IOP Publishing Mach. Learn.: Sci. Technol. 5 (2024) 015048 https://doi.org/10.1088/2632-2153/ad2fec

MACHINE LEARNING
Science and Technology

OPEN ACCESS

RECEIVED
20 July 2023

REVISED
25 December 2023

ACCEPTED FOR PUBLICATION
4 March 2024

PUBLISHED
14 March 2024

PAPER

Transforming two-dimensional tensor networks into quantum circuits for supervised learning

Zhihui Song[1], **Jinchen Xu**[1,2], **Xin Zhou**[1], **Xiaodong Ding**[1] and **Zheng Shan**[1,2,*]

[1] Information Engineering University, Zhengzhou 450001, People's Republic of China
[2] Songshan Laboratory, Zhengzhou 450001, People's Republic of China
[*] Author to whom any correspondence should be addressed.

E-mail: shanzhengzz@163.com

Keywords: quantum machine learning, two-dimensional tensor networks, variational data encoding, machine learning interpretability

Abstract

There have been numerous quantum neural networks reported, but they struggle to match traditional neural networks in accuracy. Given the huge improvement of the neural network models' accuracy by two-dimensional tensor network (TN) states in classical tensor network machine learning (TNML), it is promising to explore whether its application in quantum machine learning can extend the performance boundary of the models. Here, we transform two-dimensional TNs into quantum circuits for supervised learning. Specifically, we encode two-dimensional TNs into quantum circuits through rigorous mathematical proofs for constructing model ansätze, including string-bond states, entangled-plaquette states and isometric TN states. In addition, we propose adaptive data encoding methods and combine with TNs. We construct a tensor-network-inspired quantum circuit (TNQC) supervised learning framework for transferring TNML from classical to quantum, and build several novel two-dimensional TN-inspired quantum classifiers based on this framework. Finally, we propose a parallel quantum machine learning method for multi-class classification to construct 2D TNQC-based multi-class classifiers. Classical simulation results on the MNIST benchmark dataset show that our proposed models achieve the state-of-the-art accuracy performance, significantly outperforming other quantum classifiers on both binary and multi-class classification tasks, and beat simple convolutional classifiers on a fair track with identical inputs. The noise resilience of the models makes them successfully run and work in a real quantum computer.

Abbreviations list

List of abbreviations and definitions

TNQC	Tensor-network-inspired quantum circuit
QCL	Quantum circuit learning
NISQ	Noisy intermediate-scale quantum
QNN	Quantum neural network
TN	Tensor network
TNML	Tensor network machine learning
PEPS	Projected entangled pair states
MPS	Matrix product states
TTN	Tree tensor network
MERA	Multi-scale entanglement renormalization ansatz
SBS	String-bond states
EPS	Entangled-plaquette states
isoTNS	Isometric tensor network states
QMPS	MPS-inspired quantum circuit ansatz
QSBS	SBS-inspired quantum circuit ansatz

QEPS	EPS-inspired quantum circuit ansatz
QisoTNS	IsoTNS-inspired quantum circuit ansatz
AE	Angle encoding
CE	Convolutional encoding
VAE	Variational angle encoding
VTNE	Variational tensor network encoding
PQN	Parallel quantum node

1. Introduction

Quantum computing has improved by leaps and bounds in recent decades. It becomes more meaningful to seek and develop applications exerting quantum potential with the emergence of NISQ computers. In recent years, parameterized QCL [1–3] algorithms have drawn a wide range of interest for their noise tolerance and low qubit requirements. In these QCL algorithms, classical data are transformed into vectors in Hilbert space and quantum entanglements are used to represent correlations between them. The learning process is executed by optimizing trainable parameters of the variational quantum circuit. Therefore, the model established is also called QNN. However, almost all QNNs struggle to achieve the accuracy performance of classical neural networks, even the simplest multi-layer perceptrons (MLPs). Researchers are looking for ways to further improve QNNs performance.

Variational quantum circuits inspired by TNs have been applied to machine learning [4–9] and optimization problems [10–12] in recent studies and have become one of the most effective architectures in quantum machine learning. The so-called TN is a framework that approximates higher-order tensors using the contraction of lower-order tensors, whose entanglement entropy satisfies the area law [13, 14]. It has been widely used to simulate quantum many-body systems [15, 16] and has been employed in various fields such as building new frameworks for machine learning [17–19] and constructing quantum circuit simulators [20]. In classical TNML, the two-dimensional PEPS-based model shows a huge improvement in accuracy performance compared to the MPS-based model, and the fusion with convolution feature map further improves the performance of CNN classifier. Such performance is due to the direct reflection of two-dimensional spatial correlations and structural prior knowledge of natural images on PEPS [19]. Because of the natural compatibility of TN with quantum mechanics, quantum computing can benefit from mature TN algorithms [21], which can be encoded into quantum circuits for machine learning. There have been several TNQCs machine learning algorithms proposed, such as MPS [5, 7], TTNs [4, 5], MERA [5, 9]. Classical TNs require exponential bond dimensions to achieve the performance of their quantum version. Although these quantum models have been shown to be effective and hardware-efficient [5], they still significantly underperform classical neural networks in accuracy performance. Given the excellent performance of 2D TN in TNML, it will be interesting to explore whether it can bring similar performance improvements to QCL models and even enable them to challenge classical ones.

In this paper, we aim to apply 2D TN to QCL and improve the accuracy performance of quantum models. To this end, we mainly face two questions: first, how to encode 2D TN into quantum circuit for being applied to QCL just as it does in TNML? Second, how to use 2D TN to improve the accuracy performance of QCL to meet or even exceed that of classical classifiers?

To solve the first question, we first encode 2D TNs into quantum circuits of unitary gates through rigorous mathematical proofs, including SBS [22, 23], EPS [24], and isoTNS [25]. Then, to allow these 2D TNQCs to be used in QCL to construct ansätze or encoders, we construct a TNQC supervised learning framework transferring TNML from classical to quantum. Any circuit encoded from a TN can be applied to QCL through this framework. To solve the second question, we not only use 2D TNQCs as ansätze of quantum models, but also propose several variational encoding methods, which can be combined with CNN feature map or TNQCs to transform the original data to adaptive quantum state features. We integrate our proposed variational encoders and 2D TNQC ansätze on the basis of TNQC supervised learning framework to build 9 novel 2D TNQC classifiers. These classifiers are validated and compared with existing quantum classifiers in terms of performance on the same dataset. Considering that classical classifiers can easily perform multi-class classification tasks, the quantum models should be compared with classical ones in the same multi-class classification task, so we propose a parallel quantum machine learning method for multi-class classification, on which we build our 2D TNQC multi-class classifier.

We evaluate effectiveness of the proposed method and the performance of these models on the MNIST [26] image benchmark dataset through classical simulations as many QNN models do. The results show that our models achieve the state-of-the-art accuracy performance among quantum models, and beat classical neural network classifiers on the fair track. Without adding any classical network layer, our proposed models achieve test accuracy of over 99% in almost all MNIST pairwise subset classification tasks and still perform at a high level on another dataset Fashion-MNIST [27]. Such performances are significantly better than the

MPS-inspired QCL model and outperform other quantum classifiers. The 99.18% test accuracy in quaternary classification exceeds that of a simple CNN classifier with identical inputs. Such results demonstrate the effectiveness of our proposed encoders, ansätze, framework and models. 2D TN successfully helps the QCL models to improve their accuracy performance by 18.38%, making them outperform the classical classifiers in some cases. Moreover, performing 2D TNQC ansätze on quantum computers instead of computing the contraction of 2D TN or simulating these ansätze on classical computers can alleviate memory bottleneck problem. This advantage allows larger scale 2D TNQCs to be constructed to build QNN models with larger data input dimensions and higher accuracy and promote their practical benefits. Thus, we hope that these models are not only useful on simulators but also on real quantum machines. To this end we test noise resilience of the model, the results show that the models have certain robustness to thermal relaxation noise, which encourages us to run a minimum example successfully on the ibmq_nairobi quantum computer.

In summary, this paper makes the following contributions:

- We encode two-dimensional TNs into quantum circuits of unitary gates using rigorous mathematical proofs.
- We propose 3 novel and effective quantum variational encoding methods.
- We construct a supervised learning framework for transferring TNML from classical to quantum.
- We design 9 novel two-dimensional TN inspired QCL models.
- We propose a parallel quantum machine learning method called PQN for multi-class classification.
- The accuracies of the models extend the performance boundary of QCL, and our research promotes the application of TNs in quantum machine learning.

2. Methods

2.1. Introduction to two-dimensional TNs

TN is a framework that approximates higher-order tensors using the contraction of lower-order tensor. In a tensor diagram, each index of a tensor is represented by a line, and the tensor itself is depicted as a node. The edges between nodes represent the contraction of virtual indices, whose dimension is called bond (or virtual) dimension D and generally bounded as $D \leqslant \chi$ to reduce the computational cost. TN contracts in a certain direction, thus forming some specific TN architectures.

MPS can effectively describe the ground state in one-dimensional quantum spin system, which is due to the fact that MPS can fully capture local entanglement characteristics in the system. As shown in figure 1(a), an MPS with N nodes is described as

$$|\psi_{MPS}\rangle = \sum_{\{s\}}\sum_{\{a\}}\left(A^{s_1}_{a_1}A^{s_2}_{a_1a_2}\cdots A^{s_N}_{a_{N-1}}\right)\bigotimes_{i=1}^{N}|s_i\rangle \tag{1}$$

where each third-order tensor $A^{s_n}_{a_{n-1}a_n}$ has a_{n-1} and a_n which are related to the left and right virtual indices and has a physical index s_n. All the virtual indices are contracted to form a tensor with N physical indices, so it can be used to describe the 1D quantum-many body systems.

However, the applicability of MPS is difficult to extend to higher-dimensional systems. The same problem also occurs in machine learning models based on MPS and its quantum version. For example, when the model involves processing two-dimensional data such as natural images, as shown in figure 1(b), pixels are rearranged into one-dimensional vectors in a certain order to meet the requirements of MPS, such processing loses the original correlation information between pixels. To solve this problem, a natural idea is to construct models using 2D TNs with the same geometric structure as data, the most typical of which is the PEPS [15, 16], which is the high-dimensional generalization of MPS. A PEPS with open boundary on a two-dimensional lattice of size $H \times V = 4 \times 4$ shown in figure 1(c) can be written as

$$|\psi_{PEPS}\rangle = \sum_{\{s\}}\mathcal{F}\left(M^{s_{(1,1)}}_{(1,1)}, M^{s_{(1,2)}}_{(1,2)}\cdots, M^{s_{(H,V)}}_{(H,V)}\right)\bigotimes_{i=1}^{H}\bigotimes_{j=1}^{V}|s_{(i,j)}\rangle \tag{2}$$

where $M^{s_{(m,n)}}_{(m,n)}$ is a fifth-order tensor with four virtual indices $\alpha, \beta, \gamma, \varepsilon$ (corresponding to the left, up, right, down direction) connecting neighbors and a physical index s. $\mathcal{F}$ denotes the contraction of virtual indices of all tensors. However, it is more difficult to simulate PEPS than MPS in both classical or quantum way [10, 28], and the expansion of model is limited by computational cost. Here, we introduce other types of 2D TN states, including SBS, EPS and isoTNS, which are described as PEPS subclasses with certain limitations [25, 29]. They can reduce computational cost and the quantum circuit ansätze inspired by them can efficiently describe 2D TN.

A SBS [21] defines a set S of ordered strings, and a one-dimensional MPS for each string. By covering all nodes on the two-dimensional lattice with overlapping MPS, the correlations of two-dimensional nodes are obtained while retaining the advantages of one-dimensional structure. The description power of SBS

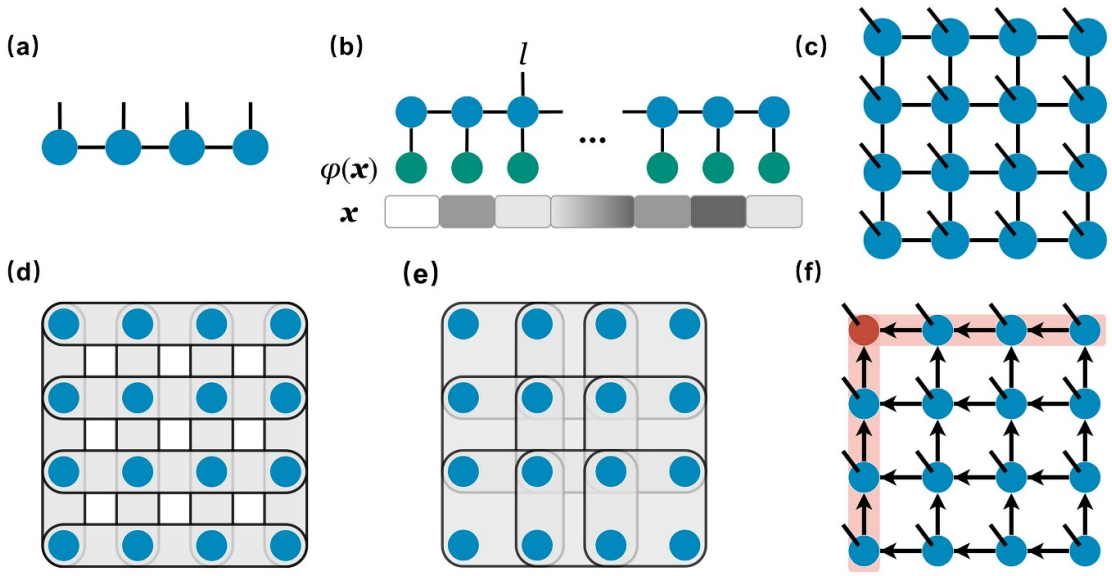

Figure 1. Classical TNs. (a) Tensor diagram of a matrix product state with virtual indices of bond dimension $D = \chi$, and physical indices (lines sticking out) of dimension d. (b) A tensor network machine learning model for image classification. The data is expanded into one dimension and mapped into vectors. The parameterized TN contracts with vectors to obtain a probability distribution over labels. (c) A PEPS with open boundary condition on a 4×4 lattice. (d) An SBS composed of vertical and horizontal strings overlaps. (e) An EPS composed of small overlapping plaquettes containing 4 physical indices. (f) IsoTNS. These arrows indicate the isometry conditions of tensors. The red node in the upper left corner is the orthogonality center, and its row and column are orthogonality hypersurfaces.

depends on the choice of strings. In image learning tasks, two-dimensional lattice is usually covered by strings on multiple rows and columns shown in figure 1(d), and overlapping snake strings can also work, which is called Snake-SBS.

Another efficient method for 2D TN is EPS, or called correlator product states. The basic approach of the EPS is to describe a wave function by product of multiple sub-plaquette tensors in a 2D lattice, and to describe the short-range correlation by overlap of sub-plaquettes whose wave functions are constructed exactly [24]. An EPS composed of $P = 9$ plaquettes with 4 physical indices is shown in figure 1(e). The description power of EPS depends on the size of sub-plaquettes. Larger plaquettes bring higher accuracy but also increase computational cost.

An isoTNS describes a TN state with isometry conditions, which allows the two-dimensional network to be reduced to the canonical form of 1D MPS when contracting rows and columns [25]. For 2D isoTNS, all tensors that make up isoTNS are isometries. The physical index of each tensor has an incoming arrow, and the virtual indices have incoming and outgoing arrows. All arrows point in the direction of the center node (called orthogonality center) or the rows and columns where it is located (called orthogonality hypersurfaces), and the directions of theses arrows are opposite to those of tensor contractions. When the incoming virtual indices and physical index of these tensors contract with the corresponding indices of their complex conjugate tensors, the remaining indices yield the identity. For example, the tensor outside the orthogonality hypersurfaces in figure 1(f) satisfies

$$\sum_{s\gamma\varepsilon} A^{s}_{\alpha\beta\gamma\varepsilon} A^{\dagger s}_{\alpha'\beta'\gamma\varepsilon} = I_{\alpha\alpha'} I_{\beta\beta'} \tag{3}$$

where I is the identity and the tensor on the orthogonality hypersurfaces satisfies

$$\sum_{s\beta\gamma\varepsilon} A^{s}_{\alpha\beta\gamma\varepsilon} A^{\dagger s}_{\alpha'\beta\gamma\varepsilon} = I_{\alpha\alpha'}. \tag{4}$$

IsoTNS can be more easily implemented on quantum circuit by efficiently moving the orthogonality center to the corner of two-dimensional lattice [25]. Note that all of the above TNs satisfy equation (2). They are applicable to classical computing, and they inspire circuit ansätze in QCL. The most obvious difference is that the quantum gates used to describe the states on quantum circuits are unitary, meaning that their corresponding classical 2D TN tensors are also required to be unitary.

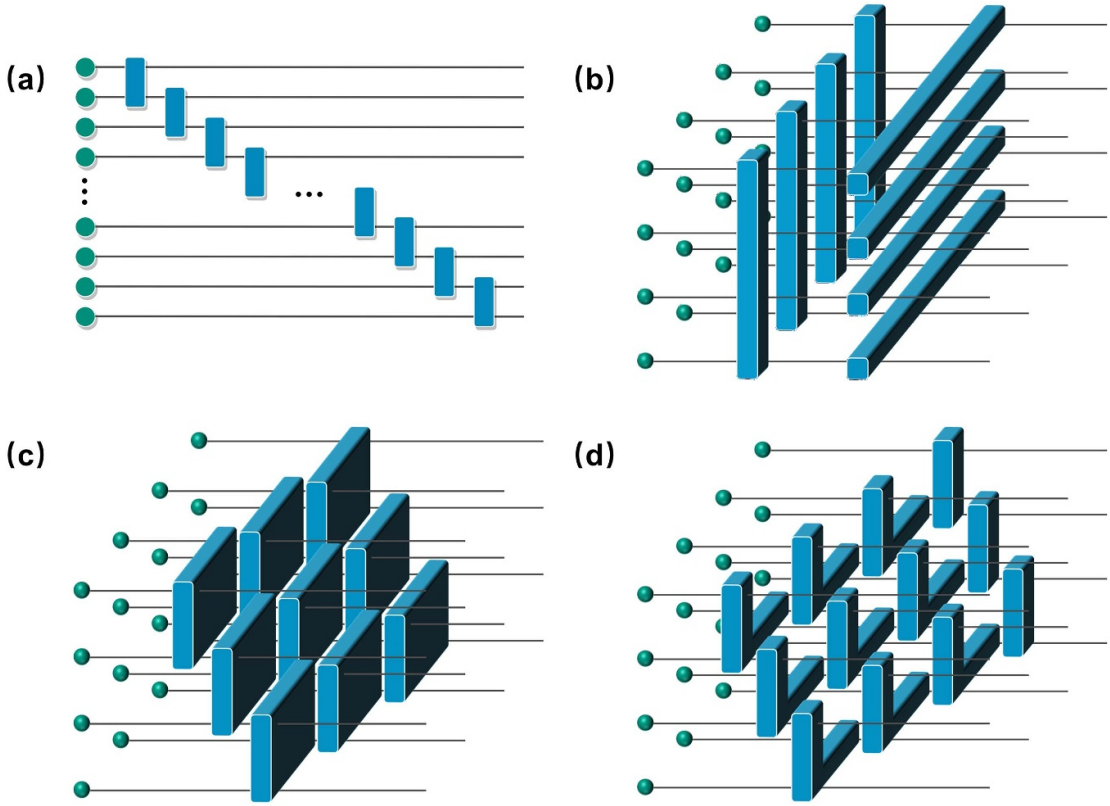

Figure 2. Quantum circuits inspired by different TNs. There are 16 green nodes in each diagram that represent encoded qubits and are arranged in a 4×4 lattice in 2D TNQC. Each blue block represents a multi-qubit unitary, except in QSBS where a block represents a QMPS. (a) A 16-qubit QMPS circuit consists of fifteen two-qubit unitaries sequentially applied to adjacent qubits (b) A 16-qubit QSBS circuit consists of four vertical QMPSs first prepared and four horizontal QMPSs followed. (c) A 16-qubit QEPS circuit consists of nine sequentially applied four-qubit unitaries. (d) A 16-qubit QisoTNS circuit consists of nine three-qubit unitaries and three two-qubit unitaries, these unitaries are applied in the order of their layers.

2.2. Ansätze: generating quantum circuit ansätze from 2D TNs through mathematical proofs

We introduced three classical 2D TNs in the previous section, since our goal is to apply 2D TNs to QCL, the first thing we need to do is to encode them into quantum circuits. In this section we propose 3 different 2D TNQC ansätze and we show how they are generated from classical TNs with rigorous mathematical proofs.

2.2.1. Circuit ansätze

QMPS: Before introducing 2D TNQC ansätze, we first review the 1D MPS circuit ansatz. As shown in figure 2(a), a 16-qubit QMPS can be implemented by sequentially applying a two-qubit unitary to adjacent qubits [30]. Each two-qubit unitary entangles the last qubit obtained from a previous unitary with the next one. Starting from the encoded product state $\bigotimes_{i=1}^{N} |\phi_i\rangle$, QMPS returns a state

$$|\psi_{QMPS}\rangle = U_{N-1}^{[2]} U_{N-2}^{[2]} \cdots U_2^{[2]} U_1^{[2]} \bigotimes_{i=1}^{N} |\phi_i\rangle \tag{5}$$

where $U_n^{[2]}, n = \{1, 2, \ldots, N-2, N-1\}$ denotes a two-qubit unitary acts on the nth and $(n+1)$th qubit.

The structure of QMPS shows that entanglement only occurs between one-dimensional adjacent qubits, while 2D TNQC ansätze extend it to two dimensions.

QSBS: We construct a 2D lattice of qubits of size $N = H \times V$ to demonstrate the spatial structure of 2D TNQC. The first ansatz shown in figure 2(b) is inspired by SBS (called QSBS), which consists of multiple QMPSs. QSBS first applies a set of vertically oriented QMPSs on different columns in parallel on the lattice, followed by a set of horizontally oriented QMPSs on all rows in the next layer. The QMPSs corresponding to

the strings in the same direction are set as canonical form with the same orthogonal direction to generate linear sequential circuits [31]. So QSBS returns a state

$$|\psi_{QSBS}\rangle = \prod_{m=1}^{H} U^{[2]}_{(m,V-1)} \cdots U^{[2]}_{(m,1)} \prod_{n=1}^{V} U^{[2]}_{(H-1,n)} \cdots U^{[2]}_{(1,n)} \bigotimes_{i=1}^{H} \bigotimes_{j=1}^{V} |\phi_{(i,j)}\rangle \tag{6}$$

where $U^{[2]}_{(m,n)}$ is a two-qubit unitary acting on the qubits at (m,n) and $(m+1,n)$ for vertical QMPSs or (m,n) and $(m,n+1)$ for horizontal QMPSs of the 2D lattice.

QEPS: The EPS-inspired quantum circuit (called QEPS) sequentially applies a unitary $U^{[4]}_{(m,n)}$ to the four qubits of each 2×2 plaquette from (m,n) to $(m+1,n+1)$ on the lattice. It returns a state

$$|\psi_{QEPS}\rangle = \prod_{\tau\in\overrightarrow{\mathcal{S}}} U^{[4]}_{\tau} \bigotimes_{i=1}^{H} \bigotimes_{j=1}^{V} |\phi_{(i,j)}\rangle \tag{7}$$

where $\tau = (m,n) \in \overrightarrow{\mathcal{S}}$, $\overrightarrow{\mathcal{S}}$ denotes the order of unitary gates. Figure 2(c) shows a QEPS consisting of 9 four-qubit unitaries acting on different plaquettes, and the unitaries are applied sequentially in the order $\mathcal{S} = ((1,1),(1,2),\ldots(3,2),(3,3))$.

QisoTNS: The isoTNS-inspired quantum circuit (called QisoTNS) starts from the corner of the 2D lattice and applies a three-qubit unitary $U^{[3]}_{(1,1)}$ on a qubit group including qubits located at $(1,1)$, $(2,1)$ and $(2,2)$. Then in the next layer, we apply a three-qubit unitary $U^{[3]}_{(m,n)}$ acting on (m,n), $(m+1,n)$ and $(m+1,n+1)$ in parallel at each location offset by $+1$ row or column relative to the qubit group in the previous layer. This process is repeated up to the boundary of the 2D lattice. A qubit group beyond the rightmost boundary of the lattice only applies a two-qubit unitary $U^{[2]}_{(m,n)}$ on the qubits located at (m,n) and $(m+1,n)$. Such a design follows the tensor contraction direction and satisfies isometry conditions in isoTNS. A QisoTNS can be described as

$$|\psi_{QisoTNS}\rangle = \prod_{l\in\overrightarrow{\mathcal{L}}} \prod_{\tau\in\mathcal{S}_l} U_{\tau} \bigotimes_{i=1}^{H} \bigotimes_{j=1}^{V} |\phi_{(i,j)}\rangle \tag{8}$$

where l represents the layer number of circuit, $\overrightarrow{\mathcal{L}}$ is an ordered set of layers ranging from 1 to $H+V-2$, and these layers are applied sequentially. $\mathcal{S}_l$ represents the unitaries applied in the lth layer, which contains $\min(l, H+V+1-l)$ unitaries of three-qubit or two-qubit, and these unitaries are applied in parallel. In total, a QisoTNS circuit contains $(H-1)\times V$ unitaries. For example, a 16-qubit QisoTNS ansatz shown in figure 2(d) has 6 layers, namely, $\{U^{[3]}_{(1,1)}\}_1$, $\{U^{[3]}_{(2,1)}, U^{[3]}_{(1,2)}\}_2$, $\{U^{[3]}_{(3,1)}, U^{[3]}_{(2,2)}, U^{[3]}_{(1,3)}\}_3$, $\{U^{[3]}_{(3,2)}, U^{[3]}_{(2,3)}, U^{[2]}_{(1,4)}\}_4$, $\{U^{[3]}_{(3,3)}, U^{[2]}_{(2,4)}\}_5$, $\{U^{[2]}_{(3,4)}\}_6$.

These circuit ansätze are generated from TNs. Recalling the classical TNs, all the indices are set to 2 so that each tensor that makes up them is described by a unitary gate on the quantum circuit. Tensors in MPS and isoTNS first needs to balance the number of incoming and outgoing indices due to unitary gate constraints, then unitary gates are applied in a specific order according to the adjusted directed MPS and isoTNS diagrams to generate QMPS and QisoTNS having the same tensor diagrams. SBS and EPS provide specific methods to represent a 2D TN by overlapping local tensors in a lattice, which allows us to construct quantum circuits with the same rules, so QSBS and QEPS are inspired by the generating rules rather than specific states, and their corresponding TNs can be described by transforming unitaries back to tensors.

2.2.2. Mathematical proofs

Mathematical proof for QMPS: Now we further prove that ansätze introduced above are generated from TNs. First, a QMPS ansatz is a left (or right)-orthogonal MPS with $D = d = 2$, and each tensor of the MPS satisfies the left (or right)-orthogonal condition, i.e.,

$$\sum_{sa_n} A^{s}_{(n)a_{n-1}a_n} A^{\dagger s}_{(n)a_{n-1}{}'a_n} = I_{a_{n-1}a_{n-1}{}'}. \tag{9}$$

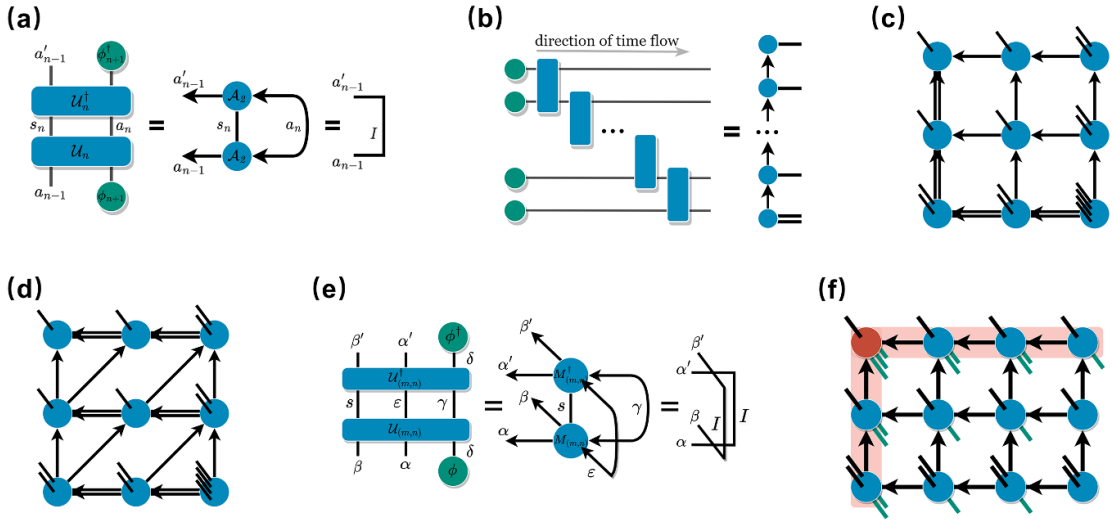

Figure 3. (a) The diagrammatic proof that each combination of a two-qubit unitary and $|\phi\rangle$ is equivalent to an MPS tensor, and it satisfies the isometry condition. (b) A QMPS circuit is equivalent to a special MPS. (c) The TN diagram corresponding to the 16-qubit QSBS circuit. (d) The TN diagram corresponding to the 16-qubit QEPS circuit. (e) The diagrammatic proof that each combination of a three-qubit unitary and $|\phi\rangle$ is equivalent to an isoTNS tensor, and it satisfies the isometry condition. (f) The TN diagram corresponding to the 16-qubit QisoTNS circuit.

While two-qubit unitary gates applied in a QMPS can be written as

$$\begin{aligned} U_1^{[2]} &= \sum_{s_1,a_1} B^{s_1}_{(1)a_1\delta_1\delta_2}|s_1,a_1\rangle\langle\delta_1,\delta_2| \\ U_n^{[2]} &= \sum_{s_n,a_n} B^{s_n}_{(n)a_{n-1}a_n\delta_{n+1}}|s_n,a_n\rangle\langle a_{n-1},\delta_{n+1}| \\ U_{N-1}^{[2]} &= \sum_{s_{N-1}s_N} B^{s_{N-1}s_N}_{(N-1)a_{N-2}\delta_N}|s_{N-1},s_N\rangle\langle a_{N-2},\delta_N| \end{aligned} \tag{10}$$

with $1 < n < N-1$ and $\dim(a_n) = \dim(s_n) = \dim(\delta_n) = 2$ where δ_n is the index connecting $|\phi_n\rangle$. Since U_n is a unitary, it has $U_n^\dagger U_n = I_{a_{n-1}a'_{n-1}} I_{\delta_{n+1}\delta'_{n+1}}$. Then according to equation (10) and $|\phi_{n+1}\rangle = \sum_{\delta_{n+1}} \nu_{(n+1)\delta_{n+1}}|\delta_{n+1}\rangle$, we have

$$\sum_{s_n a_n}\sum_{\delta_{n+1}} \nu^\dagger_{\delta_{n+1}} B^{\dagger s_n}_{(n)a'_{n-1}a_n\delta'_{n+1}} B^{s_n}_{(n)a_{n-1}a_n\delta_{n+1}} \nu_{\delta_{n+1}} = I_{a_{n-1}a'_{n-1}} \tag{11}$$

as shown in figure 3(a) with a tensor diagram. Let $\sum_{\delta_{n+1}} B^{s_n}_{(n)a_{n-1}a_n\delta_{n+1}} \nu_{(n+1)\delta_{n+1}} = A^{s_n}_{(n)a_{n-1}a_n}$, then tensor $A^{s_n}_{(n)a_{n-1}a_n}$ satisfies the isometry condition equation (9). Therefore, each unitary gate can be identified as an MPS tensor. Combining equation (5), we get

$$\begin{aligned} |\psi_{QMPS}\rangle &= U_{N-1}^{[2]} U_{N-2}^{[2]} \cdots U_2^{[2]} U_1^{[2]} \bigotimes_{i=1}^{N} |\phi_i\rangle \\ &= \sum_{\{s\}}\sum_{\{a\}} B^{s_{N-1}s_N}_{(N-1)a_{N-2}\delta_N} \cdots B^{s_1}_{(1)a_1\delta_1\delta_2} |s_1\cdots s_N\rangle\langle\delta_1\cdots\delta_N| \bigotimes_{i=1}^{N} \left(\sum_{\delta_i} \nu_{(i)\delta_i}|\delta_i\rangle\right) \\ &= \sum_{\{s\}}\sum_{\{a\}}\sum_{\{\delta\}} B^{s_{N-1}s_N}_{(N-1)a_{N-2}\delta_N} \nu_{(N)\delta_N} \cdots B^{s_1}_{(1)a_1\delta_1\delta_2} \nu_{(2)\delta_2} \nu_{(1)\delta_1} \bigotimes_{i=1}^{N} |s_i\rangle \\ &= \sum_{\{s\}}\sum_{\{a\}} A^{s'_{N-1}}_{(N-1)a_{N-2}} \cdots A^{s_1}_{(1)a_1} \bigotimes_{i=1}^{N} |s_i\rangle = |\psi_{MPS}\rangle \end{aligned} \tag{12}$$

where $s'_{N-1} = s_{N-1}s_N$. This means that a QMPS is generated from a special MPS. For example, as shown in figure 3(b), a QMPS with n unitaries is generated from a left-orthogonal MPS having n tensors. Each tensor has a physical index of dimension 2, except for the last tensor which has a physical index of dimension 2^2, this

can be seen as a contraction between A_{N-1} and an additional tensor [11]. The dimensions of all indices are 2. The temporal sequence of applying unitaries on quantum circuit is the opposite of the arrows' directions.

Mathematical proof for QSBS: Now back to 2D TNQC ansätze. The unitary gates applied in QSBS are the same as those in QMPS. So according to equation (6), we can get

$$
\begin{aligned}
|\psi_{QSBS}\rangle &= \prod_{m=1}^{H} U^{[2]}_{(m,V-1)} \cdots U^{[2]}_{(m,1)} \prod_{n=1}^{V} U^{[2]}_{(H-1,n)} \cdots U^{[2]}_{(1,n)} \bigotimes_{i=1}^{H}\bigotimes_{j=1}^{V} |\phi_{(i,j)}\rangle \\
&= \prod_{m=1}^{H} \sum_{\{s_{(m,)}\}} \sum_{\{a'_{(m,)}\}} B^{s_{(m,V-1)}s_{(m,V)}}_{(m,V-1)a^{horiz}_{(m,V-2)}\delta'_{(m,V)}} \cdots B^{s_{(m,1)}}_{(m,1)a^{horiz}_{(m,1)}\delta'_{(m,1)}\delta'_{(m,2)}} \bigotimes_{j=1}^{V} |s_{(m,j)}\rangle \bigotimes_{j=1}^{V} \langle\delta'_{(m,j)}| \\
&\quad \times \prod_{n=1}^{V} \sum_{\{\delta'_{(,n)}\}} \sum_{\{a_{(,n)}\}} A^{\delta'_{(H-1,n)}\delta'_{(H,n)}}_{(H-1,n)a^{vert}_{(H-2,n)}} \cdots A^{\delta'_{(1,n)}}_{(1,n)a^{vert}_{(1,n)}} \bigotimes_{i=1}^{H}\bigotimes_{j=1}^{V} |\delta'_{(i,j)}\rangle \\
&= \sum_{\{s\}} \sum_{\{a'_{(m,)}\}} \sum_{\{a_{(,n)}\}} \sum_{\{\delta'\}} B^{s_{(H-1,V-1)}s_{(H-1,V)}}_{(H-1,V-1)a^{horiz}_{(H-1,V-2)}\delta'_{(H-1,V)}} B^{s_{(H,V-1)}s_{(H,V)}}_{(H,V-1)a^{horiz}_{(H,V-2)}\delta'_{(H,V)}} A^{\delta'_{(H-1,V)}\delta'_{(H,V)}}_{(H-1,V)a^{vert}_{(H-2,V)}}, \ldots, \\
&\quad \times B^{s_{(1,1)}}_{(1,1)a^{horiz}_{(1,1)}\delta'_{(1,1)}\delta'_{(1,2)}} A^{\delta'_{(1,1)}}_{(1,1)a^{vert}_{(1,1)}} A^{\delta'_{(1,2)}}_{(1,2)a^{vert}_{(1,2)}} \bigotimes_{i=1}^{H}\bigotimes_{j=1}^{V} |s_{(i,j)}\rangle \\
&= \sum_{\{s\}} \sum_{\{a'_{(m,)}\}} \sum_{\{a_{(,n)}\}} M^{s_{(H-1,V-1)}s_{(H-1,V)}s_{(H,V-1)}s_{(H,V)}}_{(H-1,V-1)a^{horiz}_{(H-1,V-2)}a^{horiz}_{(H,V-2)}a^{vert}_{(H-2,V)}}, \ldots, M^{s_{(1,1)}}_{a^{horiz}_{(1,1)}a^{vert}_{(1,1)}a^{vert}_{(1,2)}} \bigotimes_{i=1}^{H}\bigotimes_{j=1}^{V} |s_{(i,j)}\rangle \\
&= \sum_{\{s\}} \sum_{\{a'_{(m,)}\}} \sum_{\{a_{(,n)}\}} M^{s'_{(H-1,V-1)}}_{(H-1,V-1)a'^{horiz}_{(H-1,V-2)}a^{vert}_{(H-2,V)}}, \ldots, M^{s_{(1,1)}}_{a^{horiz}_{(1,1)}a'^{vert}_{(1,1)}} \bigotimes_{i=1}^{H}\bigotimes_{j=1}^{V} |s_{(i,j)}\rangle = |\psi_{PEPS}\rangle
\end{aligned}
\tag{13}
$$

where δ' denes the indices of the first layer tensors connected with the second layer tensors. a^{horiz} and a^{vert} are virtual indexes in the horizontal and vertical directions. $s'_{(H-1,V-1)} = s_{(H-1,V-1)}s_{(H-1,V)}s_{(H,V-1)}s_{(H,V)}$ and $a'^{horiz}_{(H-1,V-2)} = a^{horiz}_{(H-1,V-2)}a^{horiz}_{(H,V-2)}$ mean that tensor $M_{(H-1,V-1)}$ has a physical index of dimension 2^4, and it has a virtual index of dimension 2^2 in the horizontal direction. Our proposed 16-qubit QSBS using the same generating rule as SBS but is equivalent to the TN shown in figure 3(c), which indicates that the SBS-inspired circuit is a special PEPS with non-uniform virtual and physical bond dimensions. Moreover, $\sum_{\delta'_{(i,j)}} B^{s_{(i,j)}}_{(i,j)a'_{(i,j-1)}a'_{(i,j)}\delta'_{(i,j+1)}} A^{\delta'_{(i,j+1)}}_{(i,j+1)a_{(i-1,j+1)}a_{(i,j+1)}} = M^{s_{(i,j)}}_{(i,j)a'_{(i,j-1)}a'_{(i,j)}a_{(i-1,j+1)}a_{(i,j+1)}}$ means that two unitaries (three or four in some cases) from two layers create a PEPS tensor. In addition, since indices have directions, QSBS can actually be further classified as the isoTNS.

Mathematical proof for QEPS: The four-qubit unitary gate used in QEPS can be written as

$$
U^{[4]}_{\tau} = \sum_{\{s^\tau\},\{a^\tau\}} T^{\{s^\tau\}}_{(\tau)\{\delta^\tau\}\{a^\tau\}} |\{s^\tau\},\{a^\tau_\mu\}\rangle\langle\{a^\tau_\eta\},\{\delta^\tau\}|
\tag{14}
$$

where $T^{\{s^\tau\}}_{(\tau)\{\delta^\tau\}\{a^\tau\}}$ is an eighth-order tensor (i.e. a $2^4 \times 2^4$ matrix), and $\{s^\tau\},\{a^\tau\},\{\delta^\tau\}$ are the physical indices, virtual indices, and indices connecting encoded states $|\phi\rangle$ of this tensor, $\{a^\tau_\mu\}$ and $\{a^\tau_\eta\}$ are the set of its incoming and outgoing virtual indexes. According to equation (7), we have

$$
\begin{aligned}
|\psi_{QEPS}\rangle &= \prod_{\tau\in\overrightarrow{\mathcal{S}}} U^{[4]}_{\tau} \bigotimes_{i=1}^{H}\bigotimes_{j=1}^{V} |\phi_{(i,j)}\rangle \\
&= \prod_{\tau\in\overrightarrow{\mathcal{S}}} \sum_{\{s^\tau\},\{a^\tau\}} T^{\{s^\tau\}}_{(\tau)\{\delta^\tau\}\{a^\tau\}} |\{s^\tau\},\{a^\tau_\mu\}\rangle\langle\{a^\tau_\eta\},\{\delta^\tau\}| \bigotimes_{i=1}^{H}\bigotimes_{j=1}^{V} \sum_{\{\delta\}} \nu_{(i,j)\delta_{(i,j)}} |\delta_{(i,j)}\rangle \\
&= \sum_{\{s\}}\sum_{\{a\}} M^{\{s^{(H-1,V-1)}\}}_{(H-1,V-1)\{a^{(H-1,V-1)}\}}, \ldots, M^{\{s^{(1,1)}\}}_{(1,1)\{a^{(1,1)}\}} \bigotimes_{i=1}^{H}\bigotimes_{j=1}^{V} |s_{(i,j)}\rangle
\end{aligned}
\tag{15}
$$

with $\sum_{\{\delta^\tau\}} T^{\{s^\tau\}}_{\{\delta^\tau\}\{a^\tau\}}\{\nu_\delta\} = M^{\{s^\tau\}}_{\{a^\tau\}}$, which shows that each four-qubit unitary gate creates a tensor on the 2D TN. However, the EPS-inspired circuit is not a PEPS, because, as shown in figure 3(d) where lies the corresponding tensor diagram of QEPS, its virtual bonds are not always between adjacent tensors, QEPS is a special 2D TN with non-vertical connections.

Mathematical proof for QisoTNS: Now moving to QisoTNS, a three-qubit unitary gate outside the orthogonality hypersurfaces is described as

$$U^{[3]}_{(m,n)} = \sum_{s\gamma\varepsilon} D^{s}_{(m,n)\alpha\beta\gamma\varepsilon\delta} |s,\varepsilon,\gamma\rangle\langle\beta,\alpha,\delta| \tag{16}$$

where $D^{s}_{(m,n)\delta\alpha\beta\gamma\varepsilon}$ is a sixth-order tensor (i.e. a $2^3 \times 2^3$ matrix), and $\alpha,\beta,\gamma,\varepsilon$ are its virtual indices. Since $U^{\dagger[3]}_{(m,n)} U^{[3]}_{(m,n)} = I_{\alpha\alpha'} I_{\beta\beta'} I_{\delta\delta'}$, we can get

$$\sum_{s\gamma\varepsilon}\sum_{\delta} v^{\dagger}_{(m+1,n+1)\delta} D^{\dagger s}_{(m,n)\alpha'\beta'\gamma\varepsilon\delta} D^{s}_{(m,n)\alpha\beta\gamma\varepsilon\delta} v_{(m+1,n+1)\delta} = I_{\alpha\alpha'} I_{\beta\beta'}. \tag{17}$$

Let $\sum_{\delta} D^{s}_{(m,n)\alpha\beta\gamma\varepsilon\delta} v_{(m+1,n+1)\delta} = M^{s}_{(m,n)\alpha\beta\gamma\varepsilon}$, then tensor $M^{s}_{(m,n)\alpha\beta\gamma\varepsilon}$ satisfies the isometry condition equation (3), which is shown in figure 3(e). For a three-qubit unitary gate on the orthogonality hypersurfaces, $\sum_{\delta_1\delta_2} D^{s}_{\alpha\gamma\varepsilon\delta_1\delta_2} v_{\delta_1} v_{\delta_2} = M^{s}_{\alpha\gamma\varepsilon}$, and $M^{s}_{\alpha\gamma\varepsilon}$ satisfies the isometry condition equation (4). In addition, the two-qubit unitary gate in QisoTNS has similar properties to that in QMPS. These lead to the fact that each unitary gate can be viewed as an isoTNS tensor. Combined with equation (8), we have

$$\begin{aligned} |\psi_{QisoTNS}\rangle &= \prod_{l\in\overrightarrow{\mathcal{L}}} \prod_{\tau\in\mathcal{S}_l} U_{\tau} \bigotimes_{i=1}^{H}\bigotimes_{j=1}^{V} |\phi_{(i,j)}\rangle \\ &= \sum_{\{s\}}\sum_{\{a\}} M^{s_{(H-1,V)}s_{(H,V)}}_{(H-1,V)\{a^{(H-1,V)}\}},\ldots,M^{s_{(1,1)}}_{(1,1)\{a^{(1,1)}\}} \bigotimes_{i=1}^{H}\bigotimes_{j=1}^{V} |s_{(i,j)}\rangle \\ &= \sum_{\{s\}}\sum_{\{a\}} M^{s'_{(H-1,V)}}_{(H-1,V)\{a^{(H-1,V)}\}},\ldots,M^{s_{(1,1)}}_{(1,1)\{a^{(1,1)}\}} \bigotimes_{i=1}^{H}\bigotimes_{j=1}^{V} |s_{(i,j)}\rangle = |\psi_{PEPS}\rangle \end{aligned} \tag{18}$$

where $s'_{(H-1,n)} = s_{(H-1,n)} s_{(H,n)}$ means that each tensor in the bottom row has a physical index of dimension 2^2. And a QisoTNS is generated from a special PEPS satisfying the isometry conditions, namely, an isoTNS. The 16-qubit QisoTNS circuit is equivalent to an isoTNS with 12 tensors as shown in figure 3(f). Each tensor has a physical index of dimension 2, except tensors in the bottom row, which is similar to the QMPS. The green lines indicate indices connected to the encoded states. Such directed tensor diagram is adapted from the original isoTNS. As mentioned before, to construct a QisoTNS equivalent to isoTNS, each tensor is connected by a different number of green lines to balance indices before generating QisoTNS. In addition, making the upper-left tensor the orthogonality center is easier for generating QisoTNS.

These three types of ansätze are generated from special 2D TNs. QSBS and QisoTNS belong to the same type of TN, and QisoTNS has more tensors and more uniform dimensions of indices. Although QSBS is also a 2D TN, not all its connections are between neighbors. Their TN structures may affect the classification performance.

Note that we only show single-layer circuit ansätze with mathematical proofs in this section, which are formally equivalent to TNs with $D = d = 2$. However, when constructing QCL models, the models can follow the proposal of [11] to use multi-layer circuit ansätze to construct TNs with a larger bond dimension D. These circuit structures can also be used in encoders, as we will mention in section 2.4.

2.3. Framework: from TNML to TN-inspired QCL

Our goal is to apply 2D TN to QCL, now that 2D TN has been transformed into 2D TNQC ansätze, the next thing that needs to be done is to apply these ansätze to QCL, so here we propose a TNQC supervised learning framework shifting the perspective from classical TN to quantum one.

Typically, a classification task learns from a dataset of existing classification labels and establishes a mapping from the input data space to the classification label space. Here, we consider the training data with an N-dimensional real number vector $\mathbf{x} = (x_1, x_2, \ldots, x_N)$ from a grayscale image, where $x_i \in [0,1]$, which represents the normalized value of this pixel. It first needs to map data vectors to a high-dimensional space. A typical way in classical TNML is to use the local feature map $\phi(x_i) = \cos(\frac{\pi}{2}x_i)|0\rangle + \sin(\frac{\pi}{2}x_i)|1\rangle$ for each element of a vector, then we obtain a global feature map $\Phi(\mathbf{x}) = \bigotimes_{i=1}^{N} \phi(x_i)$. This can be implemented on quantum circuit by applying a single-qubit RY rotation on each of the N qubits, which is called AE. Starting from the product state $|0\rangle^{\otimes N}$, the quantum state after feature mapping is

$$|\Phi\rangle = \bigotimes_{i=1}^{N} RY(x_i)|0\rangle. \tag{19}$$

In order to meet the requirements of the geometric position of a 2D TN, the encoded (or mapped) data is placed on a two-dimensional lattice according to the original positions of pixels, which is expressed as the product of N order-1 tensors.

In TNML, the data tensors after feature mapping are multiplied by a $(N+1)$-order weight tensor W^l containing the label index l. The next step is to decompose W^l into TN form and optimize the tensor by TN method [8, 17]. Finally, we choose l for which $f(\mathbf{x}) = W^l\Phi(\mathbf{x})$ is largest as the label of input $\mathbf{x}$. While in our TN-inspired QCL, the weight tensor is constructed by a TN-inspired unitary U_{TN} containing trainable parameters, which makes the weight tensor actually a $2N$-order tensor. Then we predict labels based on measurement results of quantum circuit. Usually, the measurement operator set $\{\mathcal{M}_i\}$ with completeness is used for measurement, the index i represents the possible results of measurement. These operators act on the state space of the system to be measured, so a TN-inspired quantum classifier can be interpreted as a contraction of input data tensors, weight tensor, measurement operator tensor and their corresponding conjugate transpose tensor from the TN perspective. After measurement, the circuit will output the probabilities of 2^N different results, the probability of the result i is

$$p_i = \langle\Phi|\mathcal{U}_{TN}^{\dagger}\left(U_j(\theta_k)\right)\mathcal{M}_i\mathcal{U}_{TN}\left(U_j(\theta_k)\right)|\Phi\rangle \tag{20}$$

where $\mathcal{U}_{TN}(U_j(\theta_k))$ represents the quantum circuit composed of a set of unitaries U_j and their trainable parameters θ_k, and the self-adjoint measurement operator $\mathcal{M}_i$ is the feature space projection of the observable $\sigma_z^{\otimes N}$. In our models, given an input $\mathbf{x}$, the output logits E can be obtained by taking a linear combination of the square roots of the probabilities of all results, specifically $E(\mathbf{x},\theta) = \sum_{i=0}^{2^{N-1}-1}\sqrt{p_i} - \sum_{i=2^{N-1}}^{2^N-1}\sqrt{p_i}$, which differs from $f(\mathbf{x})$ in TNML. The meaning of $E(\mathbf{x},\theta)$ is to divide the observation bases equally into two sets and calculate the difference value between the sum of amplitude absolute values over the bases in the two sets. The result of the optimization will maximize one of the sets' sum of amplitude absolute values, which also means maximizing the sum of probabilities of this sets (Note that using the sum of amplitude absolute values makes the network converge more stably than that of probabilities). Thus $E(\mathbf{x},\theta)$ only compares probabilities of two sets that the measurement results occur on. Using a finite number of shots ($\sqrt{p_i} = 0$ for some i) could also normally evaluate $E(\mathbf{x},\theta)$ and make it work on real quantum machines or simulators obtaining probabilities of results by repeated measurements. We show this in section 3.4.

Next, the optimization of weight U_{TN} follows the QCL framework under the classical-quantum hybrid hardware architecture [1], it feeds the logits $E(\mathbf{x},\theta)$ back to classical computer, and then use the adaptive moment estimation (Adam) optimization method to adjust the parameters θ to minimize the difference between the predicted labels and the real ones. In binary classification, the loss function is defined as

$$\mathcal{L} = -\frac{1}{|D|}\sum_{\mathbf{x}\in D}\log\frac{1}{1+\exp\left(-E(\mathbf{x},\theta)\right)} \tag{21}$$

where D represents a batch of data, logits $E(\mathbf{x},\theta)$ is processed by the *sigmoid* function and then used to calculate the cross entropy with the true labels. In multi-class classification, circuits of the same structure containing different parameters are repeated k times to produce k outputs. Its loss function is

$$\mathcal{L} = -\frac{1}{|D|}\sum_{\mathbf{x}\in D}\log\frac{\exp E_c(\mathbf{x},\theta)}{\sum_{i=1}^{k}\exp E_i(\mathbf{x},\theta)} \tag{22}$$

where c is the sample label, $E_i(\mathbf{x},\theta)$ is the output of the ith circuit, which is used to calculate the probability that the predicted label of $\mathbf{x}$ is c through the *softmax* function. Then we calculate the cross entropy of model's probabilities and data's labels.

Each step of training will return an average loss of the batch of data, and the gradient of loss can be calculated exactly using automatic differentiation software for QCL. The parameters are adjusted by Adam algorithm to generate a new quantum circuit for the next training step. This process is iterated to minimize loss and finally obtain the optimal parameters, which correspond to the optimal weight tensor in TNML. Finally, we predict the label of $\mathbf{x}$ as $\mathrm{sign}(E(\mathbf{x},\theta) - \frac{1}{2})$ in binary classification. In multi-class classification, we choose i for which $E_i(\mathbf{x},\theta)$ is largest as the label of input $\mathbf{x}$.

2.4. Encoders: novel variational encoding methods

In order to improve the accuracy performance of QCL, in addition to applying 2D TN to QCL, considering the effect of data encoders also matters. Although AE maps data to high-dimensional Hilbert space, the number of qubits it requires is equal to the classical data dimension, which makes it difficult to use complete

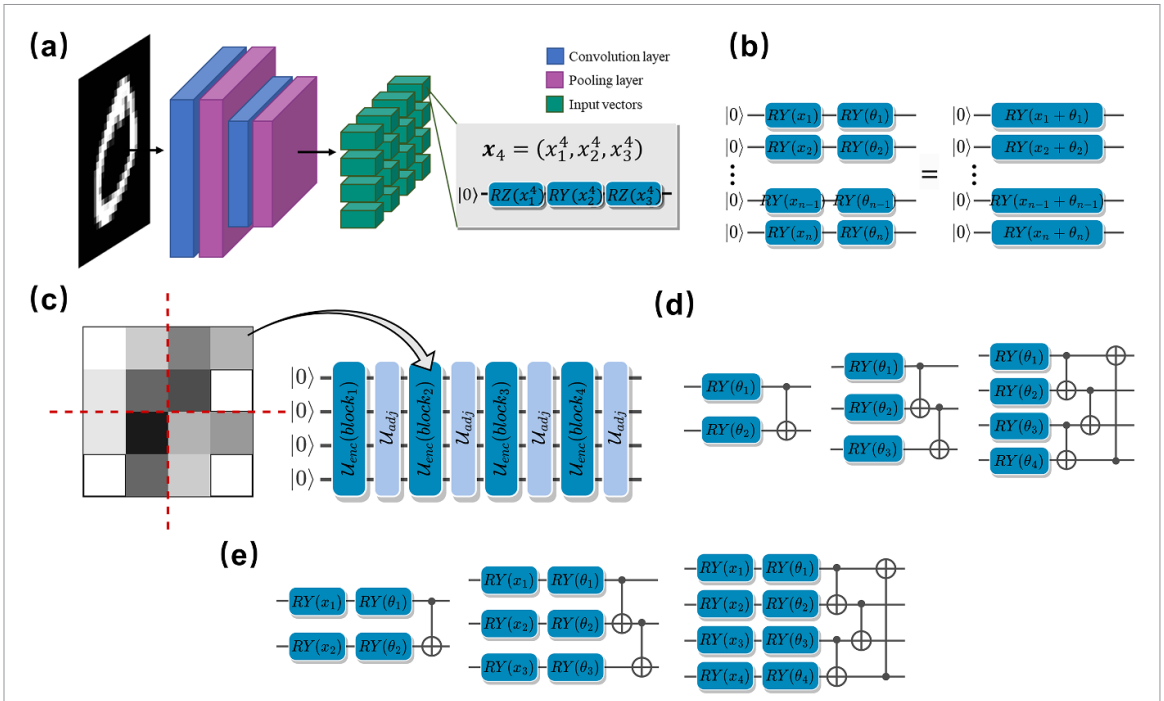

Figure 4. (a) Convolution encoding. The tensor after convolution feature mapping is re-divided into $L \times L$ c-dimensional input vectors, each vector is encoded by c single-qubit rotation on the same qubit to map into a two-dimensional vector. (b) Variational angle encoding. (c) Variational tensor network encoding. The image is divided into several parts of equal size, each part is encoded by a 2D TNQC encoding layer U_{enc}, and an adjustment layer U_{adj} is used to adjust the correlation between pixel blocks. (d) Implementations of three types of multi-qubit unitaries. (e) Implementations of unitaries used for variational tensor network encoding.

data due to the constraints of a limited number of qubits on NISQ hardware. We have to use dimensionality reduction methods to pre-process data before training. Besides, the essence of AE is a product state feature map, which is untrainable. This may result in the mapped data not being in the optimal position in the Hilbert space for classification using a TNQC ansatz. A better choice is to use a variational feature map with trainable parameters.

To address above two issues, we propose a data encoding method based on convolutional feature map, called CE, which naturally conforms to the spatial arrangement of 2D TNQC ansätze. As shown in figure 4(a), the input image data with shape $L_0 \times L_0$ is processed through convolutional and pooling layers to generate a third-order feature tensor with dimensions $L \times L \times c$, where c represents the number of convolutional channels. Instead of flattening this feature tensor, we group the data at the same position in different channels into a c-dimensional vector to inherit the spatial features of the image. Therefore, it can be regarded as a product state in the space $c^{L\times L}$. Next, c single qubit rotations selected from $\{RY, RZ\}$ are applied on each of the $L \times L$ qubits, so the convolutional features are further transformed into a product state in the Hilbert space of dimension $2^{L\times L}$. For 2D image data, adopting a trainable CE method can effectively extract 2D features from multiple channels of the images, which can be continuously adjusted during training to better fit the 2D TNQC classifiers. Meanwhile, the CE reduces the dimensionality of original high-dimensional data to a size that can be used by QNN.

The key to the effectiveness of CE lies in its parameters being learned jointly with those in TNQC during the training process, resulting in a hybrid classical-quantum architecture. It is more desirable to see QNNs perform without the aid of classical network layers, thus eliminating the possibility that the performance of the classifier comes from classical network layers. Following this idea, we introduce trainable parameters into AE, which is called VAE. As shown in figure 4(b), it begins with $|0\rangle$ on each qubit and sequentially applies two single-qubit rotations RY with angles determined by the data x_i and a trainable parameter θ respectively. This is equivalent to applying an RY gate with a rotation angle $x_i + \theta$. Therefore, VAE yields a product state $|\psi\rangle = \bigotimes_{i=1}^{N} RY(x_i + \theta)|0\rangle$, and it is trained together with the 2D TNQC classifier. Note that scaling the normalized data $\mathbf{x} = (x_1, x_2, \ldots, x_N)$ before training can further improve its adaptability.

Although VAE incorporates trainable parameters, it still requires N qubits to encode N-dimensional data which makes it difficult to make full use of the data's power. Moreover, the linear transformation of VAE limits its adaptive capability. Therefore, we propose a VTNE method to act as a scalar between the number of qubits and the data dimension, and to perform a nonlinear transformation. Specifically, the original image is

partitioned into several equally sized 2D image blocks. For each block, a 2D TNQC ansatz layer using special unitaries that we will introduce later is applied to encode it, and these encoding layers are applied to quantum circuit in sequence. This method ensures the expression of spatial correlation in image blocks, and we can also add an adjustment layer between two encoding layers to adjust the correlation between pixel blocks. The two types of layers apply the same kind of 2D TNQC ansatz but with different unitaries, and each of them can be viewed as a 2D TN operator. From this perspective, data of any dimension can be encoded as a TN state [11], and measurement operation on the circuit results in a contraction of the TN state and its conjugate transpose. This encoding method is presented in figure 4(c). In addition, when using VTNE, the encoder circuit architecture and the circuit of QCL ansatz used as trainable weights are inspired by the same TN (1D MPS or a 2D TN). Thus the circuits of VTNE-based QCL models always consist of multiple layers of QMPS or 2D TNQC, and a L-layer circuit forms a TN of $D = 2^L$.

Now, we can integrate our proposed variational encoders and 2D TNQC ansätze on the basis of TNQC supervised learning framework to build several novel 2D TNQC classifiers, which we will show in the section 3.

2.5. Unitary implementations

There are three unitaries used to construct distinct circuit ansätze, and these ansätze can be employed for data encoding or describing weight tensor. For ease of distinction, a TNQC ansatz layer used to describe the weight tensor is referred to as a QNN layer. While in VTNE, TNQC ansätze are also used to construct encoding layers and adjustment layers. Here we introduce unitary implementation in different layers and TNQC ansätze. First, TNQC ansätze in the QNN layer and the adjustment layers use the same unitaries including

$$\begin{aligned}
&U^{[2]}_{n}(\theta) = CNOT_{n,n+1}\overset{n+1}{\underset{i=n}{\otimes}}(RY_i(\theta_i)),\\
&U^{[2]}_{(m,n)}(\theta) = CNOT_{(m,n),(m+1,n)}\overset{m+1}{\underset{i=m}{\otimes}}(RY_{(i,n)}(\theta_{(i,n)})) \text{ or } CNOT_{(m,n),(m,n+1)}\overset{n+1}{\underset{j=n}{\otimes}}(RY_{(m,j)}(\theta_{(m,j)})),\\
&U^{[3]}_{(m,n)}(\theta) = CNOT_{(m+1,n),(m+1,n+1)}CNOT_{(m,n),(m+1,n)}\underset{(i,j)\in\{(m,n),(m+1,n),(m+1,n+1)\}}{\otimes}(RY_{(i,j)}(\theta_{(i,j)})),\\
&U^{[4]}_{(m,n)}(\theta) = \prod_{i=m}^{m+1}CNOT_{(i,n+m-i+1),(i,n-m+i)}\prod_{j=n}^{n+1}CNOT_{(m,j),(m+1,j)}\overset{m+1}{\underset{i=m}{\otimes}}\overset{n+1}{\underset{j=n}{\otimes}}(RY_{(i,j)}(\theta_{(i,j)}))
\end{aligned} \tag{23}$$

Figure 4(d) shows all three multi-qubit unitaries used in different circuit ansätze in these two layers, which consist of single-qubit RY rotations containing trainable parameters and $CNOT_{i,j}$ gates. Where i is control qubit and j is target qubit, it is set to always $i < j$ in two-qubit and three-qubit unitaries to ensure that CNOT gates are in the same direction in all unitaries. The four-qubit unitary is required to apply CNOT gate counterclockwise from the upper left corner of the local 2×2 lattice.

Second, in the encoding layers, we use special unitaries to construct TNQC ansätze compared to the origin ones in QNN or adjustment layer, including

$$\begin{aligned}
&U_{enc}{}^{[2]}_{(n)}(\theta,x) = U^{[2]}_{n}(\theta)\overset{n+1}{\underset{i=n}{\otimes}}(RY_i(x_i)),\\
&U_{enc}{}^{[2]}_{(m,n)}(\theta,x) = U^{[2]}_{(m,n)}(\theta)\overset{m+1}{\underset{i=m}{\otimes}}(RY_{(i,n)}(x_{(i,n)})) \text{ or } U^{(2)}_{[m,n]}(\theta)\overset{n+1}{\underset{j=n}{\otimes}}(RY_{(m,j)}(x_{(m,j)})),\\
&U_{enc}{}^{[3]}_{(m,n)}(\theta,x) = U^{[3]}_{(m,n)}(\theta)\underset{(i,j)\in\{(m,n),(m+1,n),(m+1,n+1)\}}{\otimes}(RY_{(i,j)}(x_{(i,j)})),\\
&U_{enc}{}^{[4]}_{(m,n)}(\theta,x) = U^{[4]}_{(m,n)}(\theta)\overset{m+1}{\underset{i=m}{\otimes}}\overset{n+1}{\underset{j=n}{\otimes}}(RY_{(i,j)}(x_{(i,j)}))
\end{aligned} \tag{24}$$

To be specific, each special unitary for encoding applies an RY rotation on every qubit prior to the original unitary being applied, with the angle of RY being the corresponding pixel value x_i on the 2D lattice location of the qubit. And the RY rotations of different encoding unitaries applied on the same qubit use the same pixel as angle. Figure 4(e) illustrates three multi-qubit unitaries used for VTNE.

2.6. PQN: a novel parallel quantum machine learning method for multi-class classification

Classical neural networks can easily perform multi-class classification tasks, given that our goal is to compare the accuracy performance of quantum and classical classifiers on a fair track, limiting the classification task to binary classification is obviously not fair to classical classifiers. However, quantum classifiers often need to perform multi-class classification tasks with the help of adding a classical dense layer or an MLP classifier,

forming the so-called classical-quantum hybrid classifiers. That's not a good thing because it is difficult to determine which part of the network really plays a role. The classical layer can learn and classify on its own, and sometimes using only the classical part of the hybrid model even outperforms the full hybrid classifier. Using the probabilities of the quantum measurement results or the expectation of each qubit as the logits for classification, we can perform multi-class classification without resorting to classical neural networks, but we will get a poor classification accuracy.

In order to enable quantum classifiers to efficiently perform multi-class classification tasks without the help of classical networks, we propose a parallel quantum machine learning method called PQNs for multi-class classification. For a k-class classification task, we create k quantum circuits having the same architecture. Each circuit has independent parameters for learning and iteration. They can be considered as k 'quantum nodes' of the classifier. These 'quantum nodes' can be executed in parallel on multiple different quantum machines. For an identical input, they will generate k outputs after measurement, which are then used as logits to calculate loss according to equation (22), and parameters in all k quantum nodes are then simultaneously optimized according to the gradient calculated by loss. Such quantum nodes act similarly to the output neurons in the last layer of the MLP. Note that such a multi-class classifier is still composed of pure QNNs.

3. Results

Based on the TNQC supervised learning framework, 9 different 2D TNQC classifiers can be obtained by combining the different encoders and ansätze we propose. However, it is not yet clear the specific performance of these classifiers. We evaluated the accuracy performance of the 2D TNQC classifiers on the most commonly used MNIST benchmark dataset for testing QCL models. Our simulations and experiments are geared towards answering the following research questions (RQs):

- RQ1. Can 2D TNQC ansätze and new data encoding methods improve the accuracy performance of QCL?
- RQ2. Which is the best of the new models built with different ansätze and encoders?
- RQ3 Whether the accuracy performances of the models are affected by some TN features of ansätze?
- RQ4. Can QCL classifiers be used for multi-class classification tasks without the aid of classical dense layers? How does it perform?
- RQ5. How do the best quantum classifiers perform compared to classical classifiers?
- RQ6. Do our models work on a real quantum machine?

In order to answer these questions, we implement the 2D TNQC models' simulations using Tensorflow [32] deep learning framework and Tensorcircuit [33] quantum simulator. Tensorcircuit claims to provide significant speedup by constructing simulator using a TN engine. It is well-compatible with tensorflow, with which enabling automatic differentiation and GPU acceleration, so it is suitable for variational quantum algorithms. The actual experiments are carried out on ibmq_nairobi quantum computer to verify the feasibility of the model on a real quantum machine. Since Tensorcircuit does not provide direct support for IBM hardware, this part is done using Qiskit [34]. And training of the model for the experiments uses FakeNairobi simulator backend provided by it. Note that circuits simulated by Tensorcircuit allow to return the ideal state vector and the exact probabilities of the measurement results, and it automatically computes the exact gradients. While our circuits simulated by Qiskit only estimate the probabilities of the measurement results by repeating the measurement multiple times and counting the frequencies, and use the finite-difference method to compute gradients. Thus logits $E(\mathbf{x}, \theta)$ in section 2.3 can be computed directly on Tensorcircuit from the exact probabilities returned by the simulator, and are evaluated on Qiskit based on the probabilities obtained from finite shots (repeated measurements).

We perform simulations on a server with an 8-core 3.60 GHz Intel(R) Core(TM) i7-7820X CPU and a TITAN RTX GPU. The server has 16 GB RAM and 24 GB VRAM.

We carried out 4 tasks for above RQs, which we will present in the following 4 subsections. All model training is done on the simulators. We list in advance the calculation time and number of qubits required to carry out these tasks in table 1.

We use 16 models of different combinations in section 3.1 while all other tasks only use the VTNE-QisoTNS model. The ansatz we use in section 3.2 varies in the bond dimension D while all other tasks have $D = 2$. We perform a multiclassification task in section 3.3 while all others are binary classification. Our training on FakeNairobi in section 3.4 is based on Qiskit with CPU, while others are based on Tensorcircuit with GPU. Other settings on the dataset and training can be found below. These tasks are huge amount of computation for classical simulators, although the excellent performance of Tensorcircuit greatly accelerates our training process by more than 10 times compared to using any other simulators, they still take some

Table 1. Calculation time and number of qubits in each simulation.

Task	Simulation	Time (s)	Qubits number	Simulation	Time (s)	Qubits number
Section 3.1	AE-QMPS	115	16	CE-QMPS	1473	16
	AE-QSBS	173		CE-QSBS	1543	
	AE-QEPS	144		CE-QEPS	1559	
	AE-QisoTNS	170		CE-QisoTNS	1534	
	VAE-QMPS	173	16	VTNE-QMPS	1462	16
	VAE-QSBS	208		VTNE-QSBS	1671	
	VAE-QEPS	181		VTNE-QEPS	2877	
	VAE-QisoTNS	206		VTNE-QisoTNS	1979	
Section 3.2	$D = 0$	1362	16	$D = 2^3$	5314	16
	$D = 2$	1979		$D = 2^4$	8476	
	$D = 2^2$	3589		—	—	—
Section 3.3	Multiclassification	16 256	16	—	—	—
Section 3.4	Noise resilience	9157	9	—	—	—
	FakeNairobi	2260	7	—	—	—

time. The statistics in table 1 show that simulating TNs with larger bond dimensions (section 3.2), performing multiclassification tasks (section 3.3) and simulating noise model (section 3.4) is very time-consuming. Because they require the simulator to use a large amount of memory to simulate deep circuits that record precise quantum states. Deeper (as D increases) and more (non-parallel multiclassification tasks on the simulator) circuits, as well as the addition of noise channels increase the memory and time consumption of the simulations substantially. The use of quantum hardware would be an important means of solving these problems.

3.1. Classical simulation results of binary classification for RQ1 & 2

In this section, we construct 16 QCL classifiers by combining 4 encoding methods and 4 circuit ansätze under the TNQC supervised learning framework. Among them, encoder baseline and ansätze baseline are the existing QMPS (also known as hardware-efficient ansatz) and basic angle encoding methods, respectively. To answer RQ1 and RQ2, we use a simulator to run quantum classifiers to perform binary classification tasks. The simulation results are used to analyze whether 2D TN and variational encoding help improve accuracy, and to analyze performance differences between classifiers. All simulations below are repeated 10 times with randomly initialized condition.

The number of pixels encoded by the 4 encoding methods is not the same while using the same number of qubits. In order to ensure the fairness of the comparison, we first use AE and VAE on 16 qubits to encode data and test the performance of different ansätze, because they encode the same number of pixels. The simulations are based on MNIST dataset which is a handwritten digital image dataset containing a training set of 60 000 samples and a test set of 10 000 samples, and each sample is a 28×28 grayscale image and belongs to one of 10 classes. We choose three binary classification tasks of increasing difficulty including 01, 27, and 49 classification tasks. These images are resized to 4×4 using area-based resampling method, which allows them to retain two-dimensional connections between pixels and be encoded to 16 qubits. All simulations use identical hyperparameters, the batch size is 100 and Adam optimizer is used with an initial learning rate of $\lambda = 0.01$ and a decay rate of $\alpha = 0.1$. The learning rate is decayed after 15 epochs, and the total number of training epochs is 30.

Table 2 shows the mean test accuracy and standard deviation of 10 simulations with randomly initialized parameters in different tasks. Here, the 'Ansatz' column describes the network architecture. QMPS with AE is chosen as a baseline for the simulations, and all ansätze use the same number of layers fairly in the sense of TN to compare their performance. The 'Encoding' column describes the encoding method. The bold values indicate the best result for each classificaiton task.

The results lead to following conclusions. First, 2D TNQC ansätze can improve the accuracy performance of QCL since 2D TNQC classifiers outperform 1D QMPS classifier with the same encoder in all three tasks of varying difficulty. For instance, in the '2 or 7' classification task, QisoTNS classifier using AE achieves an accuracy 3.06% higher than that of QMPS. Its advantage stems from the fact that the two-dimensional entanglements can capture the correlations between adjacent pixels and image's overall structural information. From the perspective of TN, 2D TN has a higher-dimensional entanglement than 1D MPS with the same bond dimension, which enables it to represent a larger subspace in the Hilbert space. This property

Table 2. Binary classification accuracy on MNIST dataset.

Ansatz	Encoding	0 or 1	2 or 7	4 or 9
QMPS	AE	96.86 ± 0.64	93.48 ± 0.61	80.31 ± 0.37
QMPS	VAE	97.11 ± 0.88	94.86 ± 1.34	80.72 ± 1.57
QSBS	AE	97.34 ± 0.43	96.32 ± 0.19	81.09 ± 0.95
QSBS	VAE	**98.47** ± 0.20	96.56 ± 0.45	**83.14** ± 1.05
QEPS	AE	97.45 ± 0.40	96.62 ± 0.25	81.81 ± 0.67
QEPS	VAE	98.34 ± 0.29	96.89 ± 0.50	82.56 ± 0.77
QisoTNS	AE	96.73 ± 0.59	96.54 ± 0.21	80.79 ± 0.94
QisoTNS	VAE	97.28 ± 0.95	**96.89** ± 0.54	80.42 ± 0.93

endows the 2D TNQC classifier with a larger solution space for learning global optimal parameters. Second, QCL benefits from variational encoding methods since VAE achieves higher accuracy than AE in the same classifier in almost all cases (except QisoTNS), especially in simpler or harder classification tasks. This illustrates the effectiveness of adaptive feature map. Effective improvements can be achieved by simply adding trainable bias to individual data when encoding. Additionally, there are performance differences among 2D TNQC classifiers with the same encoder. This is not only because these ansätze are generated from different TNs, but also because the specific implementations of unitaries that make up the ansätze are also different.

Quantum classifiers using AE and VAE require $O(n)$ qubits to encode n pixels, so we have to compress the images significantly, resulting in a loss of information, which clearly prevents them from being the best classifiers. Also, it may be more effective to extract higher dimensional adaptive features than VAE to transform individual data into adaptive features separately. So next we use CE and VTNE based classifiers on 16 qubits to conduct the 49-classification task which is the most difficult one in 3 tasks done above. Note that in the simulations using CE, the images are not resized, while in the simulations using VTNE, all images are resized to 12×12 to prevent the circuit from being too deep. So we are not going to compare these two encodings to each other. The simulations using CE share the same hyperparameters as the AE simulations. While in VTNE simulations, epoch is set to 10, the batch size is 50, and the learning rate starts to decay from the 8th epoch.

Figure 5 display the average accuracy and loss on training dataset and test dataset during the training process with 10 different random parameter initializations. Analyzing the results, we observe that with the aid of CE, all QC classifiers achieve improved accuracy at least 16.26% owing to the availability of larger dimensions of data and efficient nonlinear transformation. With the same bond dimension, 2D TNQC still performs slightly better than 1D QMPS. Specifically, the test accuracy of QMPS reaches 99.09%, to which QEPS is similar (99.17%), and QSBS has the highest test accuracy of 99.40%. So CE-QSBS is the best classifier in classical-quantum hybrid networks.

We also notice that CE-QisoTNS stabilizes at 100% in accuracy and has the lowest train loss during the training stage, but it suffers from overfitting with a test accuracy of only 99.32%. In order to explore the cause of overfitting and figure out whether CE-QisoTNS has more expressive power comparing to other TNs, we test the overfitting of the model by using L2 regularization at the CE and QisoTNS layers of the network respectively. After training, the training and test accuracy and the difference value Δ between them are as following table 3.

From the results, it can be found that the accuracy difference of the classifiers with CE regularization layer is reduced to a certain extent compared to the model without regularization (overfitting reduction), and the reduction $\Delta_{None} - \Delta_{CE}$ is greater than the reduction $\Delta_{None} - \Delta_{QisoTNS}$ in difference value of the classifiers with the QisoTNS regularization layer compared to the model without regularization, which indicates that the overfitting of the model mainly occurs in the CE layer.

The reason for this situation may be that the classical layer continues to learn after the quantum layer convergence. All our classical quantum hybrid models will have overfitting, and the sequence of their overfitting depends on the convergence speed of the quantum layer. For example, our results show that the training accuracy of CE-QMPS reaches 100% at the 37th epoch, while the training accuracy of CE-QSBS reaches 100% at the 51epoch, this situation is not observed with CE-QEPS. But their test accuracies do not improve further. This shows that after the convergence of the quantum layer, the classical CE layer still adjust the features adaptively to the quantum classifier and finally achieve 100% training accuracy.

According to the above inference, we believe that QisoTNS has certain advantages in the convergence speed. In order to further test whether it has advantages in accuracy, regularization is used on both CE and QisoTNS layers. $\Delta_{CE\&QisoTNS}$ decreases significantly, which indicates that overfitting is mitigated, but the test

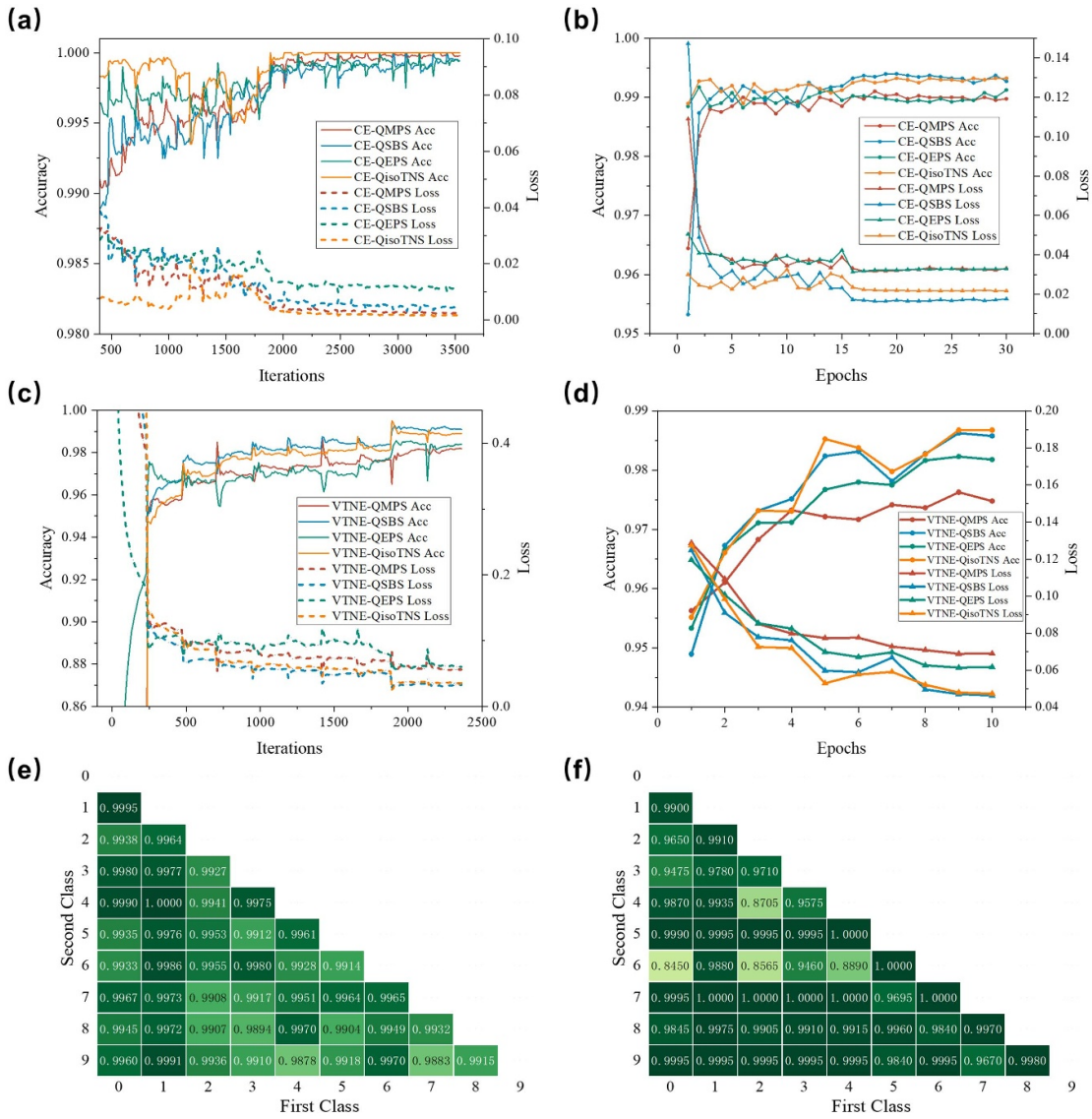

Figure 5. Binary classification results. The training and testing results of different models on MNIST dataset are shown in (a)–(d), here all models are subjected to 10 simulations with random initialization, and their average accuracy and loss are presented, including (a) training results and (b) test results of models applying convolutional encoding, (c) training results and (d) test results of models applying variational tensor network encoding. We present more results including (e) the best test accuracy of VTNE-QisoTNS classifier on all MNIST pairwise subsets and (f) on all Fashion-MNIST pairwise subsets.

Table 3. Accuracy and difference value of CE-QisoTNS when using L2 regularization at different layers.

Regularization layer	None	CE	QisoTNS	CE&QisoTNS
Training accuracy	100%	99.87%	99.93%	99.64%
Test accuracy	99.32%	99.38%	99.30%	99.35%
Difference value Δ	0.68%	0.49%	0.63%	0.29%

accuracy of the model only increases slightly, which is still not enough to exceed CE-QSBS. Therefore, CE-QSBS is still the best hybrid classifier in accuracy.

In the models using VTNE, the 2D TNQC classifiers still have higher performance. The accuracy of QisoTNS classifier is 98.69%, higher than the other three classifiers including QSBS(98.63%), QEPS(98.20%) and QMPS (97.63%). This represents a 17.9% improvement in accuracy compared to AE-QisoTNS. We can see that QEPS performs slightly less well than the other two 2D TNQCs. This could be due to the fact that both QisoTNS and QSBS can be identified as isoTNS (or PEPS), they have similar structures. While QEPS is not a PEPS, its TN has non-vertical indices. Ansätze with similar TN structures perform similarly. Considering that the dimension of the data used is only 12×12, the performance of the VTNE classifier is already excellent. Among them, VTNE-QisoTNS is the best classifier in quantum models.

Table 4. Performance of VTNE-QisoTNS under different virtual bond dimensions of single-layer ansatz.

D	0	2	2^2	2^3	2^4	2^5
Parameters	0	132	264	396	528	—
Ansatz depth	0	17	34	51	68	—
Test accuracy	97.88%	98.69%	98.83%	98.70%	98.61%	—

We test the performance of the QisoTNS classifier on all MNIST pairwise subset classification tasks and achieve a test accuracy of over 99% for almost all classifications. In order not to lose generality, we also perform the same evaluation simulations with another Fashion-MNIST dataset, which is a grayscale image dataset of clothing. It has the same size, format, number of classes, and dataset partitioning rules as the MNIST dataset but is more challenging. The best test accuracy of all pairwise classification tasks is shown in figures 5(e) and (f).

In summary, for RQ1 & 2, we have:

Insight 1. 2D TNQC can improve the accuracy of QCL models, and the variational encoding methods are more effective than the ordinary methods. Under the premise of using the same number of qubits, QisoTNS using TN variational encoding improves the accuracy of AE-QMPS baseline model by 18.38% in 49 classification tasks. VTNE-QisoTNS is the best quantum classifier, CE-QSBS is the best classical-quantum hybrid classifier.

3.2. Classical simulation results of binary classification for RQ3

We want to explore whether the models' accuracy performances are affected by some TN features of ansätze. Following the proposal of [11], we use deep quantum circuit to increase the bond dimension of a TN state to explore the relationship between accuracy and bond dimension. Specifically, starting from the encoded quantum state $\bigotimes_{i=1}^{N} |\phi_i\rangle$, we apply a series of layers $\{U_t\}(t = 1, 2, \ldots, L)$ having the same structure, and finally form a TN of $D = 2^L$. In this process, each layer U_t is a TN operator with $D = 2$, and all operators are contracted to form a TN with a larger bond dimension. Therefore, we can see that compared to classical 2D TNs, quantum computers require only $\log_2 D$ layers of circuits to construct TNs of bond dimension D, which alleviates the memory bottleneck problem that exists in classical TNs [11] and allows the construction of larger scale 2D TNs. We choose VTNE-QisoTNS as an example to measure the performance of the classifier under different virtual bond dimensions D of single ansatz used in adjustment layers by increasing the number of layers of TNQC, where the physical bond dimension of ansatz is 2.

As shown in table 4, even if the ansatz virtual bond dimension is 0 (without adding ansatz), the model can still achieve 97.88% accuracy, which is higher than VTNE-QMPS, because the VTNE encodes the data into an isoTNS. As the bond dimension increases, the total parameters of adjustment layers and depth of single ansatz also increase. When $D = 2^2$, VTNE-QisoTNS reaches the best accuracy, but the accuracy decreases as virtual bond dimension continues to increase, which also happens in classical TNML [19]. This situation is also similar to classical TNML [19]. Situation above shows the accuracy performance of model can be affected by some TN features of ansatz, and increasing the virtual bond dimension of ansatz appropriately can improve the accuracy performance, but this will also increase the circuit depth and the training difficulty. It is difficult to simulate the circuit with $D > 2^4$ in the simulator limited by memory. Note one will not have this issue for quantum computations.

In summary, for RQ3, we have:

Insight 2. The accuracy performance of the model is affected by some TN features of the ansatz. The accuracy of the model will first increase and then decrease as the virtual bond dimension of ansatz increases.

3.3. Classical simulation results of multi-class classification for RQ4 & 5

In order to answer RQ4 and enable quantum classifiers to efficiently perform multi-class classification tasks without the help of classical networks, we combine PQN and VTNE-QisoTNS to construct a multi-class classifier, the model accuracy performance is tested and compared with classical neural networks. The principle of this quantum multi-class classifier is that for an input, the same backup is passed to k QNNs with the same architecture but different trainable parameters, resulting in k results, and the serial number of the largest of the k results is selected as its label. We select all handwritten images containing digits 0–3 from the MNIST dataset. All images are resized to a size of 8×8 to enable normal simulation of the quantum classifier. Such processing is due to the memory limitations of classical simulators, while quantum computers can use full-size images to further improve the model performance, and they support multi-classification

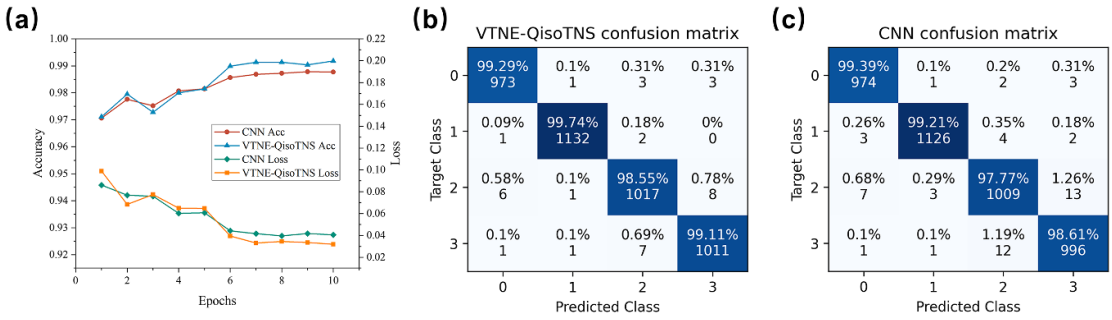

Figure 6. Multi-class classification results. (a) Mean performance of VTNE-QisoTNS and CNN classifier on 10 random initializations on 8×8 image test set. (b) Confusion matrix of VTNE-QisoTNS for MNIST classification. (c) Confusion matrix of CNN for MNIST classification.

tasks with more categories. For comparison, we choose the CNN classifier which contains a convolutional layer, a max pooling layer and a fully connected layer with 512-64-4 neurons. The convolution layer contains three 3×3 kernels with a stride of 1, and it is activated by ReLU. The pooling layer has a pool size of 2×2. The batch size of 10 random initialization training is set to 10, and Adam optimizer is used with a learning rate of 0.01 until the fifth epoch, after which the learning rate is decreased to 0.001.

Figure 6(a) shows the performance of CNN classifier and VTNE-QisoTNS classifier on the same test set. The VTNE-QisoTNS classifier achieves lower test loss and gives a higher test accuracy of up to 99.18%, outperforming the CNN classifier which has a test accuracy of 98.79%. This means that 2D TNQC classifier has certain advantages in performance compared to classical simple classifiers when using identical inputs. In figures 6(b) and (c), we provide the confusion matrices of two classifiers on test set, which show their specific differences in classification.

In summary, for RQ4 & 5, we have:

Insight 3. Based on parallel quantum machine learning method for multi-class classification, quantum classifiers can perform multi-class classification effectively. We use the best VTNE-QisoTNS quantum classifier to build its multi-class version. Compared to classical simple classifiers, the best quantum classifiers can achieve better accuracy performance on a fair track with the same inputs.

3.4. Noise resilience and running on a quantum computer for RQ6

Tensorcircuit simulator can only simulate up to 30 quantum qubits on our hardware and cannot support sufficiently deep circuits (e.g. QisoTNS ansatz with $D = 2^5$). It means that it is hard to further expand the size of 2D TNs and carry out larger-scale tasks using simulator (the largest picture size in our VTNE simulations is only 12×12). And it is difficult to simulate a TN with a higher bond dimension. Some models are very time-consuming using simulators. Therefore, long training time with simulators, difficulty in scaling task sizes and simulating TNs with high bond dimensions are all motivations for running these ansätze on quantum computers. It is necessary to use quantum computers in order to construct larger scale 2D TNQCs for building QNN models with larger data input dimensions and higher accuracy to reduce training time and promote their practical benefits. Therefore, the requirements for evaluating the effectiveness and usability of our methods and models are not only improvements in accuracy performance, but also that the models can function on real quantum machines as mentioned in RQ6. Noise exists on current NISQ machines, so algorithms that can truly execute on them should be able to be noise resilient. QCL is expected to be noise-tolerant for implementation on near-term noisy hardware. In this experiment, we test the impact of noise on the performance of the proposed model by simulating thermal relaxation noise. Subsequently, we train the classifier using a backend that simulates IBM Quantum's real hardware noise model. Finally, the trained model is deployed on a real quantum computer. Due to the limited number of qubits on quantum hardware, here we use the VTNE-QisoTNS classifier for the classification task on 01 dataset of size 3×3 as a minimal example for testing. While further achievement of practical benefits of 2D TNQC quantum models requires NISQ devices with further improvements in qubit numbers and fidelity.

Thermal relaxation describes a non-unitary evolution of a high-energy state system that spontaneously releases energy towards ground state, which originates from the energy exchange between physical qubits and environment. Thermal relaxation noise causes the quantum system to transition from a pure state to a mixed state. For the state ρ of a quantum system, the model describing noise (called channel ε) can be expressed as $\varepsilon(\rho) = \sum_k E_k \rho E_k^\dagger$ using Kraus operator, where $\{E_k\}$ are Kraus operators, which need to satisfy the completeness condition $\sum_k E_k^\dagger E_k = I$, and $\varepsilon(\rho)$ is the quantum state after evolution. The thermal relaxation

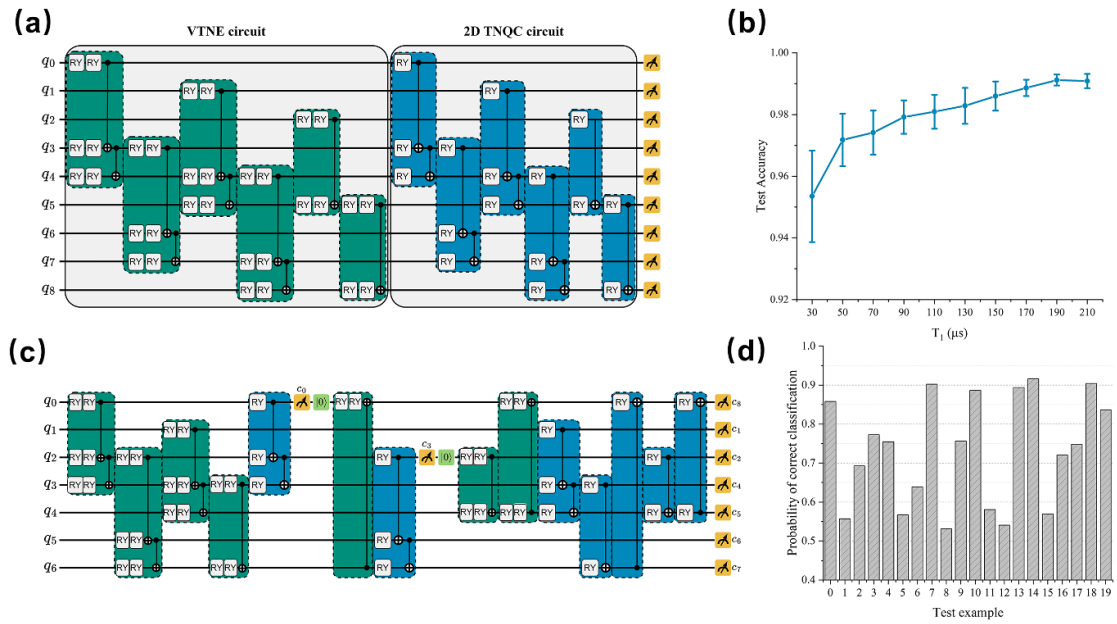

Figure 7. (a) The circuit of VTNE-QisoTNS classifier on a lattice of 3×3 qubits. It consists of a VTNE encoding layer and a 2D TNQC circuit layer. (b) Effect of thermal relaxation noise on the accuracy of VTNE-QisoTNS for 01 classification. The mean accuracy remains above 97% for thermal relaxation noise with $T_1 > 50\mu s$ indicating that the model exhibits certain noise resilience. (c) The qubit-efficient version of the VTNE-QisoTNS classifier, qubits q_0 and q_2 are measured in the middle of the circuit and reset to $|0\rangle$ for reuse. (d) The correct classification probability of a single experiment for each test samples.

channel ε can be implemented by applying the dephasing channel [35] ε_d and the amplitude damping channel [35] ε_a after each unitary, i.e. $\varepsilon(\rho) = \varepsilon_a(\varepsilon_d(\rho))$. The Kraus operators of the amplitude damping channel are

$$E_0 = \begin{bmatrix} 1 & 0 \\ 0 & \sqrt{1-p_a} \end{bmatrix}, E_1 = \begin{bmatrix} 0 & \sqrt{p_a} \\ 0 & 0 \end{bmatrix}. \tag{25}$$

While the Kraus operators of the dephasing channel are

$$E_1 = \begin{bmatrix} \sqrt{1-p_d} & 0 \\ 0 & \sqrt{1-p_d} \end{bmatrix}, E_2 = \begin{bmatrix} \sqrt{p_d} & 0 \\ 0 & -\sqrt{p_d} \end{bmatrix}. \tag{26}$$

The parameters p_a and p_d are used to characterize the strength of a quantum channel, with larger values indicating stronger channel effects and faster degradation of the fidelity of quantum information. In addition, the thermal relaxation channel can be characterized by the unitary gate duration T_g, the coherence time T_1 and dephasing time T_2 of the qubits. Specifically, p_a and p_d can be expressed as

$$P_a = 1 - e^{-\frac{t_g}{T_1}}, P_d = \frac{1}{2} \times \left(1 - e^{-\left(\frac{t_g}{T_2} - \frac{t_g}{2T_1}\right)}\right). \tag{27}$$

Our noise model applies thermal relaxation channel to each qubit of each unitary gate in the circuit. To assess the model's practical performance on real quantum hardware, we fix the duration of each three-qubit gate at 450 ns and each two-qubit gate at 250 ns, and set T_2 to $0.7 \times T_1$. We test the model's performance with T_1 ranging from 30 μs to 210 μs in 20 μs increments, which represents varying levels of noise impact from high to low. The parameter settings are in line with the real situation of current quantum devices. The classifier used for testing includes a VTNE layer and a 2D TNQC layer, its specific circuit is shown in figure 7(a).

Figure 7(b) shows the model's performance on 01 classification task, we see that the best test accuracy decreases as T_1 decreases, which is due to the increasing noise. As a benchmark, the mean test accuracy reaches 99.76% under ideal simulated condition. At $T_1 = 210\mu s$, $p_a = 0.0021$ and $p_d = 0.0010$ can be calculated at this time, the test accuracy reaches 99.08%; while at $T_1 = 130\,\mu s$, which is the current average level of hardware on IBM Quantum [36], the test accuracy is 98.28%, decreasing by only 1.48% compared to the ideal state. This indicates that the model can work on current quantum hardware. As the noise increases, at $T_1 = 30\,\mu s$, for the thermal relaxation noise with $p_a = 0.0149$ and $p_d = 0.0069$, the model's accuracy remains at 95.35%, proving that the model has some level of noise resilience.

Table 5. Classification results of a single experiment versus voting from 501 experiments.

Method	Label
Sampling of points	0 0 1 1 1 1 0 1 0 0 0 0 0 0 0 1 1 0 0 1
Single experiment result	0 0 **0** 1 **0 0** 0 1 0 0 0 0 **1** 0 0 1 1 0 0 1
Majority vote from 501 results	0 0 1 1 1 1 0 1 0 0 0 0 0 0 0 1 1 0 0 1

Further, we demonstrate the usability of the model on a real quantum machine. Training a model directly with a quantum machine is difficult, with system scheduling and shutting down network connections leading to interruptions in the training process. Therefore, we used Qiskit to train the model in the FakeNairobi backend simulated with the noise model based on the real hardware ibmq_nairobi in IBM Quantum and deployed the classifier to the ibmq_nairobi quantum computer. Considering the limited resources of quantum computing and in order to follow the general rules of machine learning, we randomly selected 120 data to construct a mini-dataset and divided it into training and test set with a ratio of 5:1. Such a setup reduces the information leakage and thus more accurately reflects to the performance of the model. A test set of 20 examples is used to determine accuracy. Using only 100 shots per execution to train for our circuits having 2^9 possible results, our model achieves 100% test accuracy on FakeNairobi backend. This shows that our framework and logits $E(\mathbf{x}, \theta)$ can work using repeated measurements with finite shots (although $100 < 2^9$). However, to make the result more stable, the model we finally used on the real machine was trained using 8192 shots per execution. For each test example in a real quantum computer, the circuit is set to 8192 shots to obtain results. Due to the limited number of available qubits (only 7), we adopt the method proposed in [5] and use measure and reset operations to prepare a qubit-efficient scheme for the circuit in figure 7(a). To be specific, the qubits q_0 and q_3 in the original circuit perform the measurement operation immediately after applying the last unitary acting on them respectively and are reset to $|0\rangle$, they will be reused as q_8 and q_2. The specific circuit is shown in figure 7(c). The circuit correctly predicts the class of 16 out of 20 test data, which is not as expected. One reason for this is that the reset operation has a long duration of up to 5696 ns, making it the longest operation on the hardware, which brings thermal relaxation noise beyond expectations. Another reason is that the noise model of the simulation backend is not exactly the same as the noise on a real quantum computer. The trained model has good robustness to the noise of the simulator (achieving a test accuracy of 100%), but it cannot fully absorb the effects of other different noises.

To mitigate these two problems, we use the method proposed in [4, 5] to repeat experiments on quantum circuits and predict labels, one could take a majority vote from 501 experimental results of each test example to obtain the most probable label.

As shown in table 5, by voting after multiple experiments, all the test samples are correctly classified compared with only 16 labels correctly predicted by a single experiment (the bold values indicate incorrectly predicted samples). Figure 7(d) shows the correct classification probability of a single experiment for each of the 20 test samples, and the average is 0.731. The final classification accuracy of test samples is 100%.

In summary, for RQ6, we have:

Insight 4. 2D TNQC classifiers have some level of noise resilience, which enables them to run and function on real quantum machines.

4. Discussion

Many QNNs have been proposed, but they are difficult to achieve the accuracy of classical neural networks. In this paper, motivated by the huge improvement in model accuracy of PEPS classifiers compared to MPS classifiers in classical TNML, we are committed to applying 2D TN to quantum circuits to extend the performance boundary of quantum machine learning.

Based on our goals, we want to solve two questions: (1) how to encode 2D TN into quantum circuit for being applied to QCL just as it does in TNML? (2) How to use 2D TN to improve the accuracy performance of QCL to meet or even exceed that of classical classifiers?

We come up with solutions to the questions and make the following innovations: (1) it is the first time to use rigorous mathematical proofs to construct quantum circuits generated by 2D TN operators (including three 2D TNQC ansätze: QSBS, QEPS and QisoTNS) to evolve the product states into different 2D TN states, which realizes encoding 2D TN into quantum circuits. These ansätze can be used for quantum machine learning or solving quantum many-body problems. (2) In this paper, we construct a TNQC supervised learning framework that transfers TNML from classical to quantum. Based on this framework, we can apply any circuits encoded by TN to quantum machine learning. (3) We present an adaptive variational encoding method, which can be combined with convolutional feature map (CE) for hybrid neural networks, and with

Table 6. The performance comparison between the proposal classifier against others on MNIST.

Acc (%) Model \ Subset	{0,1}	{2,7}	{3,6}	{3,7}	{4,9}	{0,3,6}	{0,2,9}	{0,1,2,3}
QF-Net [37]	—	—	98.27	—	—	90.40	—	—
QCNN [9, 38]	98.7	—	—	—	—	—	—	—
TRVQC [39]	—	—	—	83.73	—	—	—	—
QMPS [40]	98.77	—	—	—	—	—	—	—
QTTN [5]	99.6	95.7	97.2	94.4	88.0	—	—	—
QTNN [4]	99.79	97.64	—	—	—	—	—	—
QMERA [4]	99.87	98.86	—	—	—	—	—	—
VQTN [6]	—	96.92	97.92	—	—	—	83.36	—
Hybrid TN-VQC [41, 42]	—	—	99.44	—	—	98.0	—	—
MERA [43]	—	—	—	—	—	—	—	93.0
Our VTNE-QisoTNS	**99.95**	**99.08**	**99.80**	**99.17**	**98.78**	**99.25**	**99.17**	**99.18**

two-dimensional tensor networks (VTNE) for QNNs. (4) In this paper, we construct and implement 9 novel 2D TN-inspired QCL models based on the TNQC supervised learning framework using the above ansätze (including QSBS, QEPS and QisoTNS) and encoders (including VAE, CE and VTNE). (5) In this paper, we propose a parallel quantum machine learning method PQN for multi-class classification, and implement a quantum multi-class classifier. PQN allows quantum classifiers to complete multi-class classification tasks with high accuracy without using any classical network layer.

To test the effectiveness of the solutions and the performance of the models, we conduct a wide range of classical simulations and actual experiments which are geared towards answering the following five questions: (1) can 2D TNQC ansätze and new data encoding methods improve the accuracy performance of QCL? (2) Which is the best of the new models built with different ansätze and encoders? (3) Whether the accuracy performances of the models are affected by some TN features of ansätze? (4) Can QCL classifiers be used for multi-class classification tasks without the aid of classical dense layers? How does it perform? (5) How do the best quantum classifiers perform compared to classical classifiers? (6) Do our models work on a real quantum machine?

For these simulations and experiments on the MNIST and Fashion-MNIST datasets, we present the results of all models and baseline in section section 3, demonstrating the significant improvement of our approach on baseline. Not only that, our best model achieves the state-of-the-art accuracy performance of the current QNNs, which extends the performance boundary of quantum machine learning in the field of image classification. And the best model beats the classical simple CNN on a fair track with the same inputs. Here, we present in table 6 the performance of various QNN classifiers for image classification, including quantum versions of MLPs and convolutional neural networks (i.e. QF-Net and QCNN), and several TN-inspired QNNs. The results of these classifiers contain the best accuracies obtained from simulations using other automatic differentiation software (Pennylane [44], Tensorflow, etc). Due to the different datasets used, the difficulty of different classification tasks varies. The data show that our proposed 2D TNQC classifiers achieve more accurate classification results in the same classification task. The VTNE-QisoTNS achieves the state-of-the-art performance of TN-inspired quantum classifiers on the MNIST dataset, and is among the best results of QML methods reported [45].

Additionally, the models are proved to have some resilience to thermal relaxation noise, and a trained model is successfully executed on ibmq_nairobi quantum computer. It is worth noting that 2D TNQC may run directly on current several quantum hardware with the same geometric structure [46], which may be a potential advantage.

Analyzing these results, we get the following **Insight:** (1) 2D TNQC can improve the accuracy of QCL models, and the variational encoding methods are more effective than the ordinary methods. Under the premise of using the same number of qubits, QisoTNS using TN variational encoding improves the accuracy of AE-QMPS baseline model by 18.38% in 49 classification tasks. VTNE-QisoTNS is the best quantum classifier, CE-QSBS is the best classical-quantum hybrid classifier. (2) The accuracy performance of the model is affected by some TN features of the ansatz. The accuracy of the model will first increase and then decrease as the virtual bond dimension of ansatz increases. (3) Based on parallel quantum machine learning method for multi-class classification, quantum classifiers can perform multi-class classification effectively. We use the best VTNE-QisoTNS quantum classifier to build its multi-class version. Compared to classical simple classifiers, the best quantum classifiers can achieve better accuracy performance on a fair track with

the same inputs. (4) The 2D TNQC classifiers have some level of noise resilience, which enables them to run and function on real quantum machines.

However, there are some problems with 2D TNQC classifiers. First, the quantum version of the 2D TN we implement is not entirely equivalent to its classical counterpart. The difference lies in the different degrees of freedom of the parameters between the unitaries applied on the quantum circuit and the tensors in a classical TN. Using a universal quantum unitary gate can theoretically make the two equivalent, but it is not clear whether it will improve the classifiers' performance, and this will result in an exponential increase in the depth of quantum circuit with respect to the number of qubits, as pointed out in [47]. Second, to exploit the advantages of the two-dimensional structure, 2D TNQC ansätze requires deeper circuits than 1D TNQC. At the same time, in order to encode more values, our encoders use N (qubit) $\times$ M (layer) to encode $M \times N$ values, which leads to the further deepening of the circuit. Deep circuit leads to higher training costs and may make it more sensitive to noise. And the model is difficult to fully deploy on current NISQ machines. Fortunately, these problems will be solved with the increase of available qubits and the improvement of fidelity in quantum computers.

These issues highlight that there is still much interesting work in researching 2D TNQC. First, the performance of 2D TNQC classifiers can still be further improved, as the effects of unitary gate implementations, mixed use of different ansätze, and the sharing of weights between layers on the classifier have not been studied. These works can be combined with quantum circuit architecture search [48–51], one could build a search space [48] or a supernet [49] based on 2D TNQC ansätze to automatically seek a near-optimal classifier architecture. Second, through sensitivity analysis, the architecture and performance of the model can be optimized to be more robust, and the learning ability of 2D TNQC on graphical spatial features can be further studied. Third, it could be explored the relationship between the model performance and TN, including using TN to explain differences in performance across ansätze, discussing the connection between learning accuracy and some properties of TN ansätze (e.g. bond dimensions or entanglement entropy), and so on. In addition, exploring TN-inspired quantum circuits based on hardware topology may be an effective attempt to balance performance and training costs. Such research will facilitate the emergence of more practical quantum machine learning models. The exploration of TNQC construction will further promote the development of computational problem-solving methods and variational quantum eigensolvers [52] in two-dimensional quantum many-body systems.

Data availability statement

All data that support the findings of this study are included within the article (and any supplementary files).

Acknowledgments

The authors acknowledge the financial support from Major Science and Technology Projects in Henan Province, China, Grant No. 221100210600.

Conflict of interest

The authors have no conflicts to disclose.

Author contributions

Zhihui Song and Zheng Shan designed the research plan. Zhihui Song performed the simulations and experiments, and wrote the manuscript. Zheng Shan and Jinchen Xu assisted to analyze data and improved simulation designs. Xin Zhou, and Xiaodong Ding directed the studies and writing. All authors reviewed the final manuscript.

ORCID iDs

Zhihui Song https://orcid.org/0009-0006-6250-7896
Xiaodong Ding https://orcid.org/0000-0001-9947-4035

References

[1] Mitarai K, Negoro M, Kitagawa M and Fujii K 2018 Quantum circuit learning *Phys. Rev.* A **98** 032309
[2] Benedetti M, Lloyd E, Sack S and Fiorentini M 2019 Parameterized quantum circuits as machine learning models *Quantum Sci. Technol.* **4** 043001

[3] Cerezo M et al 2021 Variational quantum algorithms Nat. Rev. Phys. 3 625–44
[4] Grant E, Benedetti M, Cao S, Hallam A, Lockhart J, Stojevic V, Green A G and Severini S 2018 Hierarchical quantum classifiers npj Quantum Inf. 4 65
[5] Huggins W, Patil P, Mitchell B, Whaley K B and Stoudenmire E M 2019 Towards quantum machine learning with tensor networks Quantum Sci. Technol. 4 024001
[6] Huang R, Tan X and Xu Q 2021 Variational quantum tensor networks classifiers Neurocomputing 452 89–98
[7] Dborin J, Barratt F, Wimalaweera V, Wright L and Green A G 2022 Matrix product state pre-training for quantum machine learning Quantum Sci. Technol. 7 035014
[8] Araz J Y and Spannowsky M 2022 Classical versus quantum: comparing tensor-network-based quantum circuits on large hadron collider data Phys. Rev. A 106 062423
[9] Cong I, Choi S and Lukin M D 2019 Quantum convolutional neural networks Nat. Phys. 15 1273–8
[10] Liu J-G, Zhang Y-H, Wan Y and Wang L 2019 Variational quantum eigensolver with fewer qubits Phys. Rev. Res. 1 023025
[11] Ran S-J 2020 Encoding of matrix product states into quantum circuits of one- and two-qubit gates Phys. Rev. A 101 032310
[12] Haghshenas R, Gray J, Potter A C and Chan G K-L 2022 Variational power of quantum circuit tensor networks Phys. Rev. X 12 011047
[13] Wolf M M, Verstraete F, Hastings M B and Cirac J I 2008 Area laws in quantum systems: mutual information and correlations Phys. Rev. Lett. 100 070502
[14] Eisert J, Cramer M and Plenio M B 2010 Colloquium: area laws for the entanglement entropy Rev. Mod. Phys. 82 277–306
[15] Verstraete F, Murg V and Cirac J I 2008 Matrix product states, projected entangled pair states, and variational renormalization group methods for quantum spin systems Adv. Phys. 57 143–224
[16] Orús R 2014 A practical introduction to tensor networks: matrix product states and projected entangled pair states Ann. Phys. 349 117–58
[17] Stoudenmire E M and Schwab D J 2016 Supervised learning with tensor networks Proc. 30th Int. Conf. on Neural Information Processing Systems (Curran Associates Inc.) pp 4806–14
[18] Liu D, Ran S-J, Wittek P, Peng C, García R B, Su G and Lewenstein M 2019 Machine learning by unitary tensor network of hierarchical tree structure New J. Phys. 21 073059
[19] Cheng S, Wang L and Zhang P 2021 Supervised learning with projected entangled pair states Phys. Rev. B 103 125117
[20] Pan F and Zhang P 2022 Simulation of quantum circuits using the big-batch tensor network method Phys. Rev. Lett. 128 030501
[21] Orús R 2019 Tensor networks for complex quantum systems Nat. Rev. Phys. 1 538–50
[22] Schuch N, Wolf M M, Verstraete F and Cirac J I 2008 Simulation of quantum many-body systems with strings of operators and Monte Carlo tensor contractions Phys. Rev. Lett. 100 040501
[23] Glasser I, Pancotti N, August M, Rodriguez I D and Cirac J I 2018 Neural-network quantum states, string-bond states, and chiral topological states Phys. Rev. X 8 011006
[24] Mezzacapo F, Schuch N, Boninsegni M and Cirac J I 2009 Ground-state properties of quantum many-body systems: entangled-plaquette states and variational Monte Carlo New J. Phys. 11 083026
[25] Zaletel M P and Pollmann F 2020 Isometric tensor network states in two dimensions Phys. Rev. Lett. 124 037201
[26] Lecun Y, Bottou L, Bengio Y and Haffner P 1998 Gradient-based learning applied to document recognition Proc. IEEE Inst. Electr. Electron. Eng. 86 2278–324
[27] Xiao H, Rasul K and Vollgraf R 2017 Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms (arXiv:1708.07747)
[28] Schuch N, Wolf M M, Verstraete F and Cirac J I 2007 Computational complexity of projected entangled pair states Phys. Rev. Lett. 98 140506
[29] Wei Z-Y, Malz D and Cirac J I 2022 Sequential generation of projected entangled-pair states Phys. Rev. Lett. 128 010607
[30] Schön C, Hammerer K, Wolf M M, Cirac J I and Solano E 2007 Sequential generation of matrix-product states in cavity QED Phys. Rev. A 75 032311
[31] Schön C, Solano E, Verstraete F, Cirac J I and Wolf M M 2005 Sequential generation of entangled multiqubit states Phys. Rev. Lett. 95 110503
[32] Abadi M et al 2016 TensorFlow: a system for large-scale machine learning Proc. 12th USENIX Conf. on Operating Systems Design and Implementation (USENIX Association) pp 265–83
[33] Zhang S-X et al 2023 TensorCircuit: a quantum software framework for the NISQ era Quantum 7 912
[34] Aleksandrowicz G et al 2023 Qiskit: an open-source framework for quantum computing (https://doi.org/10.5281/zenodo.2573505)
[35] Nielsen M A and Chuang I L 2012 Quantum Computation and Quantum Information 10th Anniversary edn (Cambridge University Press)
[36] IBM Quantum Experience 2021 IBM Research (available at: www.research.ibm.com/ibm-q/) (Accessed 13 April 2023)
[37] Jiang W, Xiong J and Shi Y 2021 A co-design framework of neural networks and quantum circuits towards quantum advantage Nat. Commun. 12 579
[38] Hur T, Kim L and Park D K 2022 Quantum convolutional neural network for classical data classification Quantum Mach. Intell. 4 3
[39] Peddireddy D, Bansal V and Aggarwal V 2023 Classical simulation of variational quantum classifiers using tensor rings Appl. Soft Comput. 141 110308
[40] Wang K, Xiao L, Yi W, Ran S-J and Xue P 2021 Experimental realization of a quantum image classifier via tensor-network-based machine learning Photon. Res. 9 2332
[41] Chen S Y-C, Huang C-M, Hsing C-W and Kao Y-J 2020 Hybrid quantum-classical classifier based on tensor network and variational quantum circuit (arXiv:2011.14651)
[42] Chen S Y-C, Huang C-M, Hsing C-W and Kao Y-J 2021 An end-to-end trainable hybrid classical-quantum classifier Mach. Learn.: Sci. Technol. 2 045021
[43] Lazzarin M, Galli D E and Prati E 2022 Multi-class quantum classifiers with tensor network circuits for quantum phase recognition Phys. Lett. A 434 128056
[44] Bergholm V et al 2018 PennyLane: automatic differentiation of hybrid quantum-classical computations (arXiv:1811.04968)
[45] Wei L, Liu H, Xu J, Shi L, Shan Z, Zhao B and Gao Y 2023 Quantum machine learning in medical image analysis: a survey Neurocomputing 525 42–53
[46] Arute F et al 2019 Quantum supremacy using a programmable superconducting processor Nature 574 505–10
[47] Möttönen M, Vartiainen J J, Bergholm V and Salomaa M M 2004 Quantum circuits for general multiqubit gates Phys. Rev. Lett. 93 130502

[48] Zhang S-X, Hsieh C-Y, Zhang S and Yao H 2021 Neural predictor based quantum architecture search *Mach. Learn.: Sci. Technol.* **2** 045027
[49] Du Y, Huang T, You S, Hsieh M-H and Tao D 2022 Quantum circuit architecture search for variational quantum algorithms *npj Quantum Inf.* **8** 62
[50] Ostaszewski M *et al* 2021 Reinforcement learning for optimization of variational quantum circuit architectures *Advances in Neural Information Processing Systems* vol 34 pp 18182–94
[51] He Z, Chen C, Li L, Zheng S and Situ H 2022 Quantum architecture search with meta-learning *Adv. Quantum Technol.* **5** 2100134
[52] Kandala A, Mezzacapo A, Temme K, Takita M, Brink M, Chow J M and Gambetta J M 2017 Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets *Nature* **549** 242–6