the value of the feature map, which can be represented as

$$|f_{x',y'}\rangle = \frac{1}{2} \sum_{x_0=0}^{1} \sum_{y_0=0}^{1} |x_0\rangle |y_0\rangle |c_{x'x_0,y'y_0}\rangle |r_{x_0,y_0}\rangle. \tag{8}$$

Furthermore, (8) can be written as follows based on (3) and (6):

$$\begin{aligned} |f_{x',y'}\rangle = \alpha\ & |x_0\rangle|y_0\rangle|0\rangle_C|0\rangle_R \\ + \beta\ & |x_0\rangle|y_0\rangle|1\rangle_C|0\rangle_R \\ + \gamma\ & |x_0\rangle|y_0\rangle|1\rangle_C|1\rangle_R \end{aligned} \tag{9}$$

in which α, β, and γ are the amplitudes of obtained quantum states. Note that, when the state of qC is $|0\rangle_C$, the applied U3 gate will have no effect on qR. Hence, the amplitude of the quantum state $|x_0\rangle|y_0\rangle|0\rangle_C|1\rangle_R$ equals 0, and it is omitted in (9). The unnormalized amplitude of $|1\rangle_R$ in (9) can be computed following (10), where $k^0_{x_0,y_0}$ and $k^1_{x_0,y_0}$ indicate the values of the weight variables for the kernel as introduced before, and $\theta_{x'x_0,y'y_0}$ refers to the value of the corresponding pixel

$$\sum_{\substack{x_0=0\\y_0=0}}^{1} \left(\sin \theta_{x'x_0,y'y_0}\right) \times \left(\sin k^0_{x_0,y_0}\right) \times \left(e^{ik^1_{x_0,y_0}}\right). \tag{10}$$

Hence, the convolutional operation is performed in the quantum circuit, and the generated feature map ($|\Psi_{f_2}\rangle$) is successfully encoded using $|l_{x',y'}\rangle$ and $|f_{x',y'}\rangle$ with the preserved entanglement. This feature map can be treated as the input of the next quantum convolution layer. After several convolution layers, the feature map ($|\Psi_m\rangle$) with the required dimension is generated for further classification.

To apply multiple kernels in one quantum convolution layer, qK qubits will be used as additional controllers for the applied U3 gates, and d qKs can maximally prepare 2^d kernels for each layer. Before working as the controllers of the U3 gates, these d qKs first are initialized as $|0\rangle^{\otimes d}$, and then, a Hadamard gate is applied on each qK. Thus, we prepare a quantum state $|k\rangle$ that can be written as $|+\rangle^{\otimes d}$ to indicate the index of the applied kernels and also the generated feature maps.

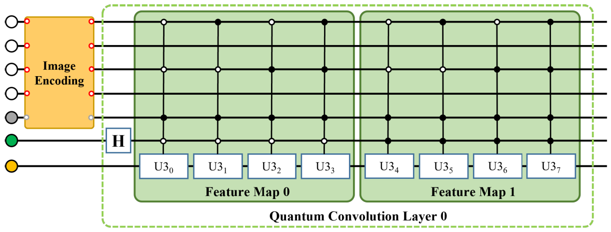


Fig. 5. Circuit example of the quantum convolution layer: 1) white for qLs, gray for qC, green for qK, and yellow for qR; 2) dot markers in the circuit indicate the controlled state: white dots for $|0\rangle$ and black dots for $|1\rangle$; and 3) H represents the Hadamard gate, and U3 represents the U3 gate.

Fig. 5 demonstrates one example of the quantum circuit for one quantum convolution layer with two kernels to process a 4×4-sized input image. In the end, two generated feature maps with the size of 2×2 are expected accordingly. The degrees in the U3 gates in this circuit example indicate the values of the weight variables for the kernels, which will be optimized during the training process.

C. Measurement Layer

The measurement layer in the model is used to obtain the feature maps from the quantum states to the classical states. Besides that, this layer will also flatten the obtained feature maps into a 1-D feature vector for the dense layer. For this purpose, the interested quantum state and the operator for the measurement need to be specified. The expectation values are taken with respect to the measurement operator M based on the quantum states $|\Psi_m\rangle$ indicating the generated quantum feature maps following

$$E(M) = \langle \Psi_m | M | \Psi_m \rangle. \tag{11}$$

The obtained expectation value, $E(M)$, will be treated as one classical feature value for classification.

Since, in this layer, the quantum state embedding the feature maps' information is of most interest, only the corresponding qLs and qR are desired. To obtain all the generated quantum feature maps, the qKs should be also taken into account.

For example, given the state $|\Psi_m\rangle$ for k quantum feature maps with the size of $m \times m$, it can be represented as

$$|\Psi_m\rangle = \frac{1}{m\sqrt{k}} \sum_{x_m=0}^{m-1} \sum_{y_m=0}^{m-1} \sum_{k=0}^{k-1} |l_{x_m,y_m}\rangle |k\rangle |f_{x_m,y_m}\rangle \tag{12}$$

where $|l_{x_m,y_m}\rangle$ indicates the location information of the feature map; $|f_{x_m,y_m}\rangle$ embeds the values of the feature map; and $|k\rangle$ indicates the index of the feature maps. To simplify, the target quantum state $|\Psi_m\rangle$ can also be rewritten as

$$|\Psi_m\rangle = \frac{1}{m\sqrt{k}} \sum_{i=1}^{k\times m^2} |i\rangle |f_i\rangle \tag{13}$$

in which $|i\rangle$ represents the index of the extracted feature and $|f_i\rangle$ embeds the value of the ith extracted feature.

Fig. 6 represents an example of the measurement layer, in which two generated feature maps with the size of 2×2 are expected. Thus, eight features will be obtained.

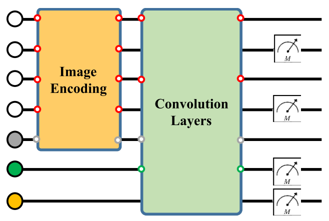


Fig. 6. Circuit example of the measurement layer: 1) white for qLs, gray for qC, green for qK, and yellow for qR; 2) one convolution layer with two kernels is applied on a 4 × 4 input image; and 3) two feature maps are generated: $|x_1 y_1\rangle$ encodes the location information, $|r\rangle$ encodes the values of the feature map, and $|k\rangle$ shows the index of the created feature maps.

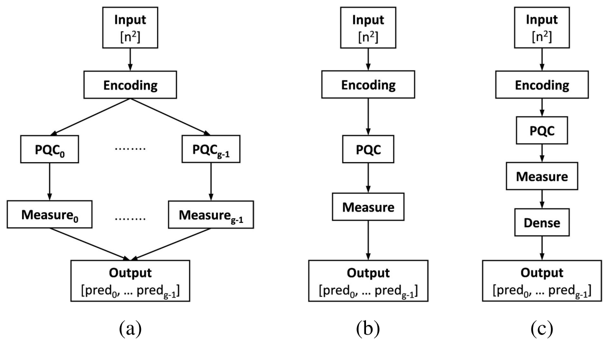


Fig. 7. Strategies for QML to deal with g-category classification tasks, including (a) using multiple binary classifiers, (b) using multiple qubits in the last layer to indicate the prediction, and (c) using a classical dense layer in the end for prediction: 1) *input*: classical gray-scale input image; 2) *encoding*: encoding layer; 3) *PQC*: quantum convolution layer with trainable parameters; 4) *measure*: measurement layer; 5) *dense*: classical dense layer with softmax activation function; and 6) *output*: ID vector for prediction.

Given the desired quantum state $|\Psi_m\rangle$ in a Hilbert space, we consider three types of measurement operators formed by Pauli-X, Pauli-Y, and Pauli-Z operators, and we measure the state $|\Psi_m\rangle$ in the X-basis, Y-basis, and Z-basis, respectively. Specifically, we define a set of operators with respect to one orthonormal basis for measurement, and each operator M is Hermitian. Regarding one specific basis formed by states $|b\rangle$, we conduct the measurement of the quantum state $|\Psi_m\rangle$ in the computational basis. For each interested state $|b_i\rangle$, the operator M_i can be defined by $|b_i\rangle\langle b_i|$. The expectation value $E(M_i)$ according to (11) is the ith obtained feature.

In this way, the feature maps' information will be obtained from the quantum states to the classical data, and it will be utilized for the final classification.

D. Dense Layer

With the classical output from the measurement layer, a classical dense layer is utilized for the final classification. The neurons in this layer are deeply connected, and each neuron encodes one expectation value. In this layer, suitable activation functions (e.g., softmax for multicategory classification and sigmoid for binary classification) will be applied in the end to achieve the nonlinear transformation and output a probability distribution to indicate the classified category.

E. Multicategory Classification

To deal with multicategory classification tasks with QML (e.g., classifying the image into g categories), there are generally two common strategies. One is to train g binary classifiers, and the structure is shown in Fig. 7(a). However, this method will inevitably increase the trained parameters and training time significantly. The other method illustrated in Fig. 7(b) employs g qubits in the model's last layer to indicate the image's category, which requires relatively fewer trainable parameters but increases the needed qubits for classification.

To improve the training efficiency and reduce the qubits, we propose to use a classical dense layer with a suitable activation function (e.g., softmax for multicategory classification) for the final prediction [see Fig. 7(c)]. Thus, the proposed model is a hybrid model containing a quantum section for the feature extraction and a classical section for the final classification output.

F. Training Process

The proposed model is based on PQCs. The rotation angles of the quantum gates in the quantum convolution layers are regarded as the trainable parameters, which will be optimized in the training process by classical algorithms (e.g., the Adam algorithm [58]), but the quantum state's evolution and measurement are conducted on a quantum computer. More precisely, the training process is composed of the following steps.

1) The training set of the classical images after the normalization process can be identified as $I = \{I_0, \dots, I_n\}$, and each element indicates one training classical image. The categorical label y_i for an arbitrary image I_i will be transformed to $[y_i^0, y_i^1, \dots y_i^{g-1}]$ for a g-category classification problem using one-hot encoding technique (note that the superscripts here represent indices instead of exponents).
2) For one input image, the model outputs a probability distribution, given as $f(I_i, \Theta) = \tilde{y}_i$, where $\tilde{y}_i$ is a vector in $\Re^g$ and Θ denotes the trainable parameters in the model.
3) The cross-entropy loss function is used to compare the output against the label $(\mathcal{L}(\tilde{y}_i, y_i))$, and the cost value will be averaged over each batch from the training dataset.
4) The trainable parameters Θ will be optimized and updated during the backpropagation of gradients by applying the Adam algorithm [58]. As for gradient calculation for the trainable parameters in the quantum circuit, there are several techniques, for instance, the adjoint method [59]. This training process will be repeated until the parameters are optimized.

V. Experiments

To evaluate the performance of our model on EO data classification tasks, we conducted experiments using multiple EO benchmarks and compared our model with different deep learning models.

Data Preparation: In this study, we experimented with five different EO datasets, i.e., Overhead-MNIST [22], So2Sat LCZ42 [23], PatternNet [24], RSI-CB256 [25], and NaSC-TG2 [26]. Due to the limited computation power of current quantum simulators, we reduced the size of the datasets in our experiments by only focusing on a subset of categories for each benchmark. Furthermore, we downscaled the labeled images in all the datasets to a size of 8 × 8 × 1 with different techniques: Lanczos algorithm [60] and convolutional autoencoder [61]. Note that the objective of our experiments was to

TABLE I
EXPERIMENTAL DATA FOR PERFORMANCE EVALUATION

Data	**Categories**	**Patch Size**	**Dimension Reduction**	**Training**	**Validation**	**Test**
Overhead-MNIST	Car, Ship, Plane, Harbor, Helicopter, Oil Gas Field	$28 \times 28 \times 1$	Lanczos	4298	800	637
So2Sat LCZ42	Compact Middle-rise, Large Low-rise, Dense Trees	$32 \times 32 \times 1$	Lanczos	3498	750	750
PatternNet	Coastal mansion, Parking lot, Swimming pool	$256 \times 256 \times 3$	AutoEncoder	1680	360	360
RSI-CB256	Dry farm, Mangrove, Residents, Snow mountain, Storage room	$600 \times 600 \times 3$	AutoEncoder	2800	600	600
NaSC-TG2	Forest, Residential, Snowberg	$128 \times 128 \times 3$	AutoEncoder	2100	450	450

assess our model's effectiveness. To avoid overpowering the autoencoders, we added Gaussian noise as the perturbation to the output of the autoencoders and evaluated our model's performance with the noisy input. The details about the data preparation process for our experiments are provided in the following, and the summarized information can be found in Table I.

Overhead-MNIST [22] contains overhead view images (28×28) of ten kinds of entities (e.g., "car," "ship," and "plane"). There are 8519 training images and 1065 testing images, which are all gray-scaled. In this study, we used all the 5098 images labeled in 6 categories ("car," "ship," "plane," "harbor," "helicopter," and "oil gas field") for training and 637 images from these categories in its test dataset for evaluation. From the training images, we randomly selected 800 samples (approximately 15%) to create a validation dataset, and the remaining samples were used for training. To downsize these labeled images, we used the Lanczos algorithm.

So2Sat LCZ42 [23] consists of 17 local climate zone (LCZ) labels of around half a million Sentinel-1 and Sentinel-2 image patches with the size of 32×32 in 42 cities. For this dataset, we used the intensity of the refined LEE-filtered VV channel from the Sentinel-1 data as the input. Then, we adopted the Lanczos algorithm to downsize the input patches. Regarding the categories, we focused on three LCZ labels ("compact middle-rise," "large low-rise," and "dense trees") and randomly selected around 5000 labeled patches in the cities of Berlin and Munich from these categories to build a balanced dataset. The sampled images were split into three sets: 70% for training, 15% for validation, and 15% for testing.

PatternNet [24] has high-resolution imagery covering 38 different classes, and there are around 800 samples of size 256×256 pixels in each class. For this dataset, we utilized a convolutional autoencoder to reduce the features of the input images to the size of 8×8. In the experiment, we focused on three classes ("coastal mansion," "parking lot," and "swimming pool") and used all the provided samples. For evaluation, we randomly separated these samples with a ratio of 70:15:15 as the training, validation, and test datasets.

RSI-CB256 [25] is a global-scale dataset, having more than 24 000 images in 35 categories. In our experiments, we set "dry farm," "mangrove," "residents," "snow mountain," and "storage room" as our target categories and randomly selected 800 RGB-image samples with a size of 256×256 for each category. The labeled images were downsized to 8×8 using a convolutional autoencoder and were also divided into three sets: 70% for training, 15% for validation, and 15% for testing, with no overlap in between.

NaSC-TG2 [26] is an EO benchmark dataset for natural scene classification, which has around 20 000 samples with the size of 128×128 in ten classes. For our experiments, we concentrated on three target classes ("forest," "residential," and "snowberg") and randomly selected 1000 samples with RGB channels for each class to build a balanced dataset. Then, we utilized a convolutional autoencoder to reduce features to 8×8 for our experiments. In the end, we randomly separated the prepared samples into the training, validation, and test datasets with the ratio 70%:15%:15%.

Model Preparation: In the experiments, we evaluated our model with two quantum convolution layers, each with two kernels for feature extraction. The main focus of our experiments is comparing our model and the classical CNN model since it serves as the classical counterpart of our approach.

Specifically, we selected two CNN models as the competitors. The first CNN model with six kernels (CNN-6) has a similar model structure as our model, and each convolution layer applies two filters. However, compared with this CNN model, our model will extract more features in the measurement layer for classification when measuring the quantum feature maps on multiple bases even though only two kernels are applied in each quantum convolution layer (24 features for the QC-CNN model versus eight features for the CNN-6). Thus, the other CNN model with 14 kernels (CNN-14), which extracts the same number of features for final classification and has a similar number of trainable parameters as our model, was also considered for performance comparison.

In addition, we also included another three CNN-based models and two quantum models for evaluation.

With respect to the selected CNN-based models, Block CNN [62], ResNet [63], and DenseNet [64] have significantly deeper structures and more parameters than our model. To ensure a fair comparison and account for the small input image size of 8×8, we simplified these models' structures and applied fewer layers and kernels for image classification. Specifically, the Block CNN model comprises two blocks, a global average pooling layer and a fully connected layer. Each block contains three convolutional layers with the same kernel size, and every layer has two filters in the first block and four filters in the second block. The ResNet model used in this study consists of two residual blocks, followed by a global average pooling layer and a fully connected layer that produces the final output. The convolutional layers in the residual blocks were configured with two and four filters, respectively, and the kernel size in the model was set to 2×2. The DenseNet architecture starts with one convolution layer and then follows two dense blocks separated by one transition layer. Each dense block includes two convolution layers, with two filters being utilized in each layer.

As for the quantum models, QCNN [54] and QNN [44], they are also hybrid, but quantum computing plays different roles in classification in these models. We evaluated them with the same input with the size of 8×8. Table II summarizes the pipelines of these models in the experiments. Different

TABLE II
PIPELINES OF THE QC-CNN MODEL AND OTHER COMPARED QUANTUM MODELS

Model	Input for Quantum Circuit	# Qubit	# Gate	Pipeline
QC-CNN	image (8×8)	10	6 H gates + 64 controlled RY gates for information encoding; 16 controlled U3 gates + 1 H gate for feature extraction;	1) quantum circuit to extract 24 features from input images; 2) classical dense layer to perform the final classification;
QCNN[54]	8 features from input	4	4 H gates + 8 RY gates + 6 CNOT gates	1) classical CNN with two convolution layers and one dense layer to extract 8 features from the input; 2) quantum circuit to transform these features and output 16 feature values; 3) classical dense layer to perform the final classification;
QNN[44]	16 patches (2×2)	4	4 RY gates to encode each patch; a random circuit to transform features;	1) quantum circuit to generate 4 feature maps with the size of 4×4 by applying the sliding window method over all the patches; 2) classical CNN with one convolution layer and a dense layer for higher-level feature extraction and final classification;

Note: the classical input data for classification is images with a size of 8×8

from our model, the quantum algorithms in these models encode and process either a small number of high-level features or local patches with low-level features. Specifically, the QCNN [54] model uses a classical deep learning model to extract high-level features first, and the quantum algorithm encodes and transforms these features, followed by a classical dense layer for classification. QNN [44] uses a quantum circuit to extract local features from every patch of the image with the sliding window method. Then, a classical CNN model is used to perform the final classification. The different ways of using quantum computing in these models result in different requirements for quantum resources.

Quantum Simulation Settings: In our experiment, we used the TFQ platform [21] to develop and train our model. As one of the widely used frameworks for quantum deep learning, it enables the usage of several types of simulators. Interested readers may refer to the work [65] for a detailed comparison among different frameworks.

Regarding the quantum machine used in our study, we adopted a noiseless simulator from TFQ for training, which outputs the analytic results after quantum computing. The reason for selecting this simulator is that the goal of our experiments is to verify the validity of our quantum algorithm, and this noiseless simulator written in C++ is faster than other simulators on the platform. As for performing backpropagation, we applied the adjoint differentiator provided by TFQ. It is compatible with the analytic output and the adopted simulator. In addition, it computes the gradients faster than others. However, note that this differentiation technique cannot currently be easily realized on a real quantum machine.

Experimental Settings: To train the aforementioned models, unless otherwise stated, we set the epoch number as 500 and the batch size as 100, and the cross-entropy loss function was used. We trained our model using the Adam optimizer [58] with a learning rate of 0.03 and the competitors with their default learning rates. Each training was repeated three times, and we calculated the average classification accuracy of the trained models with the lowest validation loss value during the training process. The resulting values, along with their corresponding standard deviations, were used for comparison and discussion purposes.

Eventually, we carried out four different experiments to evaluate the model's performance:

1) analysis of general classification performance;
2) analysis of quantum gates and measurement operators;
3) analysis of the structure of the QC-CNN model;
4) analysis of the noise effects on the QC-CNN model's performance.

Specifically, the first experiment was designed to assess the classification performance of the QC-CNN model on different datasets, in which all the prepared datasets were used. The subsequent experiments aimed to investigate the properties of the quantum component in our model. To ensure that the powerful machine learning-based approaches for feature reduction, such as autoencoders, will not diminish the impact of the studied quantum properties, we only used the Overhead-MNIST and So2Sat LCZ42 datasets for these three experiments.

A. Analysis of General Classification Performance

We present the classification accuracy achieved by different models in Table III. The table demonstrates that our model outperforms both CNN models (CNN-6 and CNN-14) in terms of test accuracy for all five EO datasets. Furthermore, despite having a simpler structure and fewer trained parameters, our model achieves comparable classification performance to the competitor with the highest test accuracy among the compared CNN-based deep learning models and quantum models for every used dataset (with a performance difference of at most 0.013). In some cases (e.g., Overhead-MNIST), our model demonstrates superior performance among all competitors.

Moreover, as shown in Table III, the difference in classification accuracy between training, validation, and testing sets for our model is relatively small compared to other models, suggesting that our model has less overfitting and higher generalizability than others.

In addition, a one-sided Wilcoxon signed-rank test [66] was performed between our model and the competitors over five EO datasets. The null hypothesis states that the classification performance (the averaged test accuracy) of our model is equal to or worse than that of the competitor over five EO datasets, and the alternative hypothesis suggests that the average test accuracy of our model over five EO datasets is greater than the competitor. As shown in Table IV, the p-values comparing our model with two CNN models are below 0.05. This indicates the rejection of the null hypothesis, i.e., in favor

TABLE III
CLASSIFICATION PERFORMANCE COMPARISON BETWEEN THE PROPOSED MODEL AND OTHERS

Dataset	**Model**	**#Parameter**	**Training Accuracy**	**Validation Accuracy**	**Test Accuracy**
Overhead-MNIST	CNN-6	82	0.552 ± 0.020	0.535 ± 0.031	0.522 ± 0.020
	CNN-14	214	0.660 ± 0.030	0.641 ± 0.031	0.620 ± 0.012
	QCNN[54]	366	0.710 ± 0.007	0.680 ± 0.006	0.650 ± 0.033
	QNN[44]	290	0.692 ± 0.009	0.670 ± 0.011	0.639 ± 0.018
	Block CNN[62]	248	0.681 ± 0.030	0.645 ± 0.037	0.645 ± 0.020
	ResNet[63]	230	0.679 ± 0.018	0.660 ± 0.011	0.659 ± 0.010
	DenseNet[64]	262	0.698 ± 0.017	0.669 ± 0.003	0.663 ± 0.023
	QC-CNN	198	0.697 ± 0.004	0.682 ± 0.008	0.672 ± 0.004
So2Sat LCZ42	CNN-6	55	0.697 ± 0.003	0.683 ± 0.009	0.689 ± 0.009
	CNN-14	139	0.712 ± 0.010	0.702 ± 0.011	0.694 ± 0.002
	QCNN[54]	315	0.719 ± 0.005	0.693 ± 0.005	0.701 ± 0.012
	QNN[44]	179	0.716 ± 0.006	0.690 ± 0.008	0.689 ± 0.014
	Block CNN[62]	233	0.721 ± 0.008	0.711 ± 0.000	0.700 ± 0.003
	ResNet[63]	215	0.721 ± 0.008	0.701 ± 0.013	0.704 ± 0.008
	DenseNet[64]	199	0.740 ± 0.006	0.729 ± 0.014	0.705 ± 0.006
	QC-CNN	123	0.718 ± 0.000	0.716 ± 0.000	0.704 ± 0.001
PatternNet	CNN-6	55	0.785 ± 0.044	0.769 ± 0.028	0.785 ± 0.056
	CNN-14	139	0.799 ± 0.035	0.807 ± 0.022	0.797 ± 0.036
	QCNN[54]	151	0.822 ± 0.029	0.793 ± 0.042	0.815 ± 0.016
	QNN[44]	177	0.896 ± 0.019	0.880 ± 0.026	0.874 ± 0.015
	Block CNN[62]	233	0.848 ± 0.017	0.823 ± 0.002	0.825 ± 0.013
	ResNet[63]	215	0.836 ± 0.023	0.811 ± 0.023	0.827 ± 0.004
	DenseNet[64]	199	0.818 ± 0.022	0.807 ± 0.006	0.780 ± 0.026
	QC-CNN	123	0.857 ± 0.020	0.860 ± 0.016	0.861 ± 0.022
RSI-CB	CNN-6	73	0.837 ± 0.023	0.839 ± 0.029	0.816 ± 0.003
	CNN-14	189	0.847 ± 0.027	0.841 ± 0.025	0.835 ± 0.007
	QCNN[54]	185	0.864 ± 0.027	0.841 ± 0.037	0.843 ± 0.021
	QNN[44]	191	0.896 ± 0.009	0.882 ± 0.013	0.858 ± 0.003
	Block CNN[62]	243	0.876 ± 0.011	0.845 ± 0.011	0.851 ± 0.011
	ResNet[63]	225	0.857 ± 0.020	0.852 ± 0.019	0.834 ± 0.018
	DenseNet[64]	241	0.885 ± 0.025	0.867 ± 0.020	0.841 ± 0.016
	QC-CNN	173	0.854 ± 0.011	0.857 ± 0.029	0.849 ± 0.014
NaSC-TG2	CNN-6	55	0.904 ± 0.007	0.905 ± 0.003	0.896 ± 0.007
	CNN-14	139	0.913 ± 0.021	0.910 ± 0.024	0.908 ± 0.012
	QCNN[54]	233	0.928 ± 0.005	0.923 ± 0.005	0.932 ± 0.008
	QNN[44]	179	0.935 ± 0.007	0.922 ± 0.005	0.922 ± 0.012
	Block CNN[62]	233	0.928 ± 0.007	0.916 ± 0.015	0.913 ± 0.014
	ResNet[63]	215	0.924 ± 0.011	0.919 ± 0.019	0.920 ± 0.010
	DenseNet[64]	199	0.940 ± 0.009	0.925 ± 0.010	0.924 ± 0.009
	QC-CNN	123	0.910 ± 0.004	0.919 ± 0.003	0.920 ± 0.002

TABLE IV
SIGNIFICANCE ANALYSIS: P-VALUE BETWEEN OUR MODEL AND OTHERS

	CNN-6	CNN-14	QCNN[54]	QNN[44]	Block CNN[62]	ResNet[63]	DenseNet[64]
QC-CNN	0.031	0.031	0.156	0.406	0.063	0.054	0.156

of the alternative hypothesis, which suggests that our model outperforms its classical counterparts. As for other competitors with more complex structures and parameters, there is insufficient evidence to support the alternative hypothesis, and its implication aligns with the previous finding based on Table III. It is important to note that, in computer vision, it is not common to use the p-value to examine the significance of the improvement, as a tiny percentage of improvement in, e.g., image classification, would lead to unprecedented practical usage. This is also the situation for EO. Thus, although we carried out this experiment for the sake of completeness, the readers are recommended to make the assessment based on the actual application scenario.

To conclude, in comparison with the CNN model (our model's classical counterpart), our model could extract critical features with fewer kernels from the input image, and it can achieve better classification performance and higher generalizability. As for the classical deep learning model with a more complex structure (e.g., ResNet), our model can have a similar performance. However, there is no guarantee that it can always outperform these classical models. In addition, it is worth mentioning that the quantum simulator used in the experiments was noiseless, so the experiments were conducted in an ideal condition. The model's performance will be compromised when adopting a noisy device or simulator.

B. Analysis of Quantum Gates and Measurement Operators

There are various types of quantum gates and measurements that can be applied in our model, as introduced in Section IV, for example:

TABLE V
QC-CNN's Accuracy With Different Quantum Gates in the Convolution Layers and Operators in the Measurement Layer

Dataset	Classes	Rotation Gate			Measurement			Test Accuracy
		U3	RX	RY	X	Y	Z	
Overhead -MNIST	Car or Plane	✓			✓			**0.955** ± 0.004
			✓		✓			0.945 ± 0.005
				✓	✓			0.945 ± 0.003
		✓				✓		**0.873** ± 0.005
			✓			✓		0.821 ± 0.004
				✓		✓		0.830 ± 0.008
		✓					✓	0.864 ± 0.014
			✓				✓	**0.866** ± 0.009
				✓			✓	0.860 ± 0.016
		✓			✓	✓	✓	0.957 ± 0.003
			✓		✓	✓	✓	0.957 ± 0.007
				✓	✓	✓	✓	**0.963** ± 0.003
So2Sat LCZ42	Compact Middle-Rise or Dense Trees	✓			✓			**0.879** ± 0.001
			✓		✓			0.877 ± 0.001
				✓	✓			**0.879** ± 0.001
		✓				✓		**0.845** ± 0.005
			✓			✓		0.842 ± 0.000
				✓		✓		0.841 ± 0.002
		✓					✓	0.848 ± 0.000
			✓				✓	0.844 ± 0.002
				✓			✓	**0.849** ± 0.001
		✓			✓	✓	✓	0.870 ± 0.000
			✓		✓	✓	✓	**0.871** ± 0.006
				✓	✓	✓	✓	**0.871** ± 0.003

1) For the rotation gate in the quantum convolution layer, U3, RX, and RY are for U3 gate, RX gate, and RY gate, respectively; 2) For the measurement, X, Y, Z indicates X-measurement, Y-measurement, Z-measurement in the measurement layer, respectively; 3) the symbol (✓) represents the applied gate or the measurement in the experiment; 4) the highest average accuracy is in bold

1) rotation gates in the quantum convolution layer (e.g., U3, RX, or RY);
2) operators in the measurement layer (e.g., Pauli-X, Pauli-Y, or Pauli-Z).

To evaluate the effects of different rotation gates and measurements on classification performance, several experiments were conducted. Specifically, we used all the samples from two categories ("Car" and "Plane" in the Overhead-MNIST; "Compact Middle-rise" and "Dense Trees" in the So2Sat LCZ42) in Table I to prepare two balanced datasets. Different types of quantum gates in the quantum convolution layers and the operators in the measurement layer have been tested, and the results are demonstrated in Table V.

1) Rotation Gates in Quantum Convolution Layer: The U3 gate makes a single qubit rotate with three Euler angles, whereas the RX and RY gate let the qubit rotate around the X-axis and the Y-axis, respectively. Thus, using U3 gates can achieve more complex rotation. As shown in Table V, adopting U3 gates can generally achieve higher accuracy for image classification, but note that the model using U3 gates has a threefold increase in the number of the trainable parameters compared with the model using RX or RY gates.

In addition, using RX or RY gates in the model can sometimes also achieve comparable performance as using U3 gates. For example, when adopting the X-measurement for the Overhead-MNIST data and the Z-measurement for the So2Sat LCZ42 data, the performance difference regarding classification accuracy when applying different gates in the quantum convolution layer is limited.

2) Operators in the Measurement Layer: In our experiments, we evaluated four measurement strategies, i.e., X, Y, Z-measurement and XYZ-measurement, to obtain the features from the quantum state by applying Pauli-X, Pauli-Y, and Pauli-Z operators. Specifically, for the first three strategies, we measure the quantum state in the X-basis, Y-basis, and Z-basis, respectively. As for the XYZ-measurement, we concatenate the values obtained based on the previous three strategies and use them together in the further classical dense layer.

As can be seen in Table V, the experimental results indicate that the model applying the XYZ-measurement can generally outperform others, but this measurement also extracts more features for the successive dense layer compared with others.

Similarly, the model with the measurement on one basis can have comparable performance to the one using the XYZ-measurement in some cases. For instance, to classify the Overhead-MNIST dataset and the So2Sat LCZ42 dataset, the X-measurement can reach a similar performance to the XYZ-measurement despite the adopted rotation gates.

C. Analysis of the Structure of the QC-CNN Model

To evaluate the impacts of the model's structure on the classification performance, we experimented with our model having a different number of kernels and quantum convolution layers using two datasets. For the Overhead-MNIST, we used all the samples from the categories "Car," "Ship," and "Plane" to build a balanced dataset. Regarding the So2Sat LCZ42, we used all the data introduced in Table I.

TABLE VI

Classification Performance Comparison of Our Model With Different Structures

Dataset	#Kernel	#Convolution Layer	Training Accuracy	Validation Accuracy	Test Accuracy
Overhead -MNIST	2	2	0.865 ± 0.001	0.849 ± 0.005	0.841 ± 0.000
	4	2	0.889 ± 0.003	0.868 ± 0.006	0.851 ± 0.007
	2	1	0.900 ± 0.008	0.880 ± 0.010	0.878 ± 0.015
	4	1	0.917 ± 0.002	0.888 ± 0.017	0.886 ± 0.008
So2Sat LCZ42	2	2	0.718 ± 0.000	0.716 ± 0.000	0.704 ± 0.001
	4	2	0.723 ± 0.002	0.707 ± 0.004	0.700 ± 0.001
	2	1	0.733 ± 0.002	0.712 ± 0.005	0.710 ± 0.002
	4	1	0.735 ± 0.011	0.712 ± 0.010	0.709 ± 0.005

TABLE VII

Classification Performance Comparison With the Noise

Dataset	Model	No Noise	Noise in Data	Noise in Model		
				Error Rate 0.01	Error Rate 0.05	Error Rate 0.10
Overhead -MNIST	QC-CNN	0.798 ± 0.008	0.741 ± 0.006	0.793 ± 0.022	0.762 ± 0.021	0.761 ± 0.005
	CNN-14	0.726 ± 0.048	0.710 ± 0.018	\	\	\
So2Sat LCZ42	QC-CNN	0.696 ± 0.000	0.668 ± 0.011	0.696 ± 0.011	0.680 ± 0.005	0.658 ± 0.007
	CNN-14	0.677 ± 0.011	0.636 ± 0.022	\	\	\

According to the results shown in Table VI, the model's structure can influence its classification performance like the classical CNN. When applying a suitable number of kernels and convolution layers for feature extraction, our model's classification accuracy can be improved.

D. Analysis of the Noise Effects on the QC-CNN Model's Performance

To evaluate our model's ability to handle the noise, we tested our model's performance given different types of noise and compared it with its classical counterpart.

For the noise in the data, to avoid the influence of the procedure for dimension reduction, we added the Gaussian noise to the downscaled input images. As for the noise in the model, we involved the noise at the end of the circuit for measured qubits. Specifically, the noisy model will additionally add one gate from Pauli-X, Pauli-Y, and Pauli-Z gates with a certain error rate on each measured qubit. In our experiments, we considered the error rate of 0.01, 0.05, and 0.10.

Considering the time needed for simulating the noisy quantum model, we simplified the tasks. In this experiment, we focused on three category classification tasks. For the Overhead-MNIST, we defined "Car," "Ship," and "Plane" as the target categories. As for the So2Sat LCZ42, we used all the categories in Table I. To train the models, we randomly selected 600 images from the training samples in these target categories for each dataset. To evaluate and compare the models' performance, we utilized all the validation and test data from the target categories listed in Table I. The models were trained with an epoch number and a batch size of 50.

The experimental results can be found in Table VII. As shown in the table, our model can have better performance compared with the CNN model when dealing with the noise in the data. Regarding the noisy model, the misclassification rate increases with the error rate, which is expected. However, the usage of the trainable classical dense layer increases the resilience of our model against the noise effects.

VI. Discussion

The scalability and efficiency of the model also play an important role in the model's estimation. The efficiency indicates the speed of the quantum algorithm for classification. As for the scalability analysis, the number of qubits needed to scale the model is discussed.

A. Network Efficiency Analysis

We analyzed our network's efficiency from two perspectives: the number of required quantum gates and the number of trainable parameters for classification. As an example for our analysis, we chose the QC-CNN model that has m sequential quantum convolution layers, and each layer applies 2^k kernels with a size of 2×2 and a convolutional stride 2. The efficiency comparison between our model and its classical counterpart for feature extraction can be found in Table VIII.

1) Number of Quantum Gates: The number of quantum gates used in a quantum algorithm is directly related to the number of operations required for computation because a quantum circuit is built up with gates, and each gate represents a specific operation. Our model's gate complexity is discussed individually for each type of layer in the model.

a) Encoding layer: To encode a gray-scale image of size $2^n \times 2^n$ using FRQI, $2n$ H gates on qLs are used to prepare the spatial information. For each pixel, a controlled RY gate with $2n$ controllers is required that can rotate a qubit with arbitrary degrees around the Y-axis based on $2n$ qubits' states. Thus, in total, this layer requires 2^{2n} RY gates controlled by $2n$ qubits and $2n$ H gates for input image encoding.

To reduce the number of gates applied in this layer, several techniques, such as those described in [67] and [68], can be applied. However, it is important to note that this study does not aim to address the issue of encoding images more efficiently, as it falls outside the scope of this work.

b) Quantum convolution layer: The convolutional computation also relies on the rotation gate with multiple controllers. In our model, we apply the U3 gate controlled by $k+3$ qubits in these layers regardless of the size of the input

TABLE VIII
FEATURE EXTRACTION EFFICIENCY COMPARISON BETWEEN QC-CNN AND CNN

Feature Extraction	#Operation	#Parameter
CNN	$\mathcal{O}(2^{k+2n})$	5×2^k
QC-CNN	$\mathcal{O}(2^{k+2})$	RX or RY Gate: 4×2^k U3 Gate: 12×2^k

TABLE IX
DEMANDED QUBITS IN THE QC-CNN MODEL FOR GRAY-SCALE IMAGES

Layer	Target		Qubit Type	#Qubits
Encoding Layer	position	$2^n \times 2^n$	qL	$2n$
	band	1	qC	1
Convolution Layer	kernel	2^k	qK	k
	layers	m	qR	m

image, and these controllers are 2 qLs, k qKs, and 1 qR (or the qC for the first convolution layer). In the end, there are $4m \times 2^k$ U3 gates with $k + 3$ controllers. To prepare 2^k kernels, k H gates for qK are also required.

c) Measurement layer: To obtain the expectation values in the measurement layer, we have to run the quantum circuit multiple times. After the quantum convolution layers, given the generated feature maps with $2^{2(n-m)+k}$ features, we need $\mathcal{O}(2^{2n-2m+k+1})$ runs of the circuit to obtain the required feature values from the quantum state.

When comparing the number of operations between our model and classical CNN models, we would like to focus on the convolution layers that are the key components responsible for feature extraction in both models. However, it is worth mentioning that, if we consider the encoding layer and the measurement layer, the overall efficiency of our model will be compromised. Still, as introduced before, there are possibilities to speed up the encoding process, and it is beyond the scope of this work. Thus, we mainly discuss and compare convolutional computation in different models.

As studied in [69], a classical convolution layer with the same settings as our QC-CNN model requires $\mathcal{O}(2^{k+2n})$ operations to process a gray-scale image of size $2^n \times 2^n$. In contrast, our quantum convolutional layer requires only 2^{k+2} U3 gates with $k + 3$ controllers, regardless of the image's spatial size. This suggests that our quantum convolution layer can speed up the convolutional operation for feature extraction, particularly when analyzing large remote sensing images.

2) Number of Trainable Parameters: Table VIII compares the number of trainable parameters of a convolution layer with 2^k kernels of size 2×2 to process gray-scale images in our QC-CNN model and a CNN model.

As shown in the table, the number of trainable parameters in the quantum convolution layer depends on the utilized quantum gate along with the number of applied kernels. Applying controlled U3 gates can manage more complex rotations, but it also has a threefold increase in the number of trainable parameters compared with using RX or RY gates.

Compared to classical CNN models, the use of U3 gates requires more trainable parameters for a given number of kernels, as shown in Table VIII. However, the experimental results in Table III indicate that our model with only four kernels can outperform the CNN-14 model with 14 kernels, suggesting that the kernels in our model are more efficient than those in classical models. As a result, our model still requires fewer parameters than CNN-14 despite the adoption of U3 gates.

Furthermore, note that incorporating RX or RY gates in our model can sometimes lead to comparable performance as U3 gates, as evidenced by the experimental results presented in Table V. As such, it is possible to further reduce the number of trainable parameters in our model without compromising classification performance. This implies that our model has the potential to extract valuable features with significantly fewer parameters, highlighting its advantage in training efficiency.

B. Scalability Analysis

The requirement of qubit resources for QML models is one of the essential criteria for quantum computing, especially in the NISQ era. Thus, it is important to analyze the qubits needed in the proposed model for the classification task.

Table IX concludes the number of qubits for each layer of the proposed model to classify gray-scale images. Concerning color images, three qubits are needed to encode the spectral information with the MCQI method [70]. Thus, the number of qubits for the model containing 2^k kernels and m successive quantum convolution layers is $2n+k+m+1$ for the gray-scale images and $2n + k + m + 3$ for the color images. As shown in Table IX, for a gray-scale image with the size of $N \times N$, our model only requires $2\log(N) + 1$ qubits. To prepare K kernels for the quantum convolution layers, $\log(K)$ qubits are sufficient. Thus, the proposed model achieves advantages in terms of information encoding.

VII. CONCLUSION AND FUTURE WORK

In this article, a new hybrid QC-CNN is proposed to classify remote sensing images into multicategories, which can accelerate feature extraction in the quantum domain for classification and achieve better performance than its classical CNN counterpart. Exclusively applying amplitude encoding in our model significantly reduces the requirement on quantum bit resources. In addition, we investigated the impacts of quantum gates, measurements, the model structure, and the noise effects on our model's classification performance. More importantly, our experimental results demonstrate a proof of concept for applying CNN in the quantum domain for image classification. Furthermore, evaluating our approach using simulators on EO benchmarks has provided us with the opportunity to explore the potential of using QNNs for EO data comprehension within the current limitations of quantum machines.

Due to the QC-CNN model's acceleration in the computation procedure and its relatively low requirement on the number of qubits, the proposed model might provide a possibility to tackle the challenges in the remote sensing domain for image classification tasks when more advanced quantum machines are available in the future.

Regardless, future research could continue to explore the following directions: 1) investigate more suitable image encoding techniques for remote sensing images; 2) further study the properties of different gates and measurements for classification; and 3) explore the potential of quantum computing for different challenges in the EO domain, such as incomplete data [71] and noisy-labeled data [72].

References

[1] M. Li, S. Zang, B. Zhang, S. Li, and C. Wu, "A review of remote sensing image classification techniques: The role of spatio-contextual information," *Eur. J. Remote Sens.*, vol. 47, no. 1, pp. 389–411, Jan. 2014.

[2] D. Ristè et al., "Demonstration of quantum advantage in machine learning," *npj Quantum Inf.*, vol. 3, no. 1, p. 16, Apr. 2017.

[3] V. Gandhi, G. Prasad, D. Coyle, L. Behera, and T. M. McGinnity, "Quantum neural network-based EEG filtering for a brain–computer interface," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 2, pp. 278–288, Feb. 2014.

[4] A. Anand, M. Lyu, P. S. Baweja, and V. Patil, "Quantum image processing," 2022, *arXiv:2203.01831*.

[5] Z. Abohashima, M. Elhosen, E. H. Houssein, and W. M. Mohamed, "Classification with quantum machine learning: A survey," 2020, *arXiv:2006.12270*.

[6] A. Pérez-Salinas, A. Cervera-Lierta, E. Gil-Fuster, and J. I. Latorre, "Data re-uploading for a universal quantum classifier," *Quantum*, vol. 4, p. 226, Feb. 2020.

[7] N. H. Nguyen, E. C. Behrman, M. A. Moustafa, and J. E. Steck, "Benchmarking neural networks for quantum computations," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 7, pp. 2522–2531, Jul. 2020.

[8] D. Konar, S. Bhattacharyya, B. K. Panigrahi, and E. C. Behrman, "Qutrit-inspired fully self-supervised shallow quantum learning network for brain tumor segmentation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 11, pp. 6331–6345, Nov. 2022.

[9] M. Wilson, R. Stromswold, F. Wudarski, S. Hadfield, N. M. Tubman, and E. G. Rieffel, "Optimizing quantum heuristics with meta-learning," *Quantum Mach. Intell.*, vol. 3, no. 1, pp. 1–14, Jun. 2021.

[10] E. C. Behrman, R. E. F. Bonde, J. E. Steck, and J. F. Behrman, "On the correction of anomalous phase oscillation in entanglement witnesses using quantum neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 9, pp. 1696–1703, Sep. 2014.

[11] J. Preskill, "Quantum computing in the NISQ era and beyond," *Quantum*, vol. 2, p. 79, Aug. 2018.

[12] Y. Ruan, X. Xue, and Y. Shen, "Quantum image processing: Opportunities and challenges," *Math. Problems Eng.*, vol. 2021, pp. 1–8, Jan. 2021.

[13] N. T. T. Nguyen and G. T. Kenyon, "Image classification using quantum inference on the D-wave 2X," in *Proc. IEEE Int. Conf. Rebooting Comput. (ICRC)*, Nov. 2018, pp. 1–7.

[14] A. Mari, T. R. Bromley, J. Izaac, M. Schuld, and N. Killoran, "Transfer learning in hybrid classical-quantum neural networks," *Quantum*, vol. 4, p. 340, Oct. 2020.

[15] D. A. Zaidenberg, A. Sebastianelli, D. Spiller, B. Le Saux, and S. L. Ullo, "Advantages and bottlenecks of quantum machine learning for remote sensing," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2021, pp. 5680–5683.

[16] M. Khoshboresh-Masouleh and R. Shah-Hosseini, "Quantum deep learning in remote sensing: Achievements and challenges," *Proc. SPIE*, vol. 11844, pp. 42–45, Jul. 2021.

[17] X. X. Zhu et al., "Deep learning in remote sensing: A comprehensive review and list of resources," *IEEE Geosci. Remote Sens. Mag.*, vol. 5, no. 4, pp. 8–36, Dec. 2017.

[18] Y. Shi, Q. Li, and X. X. Zhu, "Building footprint generation using improved generative adversarial networks," *IEEE Geosci. Remote Sens. Lett.*, vol. 16, no. 4, pp. 603–607, Apr. 2019.

[19] Y. Shi, Q. Li, and X. X. Zhu, "Building segmentation through a gated graph convolutional neural network with deep structured feature embedding," *ISPRS J. Photogramm. Remote Sens.*, vol. 159, pp. 184–197, Jan. 2020.

[20] Y. Li, R.-G. Zhou, R. Xu, J. Luo, and W. Hu, "A quantum deep convolutional neural network for image recognition," *Quantum Sci. Technol.*, vol. 5, no. 4, Jul. 2020, Art. no. 044003.

[21] M. Broughton et al., "TensorFlow quantum: A software framework for quantum machine learning," 2020, *arXiv:2003.02989*.

[22] D. Noever and S. E. M. Noever, "Overhead MNIST: A benchmark satellite dataset," 2021, *arXiv:2102.04266*.

[23] X. Xiang Zhu et al., "So2Sat LCZ42: A benchmark dataset for global local climate zones classification," 2019, *arXiv:1912.12171*.

[24] W. Zhou, S. Newsam, C. Li, and Z. Shao, "PatternNet: A benchmark dataset for performance evaluation of remote sensing image retrieval," *ISPRS J. Photogramm. Remote Sens.*, vol. 145, pp. 197–209, Nov. 2018.

[25] H. Li et al., "RSI-CB: A large-scale remote sensing image classification benchmark using crowdsourced data," *Sensors*, vol. 20, no. 6, p. 1594, Mar. 2020.

[26] Z. Zhou et al., "NaSC-TG2: Natural scene classification with Tiangong-2 remotely sensed imagery," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 14, pp. 3228–3242, 2021.

[27] J. Su, X. Guo, C. Liu, and L. Li, "A new trend of quantum image representations," *IEEE Access*, vol. 8, pp. 214520–214537, 2020.

[28] E. Ovalle-Magallanes, J. G. Avina-Cervantes, I. Cruz-Aceves, and J. Ruiz-Pinales, "Hybrid classical–quantum convolutional neural network for stenosis detection in X-ray coronary angiography," *Expert Syst. Appl.*, vol. 189, Mar. 2022, Art. no. 116112.

[29] R. Huang, X. Tan, and Q. Xu, "Learning to learn variational quantum algorithm," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Feb. 28, 2022, doi: 10.1109/TNNLS.2022.3151127.

[30] J. Zheng, Q. Gao, J. Lü, M. Ogorzałek, Y. Pan, and Y. Lü, "Design of a quantum convolutional neural network on quantum circuits," *J. Franklin Inst.*, 2022, doi: 10.1016/j.jfranklin.2022.07.033.

[31] P. Rebentrost, M. Mohseni, and S. Lloyd, "Quantum support vector machine for big data classification," *Phys. Rev. Lett.*, vol. 113, no. 13, Sep. 2014, Art. no. 130503.

[32] V. Havlíček et al., "Supervised learning with quantum-enhanced feature spaces," *Nature*, vol. 567, no. 7747, pp. 209–212, Mar. 2019.

[33] M. Ostaszewski, P. Sadowski, and P. Gawron, "Quantum image classification using principal component analysis," 2015, *arXiv:1504.00580*.

[34] Y. Ruan, X. Xue, H. Liu, J. Tan, and X. Li, "Quantum algorithm for K-nearest neighbors classification based on the metric of Hamming distance," *Int. J. Theor. Phys.*, vol. 56, no. 11, pp. 3496–3507, Nov. 2017.

[35] Y. Dang, N. Jiang, H. Hu, Z. Ji, and W. Zhang, "Image classification based on quantum K-nearest-neighbor algorithm," *Quantum Inf. Process.*, vol. 17, no. 9, pp. 1–18, Sep. 2018.

[36] I. Cong, S. Choi, and M. D. Lukin, "Quantum convolutional neural networks," *Nature Phys.*, vol. 15, no. 12, pp. 1273–1278, 2019.

[37] J. Herrmann et al., "Realizing quantum convolutional neural networks on a superconducting quantum processor to recognize quantum phases," *Nature Commun.*, vol. 13, no. 1, p. 4144, Jul. 2022.

[38] G. Chen, R. Zhou, X. Zhu, Q. Chen, Y. Chen, and Z. Yuan, "Quantum convolutional neural network on scale chaology," in *Proc. 13th Int. Congr. Image Signal Process., Biomed. Eng. Informat. (CISP-BMEI)*, Oct. 2020, pp. 243–247.

[39] Y. Lü, Q. Gao, J. Lü, M. Ogorzałek, and J. Zheng, "A quantum convolutional neural network for image classification," in *Proc. 40th Chin. Control Conf. (CCC)*, Jul. 2021, pp. 6329–6334.

[40] I. Kerenidis, J. Landman, and A. Prakash, "Quantum algorithms for deep convolutional neural networks," 2019, *arXiv:1911.01117*.

[41] S. Wei, Y. Chen, Z. Zhou, and G. Long, "A quantum convolutional neural network on NISQ devices," *AAPPS Bull.*, vol. 32, no. 1, pp. 1–11, Dec. 2022.

[42] T. Hur, L. Kim, and D. K. Park, "Quantum convolutional neural network for classical data classification," *Quantum Mach. Intell.*, vol. 4, no. 1, p. 3, 2022.

[43] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org

[44] M. Henderson, S. Shakya, S. Pradhan, and T. Cook, "Quanvolutional neural networks: Powering image recognition with quantum circuits," *Quantum Mach. Intell.*, vol. 2, no. 1, p. 2, Jun. 2020.

[45] A. Matic, M. Monnet, J. M. Lorenz, B. Schachtner, and T. Messerer, "Quantum-classical convolutional neural networks in radiological image classification," in *Proc. IEEE Int. Conf. Quantum Comput. Eng. (QCE)*, Sep. 2022, pp. 56–66.

[46] S. Y.-C. Chen, T.-C. Wei, C. Zhang, H. Yu, and S. Yoo, "Quantum convolutional neural networks for high energy physics data analysis," *Phys. Rev. Res.*, vol. 4, no. 1, Mar. 2022, Art. no. 013231.

[47] F. Riaz, S. Abdulla, H. Suzuki, S. Ganguly, R. C. Deo, and S. Hopkins, "Accurate image multi-class classification neural network model with quantum entanglement approach," *Sensors*, vol. 23, no. 5, p. 2753, Mar. 2023.

[48] M. Nielsen and I. Chuang, *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge, U.K.: Cambridge Univ. Press, 2010, doi: 10.1017/CBO9780511976667.

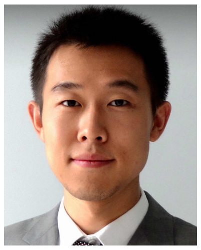
[49] Y. Zeng, H. Wang, J. He, Q. Huang, and S. Chang, "A multi-classification hybrid quantum neural network using an all-qubit multi-observable measurement strategy," *Entropy*, vol. 24, no. 3, p. 394, Mar. 2022.

[50] E. Boyda, S. Basu, S. Ganguly, A. Michaelis, S. Mukhopadhyay, and R. R. Nemani, "Deploying a quantum annealing processor to detect tree cover in aerial imagery of California," *PLoS ONE*, vol. 12, no. 2, Feb. 2017, Art. no. e0172505.

[51] G. Cavallaro, D. Willsch, M. Willsch, K. Michielsen, and M. Riedel, “Approaching remote sensing image classification with ensembles of support vector machines on the D-wave quantum annealer,” in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Sep. 2020, pp. 1973–1976.
[52] S. Otgonbaatar and M. Datcu, “A quantum annealer for subset feature selection and the classification of hyperspectral images,” *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 14, pp. 7057–7065, 2021.
[53] P. Gawron and S. Lewiński, “Multi-spectral image classification with quantum neural network,” in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Sep. 2020, pp. 3513–3516.
[54] A. Sebastianelli, D. A. Zaidenberg, D. Spiller, B. Le Saux, and S. Ullo, “On circuit-based hybrid quantum neural networks for remote sensing imagery classification,” *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 15, pp. 565–580, 2022.
[55] S. Abdel-Khalek et al., “Quantum neural network-based multilabel image classification in high-resolution unmanned aerial vehicle imagery,” *Soft Comput.*, vol. 27, pp. 13027–13038, 2023, doi: 10.1007/s00500-021-06460-3.
[56] S. Otgonbaatar and M. Datcu, “Natural embedding of the Stokes parameters of polarimetric synthetic aperture radar images in a gate-based quantum computer,” *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2022, Art. no. 4704008.
[57] P. Q. Le, F. Dong, and K. Hirota, “A flexible representation of quantum images for polynomial preparation, image compression, and processing operations,” *Quantum Inf. Process.*, vol. 10, no. 1, pp. 63–84, Feb. 2011.
[58] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2014, *arXiv:1412.6980*.
[59] X.-Z. Luo, J.-G. Liu, P. Zhang, and L. Wang, “Yao.Jl: Extensible, efficient framework for quantum algorithm design,” *Quantum*, vol. 4, p. 341, Oct. 2020.
[60] C. E. Duchon, “Lanczos filtering in one and two dimensions,” *J. Appl. Meteorol. Climatol.*, vol. 18, no. 8, pp. 1016–1022, Aug. 1979.
[61] Y. Zhang. (2018). *A Better Autoencoder for Image: Convolutional Autoencoder*. Accessed: Sep. 11, 2023. [Online]. Available: http://users.cecs.anu.edu.au/~Tom.Gedeon/conf/ABCs2018/paper/ABCs2018_paper_58.pdf
[62] M. Sharif, A. Kausar, J. Park, and D. R. Shin, “Tiny image classification using four-block convolutional neural network,” in *Proc. Int. Conf. Inf. Commun. Technol. Converg. (ICTC)*, Oct. 2019, pp. 1–6.
[63] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[64] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” in *Proc. IEEE Conf. Comput Vis. Pattern Recognit.*, Jul. 2017, pp. 4700–4708.
[65] F. Ramos Ferreira, J. P. Fernandes, and R. Abreu, “Quantum software frameworks for deep learning,” in *Quantum Software Engineering*. Cham, Switzerland: Springer, 2022, pp. 281–302.
[66] W. J. Conover, *Practical Nonparametric Statistics*, vol. 350. Hoboken, NJ, USA: Wiley, 1999.
[67] M. G. Amankwah, D. Camps, E. W. Bethel, R. Van Beeumen, and T. Perciano, “Quantum pixel representations and compression for N-dimensional images,” *Sci. Rep.*, vol. 12, no. 1, p. 7712, May 2022.
[68] V. Giovannetti, S. Lloyd, and L. Maccone, “Quantum random access memory,” *Phys. Rev. Lett.*, vol. 100, no. 16, Apr. 2008, Art. no. 160501.
[69] K. He and J. Sun, “Convolutional neural networks at constrained time cost,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Boston, MA, USA, Jun. 2015, pp. 5353–5360, doi: 10.1109/CVPR.2015.7299173.
[70] B. Sun, A. M. Iliyasu, F. Yan, F. Dong, and K. Hirota, “An RGB multi-channel representation for images on quantum computers,” *J. Adv. Comput. Intell. Intell. Informat.*, vol. 17, no. 3, pp. 404–417, May 2013.
[71] X. Luo, H. Wu, and Z. Li, “Neulft: A novel approach to nonlinear canonical polyadic decomposition on high-dimensional incomplete tensors,” *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 6, pp. 6148–6166, Jun. 2023.
[72] H. Song, M. Kim, D. Park, Y. Shin, and J.-G. Lee, “Learning from noisy labels with deep neural networks: A survey,” *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Mar. 7, 2022, doi: 10.1109/TNNLS.2022.3152527.

Fan Fan received the Bachelor of Industrial Design degree from Xi’an Jiaotong University, Xi’an, China, in 2016, and the Master of Human-Computer Interaction degree from Bauhaus University Weimar, Weimar, Germany, in 2020. He is currently pursuing the Ph.D. degree with the Chair of Data Science in Earth Observation, Technical University of Munich, Munich, Germany.

He has been with the German Aerospace Center, Weßling, Germany, as a Research Assistant, since November 2020. His research interests focus on quantum algorithms in Earth observation (EO) data analysis, including quantum machine learning, quantum optimization, and so on.



Yilei Shi (Member, IEEE) received the Dipl.Ing. degree in mechanical engineering and the Dr.Ing. degree in signal processing from the Technical University of Munich (TUM), Munich, Germany, in 2010 and 2019, respectively.

He is currently a Senior Scientist with the Chair of Remote Sensing Technology, TUM. His research interests include fast solver and parallel computing for large-scale problems, high-performance computing, computational intelligence, advanced methods of synthetic aperture radar (SAR) and interferometric SAR (InSAR) processing, machine learning, and deep learning for a variety of data sources, such as SAR, optical images, medical images, and partial differential equation (PDE)-related numerical modeling and computing.

Tobias Guggemos received the master’s degree in computer science, in 2014 and the Ph.D. degree in network security and cryptography under the chair of Prof. Kranzlmüller from the Ludwig Maximilian University of Munich (LMU), Munich, Germany, in 2020.

He has been organizing and holding introductory and practical courses on quantum computing at the Ludwig Maximilian University of Munich (LMU Munich), Munich, Germany, since 2016, where he actively changed his research toward quantum computing. He joined the German Aerospace Center, Weßling, Germany, in May 2021, to research quantum algorithms in Earth sciences and recently moved to the University of Vienna, Vienna, Austria, in June 2022, for a post-doctoral position at the Christian Doppler Laboratory for Photonic Quantum Computer of Prof. Philip Walter at the Faculty of Physics.

Xiao Xiang Zhu (Fellow, IEEE) received the M.Sc., Dr.Ing., and Habilitation degrees in signal processing from the Technical University of Munich (TUM), Munich, Germany, in 2008, 2011, and 2013, respectively.

She was the Founding Head of the Department “EO Data Science,” Remote Sensing Technology Institute, German Aerospace Center (DLR), Weßling, Germany. Since May 2020, she has been the Principal Investigator (PI) and the Director of the international future laboratory AI4EO, Munich. Since October 2020, she has been the Director of the Munich Data Science Institute (MDSI), TUM. She was a Guest Scientist or a Visiting Professor with the Italian National Research Council (CNR-IREA), Naples, Italy; Fudan University, Shanghai, China; The University of Tokyo, Tokyo, Japan; and the University of California at Los Angeles, Los Angeles, CA, USA, in 2009, 2014, 2015, and 2016, respectively. She is currently a Visiting AI Professor with Phi-Lab, European Space Agency (ESA), Paris, France. She is the Chair Professor of data science in Earth observation with TUM. Her main research interests are remote sensing and Earth observation, signal processing, machine learning, and data science, with their applications in tackling societal grand challenges, e.g., Global Urbanization, UN’s Sustainable Development Goals (SDGs), and Climate Change.