TOWARD TRAINABILITY OF QUANTUM NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Quantum Neural Networks (QNNs) have been recently proposed as generalizations of classical neural networks to achieve the quantum speed-up. Despite the potential to outperform classical models, serious bottlenecks exist for training QNNs; namely, QNNs with random structures have poor trainability due to the vanishing gradient with rate exponential to the input qubit number. The vanishing gradient could seriously influence the applications of large-size QNNs. In this work, we provide a first viable solution with theoretical guarantees. Specifically, we prove that QNNs with tree tensor architectures have gradients that vanish polynomially with the qubit number. Moreover, our result holds irrespective of which encoding methods are employed. We numerically demonstrate QNNs with tree tensor structures for the application of binary classification. Simulations show faster convergent rates and better accuracy compared to QNNs with random structures.

1 INTRODUCTION

Neural Networks (Hecht-Nielsen, 1992) using gradient-based optimizations have dramatically advanced researches in discriminative models, generative models, and reinforcement learning. To efficiently utilize the parameters and practically improve the trainability, neural networks with specific architectures (LeCun et al., 2015) are introduced for different tasks, including convolutional neural networks (Krizhevsky et al., 2012) for image tasks, recurrent neural networks (Zaremba et al., 2014) for the time series analysis, and graph neural networks (Scarselli et al., 2008) for tasks related to graph-structured data. Recently, the neural architecture search (Elsken et al., 2019) is proposed to improve the performance of the networks by optimizing the neural structures.

Despite the success in many fields, the development of the neural network algorithms could be limited by the large computation resources required for the model training. In recent years, quantum computing has emerged as one solution to this problem, and has evolved into a new interdisciplinary field known as the quantum machine learning (QML) (Biamonte et al., 2017; Havlíček et al., 2019). Specifically, variational quantum circuits (Benedetti et al., 2019) have been explored as efficient protocols for quantum chemistry (Kandala et al., 2017) and combinatorial optimizations (Zhou et al., 2018). Compared to the classical circuit models, quantum circuits have shown greater expressive power (Du et al., 2020a), and demonstrated quantum advantage for the low-depth case (Bravyi et al., 2018). Due to the robustness against noises, variational quantum circuits have attracted significant interest for the hope to achieve the quantum supremacy on near-term quantum computers (Arute et al., 2019).

Quantum Neural Networks (QNNs) (Farhi & Neven, 2018; Schuld et al., 2020; Beer et al., 2020) are the special kind of quantum-classical hybrid algorithms that run on trainable quantum circuits. Recently, small-scale QNNs have been implemented on real quantum computers (Havlíček et al., 2019) for supervised learning tasks. The training of QNNs aims to minimize the objective function f with respect to parameters $\boldsymbol{\theta}$. Inspired by the classical optimizations of neural networks, a natural strategy to train QNNs is to exploit the gradient of the loss function (Crooks, 2019). However, the recent work (McClean et al., 2018) shows that n-qubit quantum circuits with random structures and large depth $L = \mathcal{O}(\mathrm{poly}(n))$ tend to be approximately unitary 2-design (Harrow & Low, 2009), and the partial derivative vanishes to zero exponentially with respect to n. The vanishing gradient problem is usually referred to as the Barren Plateaus (McClean et al., 2018), and could affect the

trainability of QNNs in two folds. Firstly, simply using the gradient-based method like Stochastic Gradient Descent (SGD) to train the QNN takes a large number of iterations. Secondly, the estimation of the derivatives needs an extremely large number of samples from the quantum output to guarantee a relatively accurate update direction (Chen et al., 2018). To avoid the Barren Plateaus phenomenon, we explore QNNs with special structures to gain fruitful results.

In this work, we introduce QNNs with the tree tensor (TT) structure (Huggins et al., 2019) referred to as TT-QNN, and prove that for the TT-QNN, the expectation of the gradient norm of the objective function is bounded.

Theorem 1.1. *(Informal) Consider the n-qubit TT-QNN in Figure 1 and the objective function f_{TT} in (4), then we have:*

$$\frac{1+\log n}{2n} \cdot \alpha(\rho_{in}) \leq \mathbb{E}_{\boldsymbol{\theta}} \|\nabla_{\boldsymbol{\theta}} f_{TT}\|^2 \leq 2n-1, \tag{1}$$

where the expectation is taken for all parameters in $\boldsymbol{\theta}$ with uniform distributions in $[0, 2\pi]$ and $\alpha(\rho_{in}) \geq 0$ is a constant that only depends on the input state $\rho_{in} \in \mathbb{C}^{2^n \times 2^n}$. Moreover, by preparing ρ_{in} using the L-layer encoding circuit in Figure 3, the expectation of $\alpha(\rho_{in})$ could be further lower bounded as $\mathbb{E}\alpha(\rho_{in}) \geq 2^{-2L}$.

Compared to random QNNs with $2^{-\mathcal{O}(\mathrm{poly}(n))}$ derivatives, the gradient norm of TT-QNN is greater than $\mathcal{O}(1/n)$ that could lead to better trainability. Our contributions are summarized as follows:

- We prove a $\mathcal{O}(1/n)$ lower bound on the expectation of the gradient norm of TT-QNNs that guarantees the TT-QNN's trainability. Our theorem does not require the unitary 2-design assumption in existing works and is more realistic to near-term quantum computers.
- We prove that by employing the encoding circuit in (3) to prepare ρ_{in}, the expectation of term $\alpha(\rho_{\text{in}})$ is lower bounded by a constant 2^{-2L}. Thus, we further lower bounded the expectation of the gradient norm to the term independent from the input state.
- We simulate the performance of the TT-QNN and the random structure QNN on the binary classification task. All results verify proposed theorems. The TT-QNN shows better trainability and the accuracy than the random QNN.

Our proof strategy could be adopted for analyzing QNNs with other architectures as future works. With the proven assurance on the trainability of TT-QNN, we eliminate one bottleneck in front of the application of large-size Quantum Neural Networks.

The rest parts of this paper are organized as follows. We address the preliminary including the definitions, the basic quantum computing knowledge and related works in Section 2. The QNNs with TTN structure and the corresponding results are presented in Section 3. We implement the binary classification using QNNs with the results shown in Section 4. We make conclusions in Section 5.

2 Preliminary

2.1 Notations and the Basic Quantum Computing

We use $[N]$ to denote the set $\{1, 2, \cdots, N\}$. The form $\|\cdot\|$ denotes the $\|\cdot\|_2$ norm for vectors. We denote a_j as the j-th component of the vector $\boldsymbol{a}$. The tensor product operation is denoted as "$\otimes$". The conjugate transpose of a matrix A is denoted as $A^\dagger$. The trace of a matrix A is denoted as $\mathrm{Tr}[A]$. We denote $\nabla_{\boldsymbol{\theta}} f$ as the gradient of the function f with respect to the vector $\boldsymbol{\theta}$. We employ notations $\mathcal{O}$ and $\tilde{\mathcal{O}}$ to describe the standard complexity and the complexity ignoring minor terms, respectively.

Now we introduce the quantum computing. The pure state of a qubit could be written as $|\phi\rangle = a|0\rangle + b|1\rangle$, where $a, b \in \mathbb{C}$ satisfies $|a|^2 + |b|^2 = 1$, and $\{|0\rangle = (1,0)^T, |1\rangle = (0,1)^T\}$, respectively. The n-qubit space is formed by the tensor product of n single-qubit spaces. For the vector $\boldsymbol{x} \in \mathbb{R}^{2^n}$, the amplitude encoded state $|\boldsymbol{x}\rangle$ is defined as $\frac{1}{\|\boldsymbol{x}\|}\sum_{j=1}^{2^n} x_j |j\rangle$. The dense matrix is defined as $\rho = |\boldsymbol{x}\rangle\langle\boldsymbol{x}|$ for the pure state, in which $\langle\boldsymbol{x}| = (|\boldsymbol{x}\rangle)^\dagger$. A single-qubit operation to the state behaves like the matrix-vector multiplication and can be referred to as the gate —□— in the quantum circuit language.

Specifically, single-qubit operations are often used including $RX(\theta) = e^{-i\theta X}$, $RY(\theta) = e^{-i\theta Y}$, and $RZ(\theta) = e^{-i\theta Z}$:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

Pauli matrices $\{I, X, Y, Z\}$ will be referred to as $\{\sigma_0, \sigma_1, \sigma_2, \sigma_3\}$ for the convenience. Moreover, two-qubit operations, the CNOT gate and the CZ gate, are employed for generating quantum entanglement:

CNOT = $= |0\rangle\langle 0| \otimes \sigma_0 + |1\rangle\langle 1| \otimes \sigma_1$, CZ = $= |0\rangle\langle 0| \otimes \sigma_0 + |1\rangle\langle 1| \otimes \sigma_3$.

We could obtain information from the quantum system by performing measurements, for example, measuring the state $|\phi\rangle = a|0\rangle + b|1\rangle)$ generates 0 and 1 with probability $p(0) = |a|^2$ and $p(1) = |b|^2$, respectively. Such a measurement operation could be mathematically referred to as calculating the average of the observable $O = \sigma_3$ under the state $|\phi\rangle$:

$$\langle \sigma_3 \rangle_{|\phi\rangle} \equiv \langle \phi | \sigma_3 | \phi \rangle \equiv \text{Tr}[|\phi\rangle\langle\phi| \cdot \sigma_3] = |a|^2 - |b|^2 = p(0) - p(1) = 2p(0) - 1.$$

The average of a unitary observable under arbitrary states is bounded by $[-1, 1]$.

2.2 Related Works

The barren plateaus phenomenon in QNNs is first noticed by McClean et al. (2018). They prove that for n-qubit random quantum circuits with depth $L = \mathcal{O}(\text{poly}(n))$, the expectation of the derivative to the objective function is zero, and the variance of the derivative vanishes to zero with rate exponential in the number of qubits n. Later, Cerezo et al. (2020) prove that for L-depth quantum circuits consisting of 2-design gates, the gradient with local observables vanishes with the rate $\mathcal{O}(2^{-O(L)})$. The result implies that in the low-depth $L = \mathcal{O}(\log n)$ case, the vanishing rate could be $\mathcal{O}(\frac{1}{\text{poly} n})$, which is better than previous exponential results. Recently, some techniques have been proposed to address the barren plateaus problem, including the special initialization strategy (Grant et al., 2019) and the layerwise training method (Skolik et al., 2020). We remark that these techniques rely on the assumption of low-depth quantum circuits. Specifically, Grant et al. (2019) initialize parameters such that the initial quantum circuit is equivalent to an identity matrix ($L = 0$). Skolik et al. (2020) train parameters in subsets in each layer, so that a low-depth circuit is optimized during the training of each subset of parameters.

Since random quantum circuits tend to be approximately unitary 2-design[1] as the circuit depth increases (Harrow & Low, 2009), and 2-design circuits lead to exponentially vanishing gradients (McClean et al., 2018), the natural idea is to consider circuits with special structures. On the other hand, tensor networks with hierarchical structures have been shown an inherent relationship with classical neural networks (Liu et al., 2019; Hayashi et al., 2019). Recently, quantum classifiers using hierarchical structure QNNs have been explored (Grant et al., 2018), including the tree tensor network and the multi-scale entanglement renormalization ansatz. Besides, QNNs with dissipative layers have shown the ability to avoid the barren plateaus (Beer et al., 2020). However, theoretical analysis of the trainability of QNNs with certain layer structures has been little explored (Sharma et al., 2020). Also, the 2-design assumption in the existing theoretical analysis (McClean et al., 2018; Cerezo et al., 2020; Sharma et al., 2020) is hard to implement exactly on near-term quantum devices.

3 Quantum Neural Networks

In this section, we discuss the quantum neural networks in detail. Specifically, the optimizing of QNNs is presented in Section 3.1. We analyze QNNs with the tree tensor structure in Section 3.2. We introduce an approximate quantum input model in Section 3.3 which helps for deriving further theoretical bounds.

[1] We refer the readers to Appendix A for a short discussion about the unitary 2-design.

3.1 THE OPTIMIZING OF QUANTUM NEURAL NETWORKS

In this subsection, we introduce the gradient-based strategy for optimizing QNNs. Like the weight matrix in classical neural networks, the QNN involves a parameterized quantum circuit that mathematically equals to a parameterized unitary matrix $V(\boldsymbol{\theta})$. The training of QNNs aims to optimize the function f defined as:

$$f(\boldsymbol{\theta};\rho_{\text{in}}) = \frac{1}{2} + \frac{1}{2}\text{Tr}\left[O \cdot V(\boldsymbol{\theta}) \cdot \rho_{\text{in}} \cdot V(\boldsymbol{\theta})^{\dagger}\right] = \frac{1}{2} + \frac{1}{2}\langle O\rangle_{V(\boldsymbol{\theta}),\rho_{\text{in}}}, \tag{2}$$

where O denotes the quantum observable and ρ_{in} denotes the density matrix of the input quantum state. Generally, we could deploy the parameters $\boldsymbol{\theta}$ in a tunable quantum circuit arbitrarily. A practical tactic is to encode parameters $\{\theta_j\}$ as the phases of the single-qubit gates $\{e^{-i\theta_j\sigma_k}, k \in \{1,2,3\}\}$ while employing two-qubit gates $\{\text{CNOT},\ \text{CZ}\}$ among them to generate quantum entanglement. This strategy has been frequently used in existing quantum circuit algorithms (Schuld et al., 2020; Benedetti et al., 2019; Du et al., 2020b) since the model suits the noisy near-term quantum computers. Under the single-qubit phase encoding case, the partial derivative of the function f could be calculated using the parameter shifting rule (Crooks, 2019),

$$\frac{\partial f}{\partial \theta_j} = \frac{1}{2}\langle O\rangle_{V(\boldsymbol{\theta}_+),\rho_{\text{in}}} - \frac{1}{2}\langle O\rangle_{V(\boldsymbol{\theta}_-),\rho_{\text{in}}}, \tag{3}$$

where $\boldsymbol{\theta}_+$ and $\boldsymbol{\theta}_-$ are different from $\boldsymbol{\theta}$ only at the j-th parameter: $\theta_j \to \theta_j \pm \frac{\pi}{4}$. Thus, the gradient of f could be obtained by estimating quantum observables, which allows employing quantum computers for fast optimizations using stochastic gradient descents.

3.2 QUANTUM NEURAL NETWORKS WITH TREE TENSOR STRUCTURE

In this subsection, we introduce n-qubit QNNs with the tree tensor (TT) architecture (Grant et al., 2018). We prove that the expectation of the square of gradient ℓ_2-norm for the TT-QNN is lower bounded by $\tilde{\mathcal{O}}(1/n)$. Moreover, the corresponding theoretical analysis does not rely on 2-design assumptions for quantum circuits.

Now we discuss the TT-QNN in detail. We consider the $n = 2^m$-qubit quantum neural network with the tree tensor structure constructed by the single-qubit gate $W_j^{(k)} = e^{-i\theta_j^{(k)}\sigma_2}$ and the CNOT gate $\sigma_1 \otimes |1\rangle\langle 1| + \sigma_0 \otimes |0\rangle\langle 0|$. We define the k-th parameter in the j-th layer as $\theta_j^{(k)}$. The number of parameters in such a quantum circuit is $2n-1$. An illustration of the TT-QNN is shown in Figure 1 for the case $m = 3$. The objective function is given as

$$f_{\text{TT}}(\boldsymbol{\theta}) = \frac{1}{2} + \frac{1}{2}\text{Tr}[\sigma_3 \otimes I^{\otimes(n-1)} V_{\text{TT}}(\boldsymbol{\theta})\rho_{\text{in}} V_{\text{TT}}(\boldsymbol{\theta})^{\dagger}], \tag{4}$$

where $V_{\text{TT}}(\boldsymbol{\theta}) = V_{m+1}(\boldsymbol{\theta}_{m+1}) \cdot CX_m \cdot V_m(\boldsymbol{\theta}_m) \cdots CX_1 \cdot V_1(\boldsymbol{\theta}_1)$. We denote the j-th parameterized single-qubit rotation layer as $V_j(\boldsymbol{\theta}_j)$ for $j \in [m+1]$ and CX_k denotes the k-th CNOT operation layer for $k \in [m]$. The whole parameter vector is arranged as $\boldsymbol{\theta}^T = (\boldsymbol{\theta}_1^T, \cdots, \boldsymbol{\theta}_{m+1}^T)$. The main result of this section is stated in Theorem 3.1, in which we prove a $\tilde{\mathcal{O}}(1/n)$ lower bound on the expectation of the square of the gradient norm. We provide the proof of Theorem 3.1 in Appendix B and C.

Theorem 3.1. *Consider the n-qubit TT-QNN defined in Figure 1 and the corresponding objective function f_{TT} defined in (4), then we have:*

$$\frac{1+\log n}{2n} \cdot \alpha(\rho_{in}) \leq \mathbb{E}_{\boldsymbol{\theta}}\|\nabla_{\boldsymbol{\theta}} f_{TT}\|^2 \leq 2n-1, \tag{5}$$

where the expectation is taken for all parameters in $\boldsymbol{\theta}$ with uniform distributions in $[0, 2\pi]$, $\rho_{in} \in \mathbb{C}^{2^n \times 2^n}$ denotes the input state, $\alpha(\rho_{in}) = Tr\left[\sigma_{(1,0,\cdots,0)} \cdot \rho_{in}\right]^2 + Tr\left[\sigma_{(3,0,\cdots,0)} \cdot \rho_{in}\right]^2$, and $\sigma_{(i_1,i_2,\cdots,i_n)} \equiv \sigma_{i_1} \otimes \sigma_{i_2} \otimes \cdots \otimes \sigma_{i_n}$.

From the geographic view, the value $\mathbb{E}_{\boldsymbol{\theta}}\|\nabla_{\boldsymbol{\theta}} f\|^2$ characterizes the global steepness of the function surface in the parameter space. Optimizing the function f using gradient-based methods could be hard if the norm of the gradient vanishes to zero. Thus, the lower bound in (5) provides a theoretical

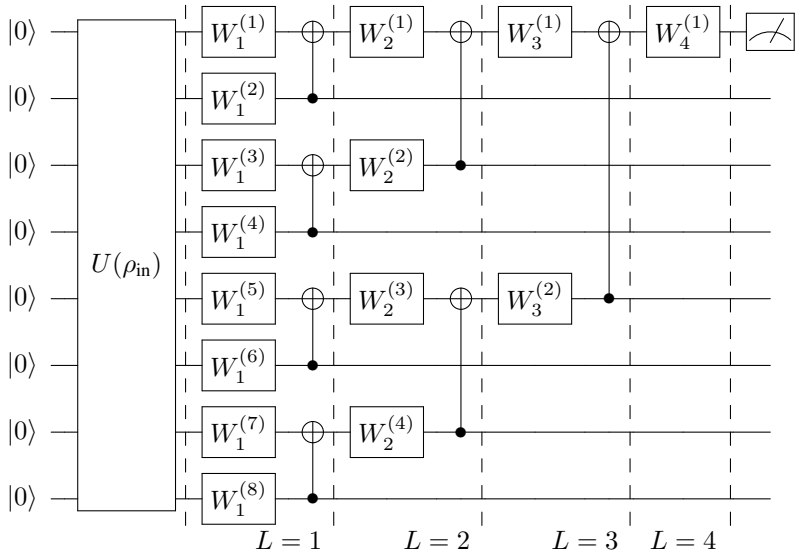

Figure 1: Quantum Neural Network with the Tree Tensor structure ($m = 3$).

guarantee on optimizing the function (4), which ensures the trainability of TT-QNNs on related machine learning tasks.

From the technical view, we provide a new theoretical framework during proving the bound in (5). Different from existing works (McClean et al., 2018; Grant et al., 2019; Cerezo et al., 2020) that define the expectation as the average of the finite unitary 2-design group, we consider the uniform distribution in which each parameter in $\boldsymbol{\theta}$ varies continuously in $[0, 2\pi]$. Our assumption suits the quantum circuits that encode the parameters in the phase of single-qubit rotations. Our framework could be extensively employed for analyzing QNNs with other different structures as future works.

3.3 Prepare the Quantum Input Model: a Variational Circuit Approach

State preparation is an essential part of most quantum algorithms, which encodes the classical information into quantum states. Specifically, the amplitude encoding $|\boldsymbol{x}\rangle = \sum_{i=1}^{2^n} x_i/\|\boldsymbol{x}\||i\rangle$ allows storing the 2^n-dimensional vector in n qubits. Due to the dense encoding nature and the similarity to the original vector, the amplitude encoding is preferred as the state preparation by many QML algorithms (Harrow et al., 2009; Rebentrost et al., 2014; Kerenidis et al., 2019). Despite the wide application in quantum algorithms, efficient amplitude encoding remains little explored. Existing work (Park et al., 2019) could prepare the amplitude encoding state in time $\mathcal{O}(2^n)$ using a quantum circuit with $\mathcal{O}(2^n)$ depth, which is prohibitive for large-size data on near-term quantum computers. In fact, arbitrary quantum amplitude encoding with polynomial gate complexity remains an open problem.

In this subsection, we introduce a quantum input model for approximately encoding the arbitrary vector $\boldsymbol{x}_{\text{in}}$ in the amplitude of the quantum state $|\boldsymbol{x}_{\text{in}}\rangle$. The main idea is to train an alternating layered circuit as summarized in Algorithm 1 and Figures 2-3. Now we explain the detail of the input model. Firstly, we randomly initialize the parameter $\boldsymbol{\beta}^{(0)}$ in the circuit 2. Then, we train the parameter to minimize the objective function defined in (6) through the gradient descent,

$$f_{\text{input}}(\boldsymbol{\beta}) = \frac{1}{n}\sum_{i=1}^{n}\langle O_i\rangle_{W(\boldsymbol{\beta})|\boldsymbol{x}_{\text{in}}\rangle} = \frac{1}{n}\sum_{i=1}^{n}\text{Tr}[O_i \cdot W(\boldsymbol{\beta})|\boldsymbol{x}_{\text{in}}\rangle\langle\boldsymbol{x}_{\text{in}}|W(\boldsymbol{\beta})^\dagger], \tag{6}$$

where $O_i = \sigma_0^{\otimes(i-1)} \otimes \sigma_3 \otimes \sigma_0^{\otimes(n-i)}, \forall i \in [n]$, and $W(\boldsymbol{\beta})$ denotes the tunable alternating layered circuit in Figure 2. Note that although the framework is given as the quantum circuit, we actually calculate and update the gradient on classical computers by considering each quantum gate operation as the matrix multiplication.

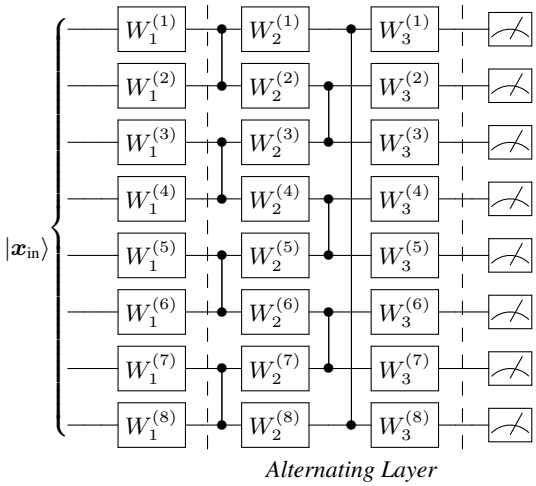

Figure 2: The parameterized alternating layered circuit with the input state $|\boldsymbol{x}_{\text{in}}\rangle$ for training the corresponding encoding circuit ($L = 1$ case).

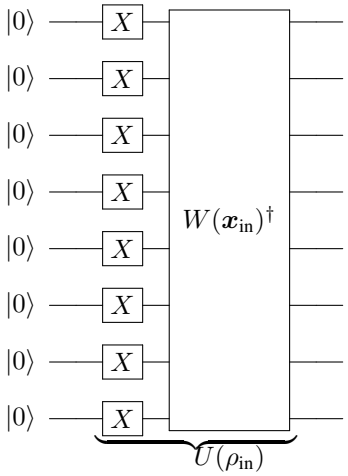

Figure 3: The encoding circuit $U(\rho_{\text{in}})$, where $W(\boldsymbol{x}_{\text{in}})$ denote the trained parameterized circuit in Figure 2.

Algorithm 1 Quantum Input Model

Require: The input vector $\boldsymbol{x}_{\text{in}} \in \mathbb{R}^{2^n}$, the number of alternating layers L, the iteration time T, and the learning rate $\{\eta(t)\}_{t=0}^{T-1}$.

Ensure: An approximate encoding circuit denoted as $U(\rho_{\text{in}})$.

1: Initialize $\{\beta_j^{(k)}\}_{j,k=1}^{n,2L+1}$ randomly in $[0, 2\pi]$. Denote the parameter vector as $\boldsymbol{\beta}^{(0)}$.
2: **for** $t \in \{0, 1, \cdots, T-1\}$ **do**
3: Run the circuit illustrated in Figure 2 to calculate the gradient $\nabla_{\boldsymbol{\beta}} f_{\text{input}}|_{\boldsymbol{\beta}=\boldsymbol{\beta}^{(t)}}$ using the parameter shifting rule (3), where the function f_{input} is defined in (6).
4: Update the parameter $\boldsymbol{\beta}^{(t+1)} = \boldsymbol{\beta}^{(t)} - \eta(t) \cdot \nabla_{\boldsymbol{\beta}} f_{\text{input}}|_{\boldsymbol{\beta}=\boldsymbol{\beta}^{(t)}}$.
5: **end for**
6: Denote the trained circuit as $W(\boldsymbol{x}_{\text{in}})$. Output the encoding circuit $U(\rho_{\text{in}}) = W(\boldsymbol{x}_{\text{in}})^\dagger \cdot X^{\otimes n}$.

Denote the trained circuit in Figure 2 as $W(\boldsymbol{x}_{\text{in}})$. Suppose we could minimize the objective function (6) to -1. Then $\langle O_i \rangle = -1, \forall i \in [n]$, which means the final output state in Figure 2 is $W(\boldsymbol{x}_{\text{in}})|\boldsymbol{x}_{\text{in}}\rangle = |1\rangle^{\otimes n}$. Thus the state $|\boldsymbol{x}_{\text{in}}\rangle$ could be prepared exactly by applying the circuit $W(\boldsymbol{x}_{\text{in}})^\dagger X^{\otimes n}$ on the state $|0\rangle^{\otimes n}$. However we could not always minimize the loss in (6) to -1, which means the framework could only prepare the amplitude encoding state approximately.

An interesting result is that by employing the encoding circuit in Figure 3 for constructing the state ρ_{in} in Section 3.2, we could bound the expectation of $\alpha(\rho_{\text{in}})$ defined in Theorem 3.1 by the constant that only relies on the layer L (Theorem 3.2).

Theorem 3.2. *Suppose the state ρ_{in} is prepared by the L-layer encoding circuit in Figure 3, then we have,*

$$\mathbb{E}_{\boldsymbol{\beta}} \alpha(\rho_{in}) \geq 2^{-2L},$$

where $\boldsymbol{\beta}$ denotes all variational parameters in the encoding circuit, and the expectation is taken for all parameters in $\boldsymbol{\beta}$ with uniform distributions in $[0, 2\pi]$.

We provide the proof of Theorem 3.2 and more details about the input model in Appendix D. Theorem 3.2 can be employed in Theorem 3.1 to derive Theorem 1.1, in which the lower bound for the expectation of the gradient norm is independent from the input state.

4 Application: QNNs for the Binary Classification

In this section, we show how to train QNNs for the binary classification in quantum computers. First of all, for the training and test data denoted as $\{(\boldsymbol{x}_i^{\text{train}}, y_i)\}_{i=1}^s$ and $\{(\boldsymbol{x}_j^{\text{test}}, y_j)\}_{j=1}^q$, where

Algorithm 2 QNNs for the Binary Classification Training

Require: Quantum encoding circuits $\{U(\rho_{i,0}^{\text{train}})\}_{i=1}^{s}$ and $\{U(\rho_{i,1}^{\text{train}})\}_{i=1}^{s}$ for the dataset $\{(\boldsymbol{x}_i^{\text{train}}, y_i)\}_{i=1}^{s}$, the quantum observable O, the parameterized quantum circuit $V(\boldsymbol{\theta})$, the iteration time T, and the learning rate $\{\eta(t)\}_{t=0}^{T-1}$.

Ensure: A quantum circuit V_{trained} that can be used for the binary classification.

1: Initialize each parameter in $\boldsymbol{\theta}^{(0)}$ randomly in $[0, 2\pi]$.
2: **for** $t \in \{0, 1, \cdots, T-1\}$ **do**
3: Randomly select an index $j \in [s]$. Calculate the gradient $\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}; \rho_{j,0}^{\text{train}})|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(t)}}$ using the parameter shifting rule (3), where the function $f(\boldsymbol{\theta}; \rho)$ is defined in (7).
4: Update the parameter $\boldsymbol{\theta}^{(t+1/2)} = \boldsymbol{\theta}^{(t)} - \eta(t) \cdot \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}; \rho_{j,0}^{\text{train}})|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(t)}}$.
5: Randomly select an index $k \in [s]$. Calculate the gradient $\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}; \rho_{k,1}^{\text{train}})|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(t+1/2)}}$ using the parameter shifting rule (3), where the function $f(\boldsymbol{\theta}; \rho)$ is defined in (7).
6: Update the parameter $\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t+1/2)} + \eta(t) \cdot \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}; \rho_{k,1}^{\text{train}})|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(t+1/2)}}$.
7: **end for**
8: Denote the trained circuit as V_{trained}. Output V_{trained}.

$y_i \in \{0, 1\}$ denotes the label, we prepare the corresponding quantum input model $\{U(\rho_{i,0}^{\text{train}})\}_{i=1}^{s}$, $\{U(\rho_{i,1}^{\text{train}})\}_{i=1}^{s}$, $\{U(\rho_{j,0}^{\text{test}})\}_{j=1}^{q}$, and $\{U(\rho_{j,1}^{\text{test}})\}_{j=1}^{q}$ using the encoding circuit presented in Section 3.3. Then, given the parameterized circuit $V(\boldsymbol{\theta})$ and the quantum observable O, we train the parameter $\boldsymbol{\theta}$ in Algorithm 2 via the stochastic gradient descent method. The parameter updating in each iteration is presented in the Step 3-6. The parameter $\boldsymbol{\theta}$ is trained by minimizing the loss defined in (7) for each input $(\rho_{i,y_i}^{\text{train}}, y_i), \forall i \in [s]$.

$$\ell(\boldsymbol{\theta}; \rho, y) = |f(\boldsymbol{\theta}; \rho) - y| = \Big|\frac{1}{2}\text{Tr}[O \cdot V(\boldsymbol{\theta})\rho V(\boldsymbol{\theta})^{\dagger})] + \frac{1}{2} - y\Big|. \tag{7}$$

After the training we obtain a quantum circuit V_{trained}, and we do test for an input ρ^{test} by calculating the objective function $f(\rho^{\text{test}}) = \frac{1}{2} + \frac{1}{2}\text{Tr}[O \cdot V_{\text{trained}}\ \rho^{\text{test}}\ V_{\text{trained}})^{\dagger}]$. We classify the input ρ^{test} as 0 if $f(\rho^{\text{test}}) < \frac{1}{2}$, or 1 if $f(\rho^{\text{test}}) > \frac{1}{2}$. In practice, the value of the objective function is estimated by measuring the output state of the QNN for several times and averaging quantum outputs.

The time complexity of the QNN training and test could be easily derived by counting resources for estimating all quantum observables. Denote the number of gates and parameters in the quantum circuit $V(\boldsymbol{\theta})$ as n_{gate} and n_{para}, respectively. Denote the number of measurements for estimating each quantum observable as n_{train} and n_{test} for the training and test stages, respectively. Then, the time complexity to train QNNs is $\mathcal{O}(n_{\text{gate}} n_{\text{para}} n_{\text{train}} T)$, and the time complexity of the test using QNNs is $\mathcal{O}(n_{\text{gate}} n_{\text{text}})$. We emphasize that directly comparing the time complexity of QNNs with classical NNs is unfair due to different parameter strategies. However, the complexity of QNNs indeed shows a polylogarithmic dependence on the dimension of the input data if the number of gates and parameters are polynomial to the number of qubits.

To analyze the practical performance of QNNs for binary classification tasks, we simulate the training and test of QNNs on the MNIST handwritten dataset. We conduct the binary classification on images with labels 0 and 1. The 28×28 size image is down-sampled into 16×16 to fit the QNN. We construct the encoding circuits in Section 3.3 with the number of alternating layers $L = 1$ for 400 training samples and 100 test samples in each class. The TT-QNN is compared to the QNN with the random structure. To make a fair comparison, we set the numbers of RY and CNOT gates in the random QNN to be the same with the TT-QNN. The objective function of the random QNN is defined as the average of the expectation of the observable σ_3 for all qubits in the circuit. The number of the training iteration is 400 for both TT-QNN and the random structure QNN, and the decayed learning rate is adopted as $\{0.4, 0.3, 0.2, 0.1\}$. We set $n_{\text{train}} = 100$ and $n_{\text{infer}} = 1000$ as numbers of measurements for estimating quantum observables during the training and test stages, respectively. All experiments are simulated through the PennyLane Python package (Bergholm et al., 2020).

Now we explain our results in Figure 4. Firstly, we train the TT-QNN and the random QNN with the stochastic gradient descent method described in Algorithm 2. The total loss is defined as the average of the single-input loss (7). We present the total error and the loss during the training iteration illustrated in Figure 4(a) and Figure 4(b). The test error about the TT-QNN after the training is

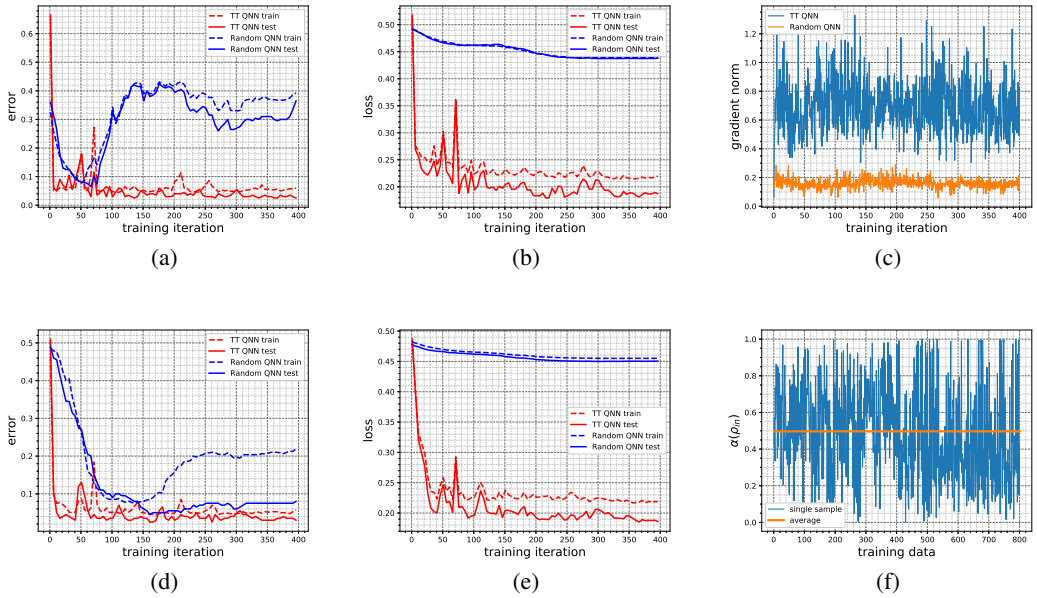

Figure 4: Simulations on the MNIST binary classification. The error and loss of QNNs trained using SGD are shown in Figures 4(a) and 4(b), respectively. The error and loss of QNNs trained using SGD with normalized gradient are shown in Figures 4(d) and 4(e), respectively. The norm of gradient and the term $\alpha(\rho_{\text{in}})$ during the training are shown in Figures 4(c) and 4(f), respectively.

lower than 3%, and the training loss converges to around the stable value after 100 iterations, both of which are better than the random QNN. The test error about the random QNN is around 36%, and the training loss converges to around the stable value after 250 iterations. We record the l_2-norm of the gradient during the training in Figure 4(c). The gradient norm for the TT-QNN is mostly distributed in $[0.5, 1.0]$, which is significantly larger than the gradient norm for the random QNN that is mostly distributed in $[0.1, 0.2]$. As shown in Figure 4(c), the gradient norm verifies the lower bound in the Theorem 1.1.

Given the results in Figure 4(c), one may consider that the random QNN could not be trained well simply because of the small gradient norm. Actually, to answer the question, we perform a similar task in which the normalized gradient $\nabla_{\boldsymbol{\theta}} f \rightarrow \nabla_{\boldsymbol{\theta}} f / \|\nabla_{\boldsymbol{\theta}} f\|$ is adopted, while other conditions remain the same. Results are shown in Figure 4(d) and Figure 4(e). We find that the test error of the random QNN drops to around 8%, but the convergence of training loss still needs 250 iterations. The bad behavior of random QNN may due to the random designed structure or the noise introduced by quantum measurements. Moreover, we calculate the term $\alpha(\rho_{\text{in}})$ defined in Theorem 3.1 and show the result in Figure 4(f). The average of $\alpha(\rho_{\text{in}})$ is around $\frac{1}{2}$, which is lower bounded by the theoretical result $\frac{1}{4}$ in Theorem 3.2 ($L = 1$). In conclusion, the TT-QNN shows better trainability and accuracy on the binary classification compared with the random structure QNN, and all theorems are verified by experiments.

5 Conclusions

In this work, we analyze the vanishing gradient problem in quantum neural networks. We prove that the gradient norm of n-qubit quantum neural networks with the tree tensor structure is lower bounded by $\mathcal{O}(\frac{1}{n})$. The bound guarantees the trainability of TT-QNNs. Our theoretical framework requires fewer assumptions than previous works and meets constraints on quantum neural networks for near-term quantum computers. Compared with the random structure QNN which is known to be suffered from the barren plateaus problem, the TT-QNN shows better trainability and accuracy on the binary classification task. We hope the paper could inspire future works on the trainability of QNNs with different architectures and other quantum machine learning algorithms.

REFERENCES

Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C Bardin, Rami Barends, Rupak Biswas, Sergio Boixo, Fernando GSL Brandao, David A Buell, et al. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779):505–510, 2019.

Kerstin Beer, Dmytro Bondarenko, Terry Farrelly, Tobias J Osborne, Robert Salzmann, Daniel Scheiermann, and Ramona Wolf. Training deep quantum neural networks. *Nature communications*, 11(1):1–6, 2020.

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, Keri McKiernan, Johannes Jakob Meyer, Zeyue Niu, Antal Száva, and Nathan Killoran. Pennylane: Automatic differentiation of hybrid quantum-classical computations, 2020.

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

Sergey Bravyi, David Gosset, and Robert König. Quantum advantage with shallow circuits. *Science*, 362(6412):308–311, 2018.

M Cerezo, Akira Sone, Tyler Volkoff, Lukasz Cincio, and Patrick J Coles. Cost-function-dependent barren plateaus in shallow quantum neural networks. *arXiv preprint arXiv:2001.00550*, 2020.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *International Conference on Machine Learning*, pp. 794–803. PMLR, 2018.

Gavin E Crooks. Gradients of parameterized quantum gates using the parameter-shift rule and gate decomposition. *arXiv preprint arXiv:1905.13311*, 2019.

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

Yuxuan Du, Min-Hsiu Hsieh, Tongliang Liu, Shan You, and Dacheng Tao. On the learnability of quantum neural networks. *arXiv preprint arXiv:2007.12369*, 2020b.

Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. *Journal of Machine Learning Research*, 20:1–21, 2019.

Edward Farhi and Hartmut Neven. Classification with quantum neural networks on near term processors. *arXiv preprint arXiv:1802.06002*, 2018.

Edward Grant, Marcello Benedetti, Shuxiang Cao, Andrew Hallam, Joshua Lockhart, Vid Stojevic, Andrew G. Green, and Simone Severini. Hierarchical quantum classifiers. *npj Quantum Information*, 4(1), Dec 2018.

Edward Grant, Leonard Wossnig, Mateusz Ostaszewski, and Marcello Benedetti. An initialization strategy for addressing barren plateaus in parametrized quantum circuits. *Quantum*, 3:214, 2019.

Aram W Harrow and Richard A Low. Random quantum circuits are approximate 2-designs. *Communications in Mathematical Physics*, 291(1):257–302, 2009.

Aram W Harrow, Avinatan Hassidim, and Seth Lloyd. Quantum algorithm for linear systems of equations. *Physical review letters*, 103(15):150502, 2009.

Vojtěch Havlíček, Antonio D Córcoles, Kristan Temme, et al. Supervised learning with quantum-enhanced feature spaces. *Nature*, 567(7747):209, 2019.

Kohei Hayashi, Taiki Yamaguchi, Yohei Sugawara, and Shin-ichi Maeda. Exploring unexplored tensor network decompositions for convolutional neural networks. In *Advances in Neural Information Processing Systems*, pp. 5552–5562, 2019.

Robert Hecht-Nielsen. Theory of the backpropagation neural network. In *Neural networks for perception*, pp. 65–93. Elsevier, 1992.

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.

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. *arXiv preprint arXiv:1911.01117*, 2019.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

Ding Liu, Shi-Ju Ran, Peter Wittek, Cheng Peng, Raul Blázquez García, Gang Su, and Maciej Lewenstein. Machine learning by unitary tensor network of hierarchical tree structure. *New Journal of Physics*, 21(7):073059, Jul 2019.

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.

Daniel K Park, Francesco Petruccione, and June-Koo Kevin Rhee. Circuit-based quantum random access memory for classical data. *Scientific reports*, 9(1):1–8, 2019.

Zbigniew Puchała and Jarosław Adam Miszczak. Symbolic integration with respect to the haar measure on the unitary group. *arXiv preprint arXiv:1109.4244*, 2011.

Patrick Rebentrost, Masoud Mohseni, and Seth Lloyd. Quantum support vector machine for big data classification. *Physical review letters*, 113(13):130503, 2014.

Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2008.

Maria Schuld, Alex Bocharov, Krysta M Svore, and Nathan Wiebe. Circuit-centric quantum classifiers. *Physical Review A*, 101(3):032308, 2020.

Kunal Sharma, Marco Cerezo, Lukasz Cincio, and Patrick J Coles. Trainability of dissipative perceptron-based quantum neural networks. *arXiv preprint arXiv:2005.12458*, 2020.

Andrea Skolik, Jarrod R McClean, Masoud Mohseni, Patrick van der Smagt, and Martin Leib. Layerwise learning for quantum neural networks. *arXiv preprint arXiv:2006.14904*, 2020.

Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*, 2014.

Leo Zhou, Sheng-Tao Wang, Soonwon Choi, Hannes Pichler, and Mikhail D Lukin. Quantum approximate optimization algorithm: performance, mechanism, and implementation on near-term devices. *arXiv preprint arXiv:1812.01041*, 2018.

The Appendix of this paper is organized as follows. We shortly introduce the notion of unitary 2-designs in Appendix A. Some useful technical lemmas are provided and proved in Appendix B. We prove the Theorem 3.1 in Appendix C. The Appendix B and Appendix C forms the theoretical framework for analyzing the gradient norm of TT-QNNs. We provide the proof of Theorem 3.2 in the main text and more detail about the proposed encoding model in Appendix D.

A NOTES ABOUT THE UNITARY 2-DESIGN

In this section, we introduce the notion of the unitary 2-design. Consider the finite gate set $S = \{G_i\}_{i=1}^{|S|}$ in the d-dimensional Hilbert space. We denote $U(d)$ as the unitary gate group with the dimension d. We denote $P_{t,t}(G)$ as the polynomial function which has the degree at most t on the matrix elements of G and at most t on the matrix elements of $G^\dagger$. Then, we could say the set S to be the unitary t-design if and only if for every function $P_{t,t}(\cdot)$, Eq. (8) holds:

$$\frac{1}{|S|}\sum_{G\in S} P_{t,t}(G) = \int_{U(d)} d\mu(G) P_{t,t}(G), \tag{8}$$

where $d\mu(\cdot)$ denotes the Haar distribution. The Haar distribution $d\mu(\cdot)$ is defined that for any function f and any matrix $K \in U(d)$,

$$\int_{U(d)} d\mu(G) f(G) = \int_{U(d)} d\mu(G) f(KG) = \int_{U(d)} d\mu(G) f(GK).$$

The form in the right side of (8) can be viewed as the average or the expectation of the function $P_{t,t}(G)$. We remark that only the parameterized gates $RY = e^{-i\theta\sigma_2}$ could not form a universal gate set even in the single-qubit space $U(2)$, thus quantum circuits employing parameterized RY gates could not form the 2-design. This is only a simple introduction about the unitary 2-design, and we refer readers to Puchała & Miszczak (2011) and Cerezo et al. (2020) for more detail.

B TECHNICAL LEMMAS

In this section we provide some technical lemmas.

Lemma B.1. *Let CNOT* $= \sigma_0 \otimes |0\rangle\langle 0| + \sigma_1 \otimes |1\rangle\langle 1|$. *Then*

$$\begin{aligned}\textit{CNOT}(\sigma_j \otimes \sigma_k)\textit{CNOT}^\dagger =& (\delta_{j0}+\delta_{j1})(\delta_{k0}+\delta_{k3})\sigma_j \otimes \sigma_k + (\delta_{j0}+\delta_{j1})(\delta_{k1}+\delta_{k2})\sigma_j\sigma_1 \otimes \sigma_k \\ &+ (\delta_{j2}+\delta_{j3})(\delta_{k0}+\delta_{k3})\sigma_j \otimes \sigma_k\sigma_3 - (\delta_{j2}+\delta_{j3})(\delta_{k1}+\delta_{k2})\sigma_j\sigma_1 \otimes \sigma_k\sigma_3.\end{aligned}$$

Further for the case $\sigma_k = \sigma_0$,

$$\textit{CNOT}(\sigma_j \otimes \sigma_0)\textit{CNOT}^\dagger = (\delta_{j0}+\delta_{j1})\sigma_j \otimes \sigma_0 + (\delta_{j2}+\delta_{j3})\sigma_j \otimes \sigma_3.$$

Proof.

$$\begin{aligned}
&\text{CNOT}(\sigma_j \otimes \sigma_k)\text{CNOT}^\dagger \\
=& \left(\sigma_0 \otimes |0\rangle\langle 0| + \sigma_1 \otimes |1\rangle\langle 1|\right)(\sigma_j \otimes \sigma_k)\left(\sigma_0 \otimes |0\rangle\langle 0| + \sigma_1 \otimes |1\rangle\langle 1|\right) \\
=& \left(\sigma_0 \otimes \frac{\sigma_0+\sigma_3}{2} + \sigma_1 \otimes \frac{\sigma_0-\sigma_3}{2}\right)(\sigma_j \otimes \sigma_k)\left(\sigma_0 \otimes \frac{\sigma_0+\sigma_3}{2} + \sigma_1 \otimes \frac{\sigma_0-\sigma_3}{2}\right) \\
=& \frac{1}{4}\left(\sigma_j \otimes \sigma_k + \sigma_1\sigma_j\sigma_1 \otimes \sigma_k + \sigma_j \otimes \sigma_3\sigma_k\sigma_3 + \sigma_1\sigma_j\sigma_1 \otimes \sigma_3\sigma_k\sigma_3\right) \\
&+ \frac{1}{4}\left(\sigma_j\sigma_1 \otimes \sigma_k + \sigma_1\sigma_j \otimes \sigma_k - \sigma_j\sigma_1 \otimes \sigma_3\sigma_k\sigma_3 - \sigma_1\sigma_j \otimes \sigma_3\sigma_k\sigma_3\right) \\
&+ \frac{1}{4}\left(\sigma_j \otimes \sigma_k\sigma_3 + \sigma_j \otimes \sigma_3\sigma_k - \sigma_1\sigma_j\sigma_1 \otimes \sigma_k\sigma_3 - \sigma_1\sigma_j\sigma_1 \otimes \sigma_3\sigma_k\right) \\
&+ \frac{1}{4}\left(\sigma_j\sigma_1 \otimes \sigma_3\sigma_k - \sigma_j\sigma_1 \otimes \sigma_k\sigma_3 + \sigma_1\sigma_j \otimes \sigma_k\sigma_3 - \sigma_1\sigma_j \otimes \sigma_3\sigma_k\right) \\
=& (\delta_{j0}+\delta_{j1})(\delta_{k0}+\delta_{k3})\sigma_j \otimes \sigma_k + (\delta_{j0}+\delta_{j1})(\delta_{k1}+\delta_{k2})\sigma_j\sigma_1 \otimes \sigma_k \\
&+ (\delta_{j2}+\delta_{j3})(\delta_{k0}+\delta_{k3})\sigma_j \otimes \sigma_k\sigma_3 - (\delta_{j2}+\delta_{j3})(\delta_{k1}+\delta_{k2})\sigma_j\sigma_1 \otimes \sigma_k\sigma_3.
\end{aligned}$$

For the case $\sigma_k = \sigma_0$, we have,

$$\text{CNOT}(\sigma_j \otimes \sigma_0)\text{CNOT}^\dagger = (\delta_{j0} + \delta_{j1})\sigma_j \otimes \sigma_0 + (\delta_{j2} + \delta_{j3})\sigma_j \otimes \sigma_3.$$

□

Lemma B.2. *Let CZ* $= \sigma_0 \otimes |0\rangle\langle 0| + \sigma_3 \otimes |1\rangle\langle 1|$. *Then*

$$\begin{aligned} CZ(\sigma_j \otimes \sigma_k)CZ^\dagger =& (\delta_{j0} + \delta_{j3})(\delta_{k0} + \delta_{k3})\sigma_j \otimes \sigma_k + (\delta_{j0} + \delta_{j3})(\delta_{k1} + \delta_{k2})\sigma_j\sigma_3 \otimes \sigma_k \\ &+ (\delta_{j1} + \delta_{j2})(\delta_{k0} + \delta_{k3})\sigma_j \otimes \sigma_k\sigma_3 - (\delta_{j1} + \delta_{j2})(\delta_{k1} + \delta_{k2})\sigma_j\sigma_3 \otimes \sigma_k\sigma_3. \end{aligned}$$

Further for the case $\sigma_k = \sigma_0$,

$$CZ(\sigma_j \otimes \sigma_0)CZ^\dagger = (\delta_{j0} + \delta_{j3})\sigma_j \otimes \sigma_0 + (\delta_{j1} + \delta_{j2})\sigma_j \otimes \sigma_3.$$

Proof.

$$\begin{aligned} &\text{CZ}(\sigma_j \otimes \sigma_k)\text{CZ}^\dagger \\ =& \left(\sigma_0 \otimes |0\rangle\langle 0| + \sigma_3 \otimes |1\rangle\langle 1|\right)(\sigma_j \otimes \sigma_k)\left(\sigma_0 \otimes |0\rangle\langle 0| + \sigma_3 \otimes |1\rangle\langle 1|\right) \\ =& \left(\sigma_0 \otimes \frac{\sigma_0 + \sigma_3}{2} + \sigma_3 \otimes \frac{\sigma_0 - \sigma_3}{2}\right)(\sigma_j \otimes \sigma_k)\left(\sigma_0 \otimes \frac{\sigma_0 + \sigma_3}{2} + \sigma_3 \otimes \frac{\sigma_0 - \sigma_3}{2}\right) \\ =& \frac{1}{4}\left(\sigma_j \otimes \sigma_k + \sigma_3\sigma_j\sigma_3 \otimes \sigma_k + \sigma_j \otimes \sigma_3\sigma_k\sigma_3 + \sigma_3\sigma_j\sigma_3 \otimes \sigma_3\sigma_k\sigma_3\right) \\ &+ \frac{1}{4}\left(\sigma_j\sigma_3 \otimes \sigma_k + \sigma_3\sigma_j \otimes \sigma_k - \sigma_j\sigma_3 \otimes \sigma_3\sigma_k\sigma_3 - \sigma_3\sigma_j \otimes \sigma_3\sigma_k\sigma_3\right) \\ &+ \frac{1}{4}\left(\sigma_j \otimes \sigma_k\sigma_3 + \sigma_j \otimes \sigma_3\sigma_k - \sigma_3\sigma_j\sigma_3 \otimes \sigma_k\sigma_3 - \sigma_3\sigma_j\sigma_3 \otimes \sigma_3\sigma_k\right) \\ &+ \frac{1}{4}\left(\sigma_j\sigma_3 \otimes \sigma_3\sigma_k - \sigma_j\sigma_3 \otimes \sigma_k\sigma_3 + \sigma_3\sigma_j \otimes \sigma_k\sigma_3 - \sigma_3\sigma_j \otimes \sigma_3\sigma_k\right) \\ =& (\delta_{j0} + \delta_{j3})(\delta_{k0} + \delta_{k3})\sigma_j \otimes \sigma_k + (\delta_{j0} + \delta_{j3})(\delta_{k1} + \delta_{k2})\sigma_j\sigma_3 \otimes \sigma_k \\ &+ (\delta_{j1} + \delta_{j2})(\delta_{k0} + \delta_{k3})\sigma_j \otimes \sigma_k\sigma_3 - (\delta_{j1} + \delta_{j2})(\delta_{k1} + \delta_{k2})\sigma_j\sigma_3 \otimes \sigma_k\sigma_3. \end{aligned}$$

For the case $\sigma_k = \sigma_0$, we have,

$$\text{CZ}(\sigma_j \otimes \sigma_0)\text{CZ}^\dagger = (\delta_{j0} + \delta_{j3})\sigma_j \otimes \sigma_0 + (\delta_{j1} + \delta_{j2})\sigma_j \otimes \sigma_3.$$

□

Lemma B.3. *Let θ be a variable with uniform distribution in $[0, 2\pi]$. Let $A, C : \mathcal{H}_2 \to \mathcal{H}_2$ be arbitrary linear operators and let $B = D = \sigma_j$ be arbitrary Pauli matrices, where $j \in \{0, 1, 2, 3\}$. Then*

$$\mathbb{E}_\theta Tr[WAW^\dagger B]Tr[WCW^\dagger D] = \frac{1}{2\pi}\int_0^{2\pi} Tr[WAW^\dagger B]Tr[WCW^\dagger D]d\theta \tag{9}$$

$$= \left[\frac{1}{2} + \frac{\delta_{j0} + \delta_{jk}}{2}\right] Tr[AB]Tr[CD] + \left[-\frac{1}{2} + \frac{\delta_{j0} + \delta_{jk}}{2}\right] Tr[AB\sigma_k]Tr[CD\sigma_k], \tag{10}$$

where $W = e^{-i\theta\sigma_k}$ and $k \in \{1, 2, 3\}$.

Proof. First we simply replace the term $W = e^{-i\theta\sigma_k} = I\cos\theta - i\sigma_k\sin\theta$.

$$\begin{aligned} &\frac{1}{2\pi}\int_0^{2\pi} d\theta \text{Tr}[WAW^\dagger B]\text{Tr}[WCW^\dagger D] \\ =& \frac{1}{2\pi}\int_0^{2\pi} d\theta \text{Tr}[(I\cos\theta - i\sigma_k\sin\theta)A(I\cos\theta + i\sigma_k\sin\theta)B] \cdot \text{Tr}[(I\cos\theta - i\sigma_k\sin\theta)C(I\cos\theta + i\sigma_k\sin\theta)D] \\ =& \frac{1}{2\pi}\int_0^{2\pi} d\theta \left\{\cos^2\theta\text{Tr}[AB] - i\sin\theta\cos\theta\text{Tr}[\sigma_k AB] + i\sin\theta\cos\theta\text{Tr}[A\sigma_k B] + \sin^2\theta\text{Tr}[\sigma_k A\sigma_k B]\right\} \\ &\cdot \left\{\cos^2\theta\text{Tr}[CD] - i\sin\theta\cos\theta\text{Tr}[\sigma_k CD] + i\sin\theta\cos\theta\text{Tr}[C\sigma_k D] + \sin^2\theta\text{Tr}[\sigma_k C\sigma_k D]\right\}. \end{aligned}$$

We remark that:

$$\frac{1}{2\pi}\int_0^{2\pi} d\theta \cos^4\theta = \frac{3}{8}, \tag{11}$$

$$\frac{1}{2\pi}\int_0^{2\pi} d\theta \sin^4\theta = \frac{3}{8}, \tag{12}$$

$$\frac{1}{2\pi}\int_0^{2\pi} d\theta \cos^2\theta \sin^2\theta = \frac{1}{8}, \tag{13}$$

$$\frac{1}{2\pi}\int_0^{2\pi} d\theta \cos^3\theta \sin\theta = 0, \tag{14}$$

$$\frac{1}{2\pi}\int_0^{2\pi} d\theta \cos\theta \sin^3\theta = 0. \tag{15}$$

Then

$$\begin{aligned}
\text{The integration term} =& \frac{3}{8}\text{Tr}[AB]\text{Tr}[CD] + \frac{3}{8}\text{Tr}[\sigma_k A\sigma_k B]\text{Tr}[\sigma_k C\sigma_k D] \\
&+\frac{1}{8}\text{Tr}[AB]\text{Tr}[\sigma_k C\sigma_k D] + \frac{1}{8}\text{Tr}[\sigma_k A\sigma_k B]\text{Tr}[CD] \\
&-\frac{1}{8}\text{Tr}[\sigma_k AB]\text{Tr}[\sigma_k CD] - \frac{1}{8}\text{Tr}[A\sigma_k B]\text{Tr}[C\sigma_k D] \\
&+\frac{1}{8}\text{Tr}[\sigma_k AB]\text{Tr}[C\sigma_k D] + \frac{1}{8}\text{Tr}[A\sigma_k B]\text{Tr}[\sigma_k CD] \\
=& \text{Tr}[AB]\text{Tr}[CD]\left[\frac{1}{2} + \frac{\delta_{j0}+\delta_{jk}}{2}\right] \\
&+\text{Tr}[AB\sigma_k]\text{Tr}[CD\sigma_k]\left[-\frac{1}{2} + \frac{\delta_{j0}+\delta_{jk}}{2}\right].
\end{aligned}$$

The last equation is derived by noticing that for $B = \sigma_j$,

$$\begin{aligned}
\text{Tr}[\sigma_k A\sigma_k B] &= \text{Tr}[A\sigma_k\sigma_j\sigma_k] \\
&= [2(\delta_{j0}+\delta_{jk}) - 1]\cdot\text{Tr}[A\sigma_j] \\
&= [2(\delta_{j0}+\delta_{jk}) - 1]\cdot\text{Tr}[AB], \\
\text{Tr}[\sigma_k AB] - \text{Tr}[A\sigma_k B] &= \text{Tr}[A\sigma_j\sigma_k] - \text{Tr}[A\sigma_k\sigma_j] \\
&= 2(1-\delta_{j0}-\delta_{jk})\cdot\text{Tr}[A\sigma_j\sigma_k] \\
&= 2(1-\delta_{j0}-\delta_{jk})\cdot\text{Tr}[AB\sigma_k],
\end{aligned}$$

while similar forms hold for $D = \sigma_j$,

$$\begin{aligned}
\text{Tr}[\sigma_k C\sigma_k D] &= \text{Tr}[C\sigma_k\sigma_j\sigma_k] \\
&= [2(\delta_{j0}+\delta_{jk}) - 1]\cdot\text{Tr}[C\sigma_j] \\
&= [2(\delta_{j0}+\delta_{jk}) - 1]\cdot\text{Tr}[CD], \\
\text{Tr}[\sigma_k CD] - \text{Tr}[C\sigma_k D] &= \text{Tr}[C\sigma_j\sigma_k] - \text{Tr}[C\sigma_k\sigma_j] \\
&= 2(1-\delta_{j0}-\delta_{jk})\cdot\text{Tr}[C\sigma_j\sigma_k] \\
&= 2(1-\delta_{j0}-\delta_{jk})\cdot\text{Tr}[CD\sigma_k].
\end{aligned}$$

□

Lemma B.4. *Let θ be a variable with uniform distribution in $[0, 2\pi]$. Let $A, C : \mathcal{H}_2 \to \mathcal{H}_2$ be arbitrary linear operators and let $B = D = \sigma_j$ be arbitrary Pauli matrices, where $j \in \{0, 1, 2, 3\}$. Then*

$$\mathbb{E}_\theta Tr[GAW^\dagger B]Tr[GCW^\dagger D] = \frac{1}{2\pi}\int_0^{2\pi} Tr[GAW^\dagger B]Tr[GCW^\dagger D]d\theta$$

$$= \left[\frac{1}{2} - \frac{\delta_{j0}+\delta_{jk}}{2}\right] Tr[AB]Tr[CD] + \left[-\frac{1}{2} - \frac{\delta_{j0}+\delta_{jk}}{2}\right] Tr[AB\sigma_k]Tr[CD\sigma_k],$$

where $W = e^{-i\theta\sigma_k}$, $G = \frac{\partial W}{\partial\theta}$ and $k \in \{1, 2, 3\}$.

Proof. First we simply replace the term $W = e^{-i\theta\sigma_k} = I\cos\theta - i\sigma_k\sin\theta$ and $G = \frac{\partial W}{\partial\theta} = -I\sin\theta - i\sigma_k\cos\theta$.

$$
\begin{aligned}
&\frac{1}{2\pi}\int_0^{2\pi} d\theta \mathrm{Tr}[GAW^\dagger B]\mathrm{Tr}[GCW^\dagger D] \\
=&\frac{1}{2\pi}\int_0^{2\pi} d\theta \mathrm{Tr}[(-I\sin\theta - i\sigma_k\cos\theta)A(I\cos\theta + i\sigma_k\sin\theta)B]\mathrm{Tr}[(-I\sin\theta - i\sigma_k\cos\theta)C(I\cos\theta + i\sigma_k\sin\theta)D] \\
=&\frac{1}{2\pi}\int_0^{2\pi} d\theta \left\{-\sin\theta\cos\theta\mathrm{Tr}[AB] - i\cos^2\theta\mathrm{Tr}[\sigma_k AB] - i\sin^2\theta\mathrm{Tr}[A\sigma_k B] + \cos\theta\sin\theta\mathrm{Tr}[\sigma_k A\sigma_k B]\right\} \\
&\cdot \left\{-\sin\theta\cos\theta\mathrm{Tr}[CD] - i\cos^2\theta\mathrm{Tr}[\sigma_k CD] - i\sin^2\theta\mathrm{Tr}[C\sigma_k D] + \cos\theta\sin\theta\mathrm{Tr}[\sigma_k C\sigma_k D]\right\}.
\end{aligned}
$$

The integration above could be simplified using equations 11-15,

$$
\begin{aligned}
\text{The integration term} =&\frac{1}{8}\mathrm{Tr}[AB]\mathrm{Tr}[CD] + \frac{1}{8}\mathrm{Tr}[\sigma_k A\sigma_k B]\mathrm{Tr}[\sigma_k C\sigma_k D] \\
&-\frac{1}{8}\mathrm{Tr}[AB]\mathrm{Tr}[\sigma_k C\sigma_k D] - \frac{1}{8}\mathrm{Tr}[\sigma_k A\sigma_k B]\mathrm{Tr}[CD] \\
&-\frac{3}{8}\mathrm{Tr}[\sigma_k AB]\mathrm{Tr}[\sigma_k CD] - \frac{3}{8}\mathrm{Tr}[A\sigma_k B]\mathrm{Tr}[C\sigma_k D] \\
&-\frac{1}{8}\mathrm{Tr}[\sigma_k AB]\mathrm{Tr}[C\sigma_k D] - \frac{1}{8}\mathrm{Tr}[A\sigma_k B]\mathrm{Tr}[\sigma_k CD] \\
=&\mathrm{Tr}[AB]\mathrm{Tr}[CD]\left[\frac{1}{2} - \frac{\delta_{j0}+\delta_{jk}}{2}\right] \\
&+\mathrm{Tr}[AB\sigma_k]\mathrm{Tr}[CD\sigma_k]\left[-\frac{1}{2} - \frac{\delta_{j0}+\delta_{jk}}{2}\right].
\end{aligned}
$$

The last equation is derived by noticing that for $B = \sigma_j$,

$$
\begin{aligned}
\mathrm{Tr}[\sigma_k A\sigma_k B] &= \mathrm{Tr}[A\sigma_k\sigma_j\sigma_k] \\
&= [2(\delta_{j0}+\delta_{jk}) - 1]\cdot\mathrm{Tr}[A\sigma_j] \\
&= [2(\delta_{j0}+\delta_{jk}) - 1]\cdot\mathrm{Tr}[AB], \\
\mathrm{Tr}[\sigma_k AB] + \mathrm{Tr}[A\sigma_k B] &= \mathrm{Tr}[A\sigma_j\sigma_k] + \mathrm{Tr}[A\sigma_k\sigma_j] \\
&= 2(\delta_{j0}+\delta_{jk})\cdot\mathrm{Tr}[A\sigma_j\sigma_k] \\
&= 2(\delta_{j0}+\delta_{jk})\cdot\mathrm{Tr}[AB\sigma_k],
\end{aligned}
$$

while similar forms hold for $D = \sigma_j$,

$$
\begin{aligned}
\mathrm{Tr}[\sigma_k C\sigma_k D] &= \mathrm{Tr}[C\sigma_k\sigma_j\sigma_k] \\
&= [2(\delta_{j0}+\delta_{jk}) - 1]\cdot\mathrm{Tr}[C\sigma_j] \\
&= [2(\delta_{j0}+\delta_{jk}) - 1]\cdot\mathrm{Tr}[CD], \\
\mathrm{Tr}[\sigma_k CD] + \mathrm{Tr}[C\sigma_k D] &= \mathrm{Tr}[C\sigma_j\sigma_k] - \mathrm{Tr}[C\sigma_k\sigma_j] \\
&= 2(\delta_{j0}+\delta_{jk})\cdot\mathrm{Tr}[C\sigma_j\sigma_k] \\
&= 2(\delta_{j0}+\delta_{jk})\cdot\mathrm{Tr}[CD\sigma_k].
\end{aligned}
$$

□

C THE PROOF OF THEOREM 3.1

Now we begin the proof of Theorem 3.1.

Proof. Firstly we remark that by Lemma C.1, each partial derivative is calculated as

$$
\frac{\partial f_{\mathrm{TT}}}{\partial\theta_j^{(k)}} = \frac{1}{2}\left(\mathrm{Tr}[O\cdot V_{\mathrm{TT}}(\boldsymbol{\theta}_+)\rho_{\mathrm{in}}V_{\mathrm{TT}}(\boldsymbol{\theta}_+)^\dagger] - \mathrm{Tr}[O\cdot V_{\mathrm{TT}}(\boldsymbol{\theta}_-)\rho_{\mathrm{in}}V_{\mathrm{TT}}(\boldsymbol{\theta}_-)^\dagger]\right),
$$

since the expectation of the quantum observable is bounded by $[-1, 1]$, the square of the partial derivative could be easily bounded as:

$$\left(\frac{\partial f_{\text{TT}}}{\partial \theta_j^{(k)}}\right)^2 \leq 1.$$

By summing up $2n-1$ parameters, we obtain

$$\|\nabla_{\boldsymbol{\theta}} f\|^2 = \sum_{j,k} \left(\frac{\partial f}{\partial \theta_j^{(k)}}\right)^2 \leq 2n-1.$$

On the other side, the lower bound could be derived as follows,

$$\mathbb{E}_{\boldsymbol{\theta}} \|\nabla_{\boldsymbol{\theta}} f_{\text{TT}}\|^2 \geq \sum_{j=1}^{1+\log n} \mathbb{E}_{\boldsymbol{\theta}} \left(\frac{\partial f_{\text{TT}}}{\partial \theta_j^{(1)}}\right)^2 \tag{16}$$

$$= \sum_{j=1}^{1+\log n} 4\mathbb{E}_{\boldsymbol{\theta}} \left(f_{\text{TT}} - \frac{1}{2}\right)^2 \tag{17}$$

$$\geq \frac{1+\log n}{2n} \cdot \left(\text{Tr}\left[\sigma_{(1,0,\cdots,0)} \cdot \rho_{\text{in}}\right]^2 + \text{Tr}\left[\sigma_{(3,0,\cdots,0)} \cdot \rho_{\text{in}}\right]^2\right). \tag{18}$$

Eq. (17) is derived using Lemma C.2, and Eq. (18) is derived using Lemma C.3.

□

Now we provide the detail and the proof of Lemmas C.1, C.2, C.3.

Lemma C.1. *Consider the objective function of the QNN defined as*

$$f(\boldsymbol{\theta}) = \frac{1+\langle O\rangle}{2} = \frac{1+Tr[O \cdot V(\boldsymbol{\theta})\rho_{in}V(\boldsymbol{\theta})^\dagger]}{2},$$

where $\boldsymbol{\theta}$ encodes all parameters which participate the circuit as $e^{-i\theta_j\sigma_k}, k \in 1,2,3$, ρ_{in} denotes the input state and O is an arbitrary quantum observable. Then, the partial derivative of the function respect to the parameter θ_j could be calculated by

$$\frac{\partial f}{\partial \theta_j} = \frac{1}{2}\left(Tr[O \cdot V(\boldsymbol{\theta}_+)\rho_{in}V(\boldsymbol{\theta}_+)^\dagger] - Tr[O \cdot V(\boldsymbol{\theta}_-)\rho_{in}V(\boldsymbol{\theta}_-)^\dagger]\right),$$

where $\boldsymbol{\theta}_+ \equiv \boldsymbol{\theta} + \frac{\pi}{4}\boldsymbol{e}_j$ and $\boldsymbol{\theta}_- \equiv \boldsymbol{\theta} - \frac{\pi}{4}\boldsymbol{e}_j$.

Proof. First we assume that the circuit $V(\boldsymbol{\theta})$ consists of p parameters, and could be written in the sequence:

$$V(\boldsymbol{\theta}) = V_p(\theta_p) \cdot V_{p-1}(\theta_{p-1}) \cdots V_1(\theta_1),$$

such that each block V_j contains only one parameter.

Consider the observable defined as $O' = V_{j+1}^\dagger \cdots V_p^\dagger O V_p \cdots V_{j+1}$ and the input state defined as $\rho'_{\text{in}} = V_{j-1} \cdots V_1 \rho_{\text{in}} V_1^\dagger \cdots V_{j-1}^\dagger$. The parameter shifting rule (Crooks, 2019) provides a gradient calculation method for the single parameter case. For $f_j(\theta_j) = \text{Tr}[O' \cdot U(\theta_j)\rho'_{\text{in}}U(\theta_j)^\dagger]$, the gradient could be calculated as

$$\frac{df_j}{d\theta_j} = f_j(\theta_j + \frac{\pi}{4}) - f_j(\theta_j - \frac{\pi}{4}).$$

Thus, by inserting the form of O' and ρ'_{in}, we could obtain

$$\frac{\partial f}{\partial \theta_j} = \frac{df_j}{d\theta_j} = f_j(\theta_j + \frac{\pi}{4}) - f_j(\theta_j - \frac{\pi}{4}) = \frac{1}{2}\left(\text{Tr}[O \cdot V(\boldsymbol{\theta}_+)\rho_{\text{in}}V(\boldsymbol{\theta}_+)^\dagger] - \text{Tr}[O \cdot V(\boldsymbol{\theta}_-)\rho_{\text{in}}V(\boldsymbol{\theta}_-)^\dagger]\right).$$

□

Lemma C.2. *For the objective function f_{TT} defined in Eq. (4), the following formula holds for every $j \in \{1, 2, \cdots, 1 + \log n\}$:*

$$\mathbb{E}_{\boldsymbol{\theta}} \left(\frac{\partial f_{TT}}{\partial \theta_j^{(1)}} \right)^2 = 4 \cdot \mathbb{E}_{\boldsymbol{\theta}} (f_{TT} - \frac{1}{2})^2, \tag{19}$$

where the expectation is taken for all parameters in θ with uniform distribution in $[0, 2\pi]$.

Proof. First we rewrite the formulation of f_{TT} in detail:

$$f_{\text{TT}} = \frac{1}{2} + \frac{1}{2} \text{Tr} \left[\sigma_{(3,0,\cdots,0)} \cdot V_{m+1} C X_m \cdots C X_1 V_1 \cdot \rho_{\text{in}} \cdot V_1^\dagger C X_1^\dagger \cdots C X_m^\dagger V_{m+1}^\dagger \right], \tag{20}$$

where $m = \log n$ and we denote

$$\sigma_{(i_1, i_2, \cdots, i_n)} \equiv \sigma_{i_1} \otimes \sigma_{i_2} \otimes \cdots \otimes \sigma_{i_n}.$$

The operation V_ℓ consists of $n \cdot 2^{1-\ell}$ single qubit rotations $W_\ell^{(j)} = e^{-i\sigma_2 \theta_\ell^{(j)}}$ on the $(j-1) \cdot 2^{\ell-1} + 1$-th qubit, where $j = 1, 2, \cdots, n \cdot 2^{1-\ell}$. The operation CX_ℓ consists of $n \cdot 2^{-\ell}$ CNOT gates, where each of them acts on the $(j-1) \cdot 2^\ell + 1$-th and $(j - 0.5) \cdot 2^\ell + 1$-th qubit, for $j = 1, 2, \cdots, n \cdot 2^{-\ell}$.

Now we focus on the partial derivative of the function f to the parameter $\theta_j^{(1)}$. We have:

$$\frac{\partial f_{\text{TT}}}{\partial \theta_j^{(1)}} = \frac{1}{2} \text{Tr} \left[\sigma_{(3,0,\cdots,0)} \cdot V_{m+1} C X_m \cdots \frac{\partial V_j}{\theta_j^{(1)}} \cdots C X_1 V_1 \cdot \rho_{\text{in}} \cdot V_1^\dagger C X_1^\dagger \cdots V_j^\dagger \cdots C X_m^\dagger V_{m+1}^\dagger \right] \tag{21}$$

$$+ \frac{1}{2} \text{Tr} \left[\sigma_{(3,0,\cdots,0)} \cdot V_{m+1} C X_m \cdots V_j \cdots C X_1 V_1 \cdot \rho_{\text{in}} \cdot V_1^\dagger C X_1^\dagger \cdots \frac{\partial V_j^\dagger}{\theta_j^{(1)}} \cdots C X_m^\dagger V_{m+1}^\dagger \right] \tag{22}$$

$$= \text{Tr} \left[\sigma_{(3,0,\cdots,0)} \cdot V_{m+1} C X_m \cdots \frac{\partial V_j}{\theta_j^{(1)}} \cdots C X_1 V_1 \cdot \rho_{\text{in}} \cdot V_1^\dagger C X_1^\dagger \cdots V_j^\dagger \cdots C X_m^\dagger V_{m+1}^\dagger \right]. \tag{23}$$

The Eq. (23) holds because both terms in (21) and (22) except ρ_{in} are real matrices, and $\rho_{\text{in}} = \rho_{\text{in}}^\dagger$. The key idea to derive $\mathbb{E}_{\boldsymbol{\theta}} \left(\frac{\partial f_{\text{TT}}}{\partial \theta_j^{(1)}} \right)^2 = 4 \cdot \mathbb{E}_\theta (f_{\text{TT}} - \frac{1}{2})^2$ is that for cases $B = D = \sigma_j \in \{\sigma_1, \sigma_3\}$, the term $\frac{\delta_{j0} + \delta_{j2}}{2} = 0$, which means Lemma B.3 and Lemma B.4 collapse to the same formulation:

$$\begin{aligned} \mathbb{E}_\theta \text{Tr}[W A W^\dagger B] \text{Tr}[W C W^\dagger D] &= \mathbb{E}_\theta \text{Tr}[G A W^\dagger B] \text{Tr}[G C W^\dagger D] \\ &= \frac{1}{2} \text{Tr}[A \sigma_1] \text{Tr}[C \sigma_1] + \frac{1}{2} \text{Tr}[A \sigma_3] \text{Tr}[C \sigma_3]. \end{aligned}$$

Now we write the analysis in detail.

$$\mathbb{E}_{\boldsymbol{\theta}} \left[\left(\frac{\partial f_{\text{TT}}}{\partial \theta_j^{(1)}} \right)^2 - 4 (f_{\text{TT}} - \frac{1}{2})^2 \right] \tag{24}$$

$$= \mathbb{E}_{\boldsymbol{\theta}_1} \cdots \mathbb{E}_{\boldsymbol{\theta}_m} \mathbb{E}_{\theta_{m+1}^{(1)}} \text{Tr} \left[\sigma_{(3,0,\cdots,0)} \cdot V_{m+1} C X_m A_m C X_m V_{m+1}^\dagger \right]^2 \tag{25}$$

$$- \mathbb{E}_{\boldsymbol{\theta}_1} \cdots \mathbb{E}_{\boldsymbol{\theta}_m} \mathbb{E}_{\theta_{m+1}^{(1)}} \text{Tr} \left[\sigma_{(3,0,\cdots,0)} \cdot V_{m+1} C X_m B_m C X_m V_{m+1}^\dagger \right]^2 \tag{26}$$

$$= \mathbb{E}_{\boldsymbol{\theta}_1} \cdots \mathbb{E}_{\boldsymbol{\theta}_m} \left\{ \frac{1}{2} \text{Tr} \left[\sigma_{(3,0,\cdots,0,3,0,\cdots,0)} \cdot A_m \right]^2 + \frac{1}{2} \text{Tr} \left[\sigma_{(1,0,\cdots,0,0,0,\cdots,0)} \cdot A_m \right]^2 \right\} \tag{27}$$

$$- \mathbb{E}_{\boldsymbol{\theta}_1} \cdots \mathbb{E}_{\boldsymbol{\theta}_m} \left\{ \frac{1}{2} \text{Tr} \left[\sigma_{(3,0,\cdots,0,3,0,\cdots,0)} \cdot B_m \right]^2 + \frac{1}{2} \text{Tr} \left[\sigma_{(1,0,\cdots,0,0,0,\cdots,0)} \cdot B_m \right]^2 \right\}, \tag{28}$$

where

$$A_m = V_m CX_{m-1} \cdots \frac{\partial V_j}{\theta_j^{(1)}} \cdots CX_1 V_1 \cdot \rho_{\text{in}} \cdot V_1^\dagger CX_1^\dagger \cdots V_j^\dagger \cdots CX_{m-1}^\dagger V_m^\dagger,$$
$$B_m = V_m CX_{m-1} \cdots V_j \cdots CX_1 V_1 \cdot \rho_{\text{in}} \cdot V_1^\dagger CX_1^\dagger \cdots V_j^\dagger \cdots CX_{m-1}^\dagger V_m^\dagger.$$

Eq. (27) and Eq. (28) are derived using the collapsed form of Lemma B.1:

$$\text{CNOT}(\sigma_1 \otimes \sigma_0)\text{CNOT}^\dagger = \sigma_1 \otimes \sigma_0,\ \text{CNOT}(\sigma_3 \otimes \sigma_0)\text{CNOT}^\dagger = \sigma_3 \otimes \sigma_3,$$

and $\boldsymbol{\theta}_\ell$ denotes the vector consisted with all parameters in the ℓ-th layer. The integration could be performed for parameters $\{\boldsymbol{\theta}_m, \boldsymbol{\theta}_{m-1}, \cdots, \boldsymbol{\theta}_{j+1}\}$. It is not hard to find that after the integration of the parameters $\boldsymbol{\theta}_{j+1}$, the term $\text{Tr}[\sigma_{\boldsymbol{i}} \cdot A_j]^2$ and $\text{Tr}[\sigma_{\boldsymbol{i}} \cdot B_j]^2$ have the opposite coefficients. Besides, the first index of each Pauli tensor product $\sigma_{(i_1, i_2, \cdots, i_n)}$ could only be $i_1 \in \{1, 3\}$ because of the Lemma B.3. So we could write

$$\mathbb{E}_{\boldsymbol{\theta}} \left[\left(\frac{\partial f_{\text{TT}}}{\partial \theta_j^{(1)}} \right)^2 - 4(f_{\text{TT}} - \frac{1}{2})^2 \right] \tag{29}$$

$$= \mathbb{E}_{\boldsymbol{\theta}_1} \cdots \mathbb{E}_{\boldsymbol{\theta}_j} \left\{ \sum_{i_1 \in \{1,3\}} \sum_{i_2=0}^{3} \cdots \sum_{i_n=0}^{3} a_{\boldsymbol{i}} \text{Tr}\left[\sigma_{\boldsymbol{i}} \cdot A_j\right]^2 - a_{\boldsymbol{i}} \text{Tr}\left[\sigma_{\boldsymbol{i}} \cdot B_j\right]^2 \right\} \tag{30}$$

where

$$\begin{aligned} A_j &= \frac{\partial V_j}{\partial \theta_j^{(1)}} CX_{j-1} \cdots CX_1 V_1 \cdot \rho_{\text{in}} \cdot V_1^\dagger CX_1^\dagger \cdots CX_{j-1} V_j^\dagger \\ &= (G_j^{(1)} \otimes I^{\otimes(n-1)}) A_j^{/\theta_j^{(1)}} (W_j^{(1)\dagger} \otimes I^{\otimes(n-1)}), \\ B_j &= V_j CX_{j-1} \cdots CX_1 V_1 \cdot \rho_{\text{in}} \cdot V_1^\dagger CX_1^\dagger \cdots CX_{j-1} V_j^\dagger \\ &= (W_j^{(1)} \otimes I^{\otimes(n-1)}) A_j^{/\theta_j^{(1)}} (W_j^{(1)\dagger} \otimes I^{\otimes(n-1)}), \end{aligned}$$

and $a_{\boldsymbol{i}}$ is the coefficient of the term $\text{Tr}\left[\sigma_{\boldsymbol{i}} \cdot A_j\right]^2$. We denote $G_j^{(1)} = \frac{\partial W_j^{(1)}}{\partial \theta_j^{(1)}}$ and use $A_j^{/\theta_j^{(1)}}$ to denote the rest part of A_j and B_j. By Lemma B.3 and Lemma B.4, we have

$$\mathbb{E}_{\theta_j^{(1)}} \left[\text{Tr}\left[\sigma_{\boldsymbol{i}} \cdot A_j\right]^2 - \text{Tr}\left[\sigma_{\boldsymbol{i}} \cdot B_j\right]^2 \right] = 0,$$

since for the case $i_1 \in \{1, 3\}$, the term $\frac{\delta_{i_1 0} + \delta_{i_1 2}}{2} = 0$, which means Lemma B.3 and Lemma B.4 have the same formulation. Then, we derive the Eq. (19).

□

Lemma C.3. *For the loss function f defined in (4), we have:*

$$\mathbb{E}_{\boldsymbol{\theta}} (f_{TT} - \frac{1}{2})^2 \geq \frac{\left\{ Tr\left[\sigma_{(1,0,\cdots,0)} \cdot \rho_{in}\right] \right\}^2 + \left\{ Tr\left[\sigma_{(3,0,\cdots,0)} \cdot \rho_{in}\right] \right\}^2}{8n}, \tag{31}$$

where we denote

$$\sigma_{(i_1, i_2, \cdots, i_n)} \equiv \sigma_{i_1} \otimes \sigma_{i_2} \otimes \cdots \otimes \sigma_{i_n},$$

and the expectation is taken for all parameters in θ with uniform distributions in $[0, 2\pi]$.

Proof. First we expand the function f_{TT} in detail,

$$f_{\text{TT}} = \frac{1}{2} + \frac{1}{2} \text{Tr}\left[\sigma_{(3,0,\cdots,0)} \cdot V_{m+1} CX_m \cdots CX_1 V_1 \cdot \rho_{\text{in}} \cdot V_1^\dagger CX_1^\dagger \cdots CX_m^\dagger V_{m+1}^\dagger \right], \tag{32}$$

where $m = \log n$. Now we consider the expectation of $(f - \frac{1}{2})^2$ under the uniform distribution for $\theta_{m+1}^{(1)}$:

$$\mathbb{E}_{\theta_{m+1}^{(1)}}(f_{\text{TT}} - \frac{1}{2})^2 = \frac{1}{4}\mathbb{E}_{\theta_{m+1}^{(1)}}\left\{\text{Tr}\left[\sigma_{(3,0,\cdots,0)} \cdot V_{m+1}CX_m \cdots CX_1V_1 \cdot \rho_{\text{in}} \cdot V_1^\dagger CX_1^\dagger \cdots CX_m^\dagger V_{m+1}^\dagger\right]\right\}^2 \tag{33}$$

$$= \frac{1}{8}\left\{\text{Tr}\left[\sigma_{(3,0,\cdots,0)} \cdot A'\right]\right\}^2 + \frac{1}{8}\left\{\text{Tr}\left[\sigma_{(1,0,\cdots,0)} \cdot A'\right]\right\}^2 \tag{34}$$

$$= \frac{1}{8}\left\{\text{Tr}\left[\sigma_{(3,0,\cdots,0,3,0,\cdots,0)} \cdot A\right]\right\}^2 + \frac{1}{8}\left\{\text{Tr}\left[\sigma_{(1,0,\cdots,0,0,\cdots,0)} \cdot A\right]\right\}^2 \tag{35}$$

$$\geq \frac{1}{8}\left\{\text{Tr}\left[\sigma_{(1,0,\cdots,0,0,\cdots,0)} \cdot A\right]\right\}^2, \tag{36}$$

where

$$A' = CX_mV_m \cdots CX_1V_1 \cdot \rho_{\text{in}} \cdot V_1^\dagger CX_1^\dagger \cdots V_mCX_m^\dagger, \tag{37}$$

$$A = V_m \cdots CX_1V_1 \cdot \rho_{\text{in}} \cdot V_1^\dagger CX_1^\dagger \cdots V_m. \tag{38}$$

Eq. (34) is derived using Lemma B.3, and Eq. (35) is derived using the collapsed form of Lemma B.1:

$$\text{CNOT}(\sigma_1 \otimes \sigma_0)\text{CNOT}^\dagger = \sigma_1 \otimes \sigma_0,\ \text{CNOT}(\sigma_3 \otimes \sigma_0)\text{CNOT}^\dagger = \sigma_3 \otimes \sigma_3,$$

We remark that during the integration of the parameters $\{\theta_\ell^{(j)}\}$ in each layer $\ell \in \{1, 2, \cdots, m+1\}$, the coefficient of the term $\{\text{Tr}[\sigma_{(1,0,\cdots,0)} \cdot A]\}^2$ only times a factor $1/2$ for the case $j = 1$, and the coefficient remains for the cases $j > 1$ (check Lemma B.3 for detail). Since the formulation $\{\text{Tr}[\sigma_{(1,0,\cdots,0)} \cdot A]\}^2$ remains the same when merging the operation CX_ℓ with $\sigma_{(1,0,\cdots,0)}$, for $\ell \in \{1, 2, \cdots, m\}$, we could generate the following equation,

$$\mathbb{E}_{\boldsymbol{\theta}_2} \cdots \mathbb{E}_{\boldsymbol{\theta}_{m+1}}(f - \frac{1}{2})^2 \geq (\frac{1}{2})^{m+2}\left\{\text{Tr}\left[\sigma_{(1,0,\cdots,0)} \cdot V_1 \cdot \rho_{\text{in}} \cdot V_1^\dagger\right]\right\}^2, \tag{39}$$

where $\boldsymbol{\theta}_\ell$ denotes the vector consisted with all parameters in the ℓ-th layer.

Finally by using Lemma B.3, we could integrate the parameters $\{\theta_1^{(j)}\}_{j=1}^n$ in (39):

$$\mathbb{E}_{\boldsymbol{\theta}}(f_{\text{TT}} - \frac{1}{2})^2 = \mathbb{E}_{\boldsymbol{\theta}_1} \cdots \mathbb{E}_{\boldsymbol{\theta}_{m+1}}(f_{\text{TT}} - \frac{1}{2})^2 \tag{40}$$

$$\geq \frac{\left\{\text{Tr}\left[\sigma_{(1,0,\cdots,0)} \cdot \rho_{\text{in}}\right]\right\}^2 + \left\{\text{Tr}\left[\sigma_{(3,0,\cdots,0)} \cdot \rho_{\text{in}}\right]\right\}^2}{2^{m+3}} \tag{41}$$

$$= \frac{\left\{\text{Tr}\left[\sigma_{(1,0,\cdots,0)} \cdot \rho_{\text{in}}\right]\right\}^2 + \left\{\text{Tr}\left[\sigma_{(3,0,\cdots,0)} \cdot \rho_{\text{in}}\right]\right\}^2}{8n}. \tag{42}$$

□

D THE QUANTUM INPUT MODEL

For the convenience of the analysis, we consider the encoding model that the number of alternating layers is L. The model begins with the inital state $|0\rangle^{\otimes n}$, where n is the number of the qubit. Then we employ the X gate to each qubit which transform the state into $|1\rangle^{\otimes n}$. Next we employ L alternating layer operations, each of which contains four parts: a single qubit rotation layer denoted as V_{2i-1}, a CZ gate layer denoted as CZ_2, again a single qubit rotation layer denoted as V_{2i}, and another CZ gate layer with alternating structures denoted as CZ_1, for $i \in \{1, 2, \cdots, L\}$. Each single qubit gate contains a parameter encoded in the phase: $W_j^{(k)} = e^{-i\sigma_2\beta_j^{(k)}}$, and each single qubit rotation layer could be written as

$$V_j = V_j(\boldsymbol{\beta}_j) = W_j^{(1)} \otimes W_j^{(2)} \otimes \cdots \otimes W_j^{(n)}.$$

Finally, we could mathematically define the encoding model:

$$U(\rho_{\text{in}}) = V_{2L+1}U_LU_{L-1} \cdots U_1X^{\otimes n},$$

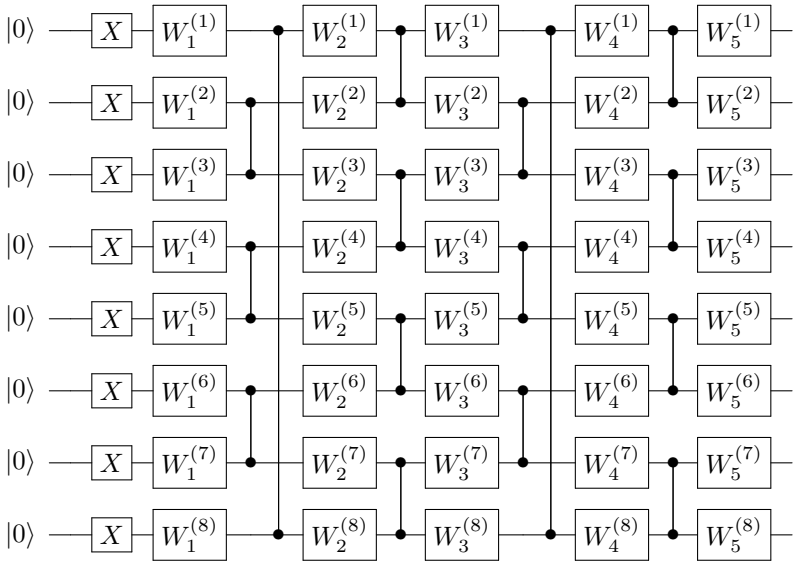

Figure 5: An example of the variational encoding model ($L = 2$ case).

where $U_j = CZ_1V_{2j}CZ_2V_{2j-1}$ is the j-th alternating layer.

By employing the encoding model illustrated in Figure 5 for the state preparation, we find that the expectation of the term $\alpha(\rho_{\text{in}})$ defined in Theorem 1.1 has the lower bound independent from the qubit number.

Theorem D.1. *Suppose the input state $\rho_{in}(\boldsymbol{\theta})$ is prepared by the L-layer encoding model illustrated in Figure 5. Then,*

$$\mathbb{E}_{\boldsymbol{\beta}}\alpha(\rho_{in}) = \mathbb{E}_{\boldsymbol{\beta}}\left(Tr\left[\sigma_{(1,0,\cdots,0)}\cdot\rho_{in}\right]^2 + Tr\left[\sigma_{(3,0,\cdots,0)}\cdot\rho_{in}\right]^2\right) \geq 2^{-2L},$$

where $\boldsymbol{\beta}$ denotes all variational parameters in the encoding circuit, and the expectation is taken for all parameters in $\boldsymbol{\beta}$ with uniform distribution in $[0, 2\pi]$.

Proof. Define $\rho_j = U_jU_{j-1}\cdots U_1X^{\otimes n}|0\rangle^{\otimes n}\langle 0|^{\otimes n}X^{\otimes n}U_1^\dagger\cdots U_{j-1}^\dagger U_j^\dagger$, for $j \in \{0, 1, \cdots, L\}$. We have:

$$\mathbb{E}_{\boldsymbol{\beta}}\left(\text{Tr}\left[\sigma_{(1,0,\cdots,0)}\cdot\rho_{\text{in}}\right]^2 + \text{Tr}\left[\sigma_{(3,0,\cdots,0)}\cdot\rho_{\text{in}}\right]^2\right) \tag{43}$$

$$= \mathbb{E}_{\boldsymbol{\beta}_1}\cdots\mathbb{E}_{\boldsymbol{\beta}_{2L+1}}\left(\text{Tr}\left[\sigma_{(1,0,\cdots,0)}\cdot V_{2L+1}\rho_L V_{2L+1}^\dagger\right]^2 + \text{Tr}\left[\sigma_{(3,0,\cdots,0)}\cdot V_{2L+1}\rho_L V_{2L+1}^\dagger\right]^2\right) \tag{44}$$

$$= \mathbb{E}_{\boldsymbol{\beta}_1}\cdots\mathbb{E}_{\boldsymbol{\beta}_{2L}}\left(\text{Tr}\left[\sigma_{(1,0,\cdots,0)}\cdot\rho_L\right]^2 + \text{Tr}\left[\sigma_{(3,0,\cdots,0)}\cdot\rho_L\right]^2\right). \tag{45}$$

$$\geq 2^{-2L}\left(\text{Tr}\left[\sigma_{(1,0,\cdots,0)}\cdot\rho_0\right]^2 + \text{Tr}\left[\sigma_{(3,0,\cdots,0)}\cdot\rho_0\right]^2\right) \tag{46}$$

$$= 2^{-2L}\cdot(0^2 + (-1)^2) = 2^{-2L}. \tag{47}$$

Eq. (44) is derived from the definition of ρ_L. Eq. (45) is derived using Lemma B.3. Eq. (46) is derived by noticing that for each $j \in \{0, 1, \cdots, L-1\}$, the following equations holds,

$$\mathbb{E}_{\boldsymbol{\beta}_1} \cdots \mathbb{E}_{\boldsymbol{\beta}_{2j+2}} \left(\text{Tr}\left[\sigma_{(1,0,\cdots,0)} \cdot \rho_{j+1}\right]^2 + \text{Tr}\left[\sigma_{(3,0,\cdots,0)} \cdot \rho_{j+1}\right]^2 \right) \tag{48}$$

$$= \mathbb{E}_{\boldsymbol{\beta}_1} \cdots \mathbb{E}_{\boldsymbol{\beta}_{2j+2}} \left(\text{Tr}\left[\sigma_{(1,0,\cdots,0)} \cdot U_{j+1}\rho_j U_{j+1}^\dagger\right]^2 + \text{Tr}\left[\sigma_{(3,0,\cdots,0)} \cdot U_{j+1}\rho_j U_{j+1}^\dagger\right]^2 \right) \tag{49}$$

$$= \mathbb{E}_{\boldsymbol{\beta}_1} \cdots \mathbb{E}_{\boldsymbol{\beta}_{2j+2}} \text{Tr}\left[\sigma_{(1,0,\cdots,0)} \cdot CZ_1 V_{2j+2} CZ_2 V_{2j+1} \rho_j V_{2j+1}^\dagger CZ_2 V_{2j+2}^\dagger CZ_1\right]^2 \tag{50}$$

$$+ \mathbb{E}_{\boldsymbol{\beta}_1} \cdots \mathbb{E}_{\boldsymbol{\beta}_{2j+2}} \text{Tr}\left[\sigma_{(3,0,\cdots,0)} \cdot CZ_1 V_{2j+2} CZ_2 V_{2j+1} \rho_j V_{2j+1}^\dagger CZ_2 V_{2j+2}^\dagger CZ_1\right]^2 \tag{51}$$

$$= \mathbb{E}_{\boldsymbol{\beta}_1} \cdots \mathbb{E}_{\boldsymbol{\beta}_{2j+2}} \text{Tr}\left[\sigma_{(1,3,0,\cdots,0)} \cdot V_{2j+2} CZ_2 V_{2j+1} \rho_j V_{2j+1}^\dagger CZ_2 V_{2j+2}^\dagger\right]^2 \tag{52}$$

$$+ \mathbb{E}_{\boldsymbol{\beta}_1} \cdots \mathbb{E}_{\boldsymbol{\beta}_{2j+2}} \text{Tr}\left[\sigma_{(3,0,\cdots,0)} \cdot V_{2j+2} CZ_2 V_{2j+1} \rho_j V_{2j+1}^\dagger CZ_2 V_{2j+2}^\dagger\right]^2 \tag{53}$$

$$\geq \mathbb{E}_{\boldsymbol{\beta}_1} \cdots \mathbb{E}_{\boldsymbol{\beta}_{2j+2}} \text{Tr}\left[\sigma_{(3,0,\cdots,0)} \cdot V_{2j+2} CZ_2 V_{2j+1} \rho_j V_{2j+1}^\dagger CZ_2 V_{2j+2}^\dagger\right]^2 \tag{54}$$

$$= \mathbb{E}_{\boldsymbol{\beta}_1} \cdots \mathbb{E}_{\boldsymbol{\beta}_{2j+1}} \left(\frac{1}{2}\text{Tr}\left[\sigma_{(1,0,\cdots,0)} \cdot CZ_2 V_{2j+1} \rho_j V_{2j+1}^\dagger CZ_2\right]^2 + \frac{1}{2}\text{Tr}\left[\sigma_{(3,0,\cdots,0)} \cdot CZ_2 V_{2j+1} \rho_j V_{2j+1}^\dagger CZ_2\right]^2 \right) \tag{55}$$

$$= \mathbb{E}_{\boldsymbol{\beta}_1} \cdots \mathbb{E}_{\boldsymbol{\beta}_{2j+1}} \left(\frac{1}{2}\text{Tr}\left[\sigma_{(1,0,\cdots,0,3)} \cdot V_{2j+1} \rho_j V_{2j+1}^\dagger\right]^2 + \frac{1}{2}\text{Tr}\left[\sigma_{(3,0,\cdots,0)} \cdot V_{2j+1} \rho_j V_{2j+1}^\dagger\right]^2 \right) \tag{56}$$

$$\geq \mathbb{E}_{\boldsymbol{\beta}_1} \cdots \mathbb{E}_{\boldsymbol{\beta}_{2j+1}} \frac{1}{2}\text{Tr}\left[\sigma_{(3,0,\cdots,0)} \cdot V_{2j+1} \rho_j V_{2j+1}^\dagger\right]^2 \tag{57}$$

$$= \mathbb{E}_{\boldsymbol{\beta}_1} \cdots \mathbb{E}_{\boldsymbol{\beta}_{2j}} \frac{1}{4} \left(\text{Tr}\left[\sigma_{(1,0,\cdots,0)} \cdot \rho_j\right]^2 + \text{Tr}\left[\sigma_{(3,0,\cdots,0)} \cdot \rho_j\right]^2 \right). \tag{58}$$

Eq. (49) is derived from the definition of ρ_{j+1}. Eq. (50-51) are derived from the definition of U_{j+1}. Eq. (52-53) and Eq. (56) are derived using Lemma B.1. Eq. (55) and Eq. (58) are derived using Lemma B.3.

□