QTN-VQC: AN END-TO-END LEARNING FRAMEWORK FOR QUANTUM NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

The advent of noisy intermediate-scale quantum (NISQ) computers raises a crucial challenge to design quantum neural networks for fully quantum learning tasks. To bridge the gap, this work proposes an end-to-end learning framework named QTN-VQC, by introducing a trainable quantum tensor network (QTN) for quantum embedding on a variational quantum circuit (VQC). The architecture of QTN is composed of a parametric tensor-train network for feature extraction and a tensor product encoding for quantum encoding. We highlight the QTN for quantum embedding in terms of two perspectives: (1) we theoretically characterize QTN by analyzing its representation power of input features; (2) QTN enables an end-to-end parametric model pipeline, namely QTN-VQC, from the generation of quantum embedding to the output measurement. Our experiments on the MNIST dataset demonstrate the advantages of QTN for quantum embedding over other quantum embedding approaches.

1 INTRODUCTION

The state-of-the-art machine learning (ML), particularly based on deep neural networks (DNN), has enabled a wide spectrum of successful applications ranging from the everyday deployment of speech recognition (Deng et al., 2013) and computer vision (Sermanet et al., 2014) through to the frontier of scientific research in synthetic biology (Jumper et al., 2021). Despite rapid theoretical and empirical progress in DNN based regression and classification (Goodfellow et al., 2016), DNN training algorithms are computationally expensive for many new scientific applications, such as new drug discovery (Smalley, 2017), which requires computational resources that are beyond the computational limits of classical hardwares (Freedman, 2019). Fortunately, the imminent advent of quantum computing devices opens up new possibilities of exploiting quantum machine learning (QML) (Biamonte et al., 2017; Schuld et al., 2015; Schuld & Petruccione, 2018; Schuld & Killoran, 2019) to improve the computational efficiency of ML algorithms in the new scientific domains.

Although the exploitation of quantum computing devices to carry out QML is still in its initial exploratory stages, the rapid development in quantum hardware has motivated advances in quantum neural networks (QNN) to run in noisy intermediate-scale quantum (NISQ) devices (Preskill, 2018; Huggins et al., 2019; Huang et al., 2021; Kandala et al., 2017). A NISQ device means that not enough qubits could be spared for quantum error correction, and the imperfect qubits have to be directly used at the physical layer. Even though, a compromised QNN approach is proposed by employing hybrid quantum-classical models that rely on the optimization of variational quantum circuits (VQC) (Benedetti et al., 2019; Mitarai et al., 2018). The resilience of the VQC based models to certain types of quantum noise errors and high flexibility concerning coherence time and gate requirements (McClean et al., 2018) admit many practical implementations of QNN on NISQ devices (Chen et al., 2020b; Yang et al., 2021; Du et al., 2020; 2021). One notable limitation in the current QNN training pipeline is that the quantum embedding is not fully realizable in a quantum computer, which may impede the learning of the QNN. Hence, this work proposes QTN-VQC to enable an end-to-end trainable QNN, including data embedding to quantum measurements, that are easily realizable in quantum devices, where QTN stands for the quantum tensor network for generating quantum embedding.

As shown in Figure 1, our QNN builds a unitary linear operator that consists of three main components: (1) quantum embedding generation; (2) variational quantum circuit; (3) measurement.

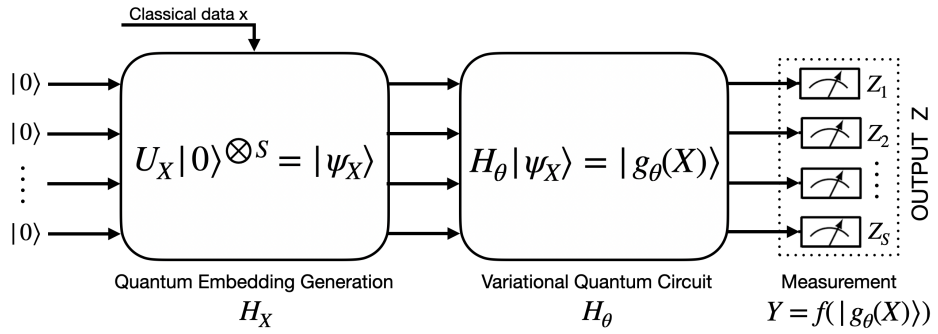


Figure 1: *An illustration of QNN based on VQC.*

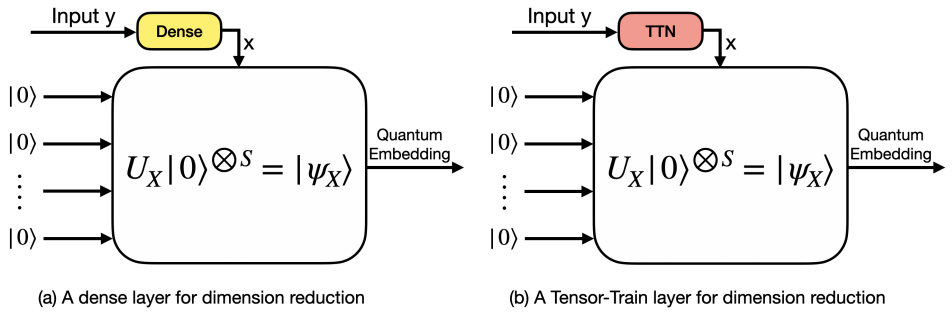


Figure 2: *Different paradigms for quantum embedding. (a) a dense layer is used to generate low-dimensional feature X from a high-dimensional one Y; (b) a TTN is used for dimension reduction.*

Quantum embedding generation, also known as quantum encoding, applies a fixed unitary linear operator H_X transforming classical data X to quantum states $|\psi_X\rangle$ in a Hilbert space. This step is an important aspect of designing quantum algorithms that directly impact the entire computation cost of VQC and owns a characteristic of quantum superposition. Moreover, the VQC comprising of learnable Pauli rotation gates composes a unitary transformation where the property of quantum entanglement is consumed. The model parameters of VQC should be optimized by using variants of gradient descent algorithms during the training process. Those parametric quantum gates of VQC are similar to the weights assigned to DNN, and such quantum circuits have been justified to be resilient to quantum noises (Farhi et al., 2014; Kandala et al., 2017; McClean et al., 2016). Besides, the measurement aims at projecting the output of quantum states $|g_\theta(X)\rangle$ to the classical output Z, which is used for further post-processing.

This work focuses on quantum embedding generation because it is quite related to the practical usage in machine learning applications in terms of computational cost and representation capability of classical input features. In particular, we design a novel quantum tensor network (QTN) for quantum embedding generation. More specifically, the QTN consists of a tensor-train network (TTN) for dimension reduction and a quantum tensor encoding framework for outputting quantum embeddings. The dimension reduction is a necessary procedure before the quantum encoding because only a small number of qubits could be supported on available NISQ computers at this moment. A typical approach for dimension reduction relies on a classical fully-connected layer, also known as a dense layer, and aims to convert high-dimensional input vectors $\mathbf{y}$ into low-dimensional features $\mathbf{x}$. However, since a dense layer cannot be physically mapped on a quantum computer, much extra overhead from classical-quantum conversion has to be considered during the end-to-end training pipeline because much computational cost can be incurred by frequently communicating between classical and quantum devices.

This work investigates the generation of quantum embedding for high-dimensional classical data. As shown in Figure 2 (b), one of our **contribution** is to leverage a tensor train network (TTN) to replace the dense layer in Figure 2 (a). The benefits of applying TTN arise from two aspects: (1) a TTN can

maintain the representation power of the dense layer, which will be justified in our theorems; (2) TTN is a tensor network and can be flexibly placed in quantum computers, which enables an end-to-end training process fully conducted in a quantum computer. Moreover, in this work, a tensor product encoding (TPE) is delicately designed for generating quantum embedding, which bridges the relationship between classical data $\mathbf{x}$ and the quantum state $|\mathbf{x}\rangle$; Besides, we further investigate the representation of QTN-VQC in terms of model size and non-linear activation function used in TTN. We denote a QTN as the combination of TTN and TPE, and utilize QTN-VQC as a genuine end-to-end training framework for QNN.

2 Related Work

The work (Schuld & Petruccione, 2018; Biamonte et al., 2017; Dunjko & Briegel, 2018) demonstrate that VQC shows great promise in surpassing the performance of classical ML. Prominent examples of VQC based models include quantum approximate optimization algorithm (QAOA) (Farhi et al., 2014), and quantum circuit learning (QCL) (Mitarai et al., 2018). Various architectures and geometries of VQC have been shown in tasks ranging from image classification (Henderson et al., 2020; Chen et al., 2020a; Kerenidis et al., 2020) to reinforcement learning (Chen et al., 2020b).

As for quantum embedding, basis encoding is the process of associating classical input data in the form of binary strings with the computational basis state of a quantum system (Leymann & Barzen, 2020). Similarly, amplitude encoding is a technique involving encoding data into the amplitudes of a quantum state (Soklakov & Schack, 2006). Unfortunately, the computational cost of both quantum embedding and amplitude encoding becomes exponentially expensive with the increasing number of qubits (Schuld & Killoran, 2019). A new technique of angle embedding makes use of the quantum gates to generate quantum states (Fu et al., 2011), but it cannot deal with the high-dimensional feature inputs. Therefore, this work investigates the use of TTN for dimension reduction followed by a TPE for generating quantum embedding.

In particular, this work employs the TTN for dimensionality reduction. The TTN model based on TT decomposition in neural networks was first proposed in (Oseledets, 2011), and it could be flexibly extended the convolutional neural network (CNN) (Garipov et al., 2016) and recurrent neural network (RNN) (Tjandra et al., 2017). The empirical study of TTN on machine learning tasks shows that TTN is capable of maintaining the DNN baseline results (Qi et al., 2020a; Yu et al., 2017; Yang et al., 2017; Jin et al., 2020). However, to our best knowledge, no existing works have applied TTN to QML.

3 Notations

We denote $\mathbb{R}^m$ as a m-dimensional real coordinate space, and $\mathbb{R}^{I_1 \times I_2 \times \cdots I_K}$ refers to a space of K-order tensors. The symbol $\mathcal{W} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_K}$ represents a K-order multi-dimensional tensor in $\mathbb{R}^{I_1 \times I_2 \times \cdots I_K}$, and the symbols $\mathbf{v} \in \mathbb{R}^m$ and $\mathbf{W} \in \mathbb{R}^{I \times J}$ represent a vector and a matrix, respectively.

For the notations of quantum computing, $\forall\, \mathbf{v} \in \mathbb{R}^m$, the symbol $|\mathbf{v}\rangle$ denotes a quantum state associated with a 2^m-dimensional vector in a Hilbert space. Particularly, $|0\rangle = [1 \;\; 0]^T$ and $|1\rangle = [0 \;\; 1]^T$.

The quantum gate $R_Y(\theta)$ means a Pauli rotation Y operator as defined in Eq. (1), which implies a qubit rotates the Bloch sphere around the Y axis by a given angle θ.

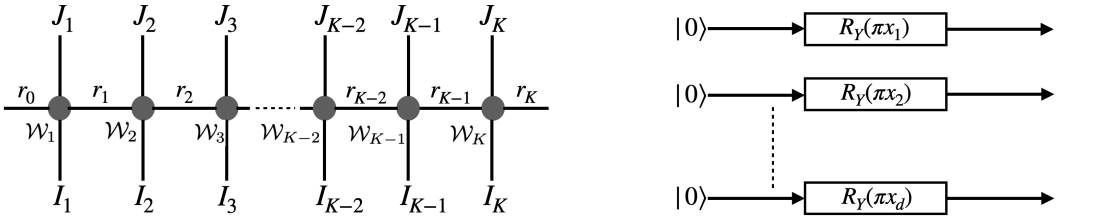
$$R_Y(\theta) = \begin{bmatrix} \cos\frac{\theta}{2} & -i\sin\frac{\theta}{2} \\ -i\sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix} \tag{1}$$

Moreover, the operator $\otimes$ is a tensor product. Given the vectors $\mathbf{v}_i \in \mathbb{R}^m$, the expression $\otimes_{i=1}^m \mathbf{v}_i$ is defined as:

$$\otimes_{i=1}^m \mathbf{v}_i = \mathbf{v}_1 \otimes \mathbf{v}_2 \otimes \cdots \otimes \mathbf{v}_m, \tag{2}$$

which results in 2^m-dimensional vector. Similarly, the symbol $|0\rangle^{\otimes S}$ means a tensor product of S quantum states of $|0\rangle$. Furthermore, for a scalar v, the quantum state $|v\rangle$ can be written as:

$$|v\rangle = \cos v|0\rangle + \sin v|1\rangle = \begin{bmatrix} \cos v \\ \sin v \end{bmatrix}. \tag{3}$$



(a) Tensor-Train Network: given a set of TT-ranks $\{r_1, r_2, \ldots, r_K\}$, a circle represents a core tensor and each line is related to a dimension.

(b) Tensor Product Encoding: $R_Y(\pi x)$ denotes the Pauli-Y rotation gate

Figure 3: *A demonstration of quantum tensor network for quantum embedding.*

4 QTN-VQC: Our Proposed End-to-End Learning Framework

This section introduces our proposed end-to-end learning framework, namely QTN-VQC in this work. As shown in Figure 3, the QTN model includes two components (a) TTN and (b) TPE, which will be separately introduced in Section 4.1 and Section 4.2. Moreover, Figure 4 illustrates the framework of VQC, and Section 4.3 is devoted to discussing the details of VQC.

4.1 Tensor Train Network for Dimension Reduction

We leverage TTN (Novikov et al., 2015) for the dimension reduction of input features. TTN relies on the TT decomposition (Oseledets, 2011) and has been commonly employed in machine learning tasks like speech processing (Qi et al., 2020b) and computer vision (Yang et al., 2017). The TT decomposition assumes that given a set of TT-ranks $\{r_0, r_1, ..., r_K\}$, a K-order tensor $\mathcal{W} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_K}$ is factorized into the multiplication of 3-order tensors $\mathcal{X}_k \in \mathbb{R}^{r_{k-1} \times r_k \times I_k}$. In more detail, given a set of indices $\{i_1, i_2, ..., i_K\}$, $\mathcal{X}(i_1, i_2, ..., i_K)$ is decomposed as:

$$\mathcal{X}(i_1, i_2, ..., i_K) = \prod_{k=1}^{K} \mathcal{X}_k(i_k), \tag{4}$$

where $\forall i_k \in [I_k]$, $\mathcal{X}_k \in \mathbb{R}^{I_k \times r_{k-1} \times r_k}$, and $\mathcal{W}_k(i_k) \in \mathbb{R}^{r_{k-1} \times r_k}$. Since $r_0 = r_1 = 1$, the term $\prod_{k=1}^{K} \mathcal{X}_k(i_k)$ is a scalar value.

TTN employs the TT decomposition in a dense layer and is explicitly demonstrated in Figure 3 (a). In more detail, for an input tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_K}$ and an output tensor $\mathcal{Y} \in \mathbb{R}^{J_1 \times J_2 \times \cdots \times J_K}$, we achieve

$$\begin{aligned}
\mathcal{Y}(j_1, j_2, ..., j_K) &= \sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_K=1}^{I_K} \mathcal{W}\left((i_1, j_1), (i_2, j_2), ..., (i_K, j_K)\right) \mathcal{X}(i_1, i_2, ..., i_K) \\
&= \sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_K=1}^{I_K} \left(\prod_{k=1}^{K} \mathcal{W}_k(i_k, j_k) \right) \cdot \prod_{k=1}^{K} \mathcal{X}_k(i_k) \\
&= \prod_{k=1}^{K} \left(\sum_{i_k=1}^{I_k} \mathcal{W}_k(i_k, j_k) \mathcal{X}_k(i_k) \right) \\
&= \prod_{k=1}^{K} \mathcal{Y}_k(j_k),
\end{aligned} \tag{5}$$

where $\mathcal{X}_k(i_k) \in \mathbb{R}^{r_{k-1} \times r_k}$, and $\mathcal{Y}_k(j_k) \in \mathbb{R}^{r_{k-1} \times r_k}$ which results in a scalar $\prod_{k=1}^{K} \mathcal{Y}_k(j_k)$ because of the ranks $r_0 = r_1 = 1$; $\mathcal{W}((i_1, j_1), (i_2, j_2), ..., (i_K, j_K))$ is closely associated with $\mathcal{W}(m_1, m_2, ..., m_K)$ as defined in Eq. (4), if each index $m_k = i_k \times j_k$ is set. The multi-dimensional tensor $\mathcal{W}$ is decomposed into the multiplication of 4-order tensors $\mathcal{W}_k \in \mathbb{R}^{J_k \times I_k \times r_{k-1} \times r_k}$. A non-linear activation function, e.g., $\tanh(\cdot)$, is imposed upon the tensor $\mathcal{Y}$. Compared with a dense layer with $\prod_{k=1}^{K} I_k J_k$ parameters, a TTN owns as few as $\sum_{k=1}^{K} r_k r_{k-1} I_k J_k$ trainable parameters.

When a TTN is utilized for the dimension reduction, the high-dimensional input vector $\mathbf{x} \in \mathbb{R}^I$ is first reshaped into a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_K}$, and then we can represent $\mathcal{X}$ as a TT format that goes through TTN. The outputs of TTN can be converted back to a tensor $\mathcal{Y} \in \mathbb{R}^{J_1 \times J_2 \times \cdots \times J_K}$, which is further reshaped to a lower dimensional vector $\mathbf{y} \in \mathbb{R}^J$. Here, we define $\prod_{k=1}^{K} I_k = I$ and $\prod_{k=1}^{K} J_k = J$.

Eq. (5) suggests that TTN is a multi-dimensional extension of a dense layer, where the trainable weight matrix of a dense layer is changed to the learnable core tensors. Additionally, many empirical studies demonstrate that a TTN is capable of maintaining the baseline results of the dense layer (Qi et al., 2020b; Yang et al., 2017; Novikov et al., 2015; Qi et al., 2020a). More significantly, since TTN can be flexibly mapped to a quantum circuit, the quantumness inherent in TTN brings great advantages over other architectures like the dense layer. In other words, although TTN is treated classically, it is possible to substitute an equivalent quantum circuit for TTN when more qubits become available. It implies that QTN-VQC stands for a genuine end-to-end QNN training architecture on a quantum computer.

4.2 Tensor Product Encoding

In this subsection, we first introduce Theorem 1, and then we derive our TPE associated with the circuits in Figure 3 (b).

Theorem 1. *Given the classical vector $\boldsymbol{x} = [x_1, x_2, ..., x_d]^T \in \mathbb{R}^d$, a TPE as shown in Figure 3 (b) can result in a quantum state $|\boldsymbol{x}\rangle$ with the following complete vector representation as:*

$$\left(\otimes_{i=1}^{d} R_Y(2x_i)\right)|0\rangle = \begin{bmatrix}\cos x_1\\ \sin x_1\end{bmatrix} \otimes \begin{bmatrix}\cos x_2\\ \sin x_2\end{bmatrix} \otimes \cdots \otimes \begin{bmatrix}\cos x_d\\ \sin x_d\end{bmatrix} = |\boldsymbol{x}\rangle. \tag{6}$$

Proof. Since each element x_i in the vector $\mathbf{x}$ can be written as $|x_i\rangle = \cos x_i|0\rangle + \sin x_i|1\rangle$, the quantum state $|\mathbf{x}\rangle$ can be written as:

$$|\mathbf{x}\rangle = \begin{bmatrix}\cos x_1\\ \sin x_1\end{bmatrix} \otimes \begin{bmatrix}\cos x_2\\ \sin x_2\end{bmatrix} \otimes \cdots \otimes \begin{bmatrix}\cos x_d\\ \sin x_d\end{bmatrix}. \tag{7}$$

When the vector $\mathbf{x}$ goes through the quantum tensor network, which implies the following as:

$$R_Y(2x_i)|0\rangle = cosx_i|0\rangle + sinx_i|1\rangle = x_i. \tag{8}$$

The preceding equation, in turn, implies that

$$\left(\otimes_{i=1}^{d} R_Y(2x_i)\right)|0\rangle = \begin{bmatrix}\cos x_1\\ \sin x_1\end{bmatrix} \otimes \begin{bmatrix}\cos x_2\\ \sin x_2\end{bmatrix} \otimes \cdots \otimes \begin{bmatrix}\cos x_d\\ \sin x_d\end{bmatrix} = |\mathbf{x}\rangle. \tag{9}$$

□

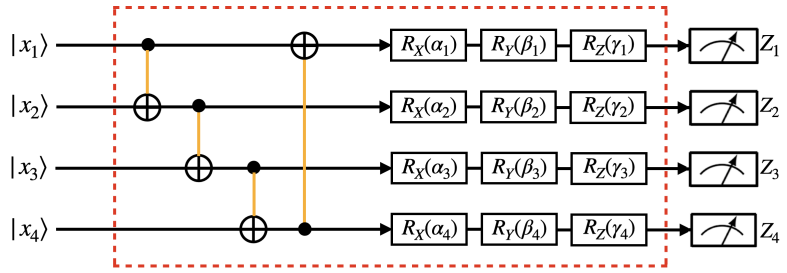
Theorem 1 builds a connection between the vector $\mathbf{x}$ and the quantum state $|\mathbf{x}\rangle$, and the resulting $|\mathbf{x}\rangle$ is taken as the quantum embedding as the inputs to VQC. Since $\otimes_{i=1}^{d} R_Y(2x_i)$ is a reversely unitary linear operator, there is no information loss incurred during the stage of quantum encoding. Furthermore, if the input is multiplied with a constant $\frac{\pi}{2}$, we obtain the following term as:

$$\left(\otimes_{i=1}^{d} R_Y(\pi x_i)\right)|0\rangle = \begin{bmatrix}\cos(\pi x_1)\\ \sin(\pi x_1)\end{bmatrix} \otimes \begin{bmatrix}\cos(\pi x_2)\\ \sin(\pi x_2)\end{bmatrix} \otimes \cdots \otimes \begin{bmatrix}\cos(\pi x_d)\\ \sin(\pi x_d)\end{bmatrix},$$

which corresponds to Figure 3 (b).

4.3 The Framework of Variational Quantum Circuit

The framework of VQC is shown in Figure 4 (a), where 4 qubit wires are taken into account, and the CNOT gates aim at mutually entangling the channels such that $|x_1\rangle$, $|x_2\rangle$, $|x_3\rangle$ and $|x_4\rangle$ lie in the same entanglement state. The Pauli rotation gates $R_X(\cdot)$, $R_Y(\cdot)$ and $R_Z(\cdot)$ with learnable parameters $(\alpha_1, \beta_1, \gamma_1)$, $(\alpha_2, \beta_2, \gamma_2)$, $(\alpha_3, \beta_3, \gamma_3)$, $(\alpha_4, \beta_4, \gamma_4)$ are built to set up the learnable part. Being similar to the rotation operator of $R_Y(\alpha)$, $R_X(\beta)$ and $R_Z(\gamma)$, which are defined in Figure 4



(a) A parametric model: the dashed square indicates repeated model with CNOT gates for entangling quantum states, and $R_X(\alpha)$, $R_Y(\beta)$, $R_Z(\gamma)$ represent Pauli rotation X,Y, Z operators with free parameters α, β, γ

$$CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \qquad R_X(\alpha) = \begin{bmatrix} \cos\frac{\alpha}{2} & -i\sin\frac{\alpha}{2} \\ -i\sin\frac{\alpha}{2} & \cos\frac{\alpha}{2} \end{bmatrix}$$

Pauli rotation X

$$R_Y(\beta) = \begin{bmatrix} \cos\frac{\beta}{2} & -\sin\frac{\beta}{2} \\ \sin\frac{\beta}{2} & \cos\frac{\beta}{2} \end{bmatrix} \qquad R_Z(\gamma) = \begin{bmatrix} \exp(-i\frac{\gamma}{2}) & 0 \\ 0 & \exp(i\frac{\gamma}{2}) \end{bmatrix}$$

Pauli rotation Y Pauli rotation Z

(b) Matrix representation for the Quantum gates used in the variational quantum circuit

Figure 4: *A framework of variational quantum circuit.*

(b), are separately associated with x- and z- rotations by the given angles of β and γ. Besides, the quantum circuits in the dash square can be repeatedly copied to compose a deeper architecture. The outputs of VQC are connected to the measurement which projects the quantum states into a certain quantum basis that become classical data $\mathbf{z} = [z_1, z_2, z_3, z_4]^T$.

As for the end-to-end training paradigm for QTN-VQC, the learnable parameters come from the VQC and TTN models, and they should be updated by applying the back-propagation algorithm based on the Adam optimizer. Given d quantum wires and H depths, there are totally $3dH$ trainable parameters for VQC. Consequently, there are $\sum_{k=1}^{K} r_{k-1}r_kI_kJ_k + 3dH$ parameters for QTN-VQC. On the other hand, the Dense-TPE-VQC model possesses more model parameters than QTN-VQC ($\prod_{k=1}^{K} r_{k-1}r_kI_kJ_k + 3dH$ vs. $\sum_{k=1}^{K} r_{k-1}r_kI_kJ_k + 3dH$).

5 Characterizing Representation Power of QTN-VQC

This section focuses on analyzing the representation power of QTN-VQC. As shown in Figure 5, given d qubits and a target quantum state $|\mathbf{z}\rangle = \otimes_{i=1}^{d}|z_i\rangle$, since H_X and H_θ are linear operators, the representation power of QTN-VQC is determined by how TTN can approximate the classical vector $H_X^{-1}H_\theta^{-1}|\mathbf{z}\rangle$. To understand the expressiveness of TTN, we first start with the discussion on the expressive capability of Dense-TPE-VQC (a dense layer is taken for dimension reduction) and then generalize it to QTN-VQC. Based on the universal approximation theorem (Cybenko, 1989; Barron, 1994) for a feed-forward neural network, we derive the following theorem as:

Theorem 2. *Given a target vector $H_X^{-1}H_\theta^{-1}|\mathbf{z}\rangle$, there exists a feed-forward neural network f_{dense} with a dense layer connecting to d qubits, then*

$$|| H_X^{-1}H_\theta^{-1}|\mathbf{z}\rangle - f_{dense}(\mathbf{y}) ||_1 \leq \frac{1}{\sqrt{d}}C, \tag{10}$$

where the activation function $\tanh(\cdot)$ is imposed upon the dense layer, and C is a constant associated with the target vector $H_X^{-1}H_\theta^{-1}|\mathbf{z}\rangle$.

Since TTN is a compact TT representation of a dense layer, by modifying Theorem 2 for TTN, we can also derive the upper bound on the approximation error as follows:

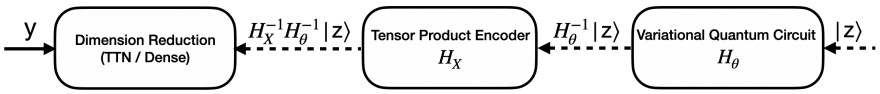


Figure 5: *An illustration of analyzing the representation power of QTN-VQC.*

Theorem 3. *Given a target vector $H_X^{-1}H_\theta^{-1}|\boldsymbol{z}\rangle$, there exists a TTN, denoted as f_{TTN}, with a TT layer connecting to d qubits, then*

$$|| \, H_X^{-1}H_\theta^{-1}|\boldsymbol{z}\rangle - f_{TTN}(\boldsymbol{y}) \, ||_1 \leq \prod_{k=1}^{K} \frac{1}{\sqrt{d_k}} C, \tag{11}$$

where $\prod_{k=1}^{K} d_k = d$, the non-linear activation function $\tanh(\cdot)$ is imposed upon the TTN model, K denotes the multi-dimensional order, C is a constant associated with the target vector $H_X^{-1}H_\theta^{-1}|\boldsymbol{z}\rangle$.

Comparing the two upper bounds, it is observed that TTN can attain an identical upper bound as the dense layer on the approximation error because $\prod_{k=1}^{K} d_k = d$. That implies that TTN can at least maintain the representation power of a dense layer. Besides, the number of qubits d is a key factor determining the upper bound on the approximation error. However, d is a small fixed number on a NISQ device, and a larger number of qubits d is expected to further improve the representation power of QTN-VQC.

6 Experiments and Results

This section aims at the empirical study of QTN-VQC as the following points:

1. Evaluate and explain the performance of the QTN-VQC models
2. Simulate the experiments with the noiseless and noisy quantum circuits, respectively
3. To corroborate our theorem on the QTN-VQC based representation power, the non-linear activation function and the number of qubits are two key factors that we need to consider

6.1 Experimental setups

We assess our QTN-VQC based end-to-end learning system on the standard MNIST. MNIST is a dataset for the task of 10 digit classification, where there are 50000 and 10000 28×28 image data assigned for training and testing, respectively. Moreover, the image data are separately reshaped into 784 dimensional input vectors. Dense-TPE-VQC and PCA-TPE-VQC are taken as our experimental baselines to compare with our QTN-VQC model. Dense-VQC denotes that a dense layer is used for dimension reduction, and PCA-TPE-VQC refers to using principal component analysis (PCA) to extract low-dimensional features before training a VQC.

As for the experiments of QTN-VQC, the image data are reshaped into 3-order $7 \times 16 \times 7$ tensors. Given the set of TT ranks $\{1, 2, 2, 1\}$, the image data are represented as the TT format according to Eq. (4) before going through the TTN model. Since 8 qubits are used for the quantum encoding, the output of TTN needs to configure the tensor format as $2 \times 2 \times 2$, which results in 8 dimensional output vectors. Besides, the model parameters of QTN-VQC are randomly initialized based on the Gaussian distribution, and the back-propagation algorithm is applied to train the models. The activation function $\tanh(\cdot)$ is used for the hidden layers of TTN.

To be consistent with QTN-VQC, the weight of the dense layer for Dense-TPE-VQC is configured as the shape of 784×8. Although Dense-TPE-VQC is a hybrid classical-quantum model, the training process of Dense-VQC can also be set as an end-to-end pipeline and the weights of the dense layer are updated during the training stage. The activation function $\tanh(\cdot)$ is also used for the dense layer. On the other hand, PCA is employed to reduce the feature dimension to 8, and the resulting low-dimensional features are further encoded into quantum states. Consequently, PCA-TPE-VQC admits the VQC parameters solely to be updated during the training stage.

Moreover, 6 VQC layers are constructed to form a deep model, and the outputs of the VQC model are connected to 10 classes with a non-trainable matrix. The back-propagation algorithm based on the Adam optimizer (Kingma & Ba, 2014) with a learning rate of 0.001 is employed for the model training. The loss of cross-entropy (CE) is utilized as the objective function during the training stage, and it is also taken as the metric to evaluate the model performance. We leverage the tools of Pennylane (Bergholm et al., 2018) and PyTorch (Paszke et al., 2019) to simulate the model performance. In particular, we separately simulate the model performance with noiseless quantum circuits and noisy quantum circuits corrupted by quantum noises from IBM Quantum machines.

6.2 Experimental Results of Noiseless Quantum Circuit

Table 1 shows the final results of the models on the test dataset. QTN-VQC owns much fewer model parameters than Dense-TPE-VQC (328 vs. 6416) and attains even higher classification accuracy than Dense-VQC (87.12% vs. 84.64%) and lower loss values than Dense-TPE-VQC (0.4792 vs. 0.6079). However, PCA-TPE-VQC with solely 18 trainable VQC parameters attains the worst performance by all metrics, which implies that a trainable quantum embedding is of significance to boost experimental performance. Although our empirical results cannot reach the state-of-the-art classification performance of classical ML algorithms, our empirical results demonstrate the advantages of QTN-VQC over the PCA-TPE-VQC and Dense-TPE-VQC counterparts. With the development of more powerful quantum devices supporting more qubits, the representation power of QTN-VQC can be boost and better better experimental results could be attained.

Models	Params	CE	Acc (%).
PCA-TPE-VQC	144	0.6079	81.45 ± 1.02
Dense-TPE-VQC	6416	0.5136	84.64 ± 0.73
QTN-VQC	328	0.4792	$\mathbf{87.12} \pm \mathbf{0.51}$

Table 1: Comparison of model performance on the MNIST test dataset.

6.3 Experimental Results of Noisy Quantum Circuit

To empirically validate the effectiveness of our proposed VQC algorithm, we proceed with the simulation of the practical experiments with noisy quantum circuits. More specifically, we follow an established noisy circuit experiment with the NISQ device suggested by (Chen et al., 2020b). One major advantage of the setups is to observe the robustness and preserve the quantum advantages of a deployed VQC with physical settings being close to quantum processing unit (QPU) experiments without an executive queuing time.

As for the detailed setup, we first use an IBM Q 20-qubit machine to collect channel noise in the real scenario for a deployed VQC and upload the machine noise into our Pennylane-Qiskit simulator (denoted as Acc_{q20}. We provide a depolarizing noisy circuit simulation (denoted as Acc_{depo}) based on a depolarizing channel attained from (Nielsen & Chuang, 2002) with a noise level of 0.1. As shown in Table 2, our proposed QTN-VQC outperforms the other VQC baseline results in the condition of noisy quantum circuits. In particular, the quantum noise collected from the IBM Q 20-qubit machine brings about a serious performance degradation.

Table 2: Comparison of model performance on the MNIST test dataset.

Models	Params	Acc_{q20} (%)	Acc_{depo} (%)
PCA-TPE-VQC	144	78.23 ± 1.34	80.12 ± 1.17
Dense-TPE-VQC	6416	81.32 ± 1.22	83.09 ± 1.04
QTN-VQC	328	$\mathbf{85.09} \pm \mathbf{1.09}$	$\mathbf{86.23} \pm \mathbf{1.07}$

6.4 Further Discussions

The above experimental results show the advantages of QTN-VQC over Dense-TPE-VQC and PCA-TPE-VQC in the scenarios with noiseless and noisy quantum circuits. Next, we will further discuss

the representation power of QTN-VQC based on two factors: (1) the activation function used in TTN; (2) the number of qubits.

6.4.1 THE ACTIVATION FUNCTION USED IN TTN

Table 3 compares the results of QTN-VQC with and without the activation function $\tanh(\cdot)$. Our simulation on noiseless quantum circuits shows that the non-linear activation function can bring more performance gain than a linear one. It also corresponds to the universal approximation property for QTN-VQC in Theorem 3.

Models	Params	CE	Acc (%).
QTN-VQC ($\tanh$)	328	0.4792	87.12 ± 0.51
QTN-VQC (linear)	328	0.4958	86.16 ± 0.65

Table 3: Comparing performance of QTN-VQC with and without activation function.

6.4.2 THE NUMBER OF QUBITS

Finally, we investigate the effects of the number of qubits on the performance of QTN-VQC by decreasing the qubits from 8 to 5. Accordingly, the model size is reduced from 328 to 274. Our experiments show that the baseline performance of QTN-VQC is degraded in terms of a higher CE value (0.4912 vs. 0.4792) and a lower accuracy (85.35% vs. 87.12%).

Models	Params	CE	Acc (%).
QTN-VQC (8 qubits)	328	0.4792	87.12 ± 0.51
QTN-VQC (5 qubits)	274	0.4912	85.35 ± 0.62

Table 4: Comparing performance of QTN-VQC with fewer qubits.

6.5 SUMMARY OF EXPERIMENTAL RESULTS

Our experiments on the MNIST dataset justify the following points:

1. Our proposed QTN-VQC exhibits the performance advantages over Dense-TPE-VQC and PCA-TPE-VQC in our simulation with noiseless and noisy quantum circuits
2. Activation function is important to enhance the representation power of QTN-VQC
3. More qubits are expected to boost the baseline performance of QTN-VQC

In particular, points 2 and 3 directly corroborate Theorem 3 for the representation power of TTN, where the number of qubits can lower the upper bound on the approximation error, and the non-linear activation ensures the universal approximation property.

7 CONCLUSIONS

This work proposes a genuine end-to-end learning framework for quantum neural network based on QTN-VQC. QTN consists of a TTN for dimension reduction and a TPE framework for generating quantum embedding. The TTN model is a compact representation of a dense layer, and it can be physically mapped on a quantum computer. Our proposed TPE encodes the classical inputs as quantum states without any information loss. Our theorem on the representation of QTN-VQC shows that the number of qubits is inversely related to the approximation error of QTN-VQC and the non-linear activation plays an important role. Our experiments compare our proposed QTN-VQC with Dense-TPE-VQC and PCA-TPE-VQC, and our simulated results demonstrate that QTN-VQC obtains better experimental performance than the other two counterparts with both noiseless and noisy quantum circuits. Besides, our simulation results justify our theorem on the representation power of QTN-VQC. We will open source our code and pre-trained models for future studies.

REFERENCES

Andrew R Barron. Approximation and Estimation Bounds for Artificial Neural Networks. *Machine learning*, 14(1):115–133, 1994.

Marcello Benedetti, Erika Lloyd, Stefan Sack, and Mattia Fiorentini. Parameterized Quantum Circuits as Machine Learning Models. *Quantum Science and Technology*, 4(4):043001, 2019.

Ville Bergholm, Josh Izaac, Maria Schuld, Christian Gogolin, M Sohaib Alam, Shahnawaz Ahmed, Juan Miguel Arrazola, Carsten Blank, Alain Delgado, Soran Jahangiri, et al. Pennylane: Automatic Differentiation of Hybrid Quantum-Classical Computations. *arXiv preprint arXiv:1811.04968*, 2018.

Jacob Biamonte, Peter Wittek, Nicola Pancotti, Patrick Rebentrost, Nathan Wiebe, and Seth Lloyd. Quantum Machine Learning. *Nature*, 549(7671):195–202, 2017.

Samuel Yen-Chi Chen, Chih-Min Huang, Chia-Wei Hsing, and Ying-Jer Kao. Hybrid Quantum-Classical Classifier Based on Tensor Network and Variational Quantum Circuit. *arXiv preprint arXiv:2011.14651*, 2020a.

Samuel Yen-Chi Chen, Chao-Han Huck Yang, Jun Qi, Pin-Yu Chen, Xiaoli Ma, and Hsi-Sheng Goan. Variational Quantum Circuits for Deep Reinforcement Learning. *IEEE Access*, 8:141007–141024, 2020b.

George Cybenko. Approximation by Superpositions of A Sigmoidal Function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

Li Deng, Jinyu Li, Jui-Ting Huang, Kaisheng Yao, Dong Yu, Frank Seide, Michael Seltzer, Geoff Zweig, Xiaodong He, Jason Williams, et al. Recent Advances in Deep Learning for Speech Research at Microsoft. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 8604–8608, 2013.

Yuxuan Du, Min-Hsiu Hsieh, Tongliang Liu, and Dacheng Tao. Expressive power of parametrized quantum circuits. *Physical Review Research*, 2(3):033125, 2020.

Yuxuan Du, Min-Hsiu Hsieh, Tongliang Liu, Dacheng Tao, and Nana Liu. Quantum noise protects quantum classifiers against adversaries. *Physical Review Research*, 3(2):023153, 2021.

Vedran Dunjko and Hans J Briegel. Machine learning & artificial intelligence in the quantum domain: A review of recent progress. *Reports on Progress in Physics*, 81(7):074001, 2018.

Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A Quantum Approximate Optimization Algorithm. *arXiv preprint arXiv:1411.4028*, 2014.

David H Freedman. Hunting for New Drugs with AI. *Nature*, 576(7787):S49–S53, 2019.

Yangguang Fu, Mingyue Ding, and Chengping Zhou. Phase Angle-Encoded and Quantum-Behaved Particle Swarm Optimization Applied to Three-Dimensional Route Planning for UAV. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 42(2):511–526, 2011.

Timur Garipov, Dmitry Podoprikhin, Alexander Novikov, and Dmitry Vetrov. Ultimate tensorization: compressing convolutional and FC layers alike. *arXiv preprint arXiv:1611.03214*, 2016.

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep Learning*, volume 1. MIT Press, 2016.

Maxwell Henderson, Samriddhi Shakya, Shashindra Pradhan, and Tristan Cook. Quanvolutional neural networks: powering image recognition with quantum circuits. *Quantum Machine Intelligence*, 2(1):1–9, 2020.

Hsin-Yuan Huang, Michael Broughton, Masoud Mohseni, Ryan Babbush, Sergio Boixo, Hartmut Neven, and Jarrod R McClean. Power of data in quantum machine learning. *Nature communications*, 12(1):1–9, 2021.

William Huggins, Piyush Patil, Bradley Mitchell, K Birgitta Whaley, and E Miles Stoudenmire. Towards quantum machine learning with tensor networks. *Quantum Science and technology*, 4 (2):024001, 2019.

Xuanyu Jin, Jiajia Tang, Xianghao Kong, Yong Peng, Jianting Cao, Qibin Zhao, and Wanzeng Kong. CTNN: A Convolutional Tensor-Train Neural Network for Multi-Task Brainprint Recognition. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 29:103–112, 2020.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly Accurate Protein Structure Prediction with AlphaFold. *Nature*, 596(7873):583–589, 2021.

Abhinav Kandala, Antonio Mezzacapo, Kristan Temme, Maika Takita, Markus Brink, Jerry M Chow, and Jay M Gambetta. Hardware-Efficient Variational Quantum Eigensolver for Small Molecules and Quantum Magnets. *Nature*, 549(7671):242–246, 2017.

Iordanis Kerenidis, Jonas Landman, and Anupam Prakash. Quantum Algorithms for Deep Convolutional Neural Networks. In *Proc. International Conference on Learning Representations*, 2020.

Diederik P Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Frank Leymann and Johanna Barzen. The Bitter Truth About Gate-Based Quantum Algorithms In the NISQ Era. *Quantum Science and Technology*, 5(4):044007, 2020.

Jarrod R McClean, Jonathan Romero, Ryan Babbush, and Alán Aspuru-Guzik. The Theory of Variational Hybrid Quantum-Classical Algorithms. *New Journal of Physics*, 18(2):023023, 2016.

Jarrod R McClean, Sergio Boixo, Vadim N Smelyanskiy, Ryan Babbush, and Hartmut Neven. Barren Plateaus in Quantum Neural Network Training Landscapes. *Nature Communications*, 9(1): 1–6, 2018.

Kosuke Mitarai, Makoto Negoro, Masahiro Kitagawa, and Keisuke Fujii. Quantum Circuit Learning. *Physical Review A*, 98(3):032309, 2018.

Michael A Nielsen and Isaac Chuang. Quantum computation and quantum information, 2002.

Alexander Novikov, Dmitry Podoprikhin, Anton Osokin, and Dmitry Vetrov. Tensorizing Neural Networks. In *Advances in Neural Information Processing Systems*. 2015.

Ivan V Oseledets. Tensor-Train Decomposition. *SIAM Journal on Scientific Computing*, 33(5): 2295–2317, 2011.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An Imperative Style, High-Performance Deep Learning Library. *Advances in neural information processing systems*, 32:8026–8037, 2019.

John Preskill. Quantum Computing in the NISQ era and beyond. *Quantum*, 2:79, August 2018. ISSN 2521-327X.

Jun Qi, Hu Hu, Yannan Wang, Chao-Han Huck Yang, Sabato Marco Siniscalchi, and Chin-Hui Lee. Exploring deep hybrid tensor-to-vector network architectures for regression based speech enhancement. *arXiv preprint arXiv:2007.13024*, 2020a.

Jun Qi, Hu Hu, Yannan Wang, Chao-Han Huck Yang, Sabato Marco Siniscalchi, and Chin-Hui Lee. Tensor-to-Vector Regression for Multi-Channel Speech Enhancement Based on Tensor-Train Network. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 7504–7508, 2020b.

Maria Schuld and Nathan Killoran. Quantum Machine Learning in Feature Hilbert Spaces. *Physical review letters*, 122(4):040504, 2019.

Maria Schuld and Francesco Petruccione. *Supervised Learning with Quantum Computers*, volume 17. Springer, 2018.

Maria Schuld, Ilya Sinayskiy, and Francesco Petruccione. An introduction to quantum machine learning. *Contemporary Physics*, 56(2):172–185, 2015.

Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun. Overfeat: Integrated Recognition, Localization and Detection Using Convolutional Networks. In *International Conference on Learning Representations*, 2014.

Eric Smalley. AI-Powered Drug Discovery Captures Pharma Interest. *Nature biotechnology*, 35(7): 604–606, 2017.

Andrei N Soklakov and Rüdiger Schack. Efficient State Preparation for A Register of Quantum Bits. *Physical review A*, 73(1):012307, 2006.

Andros Tjandra, Sakriani Sakti, and Satoshi Nakamura. Compressing Recurrent Neural Network with Tensor-Train. In *Proc. International Joint Conference on Neural Networks*, pp. 4451–4458, 2017.

Chao-Han Huck Yang, Jun Qi, Samuel Yen-Chi Chen, Pin-Yu Chen, Sabato Marco Siniscalchi, Xiaoli Ma, and Chin-Hui Lee. Decentralizing Feature Extraction with Quantum Convolutional Neural Network for Automatic Speech Recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 6523–6527, 2021.

Yinchong Yang, Denis Krompass, and Volker Tresp. Tensor-Train Recurrent Neural Networks for Video Classification. In *International Conference on Machine Learning*, pp. 3891–3900, 2017.

Rose Yu, Stephan Zheng, Anima Anandkumar, and Yisong Yue. Long-term forecasting using Tensor-Train RNNs. *Arxiv*, 2017.

A APPENDIX

The section of appendix includes the proofs for Theorem 2 and Theorem 3.

A.1 PROOF FOR THEOREM 2

Proof. Theorem 2 is derived from the modification of the universal approximation theory proposed by Barron (1994); Cybenko (1989). The universal approximation theory is shown in Lemma 1, which suggests that a feed-forward neural network with d neurons can approximate any continuous function with arbitrarily small ϵ.

Lemma 1. *Given a continuous target function $\hat{f} : \mathbb{R}^I \to \mathbb{R}$, we can employ a 2-layer neural network with a non-linear activation $f : \mathbb{R}^I \to \mathbb{R}$, such that*

$$|\hat{f} - f| \leq \frac{1}{\sqrt{d}} C_{\hat{f}}, \tag{12}$$

where J denotes the number of neurons, and $C_{\hat{f}}$ is a constant associated with $\hat{f}$. In particular, for $r \geq 1$, C_f satisfies the following condition as:

$$||\hat{f}||_\infty + \sum_{k, 1 \leq \frac{k(k-1)}{2} \leq r} ||D^k \hat{f}||_\infty \leq C_{\hat{f}}, \tag{13}$$

where $D^k \hat{f} = \left[\nabla \hat{f}, \nabla^2 \hat{f}, ..., \nabla^k \hat{f}\right]^T$.

To associate Lemma 1 with our Theorem 2, the target function is replaced with the target vector $H_X^{-1} H_\theta^{-1} |\mathbf{z}\rangle$, then there is a neural network with a dense layer connected to d qubits such that

$$||H_X^{-1} H_\theta^{-1} |\mathbf{z}\rangle - f_{dense}(\mathbf{y})||_1 \leq \frac{1}{\sqrt{d}} C, \tag{14}$$

where C is related to the target vector $H_X^{-1} H_\theta^{-1} |\mathbf{z}\rangle$. □

A.2 PROOF FOR THEOREM 3

Proof. Assume that $\mathcal{X} = f_{TTN}(\mathbf{y})$, $\hat{\mathcal{X}} = H_X^{-1} H_\theta^{-1} |\mathbf{z}\rangle$ and the TT decomposition of target vector is $\{\hat{\mathcal{X}}_1, \hat{\mathcal{X}}_2, ..., \hat{\mathcal{X}}_K\}$, then we obtain

$$||H_X^{-1} H_\theta^{-1} |\mathbf{z}\rangle - f_{TTN}(\mathbf{y})||_1 = ||\hat{\mathcal{X}} - \mathcal{X}||_1 \leq \prod_{k=1}^{K} ||\hat{\mathcal{X}}_k - \mathcal{X}_k||_1 \tag{15}$$

On the other hand, we denote vec$(\mathcal{Y}_k)$ and vec$(\mathcal{X}_k)$ as the vectorization of the tensors $\mathcal{Y}_k$ and $\mathcal{X}_k$, respectively. We also define $\prod_{k=1}^{K} I_k = I$, $\mathcal{W}_k \in \mathbb{R}^{d_k \times I_k \times r_{k-1} \times r_k}$ as the TTN parameters, and also define $\mathbf{W}_k \in \mathbb{R}^{I_k \times r_{k-1} r_k d_k}$ as the matricization of $\mathcal{W}_k$. Moreover, σ refers to a non-linear activation function.

Since vec$(\mathcal{X}_k) = \sigma(\mathbf{W}^T \text{vec}(\mathcal{Y}_k))$ that corresponds to a dense layer, we can obtain that

$$||\hat{\mathcal{X}}_k - \mathcal{X}_k||_1 \leq \frac{1}{\sqrt{d_k}} C. \tag{16}$$

In sum, we can further obtain

$$||H_X^{-1} H_\theta^{-1} |\mathbf{z}\rangle - f_{TTN}(\mathbf{y})||_1 \leq \prod_{k=1}^{K} ||\hat{\mathcal{X}}_k - \mathcal{X}_k||_1 \leq \prod_{k=1}^{K} \frac{1}{\sqrt{d_k}} C, \tag{17}$$

where $\prod_{k=1}^{K} d_k = d$. □